## Contents

### Figures ................................... vii

### Tables ..................................... ix

### About this document ................................ xi
  - Who should use this document ...................... xi
  - How this document is organized .................... xi
  - How to use this document ........................... xi
    - Determining whether a publication is current ...... xii
    - How to contact IBM service ......................... xii
  - Conventions and terminology that are used in this document ........... xii
  - Prerequisite and related information ................ xiii

### Summary of changes ............................ xix

### Chapter 1. LU 6.2 macroinstruction syntax and operands .......................... 1
  - About this chapter ................................ 1
  - How macroinstructions are described ................ 1
    - Syntax descriptions .............................. 1
    - Operand descriptions ............................. 3
    - Completion information ........................... 3
  - Coding default values ............................. 3
  - Coding comments .................................. 3
  - Coding continued lines ............................ 4
  - How to read the syntax diagrams .................... 5
  - APPCCMD CONTROL=ALLOC, QUALIFY=ALLOCD ............. 9
  - APPCCMD CONTROL=ALLOC, QUALIFY=CONVGRP .......... 21
  - APPCCMD CONTROL=ALLOC, QUALIFY=CONWIN ........... 33
  - APPCCMD CONTROL=ALLOC, QUALIFY=IMMED ............ 46
  - APPCCMD CONTROL=ALLOC, QUALIFY=WHENFREE .......... 57
  - APPCCMD CONTROL=CHECK ............................ 69
  - APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG ....... 70
  - APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV ....... 78
  - APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME ...... 85
  - APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER ....... 93
  - APPCCMD CONTROL=DEALLOC, QUALIFY=CONFIRM ........ 101
  - APPCCMD CONTROL=DEALLOC, QUALIFY=DATACON ....... 109
  - APPCCMD CONTROL=DEALLOC, QUALIFY=DATAFLU ...... 121
  - APPCCMD CONTROL=DEALLOC, QUALIFY=FLUSH .......... 132
  - APPCCMD CONTROL=DEALLOQ .......................... 139
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS ......... 148
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS ............ 154
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS ...... 163
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE ........ 168
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY ...... 174
  - APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE ...... 181
  - APPCCMD CONTROL=PREALLOC, QUALIFY=ALLOCD ...... 187
  - APPCCMD CONTROL=PREALLOC, QUALIFY=CONVGRP ..... 199
  - APPCCMD CONTROL=PREALLOC, QUALIFY=CONWIN ..... 210
  - APPCCMD CONTROL=PREALLOC, QUALIFY=IMMED ...... 222
  - APPCCMD CONTROL=PREALLOC, QUALIFY=WHENFREE .... 232
  - APPCCMD CONTROL=PREPRCV, QUALIFY=CONFIRM ...... 244
  - APPCCMD CONTROL=PREPRCV, QUALIFY=DATACON ....... 253
  - APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU ..... 265
### Chapter 2. Return codes

RCPRI and RCSEC codes .......................... 591
RTNCD, FDB2 information for LU 6.2  ............. 630

### Chapter 3. DSECTs

BIND image (ISTDBIND) .......................... 653
FMH-5 (ISTFMS) ....................................... 643
RPL extension (ISTRPL6X) .......................... 645
CNO5 session limits data structure (ISTSLCNS)  .... 653
DEFINE/DISPLAY session limits data structure (ISTSLD) 654
Restore data structure (ISTSREST) .................. 656
Status data structure (ISTSTATD) ................... 656
Feedback code data structure (ISTUSFBC) ......... 657
APPCCMD vector lists (ISTAPCVL) .................. 666
Application-ACB vector list (ISTVACBV) .......... 671
Access-method-support vector list (ISTAMSVL) .... 673
Resource-information vector list (ISTRIVL) ....... 677
Extended buffer list entry (ISTBLXEN) ............ 681

### Chapter 4. Summary of register usage

Appendix A. Architectural specifications. ............ 685
Appendix B. Accessibility ................................................................. 687

Notices ........................................................................................................ 691
Programming interface information ....................................................... 699
Policy for unsupported hardware ............................................................ 699
Trademarks ............................................................................................... 699

Bibliography ............................................................................................. 701

Index ......................................................................................................... 705

Communicating your comments to IBM ............................................... 725
Figures

1. How to code comments and continuation lines ................................................. 4
2. Layout of the RPL extension (part 1 of 3) ......................................................... 651
3. Layout of the RPL extension (part 2 of 3) ......................................................... 652
4. Layout of the RPL extension (part 3 of 3) ......................................................... 653
Tables

1. Format of WHATRCV mask ........................................ 382
2. LU 6.2 global macro variables set by ISTGAPPC ............. 579
3. Register contents upon return of control ...................... 683
About this document

This manual is designed to help customers write VTAM® application programs to use the VTAM logical unit (LU) 6.2 application programming interface (API). This manual describes the format of the macroinstructions and presents each macroinstruction in alphabetical order.

This manual explains macroinstruction syntax and parameters, return codes and responses, and identifies fields set by DSECTs. z/OS Communications Server: SNA Programmer's LU 6.2 Guide also explains how to design VTAM LU 6.2 application programs.

Who should use this document

This book is for programmers (such as application or system programmers) who code VTAM application programs. This audience can include programmers who are modifying existing programs or writing new ones.

You should be familiar with LU 6.2 architecture before you write LU 6.2 programs. z/OS Communications Server: SNA Programmer's LU 6.2 Guide provides this familiarity and is, therefore, a corequisite for using the z/OS Communications Server: SNA Programmer's LU 6.2 Reference. You should also be familiar with information in z/OS Communications Server: SNA Programming.

You should also be familiar with the information in the assembler language documentation for your operating system.

How this document is organized

This document is organized into the following topics:

- **Chapter 1, “LU 6.2 macroinstruction syntax and operands,” on page 1** describes all varieties of the VTAM APPCCMD macroinstruction.
- **Chapter 2, “Return codes,” on page 591** describes the return codes.
- **Chapter 3, “DSECTs,” on page 633** describes the LU 6.2 DSECTs.
- **Chapter 4, “Summary of register usage,” on page 683** describes the summary of register usage.
- **Appendix A, “Architectural specifications,” on page 685** lists documents that provide architectural specifications for the SNA Protocol.
- **Appendix B, “Accessibility,” on page 687** describes accessibility features to help users with physical disabilities.
- “Notices” on page 691 contains notices and trademarks used in this document.
- “Bibliography” on page 701 contains descriptions of the documents in the z/OS® Communications Server library.

How to use this document

To use this document, you should be familiar with the basic concepts of telecommunication, SNA, and VTAM.
Determining whether a publication is current

As needed, IBM® updates its publications with new and changed information. For a given publication, updates to the hardcopy and associated BookManager® softcopy are usually available at the same time. Sometimes, however, the updates to hardcopy and softcopy are available at different times. The following information describes how to determine if you are looking at the most current copy of a publication:

- At the end of a publication's order number there is a dash followed by two digits, often referred to as the dash level. A publication with a higher dash level is more current than one with a lower dash level. For example, in the publication order number GC28-1747-07, the dash level 07 means that the publication is more current than previous levels, such as 05 or 04.

- If a hardcopy publication and a softcopy publication have the same dash level, it is possible that the softcopy publication is more current than the hardcopy publication. Check the dates shown in the Summary of Changes. The softcopy publication might have a more recently dated Summary of Changes than the hardcopy publication.

- To compare softcopy publications, you can check the last 2 characters of the publication's file name (also called the book name). The higher the number, the more recent the publication. Also, next to the publication titles in the CD-ROM booklet and the readme files, there is an asterisk (*) that indicates whether a publication is new or changed.

How to contact IBM service

For immediate assistance, visit this website: http://www.software.ibm.com/network/commserver/support/

Most problems can be resolved at this website, where you can submit questions and problem reports electronically, and access a variety of diagnosis information.

For telephone assistance in problem diagnosis and resolution (in the United States or Puerto Rico), call the IBM Software Support Center anytime (1-800-IBM-SERV). You will receive a return call within 8 business hours (Monday – Friday, 8:00 a.m. – 5:00 p.m., local customer time).

Outside the United States or Puerto Rico, contact your local IBM representative or your authorized IBM supplier.

If you would like to provide feedback on this publication, see “Communicating your comments to IBM” on page 725.

Conventions and terminology that are used in this document

Commands in this book that can be used in both TSO and z/OS UNIX environments use the following conventions:

- When describing how to use the command in a TSO environment, the command is presented in uppercase (for example, NETSTAT).

- When describing how to use the command in a z/OS UNIX environment, the command is presented in bold lowercase (for example, netstat).

- When referring to the command in a general way in text, the command is presented with an initial capital letter (for example, Netstat).
All the exit routines described in this document are *installation-wide exit routines*. The installation-wide exit routines also called installation-wide exits, exit routines, and exits throughout this document.

The TPF logon manager, although included with VTAM, is an application program; therefore, the logon manager is documented separately from VTAM.

Samples used in this book might not be updated for each release. Evaluate a sample carefully before applying it to your system.

**Note:** In this information, you might see the term RDMA network interface card (RNIC) that is used to refer to the IBM 10GbE RoCE Express feature.

For definitions of the terms and abbreviations that are used in this document, you can view the latest IBM terminology at the IBM Terminology website.

**Clarification of notes**

Information traditionally qualified as Notes is further qualified as follows:

- **Note** Supplemental detail
- **Tip** Offers shortcuts or alternative ways of performing an action; a hint
- **Guideline** Customary way to perform a procedure
- **Rule** Something you must do; limitations on your actions
- **Restriction** Indicates certain conditions are not supported; limitations on a product or facility
- **Requirement** Dependencies, prerequisites
- **Result** Indicates the outcome

**Prerequisite and related information**

*z/OS* Communications Server function is described in the *z/OS Communications Server* library. Descriptions of those documents are listed in “Bibliography” on page 701, in the back of this document.

**Required information**

Before using this product, you should be familiar with TCP/IP, VTAM, MVS™, and UNIX System Services.
Softcopy information

Softcopy publications are available in the following collection.

<table>
<thead>
<tr>
<th>Titles</th>
<th>Order Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM System z® Redbooks Collection</td>
<td>SK3T-7876</td>
<td>The IBM Redbooks® publications selected for this CD series are taken from the IBM Redbooks inventory of over 800 books. All the Redbooks publications that are of interest to the zSeries® platform professional are identified by their authors and are included in this collection. The zSeries subject areas range from e-business application development and enablement to hardware, networking, Linux, solutions, security, parallel sysplex, and many others. For more information about the Redbooks publications, see <a href="http://www-03.ibm.com/systems/z/os/zos/zfavorites/">http://www-03.ibm.com/systems/z/os/zos/zfavorites/</a>.</td>
</tr>
</tbody>
</table>

Other documents

This information explains how z/OS references information in other documents.

When possible, this information uses cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see z/OS Information Roadmap (SA23-2299). The Roadmap describes what level of documents are supplied with each release of z/OS Communications Server, and also describes each z/OS publication.

To find the complete z/OS library, including the z/OS Information Center, see [www.ibm.com/systems/z/os/zos/bkserv/](http://www.ibm.com/systems/z/os/zos/bkserv/).

Relevant RFCs are listed in an appendix of the IP documents. Architectural specifications for the SNA protocol are listed in an appendix of the SNA documents.

The following table lists documents that might be helpful to readers.

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNA Formats</td>
<td>GA27-3136</td>
</tr>
<tr>
<td>TCP/IP Tutorial and Technical Overview</td>
<td>GG24-3376</td>
</tr>
<tr>
<td>Understanding LDAP</td>
<td>SG24-4986</td>
</tr>
<tr>
<td>z/OS Cryptographic Services System SSL Programming</td>
<td>SC24-5901</td>
</tr>
<tr>
<td>z/OS IBM Tivoli Directory Server Administration and Use for z/OS</td>
<td>SC23-6788</td>
</tr>
</tbody>
</table>
Redbooks publications

The following Redbooks publications might help you as you implement z/OS Communications Server.

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM z/OS V1R13 Communications Server TCP/IP Implementation, Volume 1: Base Functions, Connectivity, and Routing</td>
<td>SG24-7996</td>
</tr>
<tr>
<td>IBM z/OS V1R13 Communications Server TCP/IP Implementation, Volume 2: Standard Applications</td>
<td>SG24-7997</td>
</tr>
<tr>
<td>IBM z/OS V1R13 Communications Server TCP/IP Implementation, Volume 3: High Availability, Scalability, and Performance</td>
<td>SG24-7998</td>
</tr>
<tr>
<td>IBM z/OS V1R13 Communications Server TCP/IP Implementation, Volume 4: Security and Policy-Based Networking</td>
<td>SG24-7999</td>
</tr>
<tr>
<td>IBM Communication Controller Migration Guide</td>
<td>SG24-6298</td>
</tr>
<tr>
<td>IP Network Design Guide</td>
<td>SG24-2580</td>
</tr>
<tr>
<td>Managing OS/390® TCP/IP with SNMP</td>
<td>SG24-5866</td>
</tr>
<tr>
<td>Migrating Subarea Networks to an IP Infrastructure Using Enterprise Extender</td>
<td>SG24-5957</td>
</tr>
<tr>
<td>SecureWay Communications Server for OS/390 V2R8 TCP/IP: Guide to Enhancements</td>
<td>SG24-5631</td>
</tr>
<tr>
<td>SNA and TCP/IP Integration</td>
<td>SG24-5291</td>
</tr>
<tr>
<td>TCP/IP in a Sysplex</td>
<td>SG24-5235</td>
</tr>
<tr>
<td>TCP/IP Tutorial and Technical Overview</td>
<td>GG24-3376</td>
</tr>
<tr>
<td>Threadsafe Considerations for CICS</td>
<td>SG24-6351</td>
</tr>
</tbody>
</table>

Where to find related information on the Internet

z/OS

This site provides information about z/OS Communications Server release availability, migration information, downloads, and links to information about z/OS technology

http://www.ibm.com/systems/z/os/zos/
z/OS Internet Library

Use this site to view and download z/OS Communications Server documentation

www.ibm.com/systems/z/os/zos/bkserv/

IBM Communications Server product

The primary home page for information about z/OS Communications Server


IBM Communications Server product support

Use this site to submit and track problems and search the z/OS Communications Server knowledge base for Technotes, FAQs, white papers, and other z/OS Communications Server information


IBM Communications Server performance information

This site contains links to the most recent Communications Server performance reports.

http://www.ibm.com/support/docview.wss?uid=swg27005524

IBM Systems Center publications

Use this site to view and order Redbooks publications, Redpapers™, and Technotes

http://www.redbooks.ibm.com/

IBM Systems Center flashes

Search the Technical Sales Library for Techdocs (including Flashes, presentations, Technotes, FAQs, white papers, Customer Support Plans, and Skills Transfer information)

http://www.ibm.com/support/techdocs/atsmastr.nsf

RFCs

Search for and view Request for Comments documents in this section of the Internet Engineering Task Force website, with links to the RFC repository and the IETF Working Groups web page

http://www.ietf.org/rfc.html

Internet drafts

View Internet-Drafts, which are working documents of the Internet Engineering Task Force (IETF) and other groups, in this section of the Internet Engineering Task Force website

http://www.ietf.org/ID.html

Information about web addresses can also be found in information APAR III1334.

Note: Any pointers in this publication to websites are provided for convenience only and do not serve as an endorsement of these websites.
DNS websites

For more information about DNS, see the following USENET news groups and mailing addresses:

**USENET news groups**

comp.protocols.dns.bind

**BIND mailing lists**

https://lists.isc.org/mailman/listinfo/bind-users-request@isc.org.
Submit questions or answers to this forum by sending mail to bind-users@isc.org.

**BIND 9 Users (This list might not be maintained indefinitely.)**

- Subscribe by sending mail to bind9-users-request@isc.org.
- Submit questions or answers to this forum by sending mail to bind9-users@isc.org.

The z/OS Basic Skills Information Center

The z/OS Basic Skills Information Center is a web-based information resource intended to help users learn the basic concepts of z/OS, the operating system that runs most of the IBM mainframe computers in use today. The Information Center is designed to introduce a new generation of Information Technology professionals to basic concepts and help them prepare for a career as a z/OS professional, such as a z/OS systems programmer.

Specifically, the z/OS Basic Skills Information Center is intended to achieve the following objectives:

- Provide basic education and information about z/OS without charge
- Shorten the time it takes for people to become productive on the mainframe
- Make it easier for new people to learn z/OS

To access the z/OS Basic Skills Information Center, open your web browser to the following website, which is available to all users (no login required):

http://publib.boulder.ibm.com/infocenter/zos/basics/index.jsp
Summary of changes

This section describes the release enhancements that were made.

New in z/OS Version 2 Release 1

For specifics on the enhancements for z/OS Version 2, Release 1, see the following publications:

- z/OS Summary of Message and Interface Changes
- z/OS Introduction and Release Guide
- z/OS Planning for Installation
- z/OS Migration
Chapter 1. LU 6.2 macroinstruction syntax and operands

About this chapter

This chapter describes all varieties of the VTAM APPCCMD macroinstruction. Separate descriptions are included for each CONTROL and QUALIFY combination of the macroinstruction. This chapter also includes the ISTGAPPC and ISTRPL6 macroinstructions. Macroinstruction descriptions are arranged alphabetically.

The macroinstructions are coded as assembler instructions, using name, operation, and operand fields. Refer to the IBM High Level Assembler Language Reference for MVS and VM for coding guidelines.

How macroinstructions are described

Each macroinstruction description contains:

- The purpose of the macroinstruction
- General comments about its use
- Reference to tutorial chapters detailing its use
- The syntax of the macroinstruction and all parameters
- Input parameter descriptions
- RPL and RPL extension fields set by the macroinstruction
- Return and reason codes that can be returned for the macroinstruction

Syntax descriptions

The syntax for the macroinstruction is described using the following format:
Note: If you are not familiar with this type of syntax diagram, see "How to read the syntax diagrams" on page 5.

Name field

The name field provides a label for the macroinstruction. If you use a name, it must begin in column one of the macroinstruction, be followed by one or more blanks, and contain 1 to 8 characters in the following format:

<table>
<thead>
<tr>
<th>Character</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>First character</td>
<td>Alphabetical (A-Z) or the national characters @, #, or $</td>
</tr>
<tr>
<td>Second to eighth character</td>
<td>Alphabetical (A-Z), numerical (0-9), or the national characters @, #, or $</td>
</tr>
</tbody>
</table>

The macroinstruction field

The APPC CMD identifier is always coded exactly as shown. It begins in column ten and must be preceded and followed by one or more blanks.

Operand field

Required and optional operands direct information to the expansion program in the assembler. Generally, the information provided by the operand is made part of a parameter list provided to VTAM during program processing. All operands for a macroinstruction appear in the syntax diagram.

Operands can occupy columns 16-71. You must place one or more blanks after the last operand on a line. If the operands require more than one line, you must place a nonblank continuation character in column 72. (See "Coding continued lines" on page 4.)

Operands consist of a fixed character string (the operand keyword) followed by an equal sign (=) and one or more operand values. If the value is specified as name, it must follow the rules for a symbolic name detailed in IBM High Level Assembler Language Reference for MVS and VM.

Operands do not have to be coded in the order shown by the syntax diagram. For example, a macroinstruction having the operands AREALEN=data_length and AREA=data_area_address could be coded in either of two ways:

AREALEN=132, AREA=WORK

AREA=WORK, AREALEN=132

Keyword operands must be separated by commas. If you choose to omit a keyword operand, also omit the comma that would have been included with it.

When more than one value can be coded after the keyword, a parenthesis is required. The OPTCD keyword can specify the manner of processing (asynchronous or synchronous) and control use of the buffer list option. For example, code the following information to specify synchronous and buffer list options: OPTCD=(SYN,BUFFLST)
Operand descriptions

Each operand description begins with an explanation of the operand function. If the operand allows more than one fixed value that can be supplied with it, the operand description explains the effect that each value has on the action performed by the macroinstruction.

Most operands are coded by these general rules. If the format varies from these rules, the format is included with the description.

- When a quantity is indicated (for example, RECLEn=data_length), you can specify the value with:
  - Decimal integers.
  - An expression that is equal to such a value (for example, RECLEn=TEN, where TEN EQU 5*2).
  - The number of the register (enclosed by parentheses) to contain the quantity when the macroinstruction is executed. When specifying a quantity, Register notation is restricted to registers 2-12.

- When an address is indicated (for example, AREA=data_area_address), you can use any expression valid for an RX-type assembler instruction (for example, an LA instruction). Registers 1-12 can be specified for any operand that designates the address of an RPL. Register notation for all other address operands is restricted to registers 2-12.

For more information on operand formats, refer to the assembler language publication appropriate to your operating system.

Completion information

All executable macroinstructions pass return codes in registers, and most indicate status information in control block fields when they are posted complete. This status information is described at the end of the macroinstruction description. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information regarding the sequence of error checking.

Coding default values

The default values apply only to declarative macroinstructions ACB, EXLST, NIB, ISTRPL6, and RPL. All other macroinstructions (including APPCCMDs) use values specified in the macroinstruction itself or currently defined in the referenced control blocks. APPCCMDs do not have defaults; the defaults are in the underlying RPL and RPL extension.

Coding comments

Comments can contain any characters valid in the assembler language. You can write comments after the operand field, but they must meet the following criteria:

- Comments must begin in column 17 or beyond.
- Comments must be separated from the last operand in the field by one or more blanks.
- Comments must not extend into the continuation column, column 72.
Comments can be continued on more than one line by placing an asterisk (*) in column one as shown in Figure 1.

```
LABEL1  OP1  OPERAND1,OPERAND1,OPERAND3,OPERX
        AND4,OPERAND5  THIS IS ONE WAY

LABEL2  OP2  OPERAND1,OPERAND2, AND THIS X
        OPERAND3,OPERAND4  IS ANOTHER WAY
```

Figure 1. How to code comments and continuation lines

**Note:** A macroinstruction that has no operands cannot have comments on APPCCMD identifier line.

An entire line can be used for a comment; place an asterisk in the first column of the line. If you need several entire lines for comments, place an asterisk in the first column of each line and leave column 72 blank.

In this manual, the comments field is not shown in the macroinstruction syntax diagram.

### Coding continued lines

#### Procedure

Code VTAM macroinstructions in columns 1-71 of a line. Macroinstructions that exceed 71 columns can be continued on additional lines as shown in Figure 1. (Continuation characters are omitted from other examples in this manual.) When you need to continue on another line, the following steps apply:

1. Code the macroinstruction one of two ways:
   - Through column 71
   - Through any completed operand, stopping after the comma that separates the operand from those that follow
2. Enter a nonblank continuation character in column 72. The continuation character is not considered to be part of the macroinstruction.
3. Continue operands beginning in column 16 of the next line. Columns 1-15 must be blank. A continuation line that begins in column 17 or later is ignored. A comment line cannot follow a continuation line.
4. If you must continue on another line, proceed with Step 1.
5. Macroinstructions can be coded on as many lines as needed.
6. Comments can appear on every line of a continued macroinstruction.
7. Columns 73-80 can be used to code identification characters, macroinstruction sequence characters, or both.
How to read the syntax diagrams

Purpose

This section describes how to read the syntax diagrams used in this book.

• Read the diagrams from left-to-right, top-to-bottom, following the main path line. Each diagram begins on the left with double arrowheads (➡️) and ends on the right with two arrowheads facing each other (⬅️).

➡️ Syntax Diagram ➡️

• If a diagram is longer than one line, the first line ends with a single arrowhead (➡️) and the second line begins with a single arrowhead.

➡️ First Line ➡️

➡️ Second Line ➡️

• Required operands and values appear on the main path line.

➡️ REQUIRED_OPERAND ➡️

You must code required operands and values.
If your choices are greater than one, the choices are stacked vertically in alphanumeric order.
• Optional operands and values appear below the main path line.

You can choose not to code optional operands and values. If your choices are more than one, they are stacked vertically in alphanumeric order below the main path line.

• An arrow returning to the left above an operand or value on the main path line means that the operand or value can be repeated. The comma means that each operand or value must be separated from the next by a comma.

• An arrow returning to the left above a group of operands or values means that more than one can be selected, or a single one can be repeated.
A word in all uppercase is an operand or value you must spell exactly as shown. In this example, you must code **OPERAND**.

**Note:** VTAM commands are not case-sensitive. You can code them in uppercase or lowercase.

If an operand or value can be abbreviated, the abbreviation is discussed in the text associated with the syntax diagram.

- If a diagram shows a character that is not alphanumeric (parentheses, periods, commas, and equal signs), you must code the character as part of the syntax. In this example, you must code **OPERAND=(001,0.001)**.

- If a diagram shows a blank space, you must code the blank space as part of the syntax. In this example, you must code **OPERAND=(001 FIXED)**.

- Default operands and values appear above the main path line. VTAM uses the default if you omit the operand entirely.
- A word appearing in all lowercase italics is a *variable*. Where you see a variable in the syntax, you must replace it with one of its allowable names or values, as defined in the text.

- References to syntax notes appear as numbers enclosed in parentheses above the line. Do not code the parentheses or the number.

**Notes:**
1. An example of a syntax note.

- Some diagrams contain syntax fragments; these serve to break up diagrams that are too long, too complex, or too repetitious. Syntax fragment names appear in mixed case and are shown in the diagram and in the heading of the fragment. The fragment is placed below the main diagram.
APPCCMD CONTROL=ALLOC, QUALIFY=ALLOCD

Purpose

This macroinstruction allocates resources for a conversation and assigns a contention winner or contention loser session to the conversation. If a session is not available and session limits allow, VTAM activates a session for the conversation, if possible.

Usage

QUALIFY=ALLOCD is used when an application program allocates a conversation and wants VTAM to queue the request if the request cannot be met immediately. This macroinstruction corresponds to the ALLOCATE RETURN_CONTROL (WHEN_SESSION_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a session to the conversation or when an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

- If a session is free, VTAM assigns it to the conversation.
- If no free sessions exist and session limits allow, VTAM establishes a session and assigns it to the conversation.
- If a new session cannot be established, VTAM queues the request until a session becomes available or until the session limits are changed to allow the establishment of a new session.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on allocating a conversation.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is suspended for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=fmh-5_and_opt._pip_gds_var._add._field
AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 643 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.
**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

 Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to *z/OS Communications Server: SNA Programmer's LU 6.2 Guide* for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to *z/OS Communications Server: SNA Resource Definition Reference.* This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

 Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

 The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

 VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to *z/OS Communications Server: SNA Programmer's LU 6.2 Guide.*

**LUNAME=8-byte_lu_name**

 Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is
located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NAMEUSE
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

NAMEUSE=APNAME
The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**RECLEN=fmh-5_and_opt._pip_gds_var._len.**

**RECLEN=(fmh-5_and_opt._pip_gds_var._len._reg.)**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4-bytes_of_user_data**

**USERFLD=(user_data_register)**

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**

**VTRINA=(vector_address_register)**

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- **VTRINA=0**
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**
VTRINL=(*vector_length_register*)
  Specifies the length of the data area where VTAM places vector list information
  for the application.

  This parameter is ignored if the value for VTRINA is 0 or is not specified. This
  field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**AVFA**
  The field in the RPL extension that indicates whether the partner LU accepts
  the already-verified indicator in place of the password security access subfield
  on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL
  extension.

  **YES (B'1')**
  The partner LU accepts the already-verified indicator.

  **NO (B'0')**
  The partner LU does not accept the already-verified indicator.

**CGID**
  Specifies the 32-bit conversation group identifier.

  It is labeled RPL6CGID in the RPL extension.

**CONSTATE**
  The field in the RPL extension that indicates what state the conversation is in.

  It is labeled RPL6CCST in the RPL extension.

  This field can have the following values for half-duplex conversations:

  **X'00'**  RESET
  **X'01'**  SEND
  **X'08'**  END_CONVERSATION

  This field can have the following values for full-duplex conversations:

  **X'00'**  RESET
  **X'80'**  FDX_RESET
  **X'81'**  SEND/RECEIVE

**CONVID**
  Specifies the resource identifier of the conversation. This field is labeled
  RPL6CNVD in the RPL extension.

  **Note:** The value in this field is returned before this macroinstruction completes
  to allow the application to cancel the conversation allocation process before it
  completes. Refer to **z/OS Communications Server: SNA Programmer's LU 6.2
  Guide** for more information.

**CONVSECP**
  The field in the RPL extension that returns an indication of whether the
  partner LU accepts FMH-5s that include security subfields and indicators. The
  indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This
  field is labeled RPL6CLSA in the RPL extension.
YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.
**PRSISTVP**

Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**

The partner LU accepts persistent-verification indicators.

**NO (B'0')**

The partner LU does not accept persistent-verification indicators.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**

The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**

The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**

The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**

The session was established using session-level LU-LU verification.

**NO (B'0')**

The session was not established using session-level LU-LU verification.
Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversations state is SEND after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

| RCPRI | RCSEC | Meaning                                                                 
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=CONVGRP

Purpose

This macroinstruction allocates resources for a conversation and assigns to the conversation a session with a specified conversation group identifier. If the specific session is not available and session limits allow, VTAM queues the request until the session becomes available. If the specific session does not exist, VTAM fails the allocation request.

Usage

QUALIFY=CONVGRP is used to allocate a conversation over a specific session that already exists. It provides the ability to serially allocate a related group of conversations on a particular session. This macroinstruction corresponds to the ALLOCATE RETURN _ CONTROL (WHEN_CONVERSATION_GROUP_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when:

- VTAM assigns the specified session to the conversation.
- The specific session is deactivated.
- An error occurs that prevents VTAM from assigning the session to the conversation.

To indicate the session to be used, the application program specifies the conversation group identifier for the session on the CGID keyword. This value was returned to the application program by the CGID return field for a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction.

VTAM finds the session for the conversation as follows:
• If the specified session is available, VTAM assigns it to the conversation.
• If the specified session exists but is not available, VTAM queues the request until the session becomes available.
• If the specified session is deactivated while the request is queued, the queued request will be rejected.

As with other ALLOC macroinstructions, when the conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for details on allocating a conversation.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax

```plaintext
APPCCMD — CONTROL — ALLOC — QUALIFY — CONVGRP

(1)

RPL — rpl_address_field
(—rpl_address_register—)

(2)

AAREA — rpl_extension_address_field
(—rpl_extension_address_register—)

(3)

ACB — acb_address_field
(—acb_address_register—)

(3)

AREA — fmh-5_and_opt._pip_gds_var._odd_field
(—fmh-5_and_opt._pip_gds_var._odd_reg.—)

(3)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl\_extension\_address\_field}

\texttt{AAREA=\texttt{(rpl\_extension\_address\_register)}

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}

\texttt{ACB=\texttt{(acb\_address\_register)}

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\texttt{fmh-5\_and\_opt.\_pip\_gds\_var.\_add\_field}
AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)
Specifies the address of the data area containing a formatted FMH-5. A
formatted GDS field can follow the FMH-5 in the data area. See “FMH-5
[ISTFM5]” on page 643 for the VTAM-supplied DSECT that can be used to
create and test values for an FMH-5. Refer to z/OS Communications Server:
SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS
variable. This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application
programs running in supervisor state under a TCB. Application programs
running in TCB-mode supervisor state can use BRANCH=YES to obtain
authorized path services. The indicator resides within the RPLEXTDS field of
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs
running in problem state (non-supervisor state) under a TCB,
BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is
processed in this manner automatically, regardless of the actual setting of
the BRANCH field.

CGID=32-bit_conversation_group_id_field
CGID=(32-bit_conversation_group_id_register)
Specifies the 32-bit conversation group ID.

This value can be obtained from a previous APPCCMD CONTROL=ALLOC,
CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the
CGID operand is not specified, VTAM uses the conversation group ID that is
already in the RPL6CGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific
LUs. It provides a means by which a VTAM LU 6.2 application program and
its partner LU can share serially the same session.

CONMODE
Specifies the mode for receiving normal information upon completion of the
APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode.
It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY
can be used to receive data and that the application program is to receive
data independently of the logical-record format of the data. BUFFCA
corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It
indicates that only APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC can be used to receive data on this conversation.
When the application program issues APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be
received in terms of the logical-record format of the data, or independently
of the logical-record format of the data.
**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.
The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=\texttt{fmh-5\_and\_opt._pip\_gds\_var._len.}
RECLEN=\texttt{(fmh-5\_and\_opt._pip\_gds\_var._len._reg.)}

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLLEN in the RPL.

RPL=rpl\_address\_field
RPL=(rpl\_address\_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN

Specifies the manner in which the Request\_To\_Send\_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH

Specifies that the Request\_To\_Send\_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD

Specifies that the Request\_To\_Send\_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4\_bytes\_of\_user\_data
USERFLD=(user\_data\_register)

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector\_address\_field
VTRINA=(vector\_address\_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector\_length\_field
VTRINL=(vector\_length\_register)

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields.

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B’1’)**

The partner LU accepts the already-verified indicator.

**NO (B’0’)**

The partner LU does not accept the already-verified indicator.

**CONSTATE**

The field in the RPL extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

- X'00'  RESET
- X'01'  SEND
- X'08'  END_CONVERSATION

This field can have the following values for full-duplex conversations:

- X'00'  RESET
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE

**CONVID**

Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to **z/OS Communications Server: SNA Programmer’s LU 6.2 Guide** for more information.

**CONVSECP**

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

**YES (B’1’)**

The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B’0’)**

The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.
NONE (B'00')
   No data is to be encrypted.

SELECTIVE (B'01')
   The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
   All data is to be encrypted.

FDB2
   The field in the RPL in which a global VTAM secondary return code is
   returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
   The field in the RPL extension that returns the length of the FMH-5 waiting
   to be received by the application program. If multiple FMH-5s are waiting to be
   received, FMH5LEN specifies the length of the longest FMH-5 to be received
   by the application program. This field has meaning only when
   FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
   The field in the RPL extension that returns an indication of whether an FMH-5
   has been received. The indication is either YES or NO (RPL6RMH5 set on or
   off). This field is labeled RPL6RMH5 in the RPL extension.

   YES (B'1')
   One or more FMH-5s have been received from partner application
   programs. The FMH5RCV field continues to be set to YES as long as an
   FMH-5 is waiting to be received by the application program. The
   application program must issue APPCCMD CONTROL=RCVFMH5 in
   order to receive an FMH-5.

   NO (B'0')
   No FMH-5s are waiting to be received by the application program.

LOGMODE
   Specifies the logon mode name designating the network properties for the
   session to be allocated for this conversation. The network properties include,
   for example, the class of service to be used.

   The logon mode name cannot be blanks. The logon mode name can be up to 8
   characters in length. If it is fewer than 8 characters in length, VTAM pads it on
   the right with blanks.

   If the logon mode parameter on the APPCCMD macroinstruction specifies a
   logon mode name that does not exist in the logon mode table, VTAM uses the
   mode name of blanks to retrieve the default mode entry when processing
   session activation requests. (Refer to z/OS Communications Server: SNA
   Programmer’s LU 6.2 Guide for more information.) This logon mode name
   corresponds to a logon mode name specified in a MODEENT definition
   statement. (The MODEENT statement is used to build the logon mode table
   named in the MODETAB operand of the APPL definition statement for this
   application program.) For more information on the MODEENT
   macroinstruction, refer to z/OS Communications Server: SNA Resource
   Definition Reference This field is labeled RPL6MODE in the RPL extension.

LUAFFIN
   The field in the RPL extension that indicates the requested (on input) or actual
   (on output) ownership of a Generic Resource affinity with the partner LU, if
   one exists. A result value is returned at completion only if a requested value is
   specified when the macroinstruction is issued.
NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

**LUNAME**
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NETID**
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located.

This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
The partner LU accepts persistent-verification indicators.

**NO (B'0')**
The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.
SESSID
The field in the RPL extension that returns a session instance identifier of the
session over which the FMH-5 flows. The FMH-5 is supplied by the
application program using the AREA field. This field is labeled RPL6SSID in
the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance
identifier, which is itself returned in the SESSID field. Values in the range of
0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was
established using session-level LU-LU verification. This field is labeled
RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA
parameter:
• VTAM-to-APPL required information vector (X'10')
• PCID vector (X'17')
• Name change vector (X'18')
• Session information vector (X'19')
• Partner's application capabilities vector (X'1A')

State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversation, the conversation state is SEND after successful
processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after
successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other
return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application
program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return
codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=CONWIN

Purpose

This macroinstruction allocates resources for a conversation and, if session limits allow, assigns a contention-winner session to the conversation. If a contention-winner session is not available, VTAM queues the request for later completion.

Usage

QUALIFY=CONWIN is used when an application program allocates a conversation and wants VTAM to queue the request if no contention-winner session can be assigned. This macroinstruction corresponds to the ALLOCATE_RETURN_CONTROL (WHEN_CONTENTION_WINNER_ALLOCATED) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a contention-winner session or an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

- If a contention-winner session is currently available, VTAM assigns it to the conversation.
- If no contention-winner session is available and session limits allow, VTAM establishes a new contention-winner session and assigns it to the conversation.
If a new contention-winner session cannot be established, VTAM queues the request until a contention-winner session is available or session limits are changed to allow a new contention-winner session to be activated.

For this macroinstruction to complete successfully, the session limits must define at least one contention-winner session.

If contention-winner sessions are deactivated under normal conditions and an APPCCMD CONTROL=ALLOC, QUALIFY=CONWIN request is queued, VTAM activates another contention-winner session to meet the queued request.

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for details on allocating a conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
APPCCMD  CONTROL=ALLOC, QUALIFY=CONWIN

(1) RPL=rpl_address_field
    (rpl_address_register)

(2) AAREA=rpl_extension_address_field
    (rpl_extension_address_register)

(3) ACB=acb_address_field
    (acb_address_register)

(3) AREA=fmh-5_and_opt._pip_gds_var._add._field
    (fhm-5_and_opt._pip_gds_var._add._reg.)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ \text{AREA} = \text{rpl_extension_address_field} \]
\[ \text{AREA} = \text{(rpl_extension_address_register)} \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=fmh-5_and_opt._pip_gds_var._add._field
AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 643 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA

Specifies that the conversation is to be placed in logical-record-continue-
any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXM in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programming Guide)
This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte.lu_name**

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name
is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

- **OPTCD=SYN**
  
  Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

- **OPTCD=ASY**
  
  Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

- **OPTCD=KEEPSRB**
  
  Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

- **OPTCD=NKEEPSRB**
  
  Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLLEN=fmh-5_and_opt._pip_gds_var._len.**

**RECLLEN=(fhm-5_and_opt._pip_gds_var._len._reg.)**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.
RTSRTRN=BOOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.
This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.

CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

- X'00'  RESET
- X'01'  SEND
- X'08'  END_CONVERSATION

In addition to the half-duplex conversation states, this field can contain the following full-duplex conversation states:

- X'00'  RESET
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE

**CONVID**
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

*Note:* The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/support/docview.wss?rs=257&context=DIS&docref=agg1006) for more information.

**CONVSECP**
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

- **YES (B'1')**
  The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

- **NO (B'0')**
  The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  No data is to be encrypted.

- **SELECTIVE (B'01')**
  The application program specifies the data that is to be encrypted.

- **REQUIRED (B'11')**
  All data is to be encrypted.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B’1’)
The session was established using session-level LU-LU verification.

NO (B’0’)
The session was not established using session-level LU-LU verification.

Vectors returned
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X’10’)
- PCID vector (X’17’)
- Name change vector (X’18’)
- Session information vector (X’19’)
- Partner’s application capabilities vector (X’1A’)

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is SEND state after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See the Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.
## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNLEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNLEVEL_NOT_VALID_FOR_FULL DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL DUPLEX CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID LU_NAME OR NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FM5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE OR RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0078'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION NOT APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR OS_LEVEL DOES NOT SUPPORT REQUESTED FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNASSIGNED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNASSIGNED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=IMMED

Purpose

This macroinstruction allocates resources for a conversation and if session limits allow, assigns an active contention-winner session to it. If no session is available, the allocation request fails.

Usage

QUALIFY=IMMED is used to allocate a conversation when the application program needs an immediate response from VTAM. This macroinstruction completes successfully only when an active contention-winner session is available to be assigned to the conversation. If the request cannot be met immediately, VTAM does not queue it. VTAM neither tries to activate a new session nor bids on a contention-loser session. APPCCMD CONTROL=ALLOC, QUALIFY=IMMED corresponds to the ALLOCATE RETURN_CONTROL(IMMEDIATE) verb in the LU 6.2 architecture.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program must associate a conversation with a particular transaction by using the conversation identifier.

The application program can specify how expedited data is to be received.

For details on allocating a conversation, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax
APPCCMD CONTROL = ALLOC, QUALIFY = IMMED

RPL = rpl_address_field (rpl_address_register)

AAREA = rpl_extension_address_field (rpl_extension_address_register)

ACB = acb_address_field (acb_address_register)

AREA = fmh-5_and_opt._pip_gds_var._add._field (fmh-5_and_opt._pip_gds_var._add._reg.)

BRANCH = NO YES

CONMODE = BUFFCA CS LLCA

CONXMOD = CA CS

ECB = INTERNAL ecb_address_field (ecb_address_register)

EXIT = exit_routine_address_field (exit_routine_address_register)

LOGMODE = 8-byte_logon_mode_name

LUNAME = 8-byte.lu_name
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.
5  EXIT is meaningful only for asynchronous operations.

6  You can code more than one suboperand on OPTCD, but no more than one from each group.

7  KEEPSRB is meaningful only for synchronous operations.

8  NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AREA=**<br>Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**<br>Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=**<br>Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See “FMH-5 (ISTFM5)” on page 643 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to *z/OS Communications Server: SNA Programmer’s LU 6.2 Guide* for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

**BRANCH**<br>Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**<br>Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**<br>Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**<br>Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**<br>Specifies that the conversation is to be placed in buffer-continue-any mode.
It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=fmh-5_and_opt._pip_gds_var._len.
RECLEN=(fmh-5_and_opt._pip_gds_var._len._reg.)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

\[ \text{VTRINL=vector\_length\_field} \]
\[ \text{VTRINL=(vector\_length\_register)} \]

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B’1’)**

The partner LU accepts the already-verified indicator.

**NO (B’0’)**

The partner LU does not accept the already-verified indicator.

**CGID**

Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

\[ X'00' \text{ RESET} \]
\[ X'01' \text{ SEND} \]
\[ X'08' \text{ END\_CONVERSATION} \]

This field can have the following values for full-duplex conversations:

\[ X'00' \text{ RESET} \]
\[ X'80' \text{ FDX\_RESET} \]
\[ X'81' \text{ SEND/RECEIVE} \]

**CONVID**

Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.
CONVSECP

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')

The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')

The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')

No data is to be encrypted.

SELECTIVE (B'01')

The application program specifies the data that is to be encrypted.

REQUIRED (B'11')

All data is to be encrypted.

FDB2

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')

No FMH-5s are waiting to be received by the application program.

PRSISTVP

Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')

The partner LU accepts persistent-verification indicators.
The partner LU does not accept persistent-verification indicators.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SESSID**

The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**

The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**

The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**

The session was established using session-level LU-LU verification.

**NO (B'0')**

The session was not established using session-level LU-LU verification.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is SEND after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See the [Chapter 2, “Return codes,” on page 591](#) for state changes associated with other return codes.

### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCLEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCLEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0058'</td>
<td>X'0000'</td>
<td>UNSUCCESSFUL—SESSION_NOT_AVAILABLE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_INARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=ALLOC, QUALIFY=WHENFREE

Purpose

This macroinstruction allocates resources for a conversation and if session limits allow, assigns a session to the conversation. If a session is not available and one cannot be activated, VTAM returns control to the application program.

Usage

QUALIFY=WHENFREE is used when an application program allocates a conversation and wants VTAM to search for a session that satisfies the ALLOCATE request. This macroinstruction corresponds to the ALLOCATE RETURN_CONTROL (WHEN_SESSION_FREE) verb in the LU 6.2 architecture. This macroinstruction completes when VTAM assigns a session to the conversation or when VTAM cannot assign a session and returns control to the application program with a return code of X'0004', X'0001'.

VTAM finds a session for the conversation as follows:
1. If a session is available, VTAM assigns it to the conversation.
2. If no available sessions exist and session limits allow, VTAM establishes a session and assigns it to the conversation.
3. If a session cannot be established and session activation requests are pending, VTAM queues the ALLOCATE request until the request is satisfied or until all pending session activation requests are used. If the pending session activation requests are used before the ALLOCATE request is satisfied, VTAM fails the ALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001'.
4. If a session cannot be established and no session activation request is pending that might satisfy the ALLOCATE request, VTAM fails the ALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001' and returns control to the application program.

When a conversation is allocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on allocating a conversation.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax
Notes:

1  Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2  See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3  Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4  ECB is meaningful only for asynchronous operations.

5  EXIT is meaningful only for asynchronous operations.

6  You can code more than one suboperand on OPTCD, but no more than one from each group.
7  KEEPSRB is meaningful only for synchronous operations.
8  NKEEP SRB is meaningful only for synchronous operations.

Input parameters

**AAREA**=rpl_extension_address_field

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=fmh-5_and_opt._pip_gds_var._add._field

AREA=(fmh-5_and_opt._pip_gds_var._add._reg.)

Specifies the address of the data area containing a formatted FMH-5. A formatted GDS field can follow the FMH-5 in the data area. See "FMH-5 (ISTFM5)" on page 643 for the VTAM-supplied DSECT that can be used to create and test values for an FMH-5. Refer to  

SNA Programmer’s LU 6.2 Guide for a description of the FMH-5 and GDS variable. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.
CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.
LOGMODE=8-byte_logon_mode_name

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

LUAFFIN=APPL

The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

LUNAME=8-byte_lu_name

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NAMEUSE

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.
NAMEUSE=APNAME
The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=fmh-5_and_opt._pip_gds_var._len.
RECLEN=(fmh-5_and_opt._pip_gds_var._len._reg.)

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLLEN in the RPL.

RPL=rl_address_field
RPL=(rl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts
the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B'1')**

The partner LU accepts the already-verified indicator.

**NO (B'0')**

The partner LU does not accept the already-verified indicator.

**CGID**

Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**

The field in the RPL extension that indicates which state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

- **X'00'**  RESET
- **X'01'**  SEND
- **X'08'**  END_CONVERSATION

This field can have the following values for full-duplex conversations:

- **X'00'**  RESET
- **X'80'**  FDX_RESET
- **X'81'**  SEND/RECEIVE

**CONVID**

Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com) for more information.

**CONVSECP**

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

**YES (B'1')**

The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B'0')**

The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.
NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B’1’)**
The session was established using session-level LU-LU verification.

**NO (B’0’)**
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')
State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is SEND after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See the [Chapter 2, “Return codes,” on page 591](#) for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNCELEVEL_NOT_VALID_FOR_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FULL_DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=CHECK

#### Purpose

This macroinstruction waits for completion of an asynchronous macroinstruction request and marks the RPL and RPL extension used in the request as inactive upon completion.

#### Usage

When asynchronous handling is specified for an RPL-based request, the application program receives control when the request has been accepted by VTAM and the requested operation has been scheduled. An APPCCMD CONTROL=CHECK macroinstruction must be issued for the RPL used for the request to determine when the macroinstruction completes and to get the return code information. APPCCMD CONTROL=CHECK cannot be issued for synchronous requests. In addition, APPCCMD CONTROL=CHECK cannot be issued for an RPL that specifies a non-APPCCMD request. This macroinstruction can be issued in cross-memory mode against an active asynchronous RPL request only when the RPL’s ECB has been posted or the RPL exit has been scheduled.

When APPCCMD CONTROL=CHECK is executed for an RPL that specifies an ECB, the following actions occur:

- If the operation being checked has not been completed, VTAM suspends the execution of the application program task under which the APPCCMD CONTROL=CHECK is issued until the operation is completed.
- If the operation being checked has completed, VTAM returns control to the next sequential instruction after the APPCCMD CONTROL=CHECK macroinstruction.
- The ECB (internal or external) is cleared before VTAM returns control to the application program. (The ECB must be cleared before an RPL-based macroinstruction is issued.)

**Note:** The ECB specified in an asynchronous APPCCMD macroinstruction must not be cleared between the time it is issued and the corresponding APPCCMD CONTROL=CHECK is issued. If the ECB is cleared during this interval, control cannot be returned to the application program after the APPCCMD CONTROL=CHECK is issued.
• The RPL being checked is marked available for reuse by another APPCCMD macroinstruction. (APPCCMD CONTROL=CHECK is the only way this can be done for asynchronous APPCCMD requests.)

When APPCCMD CONTROL=CHECK is executed in an RPL exit routine for the associated RPL, the following actions occur:

• VTAM marks the RPL being checked as available for reuse by another APPCCMD macroinstruction.
• If the operation being checked has completed, VTAM returns control to the next sequential instruction after the APPCCMD CONTROL=CHECK.

**Context**

Input states are not applicable to this macroinstruction.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```plaintext
APPCCMD CONTROL=CHECK, RPL=(rpl_address_field)
```

**Notes:**

1. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

**Input parameters**

- **RPL=rpl_address_field**
- **RPL=(rpl_address_register)**

  Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

After the APPCCMD CONTROL=CHECK macroinstruction has completed, the completion information returned is for the macroinstruction being checked. Refer to the description of the particular APPCCMD being checked for a list of the parameters that are returned to the application program.

---

**APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG**

**Purpose**

This macroinstruction deallocates a conversation when an application program has detected a transaction program error.

**Usage**

QUALIFY=ABNDPROG is used when the application program detects an error in a transaction program and that error prevents further useful communication on the conversation. It corresponds to DEALLOCATE TYPE=ABEND_PROG in the LU 6.2 architecture. If the conversation is in a sending state, the SEND buffer is flushed before the conversation is deallocated.
This macroinstruction, along with the other QUALIFY=ABND* forms, can be used to cancel an outstanding APPCCMD macroinstruction, which allows the application program to recover from hung transactions. The following macroinstructions cannot be canceled:

- APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC that has not received any data from the partner LU
- APPCCMD CONTROL=RECEIVE, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVF MH5, QUALIFY=NULL|QUEUE
- APPCCMD CONTROL=RESETRCV
- APPCCMD CONTROL=OPRCNTL
- APPCCMD CONTROL=REJECT, QUALIFY=CONV
- APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL|IALL
- One of the abnormal deallocation macroinstructions
- A macroinstruction that is waiting for a response to a confirmation request
- A macroinstruction that is waiting for the arrival of an FMH-7

If any one of these macroinstructions is outstanding, the application program must wait for it to complete before issuing this macroinstruction. Alternatively, the application program can issue an APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on deallocating a conversation when an error is detected or for early deallocation of a pending APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVF MH5, QUALIFY=DATAQUE.

Context

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_SEND
- PEND_END_CONV_LOG
- PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG
This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
{name}  APPCCMD=DEALLOC, QUALIFY=ABNDPROG

(1)

RPL=rpl_address_field
    (rpl_address_register)

(2)

AAREA=rpl_extension_address_field
    (rpl_extension_address_register)

(3)

ACB=acb_address_field
    (acb_address_register)

(3)

AREA=optional_log_data_area_address_field
    (optional_log_data_area_address_register)

(3)

BRANCH=NO YES

(3)

CONVID=32-bit_resource_id_field
    (32-bit_resource_id_register)

(1)

ECB=INTERNAL
    ecb_address_field
    (ecb_address_register)

(4) (3)

EXIT=exit_routine_address_field
    (exit_routine_address_register)

(5) (3)

OPTCD=(ASY SYN KEEPSRB NKEEPSRB)

(3) (6)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA** = rpl_extension_address_field

AAREA = (rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = acb_address_field

ACB = (acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA** = optional_log_data_area_address_field

AREA = (optional_log_data_area_address_register)

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](http://www.ibm.com) for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.
BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLEN=**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

**RPL=**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields.

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
X'84' PENDING_SEND/RECEIVE_LOG
X'85' PENDING_RECEIVE-ONLY_LOG
X'86' PENDING_RESET_LOG

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the
AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](http://www.ibm.com) for more information.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](http://www.ibm.com) for more information.

**State changes**

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful completion of the macroinstruction. For full-duplex conversations, the conversation state is FDX_RESET after successful completion of the macroinstruction.

See [Chapter 2, “Return codes,” on page 591](http://www.ibm.com) for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](http://www.ibm.com) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV**

**Purpose**

This macroinstruction is used when the application program detects an error in its implementation of LU 6.2 services.

**Usage**

QUALIFY=ABNDSERV is used when the application program encounters errors related to LU 6.2 services. For example, the application program might detect an error in its support of mapped conversations or in conversation-level security that would require it to deallocate the conversation. QUALIFY=ABNDSERV corresponds to the DEALLOCATE TYPE=ABEND_SVC verb in the LU 6.2 architecture.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV can be issued against a conversation for which there is already an APPCCMD outstanding. These commands cancel the previous macroinstruction, allowing the application program to recover from a “hung” transaction. However, there are cases where it is not allowed when a prior macroinstruction is outstanding. See **Usage** on page 70 for a list of macroinstructions that cannot be canceled.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on abnormally deallocating a conversation.

**Context**

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- **PENDING_ALLOCATE**
- **SEND**
- **RECEIVE**
- **RECEIVE_CONFIRM**
- **RECEIVE_CONFIRM_SEND**
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_SEND
- PEND_END_CONV_LOG
- PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD=DEALLOC, QUALIFY=ABND
RPL=rpl_address_field
AAREA=rpl_extension_address_field
ACB=acb_address_field
AREA=optional_log_data_area_address_field
BRANCH=YES
CONVID=32-bit_resource_id_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID.
(CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=optional_log_data_area_address_field**

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH= YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. **Event_control_block_address** is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPERSB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPERSB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RECLLEN=optional_log_data_length**

Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the PRL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X’01’ SEND
- X’02’ RECEIVE
- X’03’ RECEIVE_CONFIRM
For full-duplex conversations, this field can have the following values:

- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
- X'84'  PENDING_SEND/RECEIVE_LOG
- X'85'  PENDING_RECEIVE-ONLY_LOG
- X'86'  PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**

The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**

The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://pubs.opengroup.org/onlinepubs/009695399/) for more information.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://pubs.opengroup.org/onlinepubs/009695399/) for more information.

**State changes**

The changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversation state is FDX_RESET after successful processing.

See [Chapter 2, “Return codes,” on page 591](https://pubs.opengroup.org/onlinepubs/009695399/) for state changes associated with other return codes.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, "Return codes," on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLYED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNOMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_COMPATIBLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME

Purpose

This macroinstruction is used to deallocate a conversation that has had no activity for a specified amount of time.

Usage

QUALIFY=ABNDTIME is used when the LU detects that it has not received information from one of its transaction programs within a specific amount of time. For example, an application program would use this macroinstruction if one of the conversations is in a state that allows receiving and has not received any data in an excessive amount of time. The application program must determine how long to wait before issuing this macroinstruction.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to abnormally deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME can be issued against a conversation for which there is already an APPCCMD outstanding. These commands cancel the previous macroinstruction, allowing the application program to recover from a hung transaction. However, there are cases where it is not
allowed when a prior macroinstruction is outstanding. See “Usage” on page 70 for a list of macroinstructions that cannot be canceled.

QUALIFY=ABNDTIME corresponds to the DEALLOCATE TYPE=ABEND_TIMER verb in the LU 6.2 architecture.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of abnormally terminating a conversation.

Context

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_SEND
- PEND_END_CONV_LOG
- PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
APPCCMD CONTROL = DEALLOC , QUALIFY = ABNDTIME

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

AREA = optional_log_data_area_address_field

BRANCH = NO

CONVID = 32-bit_resource_id_field

ECB = INTERNAL

EXIT = exit_routine_address_field

OPTCD = ( ASY ) SYN KEEPSRB NKEEPSRB

RECLEN = optional_log_data_length
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=(optional_log_data_area_address_register)

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=KEEP SRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.
OPTCD=NKEEPSRB
  Specifies that for a synchronous request issued in SRB mode, VTAM does
  not return to the application under the same SRB in which VTAM was
  invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
  Specifies the length of the data area indicated by the AREA field. This field is
  labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that
  the application program has chosen not to provide optional error log data to
  VTAM. If the application program specifies RECLEN=0, VTAM indicates in the
  FMH-7 it creates as a result of this APPCCMD that no error log data follows
  the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
  Specifies the address of the request parameter list that contains information to
  be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
  The field in the RPL6 extension that indicates the state of the conversation.
  This field is labeled RPL6AVFA in the RPL extension.
  For half-duplex conversations, this field can have the following values:
    X'01'   SEND
    X'02'   RECEIVE
    X'03'   RECEIVE_CONFIRM
    X'04'   RECEIVE_CONFIRM_SEND
    X'05'   RECEIVE_CONFIRM_DEALLOCATE
    X'07'   PENDING_END_CONVERSATION_LOG
    X'08'   END_CONVERSATION
    X'09'   PENDING_SEND
    X'0A'   PENDING_RECEIVE_LOG
  For full-duplex conversations, this field can have the following values:
    X'80'   FDX_RESET
    X'81'   SEND/RECEIVE
    X'82'   SEND_ONLY
    X'83'   RECEIVE_ONLY
    X'84'   PENDING_SEND/RECEIVE_LOG
    X'85'   PENDING_RECEIVE-ONLY_LOG
    X'86'   PENDING_RESET_LOG

EXPDLLEN
  The field in the RPL6 that shows the length of the expedited data waiting to be
  received. This field has meaning only when EXPDRCV=YES. This field is
  labeled RPL6EXDL in the RPL extension.
EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.
occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STD5 in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.ws/docid/57104) for more information.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.ws/docid/57104) for more information.

**State changes**

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversation state is FDX_RESET after successful processing.

See **Chapter 2, “Return codes,” on page 591** for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See **Chapter 2, “Return codes,” on page 591** for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER

**Purpose**

This macroinstruction deallocates a conversation when the application program detects an error.

**Usage**

This macroinstruction is used by an application program to deallocate a conversation and to inform the partner LU of the reason for the deallocation. To indicate the reason for the deallocation, the application program specifies a sense code on the macroinstruction. This sense code is sent to the partner LU in an FMH-7 and must be appropriate to the error. Otherwise improper processing of the macroinstruction might occur. For a list of valid sense codes, refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/docs/en/zoscommunications-server).

This macroinstruction does not correspond to any of the verbs in the LU 6.2 architecture.

An example of the use of this macroinstruction would be to report errors that the application program detects on a received FMH-5. Although VTAM performs preliminary format checks on the FMH-5 before passing it to the application program, the application program validates the FMH-5. If the application program detects an error in the FMH-5, it issues APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER and specifies the appropriate sense code. VTAM sends the conversation deallocation notification and the FMH-7 to the partner LU.

If the conversation is in a state that allows sending, the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction is executed prior to abnormally deallocating the conversation.

APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER can be issued against a conversation for which there is already an APPCCMD outstanding. It cancels the previous macroinstruction, allowing the application program to recover from a “hung” transaction. However, in some cases, it is not allowed when a prior macroinstruction is outstanding. See “Usage” on page 70 for a list of macroinstructions that cannot be canceled.

Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/docs/en/zoscommunications-server) for more information on abnormally deallocating a conversation.

**Context**

On half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
• SEND
• RECEIVE
• RECEIVE_CONFIRM
• RECEIVE_CONFIRM_SEND
• RECEIVE_CONFIRM_DEALLOCATE
• PEND_SEND
• PEND_END_CONV_LOG
• PENDING_RECEIVE_LOG

On full-duplex conversations, this macroinstruction can be issued from the following conversation states:
• PENDING_ALLOCATE
• SEND/RECEIVE
• SEND_ONLY
• RECEIVE_ONLY
• PENDING_SEND/RECEIVE_LOG
• PENDING_RECEIVE-ONLY_LOG
• PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCMD CONTROL = DEALLOC, QUALIFY = ABNDUSER

(1)

RPL = rpl_address_field (rpl_address_register)

(2)

AAREA = rpl_extension_address_field (rpl_extension_address_register)

(3)

ACB = acb_address_field (acb_address_register)

(3)

AREA = optional_log_data_area_address_field (optional_log_data_area_address_register)

(3)

BRANCH = NO

YES
```
Notes:

1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
**AAREA=**(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=**(acb_address_field)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=**(optional_log_data_area_address_field)

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/SSLTBK_380188_6.2.0/com.ibm.zos.v2r1.wixy猗 регаguide.doc) for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=**(32-bit_resource_id_field)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.
ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event control block address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=optional_log_data_length
RECLEN=(optional_log_data_length_register)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE=user-supplied_32-bit_fmh-7_sense_code
SENSE=(user-supplied_32-bit_fmh-7_sense_code_register)
Specifies the user-specified sense code that the application program requests to be placed in the FMH-7 that VTAM creates as a result of this APPCCMD macroinstruction. This sense code must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. This is the only one
of the abnormal DEALLOC macroinstructions for which this field is applicable. This field is labeled RPL6SNSO in the RPL extension. For a list of valid sense codes, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X’01’  SEND
- X’02’  RECEIVE
- X’03’  RECEIVE_CONFIRM
- X’04’  RECEIVE_CONFIRM_SEND
- X’05’  RECEIVE_CONFIRM_DEALLOCATE
- X’07’  PENDING_END_CONVERSATION_LOG
- X’08’  END_CONVERSATION
- X’09’  PENDING_SEND
- X’0A’  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X’80’  FDX_RESET
- X’81’  SEND/RECEIVE
- X’82’  SEND_ONLY
- X’83’  RECEIVE_ONLY
- X’84’  PENDING_SEND/RECEIVE_LOG
- X’85’  PENDING_RECEIVE-ONLY_LOG
- X’86’  PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received.
by the application program. This field has meaning only when
FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STD$ in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversation state is FDX_RESET after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>PARAMETER_ERROR—ABNORMAL_DEALLOCATE_REJECTED_RETRY</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002D'</td>
<td>PARAMETER_ERROR—INVALID_SENSE_CODE_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=DEALLOC, QUALIFY=CONFIRM

Purpose

This macroinstruction sends a confirmation request to a partner application program and, if the partner sends a positive confirmation response, VTAM deallocates the conversation.

Usage

QUALIFY=CONFIRM is used to ensure that the partner receives all data on a conversation before that conversation is deallocated.

VTAM sends the partner LU any remaining data in the SEND buffer, which is followed by a confirmation request. If the partner LU sends a positive response to the confirmation request, VTAM deallocates the conversation. If the partner LU sends a negative response to the confirmation request, VTAM does not deallocate the conversation. This macroinstruction completes only after a response is received from the partner LU. It corresponds to the DEALLOCATE (TYPE=CONFIRM) verb in the LU 6.2 architecture.

When this macroinstruction completes, the current conversation state is in the CONSTATE field.

Because this macroinstruction requests deallocation of the conversation, the data in the SEND buffer must complete a logical record.

For more information on sending and responding to confirmation requests, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

This macroinstruction can only be used on half-duplex conversations from the SEND conversation state.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
APPCCMD CONTROL = DEALLOC, QUALIFY = CONFIRM

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

BRANCH = NO, YES

CONMODE = BUFFCA, CS, LLCA, SAME

CONVID = 32-bit_resource_id_field

CONXMOD = CA, CS, SAME

ECB = INTERNAL

EXIT = exit_routine_address_field

OPTCD = (ASY), SYN, KEEPSRB, NKEEP SRB

Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

Operator value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPESRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=rpl_extension_address_field
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY I IANY...
can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.
ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05' RECEIVE_CONFIRM DEALLOCATE
X'07' PENDING END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
Indicates that an FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.
LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004' ALLOCATION_ERROR
- X'0014' DEALLOCATE_ABEND_PROGRAM
- X'0018' DEALLOCATE_ABEND_SERVICE
- X'001C' DEALLOCATE_ABEND_TIMER
- X'0030' PROGRAM_ERROR_NO_TRUNC
- X'0034' PROGRAM_ERROR_PURGING
- X'0038' PROGRAM_ERROR_TRUNC
- X'003C' SERVICE_ERROR_NO_TRUNC
- X'0040' SERVICE_ERROR_PURGING
- X'0044' SERVICE_ERROR_TRUNC
- X'005C' USER_ERROR_CODE_RECEIVED

NO (B'0')

Indicates either that no error indicator was received or that an error indicator was received but indicated that no log data follows.

RCPRI

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE

The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field
indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an
FMH-7 sense code that VTAM did not recognize. This field is labeled
RPL6SNSI in the RPL extension.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be
associated with a conversation. Whenever an APPCCMD completes, VTAM
places in the USERFLD field of the RPL extension the 4 bytes that were
supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
classification was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a
remote application program). This field is labeled RPL6USR in the RPL
extension.

Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](#) for
more information.

**State changes**

These changes are applicable when RCPRI indicates OK.

The conversation state is END_CONV after successful processing.

See [Chapter 2, “Return codes,” on page 591](#) for state changes associated with other
return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application
program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return
codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0002’</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0004’</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0005’</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0007’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0008’</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0009’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000A’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000B’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000D’</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X’0014’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X’0018’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X’001C’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X’0024’</td>
<td>X’0000’</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0011’</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’001F’</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0032’</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X’0030’</td>
<td>X’0000’</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=DEALLOC, QUALIFY=DATACON

#### Purpose

This macroinstruction sends data, which is supplied by the application program, and any information in the SEND buffer to a partner application program, followed by a confirmation request. If the partner LU sends a positive response to the confirmation request, VTAM deallocates the conversation normally.

#### Usage

This macroinstruction is used to send data to the partner LU and to ensure that the partner receives all the data before the conversation is deallocated.

VTAM sends any data remaining in the buffer followed by the data specified on the macroinstruction to the partner LU. This data is followed by a confirmation request. The macroinstruction completes only after the partner LU responds to the confirmation request. If the partner sends a positive confirmation response, the conversation is deallocated. If the partner LU sends a negative confirmation response, the conversation is not deallocated. This macroinstruction corresponds to the SEND_DATA and DEALLOCATE (TYPE=CONFIRM) verbs in the LU 6.2 architecture.

When this macroinstruction completes, the current conversation state is found in the CONSTATE field.

Because this macroinstruction requests deallocation of the conversation, the data sent must complete a logical record.

For more information on sending and responding to confirmation requests, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com).
Context

This macroinstruction can be used only on half-duplex conversations from the SEND conversation state.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD CONTROL=DEALLOC, QUALIFY=DATACON (1)
RPL=rpl_address_field (2)
AAREA=rpl_extension_address_field (3)
ACB=acb_address_field (3)
AREA=data_area_or_buffer_list_address_field (3)
BRANCH=NO YES (1)
CONMODE=BUFFCA CS LLCA SAME (1)
CONVID=32-bit_resource_id_field (1)
CONXMOD=CA CS SAME (1)
CRYPT=NO YES (3)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)
Specifies the address of a data buffer or buffer list.

• If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

• If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

• If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC or ISPEC.
CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

**OPTCD=NBUFFLST**

Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

**OPTCD=XBUFLST**

Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X’002C’, X’0010’ (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLEN=**

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
• If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)

• If OPTCD=XBUFFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=rpl_address_field
RPL=rpl_address_register

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values:

X'01'  SEND
X'02'  RECEIVE
X'03'  RECEIVE_CONFIRM
X'04'  RECEIVE_CONFIRM_SEND
X'05'  RECEIVE_CONFIRM_DEALLOCATE
X'07'  PENDING_END_CONVERSATION_LOG
X'08'  END_CONVERSATION
X'09'  PENDING_SEND
X'0A'  PENDING_RECEIVE_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.
YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
Indicates that an FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:
X'0004'
  ALLOCATION_ERROR
X'0014'
  DEALLOCATE_ABEND_PROGRAM
X'0018'
  DEALLOCATE_ABEND_SERVICE
X'001C'
  DEALLOCATE_ABEND_TIMER
X'0030'
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERROR_PURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
Indicates either that no error indicator was received or that an error indicator was received but indicated that no log data follows.
RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6SNSI in the RPL extension.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether the application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.
Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the RPL extension.

YES (B'1')
Indicates that a SIGNAL RU has been received from the partner application program. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
Indicates that no SIGNAL RU has been received from the partner application program. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

State changes

These changes are applicable when RCPRI indicates OK.

The conversation state is END_CONV after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.
## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'00D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'00E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'00F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'013'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_NOT_VALID_</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FOR FULL-DUPLEX_CONVERSATIONS</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=DEALLOC, QUALIFY=DATAFLU

Purpose

This macroinstruction unconditionally deallocates a conversation after sending data to a partner application program. The send function of the macroinstruction includes flushing the SEND buffer.

Usage

This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=DEALLOC, QUALIFY=FLUSH. As with all macroinstructions that both send data and deallocate a conversation, the data sent by the application program must complete a logical record.

The deallocation request on this macroinstruction is unconditional. After VTAM successfully sends the data, it deallocates the conversation. Any incoming error information received for the application program is discarded.

This macroinstruction corresponds to the SEND_DATA verb followed by the DEALLOCATE (TYPE=FLUSH) verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the SEND or PENDING_SEND conversation states.

For full-duplex conversations, this macroinstruction can be issued from the following states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** acb_address_field

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
**AREA=** `data_area_or_buffer_list_address_field`

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)
- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY
can be used to receive data and that the application program is to receive
data independently of the logical-record format of the data. BUFFCA
corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It
indicates that only APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC can be used to receive data on this conversation.
When the application program issues APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be
received in terms of the logical-record format of the data or independently
of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-
any mode. It indicates that APPCCMD CONTROL=RECEIVE,
QUALIFY=ANY | IANY can be used to receive data on this conversation
and that the application program is to receive data in terms of the
logical-record format of the data. LLCA corresponds to FILL=LL on the
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC
macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain
unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD
in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the
APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received by either a specific-type
macroinstruction or an any-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received only by a specific-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD,
QUALIFY=SPEC | ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain
unchanged at the completion of this macroinstruction.

**CRYPT**
Specifies whether data at the location indicated by the AREA is to be
encrypted before it is sent on the conversation. This field is labeled RPLTCRYP
in the RPL.

**CRYPT=NO**
Do not encrypt data before it is sent.
CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ECB_address_field
ECB=(ECB_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=EXIT_address_field
EXIT=(EXIT_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
**OPTCD=BUFLST**

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFLST is chosen, the AREA field of the RPL points to a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/support/docdisplay?rs=2956&context=TP52222&locale=en&docdisplay=true&category=advanced&suid=3625001917) for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

**OPTCD=NBUFFLST**

Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

**OPTCD=XBUFLST**

Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/support/docdisplay?rs=2956&context=TP52222&locale=en&docdisplay=true&category=advanced&suid=3625001917) for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of ‘X’002C’, X’0010’ (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLEN=data_length**

**RECLEN=(data_length_register)**

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

**RPL=rpl_address_field**
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to
   be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
   The field in the RPL6 extension that indicates the state of the conversation. It is
   labeled RPL6CCST in the RPL extension. For half-duplex conversations, this
   field can have the following values:

   X'01' SEND
   X'02' RECEIVE
   X'03' RECEIVE_CONFIRM
   X'04' RECEIVE_CONFIRM_SEND
   X'05' RECEIVE_CONFIRM_DEALLOCATE
   X'07' PENDING_END_CONVERSATION_LOG
   X'08' END_CONVERSATION
   X'09' PENDING_SEND
   X'0A' PENDING_RECEIVE_LOG
   For full-duplex conversations, this field can have the following values:

   X'80' FDX_RESET
   X'81' SEND/RECEIVE
   X'82' SEND_ONLY
   X'83' RECEIVE_ONLY
   X'84' PENDING_SEND/RECEIVE_LOG
   X'85' PENDING_RECEIVE-ONLY_LOG
   X'86' PENDING_RESET_LOG

EXPDLEN
   The field in the RPL6 that shows the length of the expedited data waiting to be
   received. This field has meaning only when EXPDRCV=YES. This field is
   labeled RPL6EXDL in the RPL extension.

EXPDRCV
   The field in the RPL6 that indicates whether expedited data is waiting to be
   received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
   The field in the RPL in which a global VTAM secondary return code is
   returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
   The field in the RPL extension that returns the length of the FMH-5 waiting to
   be received by the application program. If multiple FMH-5s are waiting to be
   received, FMH5LEN specifies the length of the longest FMH-5 to be received
   by the application program. This field has meaning only when
   FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.
**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RPLXSRV**

A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

Returns the sense code carried in the FMH-7 used in deallocating the conversation. This field is labeled RPL6SNSI in the RPL extension.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.
Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the RPL extension.

YES (B'1')
Indicates that a SIGNAL RU has been received from the partner application program. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
Indicates that no SIGNAL RU has been received from the partner application program. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.
State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation enters END_CONV after successful completion of the macroinstruction.

For full-duplex conversations, the conversation enters one of the following states after successful completion of the macroinstruction.

- RECEIVE_ONLY
- PENDING_RECEIVE-ONLY_LOG
- FDX_RESET

See [Chapter 2, “Return codes,” on page 591](#) for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>SYNC_LEVEL_NOT_SUPPORTED_BY_LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>TRANSACTION_PROGRAM_NOT_AVAILABLE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>TRANSACTION_PROGRAM_NOT_AVAILABLE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>CANNOT_RECONNECT_TRANSACTION_PROGRAM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>CANNOT_RECONNECT_TRANSACTION_PROGRAM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>RECONNECT_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR=INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR=ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR=ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR=REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR=CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0100'</td>
<td>PARAMETER_ERROR=INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0111'</td>
<td>PARAMETER_ERROR=PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'011F'</td>
<td>PARAMETER_ERROR=APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0244'</td>
<td>PARAMETER_ERROR=PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0255'</td>
<td>PARAMETER_ERROR=PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0288'</td>
<td>PARAMETER_ERROR=CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0332'</td>
<td>PARAMETER_ERROR=UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNCATION</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNCATING</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=DEALLOC, QUALIFY=FLUSH

**Purpose**

This macroinstruction flushes the SEND buffer and unconditionally deallocates a conversation.

**Usage**

For half-duplex conversations, this macroinstruction executes the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction prior to the deallocation. Any error information coming from the partner application program that is received by VTAM after the macroinstruction is issued is not reported to the application program.

This macroinstruction, when issued on a full-duplex conversation, either initiates the conversation deallocation or completes the conversation deallocation if a deallocation request has been received from the conversation partner.

This macroinstruction corresponds to the DEALLOCATE (TYPE=FLUSH) verb described in the LU 6.2 architecture.
**Context**

For half-duplex conversations, this macroinstruction can be issued from a SEND or PENDING SEND conversation state.

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction...
programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

 Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**

Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB
   Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
   Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
   Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
   The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.
   For half-duplex conversations, this field can have the following values:
   X'01' SEND
   X'02' RECEIVE
   X'03' RECEIVE_CONFIRM
   X'04' RECEIVE_CONFIRM_SEND
   X'05' RECEIVE_CONFIRM_DEALLOCATE
   X'07' PENDING_END_CONVERSATION_LOG
   X'08' END_CONVERSATION
   X'09' PENDING_SEND
   X'0A' PENDING_RECEIVE_LOG
   For full-duplex conversations, this field can have the following values:
   X'80' FDX_RESET
   X'81' SEND/RECEIVE
   X'82' SEND_ONLY
   X'83' RECEIVE_ONLY
   X'84' PENDING_SEND/RECEIVE_LOG
   X'85' PENDING_RECEIVE-ONLY_LOG
   X'86' PENDING_RESET_LOG

EXPDLEN
   The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
   The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.
**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B’1’)**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B’0’)**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.


**State changes**

These changes are applicable when RCPRI indicates OK.
For half-duplex conversations, the conversation enters END_CONV after successful completion of the macroinstruction.

For full-duplex conversations, the conversation can enter the following conversation states after successful processing:

- RECEIVE_ONLY
- PENDING_RECEIVE-ONLY_LOG
- FDX_RESET

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_ REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION RECEIVED DEALLOCATE_ ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION RECEIVED DEALLOCATE_ ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION RECEIVED DEALLOCATE_ ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION RECEIVED_ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION RECEIVED_UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION RECEIVEDRESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION RECEIVEDRESOURCE_FAILURE_NO_RETRY</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=DEALLOCQ**

**Purpose**

This macroinstruction deallocates a conversation when an application program has detected an error. This macroinstruction is queued if the conversation is in the
RECEIVE state and has not yet received data. When data is received, VTAM continues deallocation of the conversation.

**Usage**

QUALIFY=ABNDPROG is used to abnormally terminate a conversation when the application program detects an error that will prevent further useful conversation.

QUALIFY=ABNDSERV is used to abnormally terminate a conversation and alert VTAM that an LU service component has encountered an error.

QUALIFY=ABNDTIME is used to abnormally terminate a conversation when the application program detects that it has not received information from its partner for a specified amount of time.

QUALIFY=ABNDUSER is used to abnormally terminate a conversation. The command also alerts VTAM that the application program will provide a user-specified sense code to place in the FMH-7 that VTAM creates as a result of this command. The application program is responsible for the validity of the sense code.

This macroinstruction abnormally deallocates a conversation. If the conversation is in a sending state, the function is identical to the abnormal termination APPCCMD CONTROL=DEALLOC. The SEND buffer is flushed before the conversation is deallocated.

If the conversation is in a receiving state and is waiting for a first, or only element in the chain, this macroinstruction is queued until data is received from the partner LU.

To contrast this macroinstruction with DEALLOC, the DEALALLOCQ macroinstruction will never receive an RCPRI, RCSEC of X’002C’, X’0021’.

The following macroinstructions cannot be canceled by APPCCMD CONTROL=DEALLOCQ:

- APPCCMD CONTROL=RECEIVE, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY that has not been matched to a conversation
- APPCCMD CONTROL=RCVFMH5, QUALIFY=NULL|QUEUE
- APPCCMD CONTROL=RESETRCV
- APPCCMD CONTROL=OPRCNTL
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG|ABNDSERV|ABNDTIME|ABNDUSER
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDPROG|ABNDSERV|ABNDTIME|ABNDUSER
- APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL|IALL
- A macroinstruction that is waiting for a response to a confirmation request
- A macroinstruction that is waiting for the arrival of an FMH-7

If any one of these macroinstructions is outstanding, the application program can either wait for the outstanding APPCCMD to complete and then issue APPCCMD CONTROL=DEALLOCQ or issue APPCCMD CONTROL=REJECT.
Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on deallocating a conversation when an error is detected.

For early deallocation of a pending APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_SEND
- PENDING_END_CONV_LOG
- PENDING_RECEIVE_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- PENDING_ALLOCATE
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl_extension_address_field
**AAREA**=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field
**ACB**=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=optional_log_data_area_address_field
**AREA**=(optional_log_data_area_address_register)

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner application program. The application program is responsible for placing the error log data into the local system log. If the application program chooses to supply an error log GDS variable, it has to supply the entire GDS variable on the APPCCMD macroinstruction. VTAM inspects the 2-byte logical-record length (LL) field of the GDS variable to determine if the amount of data supplied is equal to the length specified in the LL field. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.
BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPRS RB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPRS RB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECL EN=optional_log_data_length
RECL EN=(optional_log_data_length_register)
Specifies the length of the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL. A value of 0 in the RECLEN field indicates that the application program has chosen not to provide optional error log data to VTAM. If the application program specifies RECLEN=0, VTAM indicates in the FMH-7 it creates as a result of this APPCCMD that no error log data follows the FMH-7, and the AREA field in the RPL is ignored.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE
Specifies the user-specified sense code that the application program requests to be placed in the FMH-7 that VTAM creates as a result of this APPCCMD macroinstruction. This sense code must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. This field is examined only if QUALIFY=ABNDUSER is issued. This field is labeled RPL6SNSI in the RPL extension. For a list of valid sense codes,

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05' RECEIVE_CONFIRM_DEALLOCATE
X'07' PENDING_END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:

- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
- X'84'  PENDING_SEND/RECEIVE_LOG
- X'85'  PENDING_RECEIVE-ONLY_LOG
- X'86'  PENDING_RESET_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, END_CONV state is entered.

For full-duplex conversations, FDX_RESET state is entered.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (DEALLOCATION IS COMPLETE)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>PARAMETER_ERROR—INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS

**Purpose**

This macroinstruction responds positively to a session establishment request.

**Usage**

This macroinstruction is issued after the application program is notified through its LOGON or SCIP exit routine that a CINIT or BIND request has been received. (For a description of when the LOGON and SCIP exits are scheduled and for the information provided in each exit, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/S55URU_2.1.0/com.ibm.zos.v2r1.o59/zos_v2r1_0.Payas.html).) The function of this command is similar to the VTAM API commands OPNDST OPTCD=ACCEPT and OPNSEC for non-LU 6.2 sessions.

When this macroinstruction is used in a LOGON exit, the RPLAREA field of the read-only RPL passed to the exit routine contains the address of a read-only copy of the CINIT. The application program can examine the parameters of the BIND in the CINIT. If the application program needs to override any of the BIND parameters, it can specify session parameters for a BIND on this macroinstruction (mapped by ISTDBIND).

**Attention:** If both the local and the partner LU are the same LU, then this macroinstruction must not be issued from the LOGON exit routine. Otherwise, the session will hang.

The partner LU can negotiate the BIND. If this occurs, VTAM verifies and accepts the negotiated BIND parameters. (For information on BIND fields and their settings, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/S55URU_2.1.0/com.ibm.zos.v2r1.o59/zos_v2r1_0.Payas.html).) However, VTAM does not return the negotiated BIND response to the application program when this macroinstruction completes.

The application program can use this macroinstruction in a SCIP exit to override some of the values received in the BIND by providing a BIND image (in

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002D'</td>
<td>PARAMETER_ERROR—INVALID_SENSE_CODE_VALUE_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
ISTDBIND format) to be used in building a response. When this macroinstruction is used in a SCIP exit, word 4 of the parameter list points to session parameters mapped by ISTDBIND. If the application program needs to override any of the BIND parameters, it can specify session parameters for a BIND response on this macroinstruction (mapped by ISTDBIND). Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for details on the values that can be overridden.

**Note:** APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS does not correspond to the ACTIVATE_SESSION verb described in the LU 6.2 architecture.

**Context**

Input states are not applicable to this macroinstruction.

**Syntax**

```
APPCCMD
  name  
  CONTROL=OPRCNTL, QUALIFY=ACTSESS  
  RPL=rpl_address_field  
  (rpl_address_register)  
  AAREA=rpl_extension_address_field  
  (rpl_extension_address_register)  
  ACB=acb_address_field  
  (acb_address_register)  
  AREA=session_parameter_address_field  
  (session_parameter_address_register)  
  ARG=4-byte_session_identifier_(cid)_field  
  (4-byte_session_identifier_(cid)_register)  
  BRANCH=NO  
  CONFTXT=NO  
  ECB=INTERNAL  
  (ecb_address_field  
  (ecb_address_register))  
  EXIT=exit_routine_address_field  
  (exit_routine_address_register)  
```
Notes:
1 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4 ECB is meaningful only for asynchronous operations.

5 EXIT is meaningful only for asynchronous operations.

6 You can code more than one suboperand on OPTCD, but no more than one from each group.

7 KEEPSRB is meaningful only for synchronous operations.

8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ AAREA= (rpl_extension_address_field) \]
\[ AAREA= (rpl_extension_address_register) \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ ACB= (acb_address_field) \]
\[ ACB= (acb_address_register) \]

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=session_parameter_address_field
AREA=(session_parameter_address_register)
Specifies the address of an area that contains a set of session parameters that VTAM uses when constructing the BIND or BIND response, which is sent to establish a session. If an address is indicated, the set of parameters specified by the application program will override the session parameters given in the CINIT or BIND (refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for information on building the session parameters). This field is labeled RPLAREA in the RPL. If you specify AREA=0, VTAM uses the set of session parameters contained in the CINIT or BIND to construct the BIND or BIND response.

Note: You should use the ISTDBIND DSECT if you include user data fields on the BIND.

ARG=4-byte_session_identifier_(cid)_field
ARG=(4-byte_session_identifier_(cid)_register)
Specifies the CID of the session that was returned to the application program in the parameter list of the LOGON or SCIP exit routine. The specified CID must identify a CINIT or BIND that is queued for this application program.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONFTXT
Indicates whether or not data sent or received on this session is to be considered “confidential” within this host. This field is labeled RPL6CFTX in the RPL extension.

CONFTXT=YES
The VTAM buffers used to hold the data are cleared before they are returned to their buffer pools.

CONFTXT=NO
No clearing is performed.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.
ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program
immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the
scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM
returns to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=session_parameter_information_length
RECLEN=(session_parameter_information_length_register)
Specifies the length of the session parameter information. This field is labeled
RPLRLEN in the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with
this operator control request. The information is returned unchanged when the
macroinstruction completes. This data cannot be used by any conversations. It
can be used for correlation purposes. This field is labeled RPL6USR in the RPL
extension.
RPL and RPL extension fields modified by macroinstruction

The following information descriptions of RPL and RPL extension fields:

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
Contains the sense code if any is returned from session initiation macroinstructions. This field is labeled RPL6SNSI in the RPL extension.

**USERFLD**
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, "Return codes," on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>PARAMETER_ERROR---INCOMPLETE_STRUCTURE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR---ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR---ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR---REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR---CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR---INVALID_DATA_ADDRESS OR LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR---PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0014'</td>
<td>PARAMETER_ERROR---INVALID_BIND_PARAMETERS</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR---CID_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR---APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'0064'</td>
<td>X'0000'</td>
<td>ACTIVATION_FAILURE</td>
</tr>
<tr>
<td>X'0068'</td>
<td>X'0000'</td>
<td>LU_MODE_SESSION_LIMIT_EXCEEDED</td>
</tr>
<tr>
<td>X'006C'</td>
<td>X'0000'</td>
<td>SESSION_NOT_PENDING</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS**

**Purpose**

This macroinstruction negotiates the session limits on a mode group between the application program and a partner application.

**Usage**

VTAM determines the new session limits by using the session limits specified on the macroinstruction and the defined session limits of the partner LU. The overall session limits, the contention-winner session limits, and the contention-loser session limits are negotiated. Other parameters, such as draining of a conversation request and responsibility for deactivation, are also negotiated by this macroinstruction.

When this macroinstruction completes, VTAM can activate or deactivate sessions to make them conform to the new session limits. However, sessions already assigned to a conversation are not deactivated.

This macroinstruction corresponds to the INITIALIZE_SESSION_LIMIT, CHANGE_SESSION_LIMIT, and RESET_SESSION_LIMIT verbs in the LU 6.2 architecture.

The APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE macroinstruction can be used by a partner LU that is capable of parallel sessions to define the session limits that can be used in the negotiation when it receives the CNOS request.

For a full discussion of this macroinstruction, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide

**Context**

Input states are not applicable to this macroinstruction.

When a mode is retained for persistent LU-LU sessions, the QUALIFY=CNOS macroinstruction is not allowed.

**Syntax**
APPCCMD CONTROL = OPRCNTL, QUALIFY = CNOS

RPL = rpl_address_field
    (rpl_address_register)

AAREA = rpl_extension_address_field
        (rpl_extension_address_register)

ACB = acb_address_field
    (acb_address_register)

AREA = 0

cnos_session_limits_structure_address_field
    (cnos_session_limits_structure_address_register)

BRANCH = NO

ECB = INTERNAL
    ecb_address_field
        (ecb_address_register)

EXIT = exit_routine_address_field
    (exit_routine_address_register)

LOGMODE = 8-byte_logon_mode_name

LUAFFIN = APPL

LUNAME = 8-byte_lu_name

NAMEUSE = APNAME

OPTCD = (ASY)
    (SYN)
    (KEEPSRB)
    (NKEEPSRB)
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl extension address field

Specify the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb address field

Specify the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
**AREA**=cnos_session_limits_structure_address_field
**AREA**=(cnos_session_limits_structure_address_register)

Specifies the address of a data area containing a CNOS session limits data structure. (See "CNOS session limits data structure (ISTSLCNS)" on page 653 for the VTAM-supplied DSECT that can be used to fill in and test values.) The specification of a session limits structure is optional (the AREA field in the RPL extension would be 0 in this case). The defaults that are used when a session limits structure is omitted are given in the description of each parameter. The fields in the data structure that apply to this macroinstruction are described in the z/OS Communications Server: SNA Programmer’s LU 6.2 Guide. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH**=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH**=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**ECB**=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB**=ecb_address_field

**ECB**=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT**=exit_routine_address_field

**EXIT**=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE**=8-byte_logon_mode_name

Specifies the logon mode name that requires the session limit and contention-winner polarity values to be changed. The mode name specified can be any mode name that is valid as the LOGMODE value on the APPCCMD CONTROL=ALLOC macroinstruction including the SNASVCMG mode name,
which is used for exchanging the CNOS request and reply when the application program and partner application are connected by parallel sessions. However, no CNOS flow occurs to the partner application program as a result of issuing this macroinstruction for the SNASVCMG mode name.

If the session limits control block specifies that SESSLIM=0 and NBRMODE=ALL, the session limit negotiation applies to all noncontrol modes between the two LUs, and this parameter is ignored.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is less than 8 characters, VTAM pads it on the right with blanks. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information). This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**LUNAME=8-byte_lu_name**

Specifies the name of the partner application program to which the change in the session limit and contention-winner polarity values applies. The LU name is a name that is valid as the LU name value on the APPC/APP macroinstruction.

The LU name can be up to 8 characters in length. If it is less than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.
**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program to which the change in the session limit and contention-winner polarity value applies.

If PARMS=(NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS=(NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier is also used to verify and update the logon mode table. It is the same as the NETID value on the APPCCMD CONTROL=ALLOC macroinstruction.

The network identifier can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**RECLEN=cnos_session_limits_structure_length**

Specifies the length of the CNOS session limits data structure supplied by the AREA field. The application program must supply the entire session limits data structure; it cannot supply a partial structure. This field is applicable only
if a CNOS session limits structure is specified by the AREA field. Otherwise, it is ignored by VTAM. This field is labeled RPLLEN in the RPL.

```
RPL=rpl_address_field
RPL=(rpl_address_register)
```

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

```
USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
```

Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

```
VTRINA=vector_address_field
VTRINA=(vector_address_register)
```

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

```
VTRINL=vector_length_field
VTRINL=(vector_length_register)
```

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

- **YES (B'1')**
  - The partner LU accepts the already-verified indicator.

- **NO (B'0')**
  - The partner LU does not accept the already-verified indicator.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**LUAFFIN**

The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if
one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

**NONE (B'00')**

GR affinity is not applicable or is unknown.

**NOTAPPL (B'01')**

GR affinity is not application-owned.

**APPL (B'10')**

GR affinity is application-owned.

**PRSISTVP**

Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**

The partner LU accepts persistent-verification indicators.

**NO (B'0')**

The partner LU does not accept persistent-verification indicators.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. This sense code is returned for the control operator session that VTAM establishes as part of processing the CNOS request. This field is labeled RPL6SNSI in the RPL extension.

**USERFLD**

Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner’s DCE capabilities vector (X'12')
- Name change vector (X'18')
- Partner’s application capabilities vector (X'1A')
## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0001'</td>
<td>OK—AS_SPECIFIED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0002'</td>
<td>OK—AS_NEGOTIATED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0003'</td>
<td>OK_AS_SPECIFIED—PARTNER_LU_KNOWN_BY_DIFFERENT_NAME</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0004'</td>
<td>OK—PARTNER LU SUPPORTS SINGLE SESSION</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0000'</td>
<td>CNOS_ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0001'</td>
<td>CNOS_ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0002'</td>
<td>CNOS_ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0003'</td>
<td>CNOS_ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0004'</td>
<td>CNOS_ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0005'</td>
<td>CNOS_ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0006'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0008'</td>
<td>X'0007'</td>
<td>NETWORK_QUALIFIED_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'000C'</td>
<td>X'0000'</td>
<td>CNOS_RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0000'</td>
<td>COMMAND_RACE_REJECT—PARTNER_GRANTED_RETRY</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0001'</td>
<td>COMMAND_RACE_REJECT—COPR_FOR_LOCAL_LU_RETRIED</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0002'</td>
<td>COMMAND_RACE_REJECT—PARTNER_CNOS_IN_PROGRESS</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0003'</td>
<td>COMMAND_RACE_REJECT—LU_IS_IN_PENDING_SINGLE_STATE</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'0004'</td>
<td>COMMAND_RACE_REJECT—PARTNER_LU_STARTING_SESSION</td>
</tr>
<tr>
<td>X'0020'</td>
<td>X'0000'</td>
<td>CNOS_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0028'</td>
<td>X'0000'</td>
<td>LU_MODE_SESSION_LIMIT_CLOSED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0004'</td>
<td>PARAMETER_ERROR—INVALID_VALUES_FOR_SNASVCMG_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0005'</td>
<td>PARAMETER_ERROR—INVALID_DRAINL_CHANGE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0006'</td>
<td>PARAMETER_ERROR—SNASVCMG_MODE_CANNOT_CURRENTLY_BE_RESET</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0007'</td>
<td>PARAMETER_ERROR—MINWINL_PLUS_MINWINR_EXCEEDS_SESSLIM</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>PARAMETER_ERROR—INCOMPLETE_STRUCTURE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO ECB FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0018'</td>
<td>PARAMETER_ERROR—INVALID_LIMIT_SPECIFIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0019'</td>
<td>PARAMETER_ERROR—SNASVCMG_MODE_ALREADY_INITIALIZED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001A'</td>
<td>PARAMETER_ERROR—ALL_MODES_SPECIFIED_ON_SINGLE_SESSION_LU</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001B'</td>
<td>PARAMETER_ERROR—SNASVCMG_OR_CPSVCMG_MODE_FOR_SINGLE_SESSION_LU</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001C'</td>
<td>PARAMETER_ERROR—SINGLE_SESSION_MODE_ALREADY_INITIALIZED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001E'</td>
<td>CID_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0054'</td>
<td>X'0000'</td>
<td>UNRECOGNIZED_MODE_NAME</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS**

**Purpose**

This macroinstruction responds negatively to a request for session establishment.

**Usage**

This command is issued after the application program is notified through its LOGON or SCIP exit routine that a CINIT or BIND request has been received. The function of this command is similar to the VTAM API commands CLSDST RELEASE and SESSIONC CONTROL=BIND for non-LU 6.2 sessions.

When this macroinstruction is used in a LOGON exit, the RPLAREA field of the read-only RPL contains a read-only copy of the CINIT. After examining the BIND image in the CINIT, the application program can issue this macroinstruction to prevent the session from being activated.

When this macroinstruction is used in a SCIP exit, the RPLAREA field of the read-only RPL contains the address of a read-only copy of the BIND. After examining the BIND, the application program can issue this macroinstruction to prevent the session from being activated.

*APPCCMD CONTROL=OPRCNTL, QUALIFY=DACTSESS* does not correspond to the DEACTIVATE_SESSION verb described in the LU 6.2 architecture.

**Context**

Input states are not applicable to this macroinstruction.

**Syntax**
**Notes:**

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPI or APPCCMD macroinstruction.
Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

ARG=4-byte_session_identifier_(cid)_field

ARG=(4-byte_session_identifier_(cid)_register)

Specifies the CID of the session that was returned to the application program in the parameter list of the LOGON or SCIP exit routine. The specified CID must identify a CINIT or BIND that is queued for this application program.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**USERFLD=4_bytes_of_user_data**

**USERFLD=(user_data_register)**

Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001E'</td>
<td>PARAMETER_ERROR—CID_INVALID</td>
</tr>
<tr>
<td>X'006C'</td>
<td>X'0000'</td>
<td>SESSION_NOT_PENDING</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE

Purpose

This macroinstruction changes the session limit values that have been defined and that are used to negotiate a CNOS request from a partner LU. It also displays selected fields from the LU-mode table in the DEFINE/DISPLAY (ISTSLD) control block.

Usage

This macroinstruction can be used to modify values in a mode name entry that were originally obtained by VTAM from the APPL definition statement or that were supplied by using this macroinstruction previously. There is no direct correlation to the DEFINE verb in the LU 6.2 architecture.

The session limit values that are defined are passed to VTAM in a DEFINE/DISPLAY control block. You must specify the address of this control block in the RPL when issuing the macroinstruction. The address is contained in the RPLAREA field, which can be set with the AREA keyword.

Most of the values specified in the DEFINE/DISPLAY control block are used to negotiate the values received in a CNOS request sent by the partner application program. The values are not affected by, nor do they have any effect upon, the values specified through the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. For example, an application program can define the session limit used for negotiating purposes to be 10, yet later issue a CNOS macroinstruction that specifies a session limit of 20. The defined value of 10 does not restrict the CNOS value of 20; the CNOS value of 20 does not cause the defined value of 10 to be changed.

When this macroinstruction is issued before a CNOS request is negotiated on a mode, VTAM creates an entry in the LU-mode table for the mode and places the defined session limits in the table. The negotiated session limits are not determined until a CNOS request is negotiated.

When issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS, the application program can elect not to specify the limits to be used for CNOS negotiation. If this occurs, VTAM uses the defined limits specified by this macroinstruction as the default for these values. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

This macroinstruction can also be used to help control VTAM’s use of storage. Specifying default limits of 0 with DELETE=ALLOW (in the DEFINE/DISPLAY session limits control block) informs VTAM that this mode name can be deleted from the LU-mode table when the mode name is no longer being used. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for an example of setting the DEFINE/DISPLAY session limits control block.) The execution of this macroinstruction involves only the application program; it does not cause any information to be sent through the network. The specified field values are in effect once the execution completes.

Context

Input states are not applicable to this macroinstruction.
When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
name APPCMD CONTROL OPRCNTL QUALIFY DEFINE (1)
RPL rpl_address_field rpl_address_register (2)
AAREA rpl_extension_address_field rpl_extension_address_register (3)
ACB acb_address_field acb_address_register (3)
AREA define/display_session_limits_structure_address_field define/display_session_limits_structure_address_register (3)
BRANCH NO YES (3)
ECB INTERNAL ecb_address_field ecb_address_register (4) (3)
EXIT exit_routine_address_field exit_routine_address_register (5) (3)
LOGMODE 8-byte_logon_mode_name (1)
LUNAME 8-byte_lu_name (1)
NETID 8-byte_network_identifier (1)
OPTCD (3) (6)
ASY SYN (3) (7) KEEKPSRB (3) (8)
RECLEN define/display_session_limits_structure_length define/display_session_limits_structure_length_register (3)
```
Notes:

1.Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3.Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

- AAREA=rpl_extension_address_field
  - AAREA=(rpl_extension_address_register)
  Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- ACB=acb_address_field
  - ACB=(acb_address_register)
  Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

- AREA=define/display_session_limits_structure_address_field
  - AREA=(define/display_session_limits_structure_address_register)
  Specifies the address of a data area containing a DEFINE/DISPLAY session limits data structure. (See “DEFINE/DISPLAY session limits data structure [ISTSLD]” on page 654 for a description of the IBM-supplied DSECT that can be used to map this storage.) A description of the fields in the control block can be found in z/OS Communications Server: SNA Programmer’s LU 6.2 Guide. This field is labeled RPLAREA in the RPL.

- BRANCH
  Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs
running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=**\(\text{ecb\_address\_field}\)

**ECB=**(\(\text{ecb\_address\_register}\))

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event\_control\_block\_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=**\(\text{exit\_routine\_address\_field}\)

**EXIT=**(\(\text{exit\_routine\_address\_register}\))

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA [Programmer's LU 6.2 Guide](https://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itg.tivoli.itg.xsw.ref/ix009978.html) for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource [Definition Reference](https://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itg.tivoli.itg.xsw.ref/ix009978.html) This field is labeled RPL6MODE in the RPL extension.
LUNAME=8-byte_lu_name
Specifies the name of the partner application program to which the change in the session limit and contention-winner polarity values applies. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program to which the change in the session limit and contention-winner polarity value applies. If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.) The network identifier also is used to find and update the contents of the logon mode table.

This network identifier is an identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RECLEN=define/display_session_limits_structure_length
RECLEN=(define/display_session_limits_structure_register)
Specifies the length of the DEFINE/DISPLAY session limits data structure supplied by the AREA field. The application program must supply the entire session limits data structure; it cannot supply a partial structure. This field is labeled RPLRLLEN in the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0007'</td>
<td>PARAMETER_ERROR—MINWINL_PLUS_MINWINR_EXCEEDS_SESSLIM</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>PARAMETER_ERROR—INCOMPLETE_STRUCTURE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0017'</td>
<td>PARAMETER_ERROR—INVALID_MODE_SPECIFIED</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY

**Purpose**

This macroinstruction returns information associated with an LU or a mode name of an LU.

**Usage**

The information returned from this macroinstruction is contained in the DEFINE/DISPLAY control block. You must specify the address of this control block in the RPL when issuing the macroinstruction. It is contained in the RPLAREA field, which can be set with the AREA keyword. Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.wss?uid=swg21277881) for a description of the control block.

The execution of this macroinstruction involves only the application program. It does not cause any information to be sent through the network. There is no direct correlation to the DISPLAY verb described in the LU 6.2 architecture.

**Context**

Conversation states are not applicable to this macroinstruction.

**Syntax**
APPCCMD — CONTROL — OPRCNTL — QUALIFY — DISPLAY

RPL — rpl_address_field

AREA — rpl_extension_address_field

ACB — acb_address_field

AREA — define/display_session_limits_structure_address_field

AREALEN — define/display_session_limits_structure_length

BRANCH — NO

ECB — INTERNAL

EXIT — exit_routine_address_field

LOGMODE — 8-byte_logon_mode_name

LUNAME — 8-byte_lu_name

NETID — 8-byte_network_identifier

OPTCD — (ASY)

KEEPSRB — (3) (7)

NKEEPSRB — (3) (8)

USERFLD — 4-bytes_of_user_data

VTRINA — vector_address_field

Chapter 1. LU 6.2 macroinstruction syntax and operands 175
Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with
   this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
   Specifies the address of an access method control block that identifies the
   application program that is issuing the APPCCMD macroinstruction. VTAM
   associates conversations with application programs using the conversation ID
   (CONVID). The application program associates conversations with transaction
   programs. Application programs cannot issue APPCCMD macroinstructions in
   address spaces other than the ACB address space. This field is labeled
   RPLDACB in the RPL.

AREA=define/display_session_limits_structure_address_field
AREA=(define/display_session_limits_structure_address_register)
   Specifies the address of a data area for the DEFINE/DISPLAY session limits
   data structure (Refer to z/OS Communications Server: SNA Programmer's LU
   6.2 Guide for a description of this control block.). This field is labeled
   RPLAREA in the RPL.

AREALEN=define/display_session_limits_structure_length
AREALEN=(define/display_session_limits_structure_length_register)
   Specifies the length of the area in which the DEFINE/DISPLAY session limits
data structure is to be returned. If a mode name is specified for the
LOGMODE field, the application program must supply an area large enough
to contain the entire session limits data structure. If LOGMODE=0 is specified, a length of 40 can be coded for this field. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be allocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.zhtml?docid=536114) for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition.
statement. (The MODEENT statement is used to build the logon mode table
tabled in the MODETAB operand of the APPL definition statement for this
application program.) For more information on the MODEENT
macroinstruction, refer to z/OS Communications Server: SNA Resource
Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name
Specifies the name of the partner application program to which the requested
session information applies. The LU name is a name that is valid as the LU
name value on the APPCCMD CONTROL=ALLOC macroinstruction and the
network name of the target LU. It can be up to 8 characters in length. If it is
less than 8 characters in length, VTAM pads it on the right with blanks. This
field is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program to which
the requested session information applies.

If PARMS= (NQNames=NO) is specified on the ACB macroinstruction and
you specify NETID, the NETID value is ignored. If PARMS=
(NQNames=YES) is specified on the ACB macroinstruction, NETID must be
supplied.

If NQNames=YES, LUNAME and NETID are used together to form the
network-qualified name of the target LU. (If NETID is specified, LUNAME
must be specified.) The network identifier also is used to find and update the
contents of the logon mode table. It is the same as the NETID value on the
APPCCMD CONTROL=ALLOC macroinstruction.

This network identifier is the identifier of the target LU. It can be up to 8
characters in length. If it is fewer than 8 characters in length, VTAM pads the
network identifier on the right with blanks. It is labeled RPL6NET in the RPL
extension.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program
immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the
scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM
returns to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual length of the session limits structure being returned by the AREA field. If the application program specified LOGMODE=0, the value 40 is returned for this field. This field is labeled RPLRLEN in the RPL.
A description of the session limits structure is found in the z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**

Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Partner’s application capabilities vector (X’1A’)

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0008’</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0010’</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0013’</td>
<td>PARAMETER_ERROR—NO_CORRESPONDING_MODE_IN_LOGMODE_TABLE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0016’</td>
<td>PARAMETER_ERROR—NO_CORRESPONDING_LU_IN_LOGMODE_TABLE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’001F’</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002B’</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002E’</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002F’</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X’0078’</td>
<td>X’0000’</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X’007C’</td>
<td>X’0000’</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X’0090’</td>
<td>X’0000’</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X’00A8’</td>
<td>X’0000’</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X’00A8’</td>
<td>X’0001’</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X’00A8’</td>
<td>X’0002’</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X’00B0’</td>
<td>X’0001’</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X’00B0’</td>
<td>X’0006’</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X’00B0’</td>
<td>X’0008’</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE

Purpose

This macroinstruction is used to restore modes and their associated persistent LU-LU sessions that are pending recovery.

Usage

This macroinstruction can be used by an application program to restore modes and associated persistent LU-LU sessions that are pending recovery. A mode is restored only after any sessions for the mode are restored. A mode without sessions also must be restored.

A single LU-mode can be restored when the LU name and logon mode are specified on the RESTORE command. All modes for a specific LU are restored when only the LU name is specified. If neither the LU name (with its NETID, if applicable) nor the logon mode is specified, all LUs and modes in the LU-mode table are restored.

The application program specifies the amount of information that is to be returned in the RESTORE control block. To do this, it uses the LIST keyword in the RESTORE macroinstruction. The application program can specify LU-mode table information, LU-mode table and session information, or no information. If the application program requests information to be returned, it must specify the address of a data area to contain that information. The application program must provide the storage area in addition to specifying the address of the storage. This address is contained in the RPLAREA field, which can be set with the AREA keyword.

When the area pointed to by RPLAREA is large enough, the macroinstruction builds multiple RESTORE blocks in this area, if necessary. The RESTORE structures are placed in the area specified until the area is filled or the command is completed, whichever comes first.

For more information about the restore process, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide. For information about the RESTORE control block, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide. For an example of retrieving information that is returned, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

Input states are not applicable to this macroinstruction.

The recovering VTAM application program can issue this macroinstruction only after it issues the SETLOGON START macroinstruction. Otherwise, this macroinstruction is rejected.

Syntax
Input parameters

The following information shows descriptions of the input parameters:

\[
\text{AAREA} = \text{rpl\_extension\_address\_field}
\]
\[
\text{AAREA} = (\text{rpl\_extension\_address\_register})
\]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[
\text{ACB} = \text{acb\_address\_field}
\]
\[
\text{ACB} = (\text{acb\_address\_register})
\]

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.
**AREA**=%(restore_structure_address_field)

Specifies the address of a data area that returns one or more RESTORE data structures. It is used only with LIST=ALL or LIST=NOSESS. It is unnecessary when LIST=NONE is specified. This field is labeled RPLAREA in the RPL.

**AREALEN**=%(restore_structure_length)

Specifies the length of the area in which the RESTORE data structure is to be returned. It is used only with LIST=ALL or LIST=NOSESS. It is unnecessary when LIST=NONE is specified. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

- **ECB=INTERNAL**
  
  Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

- **ECB=ecb_address_field**
  
  Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT**=%(exit_routine_address_field)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LIST**

Specifies the information to be returned in the RESTORE structure, which describes the LUs, modes, and sessions that have been restored. This field is labeled RPL6LIST in the RPL extension.
LIST=ALL
Specifies that all LU, mode, and session information is returned in the
RESTORE structure.

LIST=None
Specifies that no RESTORE structure is returned.

LIST=NOSESS
Specifies that all LU and mode information but no session information is
returned in the RESTORE structure.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name which should be restored. The application
program can specify a logon mode name with an LU name to give greater
granularity over the scope of the command. LOGMODE can be specified only
with LUNAME. The logon mode name cannot be blanks. The logon mode
name can be up to 8 characters in length. If it is less than 8 characters, VTAM
pads it on the right with blanks. If this operand is coded on this
macroinstruction and on the RPL extension, VTAM uses the specifications from
the macroinstruction. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name
Specifies the name of the partner whose modes must be restored. It is the same
as the LU name value on the APPCCMD CONTROL=ALLOC
macroinstruction. It is also the network name of the target LU. When the
application program does not specify the LU name, all LUs and modes are
restored. Otherwise, only the modes associated with a specified LU name are
restored. The LU name can be up to 8 characters in length. If it is less than 8
characters in length, VTAM pads the LU name on the right with blanks. This
field is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner whose modes must be restored.
If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and
you specify NETID, the NETID value is ignored.

If NQNAME=YES is specified, LUNAME and NETID together form the
network-qualified name of the target LU. (If NETID is specified, LUNAME
must be specified.) The network identifier also is used to find the appropriate
information on sessions and modes waiting to be restored. It is the same as the
NETID value on the APPCCMD CONTROL=ALLOC macroinstruction.

This network identifier is the identifier of the target LU. It can be up to 8
characters in length. If it is fewer than 8 characters in length, VTAM pads the
network identifier on the right with blanks. It is labeled RPL6NET in the RPL
extension.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program
immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of information that the application program can associate with this operator control request. The information is returned unchanged when the macroinstruction completes. This data cannot be used by any conversations. It can be used for correlation purposes. This field is labeled RPL6USR in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the length of AREA used to contain the RESTORE structure(s) returned by the AREA field. This field is labeled RPLRLEN in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Returns any unchanged user data that the application program placed in this field. This field is labeled RPL6USR in the RPL extension.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (RESTORE complete.)</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0006'</td>
<td>RESTORE_UNNECESSARY—NO_SESSIONS_TO_RESTORE</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0007'</td>
<td>RESTORE_INCOMPLETE—INPUT_WORK_AREA_TOO_SMALL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>NO_CORRESPONDING_MODE_IN_LM_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>NO_CORRESPONDING_LU_IN_LM_TABLE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0029'</td>
<td>INVALID_LIST_VALUE_SPECIFIED_ON_APPCCMD_FOR_RESTORE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'009C'</td>
<td>X'0001'</td>
<td>RESTORE_REJECTED—RESTORE_ISSUED_BEFORE_SETLOGON_START</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=PREALLOC, QUALIFY=ALLOCD

Purpose

This macroinstruction reserves a session without establishing a conversation. If a session is not available and session limits allow, VTAM activates a session for the conversation, if possible. Session related information can be passed from VTAM to the application before the application sends the FMH-5. The conversation is not active until the application issues the APPCCMD CONTROL=SENDFMH5 macroinstruction.

Usage

QUALIFY=ALLOCD is used when an application program preallocates a conversation and wants VTAM to queue the request if the request cannot be met immediately. This macroinstruction completes when VTAM reserves a session for a conversation or when an error occurs that prevents VTAM from reserving a session.

VTAM finds a session for the conversation as follows:
1. If a session is free, VTAM reserves it for a conversation.
2. If no free sessions exist and session limits allow, VTAM establishes a session and reserves it for a conversation.
3. If a new session cannot be established, VTAM queues the request until a session becomes available or until the session limits are changed to allow the establishment of a new session.

After session initiation, the conversation is reserved in PENDING ALLOCATE state and the application receives the conversation identifier in the CONVID field. The application could also receive the PCID for the session if VTRINA and VTRINL are specified on the preallocation request. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is suspended for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```
APPCCMD CONTROL=PREALLOC QUALIFY=ALLOC
RPL=rpl_address_field (rpl_address_register)
AAREA=rpl_extension_address_field (rpl_extension_address_register)
ACB=acb_address_field (acb_address_register)
BRANCH=YES NO
CONMODE=BUFFCA CS LLCA
CONXMOD=CA CS
```
Chapter 1. LU 6.2 macroinstruction syntax and operands
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCMD macroinstruction completes.
**ECB**=*ecb_address_field*

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT**=*exit_routine_address_field*

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE**=*8-byte_logon_mode_name*

Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program. For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN**=*APPL*

The application program will own the GR affinity for this LU.

**LUAFFIN**=*NOTAPPL*

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.
**LUNAME=8-byte_lu_name**

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS=(NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS=(NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of...
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPL. 

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

RPL=rpl_address_field

RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4-bytes_of_user_data**

USERFLD=(user_data_register)

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**

VTRINA=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**

VTRINL=(vector_length_register)

Specifies the length of the data area where VTAM places vector list information for the application.
This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of the RPL and RPL extension fields:

**AVFA**

The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B’1’)**

The partner LU accepts the already-verified indicator.

**NO (B’0’)**

The partner LU does not accept the already-verified indicator.

**CGID**

Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**

The field in the RPL extension that indicates what state the conversation is in.

It is labeled RPL6CCST in the RPL extension.

This field can have the following values at the completion of this macroinstruction:

- **X'00'**  RESET
- **X'08'**  END_CONV
- **X'FF'**  PENDING_ALLOCATE

**CONVID**

Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to **z/OS Communications Server: SNA Programmer’s LU 6.2 Guide** for more information.

**CONVSEC**

The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

**YES (B’1’)**

The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B’0’)**

The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.
CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.
**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**
The session was established using session-level LU-LU verification.

**NO (B'0')**
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Local nonce vector (X’13’)
- Partner’s nonce vector (X’14’)
- Send FMH_5 sequence number vector (X’15’)

Chapter 1. LU 6.2 macroinstruction syntax and operands 197
- Receive FMH_5 sequence number vector (X'16')
- PCID vector (X'17')
- Name change vector (X'18')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**

The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO ECB FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNSUSABLE_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREALLOC, QUALIFY=CONVGRP

Purpose

This macroinstruction reserves a session for a conversation with a specified conversation group identifier without establishing a conversation. If the specified session is not available and session limits allow, VTAM queues the request until the session becomes available. If the specific session does not exist, VTAM fails the preallocation request. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

Usage

QUALIFY=CONVGRP is used to preallocate a conversation over a specific session that already exists. It provides the ability to serially preallocate a related group of conversations on a particular session. This macroinstruction completes when:

- VTAM assigns the specified session to the conversation.
- The specific session is deactivated.
- An error occurs that prevents VTAM from assigning the session to the conversation.

To indicate the session to be used, the application program specifies the conversation group identifier for the session on the CGID keyword. The conversation group identifier of the session is returned to the application program by the CGID returned field for the following APPCCMD macroinstructions:

- APPCCMD CONTROL=ALLOC
- APPCCMD CONTROL=PREALLOC
- APPCCMD CONTROL=RCVFMH5

VTAM finds the session for the conversation as follows:
1. If the specified session is available, VTAM assigns it to the conversation.
2. If the specified session exists but is not available, VTAM queues the request until the session becomes available.
3. If the specified session is deactivated while the request is queued, the queued request is rejected.

After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. A conversation identifier is returned to the application in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.
Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ \text{AAREA} = \text{rpl_extension_address_field} \]
AAREA=(*rpl_extension_address_register*)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=*acb_address_field
ACB=(*acb_address_register*)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CGID=32-bit_conversation_group_id_field
CGID=(*32-bit_conversation_group_id_register*)
Specifies the 32-bit conversation group ID. This value can be obtained from a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the CGID operand is not specified, VTAM uses the conversation group ID that is already in the RPL6CGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific LUs. It provides a means by which a VTAM LU 6.2 application program and its partner LU can share serially the same session.

CONMODE
Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC can be used to receive data on this conversation.
When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the subsequent APPCCMD CONTROL=SENDMH5. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, for example, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, for example, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPIOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifications that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.
LUAFFIN=APPL
The application program will own the GR affinity for this LU.

LUAFFIN=NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

NAMEUSE
Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

NAMEUSE=APNAME
The application identifies itself to the partner LU by its application network name.

NAMEUSE=GNAME
The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM
returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4-bytes_of_user_data**

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- **VTRINA=0**
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B’1’)
The partner LU accepts the already-verified indicator.

NO (B’0’)
The partner LU does not accept the already-verified indicator.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values at the completion of this macroinstruction:

X’00’ RESET
X’08’ END_CONV
X’FF’ PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B’1’)
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B’0’)
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B’00’)
No data is to be encrypted.

SELECTIVE (B’01’)
The application program specifies the data that is to be encrypted.
REQUIRED (B'11')

All data is to be encrypted.

FDB2

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')

No FMH-5s are waiting to be received by the application program.

LOGMODE

Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUAFFIN

The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is returned at completion only if a requested value is specified when the macroinstruction is issued.

NONE (B'00')

GR affinity is not applicable or is unknown.

NOTAPPL (B'01')

GR affinity is not application-owned.
APPL (B'10')
GR affinity is application-owned.

LUNAME
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SN5 in the RPL extension.

SESSID
The field in the RPL extension that returns a session instance identifier of the
session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B’1’)**
The session was established using session-level LU-LU verification.

**NO (B’0’)**
The session was not established using session-level LU-LU verification.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Local nonce vector (X’13’)
- Partner’s nonce vector (X’14’)
- Send FMH_5 sequence number vector (X’15’)
- Receive FMH_5 sequence number vector (X’16’)
- PCID vector (X’17’)
- Name change vector (X’18’)
- Session information vector (X’19’)
- Partner’s application capabilities vector (X’1A’)

**State changes**
The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to [Chapter 2](#) “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0000’</td>
<td>X’000A’</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X’0000’</td>
<td>X’000B’</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0000’</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0001’</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000F’</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000F’</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0000’</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=PREALLOC, QUALIFY=CONWIN**

**Purpose**

This macroinstruction reserves a contention-winner session for a conversation, if session limits allow, without establishing a conversation. If a contention-winner session is not available, VTAM queues the request for later completion. After a session is reserved, Session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

**Usage**

QUALIFY=CONWIN is used when an application program preallocates a conversation and wants VTAM to queue the request if no contention-winner session can be assigned. This macroinstruction completes when VTAM reserves a contention-winner session or an error occurs that prevents VTAM from assigning a session.

VTAM finds a session for the conversation as follows:

1. If a contention-winner session is currently available, VTAM reserves it for a conversation.
2. If no contention-winner session is available and session limits allow, VTAM establishes a new contention-winner session and assigns it to the conversation.
3. If a new contention-winner session cannot be established, VTAM queues the request until a contention-winner session is available or session limits are changed to allow a new contention-winner session to be activated.

For this macroinstruction to complete successfully, the session limits must define at least one contention-winner session.
If contention-winner sessions are deactivated under normal conditions and an APPCCMD CONTROL=PREALLOC, QUALIFY=CONWIN request is queued, VTAM activates another contention-winner session to meet the queued request.

After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. A conversation identifier is returned to the application in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```plaintext
APPCCMD CONTROL=PREALLOC, QUALIFY=CONVGRP, RPL=rpl_address_field, AAREA=rpl_extension_address_field, ACB=acb_address_field, BRANCH=YES, CONMODE=BUFFCA, CONXMOD=CA
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSPRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[
\text{AAREA= rpl\_extension\_address\_field} \\
\text{AAREA=(rpl\_extension\_address\_register)} \\
\text{ACB= acb\_address\_field} \\
\text{ACB=(acb\_address\_register)} \\
\text{BRANCH} \\
\]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.
**CONMODE**

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

\textbf{EXIT=}exit\_routine\_address\_field
\textbf{EXIT=}(exit\_routine\_address\_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

\textbf{LOGMODE=}8-byte\_logon\_mode\_name
Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to \textit{z/OS Communications Server: SNA Programmer's LU 6.2 Guide} for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to \textit{z/OS Communications Server: SNA Resource Definition Reference}. This field is labeled RPL6MODE in the RPL extension.

\textbf{LUAFFIN}
Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

\textbf{LUAFFIN=}APPL
The application program will own the GR affinity for this LU.

\textbf{LUAFFIN=}NOTAPPL
VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is meaningful only when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to \textit{z/OS Communications Server: SNA Programmer's LU 6.2 Guide}

\textbf{LUNAME=}8-byte\_lu\_name
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is
located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS=(NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS=(NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field

RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data

USERFLD=(user_data_register)

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field

VTRINA=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field

VTRINL=(vector_length_register)

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.

CGID
Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values at the completion of this macroinstruction:

X'00'   RESET
X'08'   END_CONV
X'FF'   PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.
No data is to be encrypted.

The application program specifies the data that is to be encrypted.

All data is to be encrypted.

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

No FMH-5s are waiting to be received by the application program.

The field in the RPL extension that indicates the requested (on input) or actual (on output) ownership of a Generic Resource affinity with the partner LU, if one exists. A result value is only returned at completion if a requested value is specified when the macroinstruction is issued.

GR affinity is not applicable or is unknown.

GR affinity is not application-owned.

GR affinity is application-owned.

Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

The partner LU accepts persistent-verification indicators.

The partner LU does not accept persistent-verification indicators.

The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**
The session was established using session-level LU-LU verification.

**NO (B'0')**
The session was not established using session-level LU-LU verification.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner’s DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner’s nonce vector (X'14')
- Send FMH_5 sequence number vector (X'15')
- Receive FMH_5 sequence number vector (X'16')
- PCID vector (X'17')
• Name change vector (X'18')
• Session information vector (X'19')
• Partner's application capabilities vector (X'1A')

**State changes**

The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to [Chapter 2](#) "Return codes," on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_ORRESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_DIFFERS_FROM_ASSOCIATED_NAME</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNUSABLE_NAME_ENTRY</td>
</tr>
</tbody>
</table>

Chapter 1. LU 6.2 macroinstruction syntax and operands 221
APPCCMD CONTROL=PREALLOC, QUALIFY=IMMED

Purpose

This macroinstruction reserves an active contention-winner session for a conversation, if session limits allow, without establishing a conversation. If no session is available, the preallocation request fails. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

Usage

QUALIFY=IMMED is used to preallocate a conversation when the application program needs an immediate response from VTAM. This macroinstruction completes successfully only when an active contention-winner session is available to be reserved for a conversation. If the request cannot be met immediately, VTAM does not queue it. VTAM neither tries to activate a new session nor bids on a contention-loser session.

When a conversation is preallocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program must associate a conversation with a particular transaction by using the conversation identifier.

The application program can specify how expedited data is to be received.

Context

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

Syntax
APPCCMD CONTROL = PREALLOC, QUALIFY = IMMED
RPL = rpl_address_field (rpl_address_register)
AAREA = rpl_extension_address_field (rpl_extension_address_register)
ACB = acb_address_field (acb_address_register)
BRANCH = NO YES
CONMODE = BUFFCA CS LLCA
CONXMOD = CA CS
ECB = INTERNAL ecb_address_field (ecb_address_register)
EXIT = exit_routine_address_field (exit_routine_address_register)
LOGMODE = 8-byte_logon_mode_name
LUNAME = 8-byte.lu_name
NETID = 8-byte_network_identifier

Chapter 1. LU 6.2 macroinstruction syntax and operands 223
Notes:
1  Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2  See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3  Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4  ECB is meaningful only for asynchronous operations.
5  EXIT is meaningful only for asynchronous operations.
6  You can code more than one suboperand on OPTCD, but no more than one from each group.
7  KEEPSRB is meaningful only for synchronous operations.
8  NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the
logical-record format of the data. LLCA corresponds to FILL=LL
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC
macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the
APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received by either a specific-type
macroinstruction or an any-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received only by a specific-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD,
QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

LOGMODE=8-byte_logon_mode_name
Specifies the logon mode name designating the network properties for the
session to be preallocated for this conversation. The network properties
include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8
characters in length. If it is fewer than 8 characters in length, VTAM pads it on
the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a
logon mode name that does not exist in the logon mode table, VTAM uses the
mode name of blanks to retrieve the default mode entry when processing
session activation requests. (Refer to z/OS Communications Server: SNA
Programmer's LU 6.2 Guide for more information.) This logon mode name
corresponds to a logon mode name specified in a MODEENT definition
statement. (The MODEENT statement is used to build the logon mode table
named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

LUNAME=8-byte_lu_name
Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

NETID=8-byte_network_identifier
Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.
RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

YES (B'1')
The partner LU accepts the already-verified indicator.

NO (B'0')
The partner LU does not accept the already-verified indicator.
CGID
Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

CONSTATE
The field in the RPL extension that indicates what state the conversation is in.
It is labeled RPL6CCST in the RPL extension.
This field can have the following value at the completion of this macroinstruction:

X'00'    RESET
X'08'    END_CONV
X'FF'    PENDING_ALLOCATE

CONVID
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

Note: The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

CONVSECP
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLS in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

YES (B'1')
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

NO (B'0')
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

CRYPTLVL
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

NONE (B'00')
No data is to be encrypted.

SELECTIVE (B'01')
The application program specifies the data that is to be encrypted.

REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received.
by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**PRSISTVP**
Indicates that the partner LU accepts requests for persistent verification. This field is labeled RPL6PV in the RPL extension.

**YES (B'1')**
The partner LU accepts persistent-verification indicators.

**NO (B'0')**
The partner LU does not accept persistent-verification indicators.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension. The format of the session instance identifier is described in the [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/SFZTHD_6.2.0/com.ibm.zos.r62.sna.doc/guide/r62sna0942.htm).

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.
YES (B’1’)
Indicates that the session was established using session-level LU-LU
verification.

NO (B’0’)
Indicates that the session was not established using session-level LU-LU
verification.

Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA
parameter:
• VTAM-to-APPL required information vector (X’10’)
• Partner’s DCE capabilities vector (X’12’)
• Local nonce vector (X’13’)
• Partner’s nonce vector (X’14’)
• Send FMH_5 sequence number vector (X’15’)
• Receive FMH_5 sequence number vector (X’16’)
• PCID vector (X’17’)
• Name change vector (X’18’)
• Session information vector (X’19’)
• Partner’s application capabilities vector (X’1A’)

State changes

The conversation state is PENDING_ALLOCATE after successful completion of this
macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application
program when it issues this APPCCMD macroinstruction. Refer to Chapter 2,
“Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0004’</td>
<td>X’000E’</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0000’</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0001’</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’001F’</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002B’</td>
<td>PARAMETER_ERROR—NETWORK-QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002E’</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’002F’</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X’0058’</td>
<td>X’0000’</td>
<td>UNSUCCESSFUL_SESSION_NOT_AVAILABLE</td>
</tr>
<tr>
<td>X’0070’</td>
<td>X’0000’</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X’0074’</td>
<td>X’0000’</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X’0078’</td>
<td>X’0000’</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X’007C’</td>
<td>X’0000’</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X’0090’</td>
<td>X’0000’</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREALLOC, QUALIFY=WHENFREE

Purpose

This macroinstruction reserves a session for a conversation, if session limits allow, without establishing a conversation. If a session is not available and one cannot be activated, VTAM returns control to the application program. After a session is reserved, session related information can be passed between the application program and VTAM. The conversation is not active until the APPCCMD CONTROL=SENDFMH5 is issued.

Usage

QUALIFY=WHENFREE is used when an application program preallocates a conversation and wants VTAM to search for a session that satisfies the ALLOCATE request. This macroinstruction completes when VTAM reserves a session for a conversation or when VTAM cannot reserve a session and returns control to the application program with a return code of X'0004', X'0001'.

VTAM finds a session for the conversation as follows:
1. If a session is available, VTAM reserves it for a conversation.
2. If no available sessions exist and session limits allow, VTAM establishes a session and reserves it for a conversation.
3. If a session cannot be established and session activation requests are pending, VTAM queues the PREALLOCATE request until the request is satisfied or until all pending session activation requests are used. If the pending session activation requests are used before the PREALLOCATE request is satisfied, VTAM fails the PREALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001'.
4. If a session cannot be established and no session activation request is pending that might satisfy the PREALLOCATE request, VTAM fails the PREALLOCATE request with an RCPRI, RCSEC code of X'0004', X'0001' and returns control to the application program.
After session initiation, the session is reserved to receive session related information if necessary and is assigned to a conversation. When a conversation is preallocated, VTAM assigns a conversation identifier to it. This identifier is returned in the CONVID field. The application program associates a conversation with a particular transaction by using the conversation identifier (CONVID).

The application program can specify how expedited data is to be received on this conversation.

**Context**

This macroinstruction is independent of conversation states when it is issued. The initial conversation state is created after this macroinstruction completes.

When a mode is retained for persistent LU-LU sessions, this macroinstruction is not allowed.

**Syntax**

```plaintext
APPCCMD CONTROL = PREALLOC, QUALIFY = WHENFREE

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

BRANCH = NO

CONMODE = BUFFCA

CONXMOD = CA

ECB = INTERNAL

EXIT = exit_routine_address_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

ECB is meaningful only for asynchronous operations.

EXIT is meaningful only for asynchronous operations.

You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA
corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled.
When the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**LOGMODE=8-byte_logon_mode_name**

Specifies the logon mode name designating the network properties for the session to be preallocated for this conversation. The network properties include, for example, the class of service to be used.

The logon mode name cannot be blanks. The logon mode name can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads it on the right with blanks.

If the logon mode parameter on the APPCCMD macroinstruction specifies a logon mode name that does not exist in the logon mode table, VTAM uses the mode name of blanks to retrieve the default mode entry when processing session activation requests. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.) This logon mode name corresponds to a logon mode name specified in a MODEENT definition statement. (The MODEENT statement is used to build the logon mode table named in the MODETAB operand of the APPL definition statement for this application program.) For more information on the MODEENT macroinstruction, refer to z/OS Communications Server: SNA Resource Definition Reference. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte_lu_name**

Specifies the name of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is located. This LU name is the network name of the target LU. It can be up to 8 characters in length. If it is less than 8 characters in length, VTAM pads the LU name on the right with blanks. It is labeled RPL6LU in the RPL extension.

**NAMEUSE**

Specifies the preferred type of name identifying the application to the partner LU in the PLU name structured user data subfield in the BIND requests or in
the SLU name structured user data subfield in BIND responses sent while the application is acting as a generic resource.

**NAMEUSE=APNAME**

The application identifies itself to the partner LU by its application network name.

**NAMEUSE=GNAME**

The application identifies itself to the partner LU by a generic resource name.

The NAMEUSE value is honored if no sessions currently exist with the partner LU and if no partner affinity is being retained. If any active or pending sessions exist or a partner affinity is being retained, the previous type of name is used for new sessions. If NAMEUSE is not specified, the generic resource name will be the preferred name used when starting sessions as a generic resource.

**NETID=8-byte_network_identifier**

Specifies the network identifier of the partner application program at which the remote transaction program, specified in the FMH-5 supplied in the AREA field, is found.

If PARMS= (NQNAMES=NO) is specified on the ACB macroinstruction and you specify NETID, the NETID value is ignored. If PARMS= (NQNAMES=YES) is specified on the ACB macroinstruction, NETID must be supplied.

If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified name of the target LU. (If NETID is specified, LUNAME must be specified.)

This network identifier is the identifier of the target LU. It can be up to 8 characters in length. If it is fewer than 8 characters in length, VTAM pads the network identifier on the right with blanks. It is labeled RPL6NET in the RPL extension.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

RTSRTRN
Specifies the manner in which the Request_To_Send_Received indication is to
be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH
Specifies that the Request_To_Send_Received indication can be reported in
the SIGRCV and SIGDATA fields on all APPCCMDs that return these
parameters.

RTSRTRN=EXPD
Specifies that the Request_To_Send_Received indication can be reported in
the SIGRCV and SIGDATA fields on an APPCCMD
CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4-bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data to be associated with the new conversation.
Whenever an APPCCMD macroinstruction completes for this conversation,
VTAM places in the USERFLD field of the RPL extension the 4 bytes that were
supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a
remote application program). This field is labeled RPL6USR in the RPL
extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list
information for the application.

This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return
the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information
for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This
field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

AVFA
The field in the RPL extension that indicates whether the partner LU accepts
the already-verified indicator in place of the password security access subfield on the FMH-5s that it receives. This field is labeled RPL6AVFA in the RPL extension.

**YES (B'1')**
The partner LU accepts the already-verified indicator.

**NO (B'0')**
The partner LU does not accept the already-verified indicator.

**CGID**
Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**
The field in the RPL extension that indicates what state the conversation is in.

It is labeled RPL6CCST in the RPL extension.

This field can have the following value at the completion of this macroinstruction:

- X'00'  RESET
- X'08'  END_CONV
- X'FF'  PENDING_ALLOCATE

**CONVID**
Specifies the resource identifier of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**Note:** The value in this field is returned before this macroinstruction completes to allow the application to cancel the conversation allocation process before it completes. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

**CONVSECP**
The field in the RPL extension that returns an indication of whether the partner LU accepts FMH-5s that include security subfields and indicators. The indication is either YES or NO (RPL6CLSA in RPL6RTUN set on or off). This field is labeled RPL6CLSA in the RPL extension.

**YES (B'1')**
The partner LU accepts FMH-5s with security subfields and indicators. The subfields allow the application program to include a password, user ID, and profile on allocation requests.

**NO (B'0')**
The partner LU does not accept FMH-5s with security subfields. If this is the case, VTAM strips out any security subfields and indicators that might be included on an allocation request.

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

**NONE (B'00')**
No data is to be encrypted.

**SELECTIVE (B'01')**
The application program specifies the data that is to be encrypted.
REQUIRED (B'11')
All data is to be encrypted.

FDB2
The field in the RPL in which a global VTAM secondary return code is
returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to
be received by the application program. If multiple FMH-5s are waiting to be
received, FMH5LEN specifies the length of the longest FMH-5 to be received
by the application program. This field has meaning only when
FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5
has been received. The indication is either YES or NO (RPL6RMH5 set on or
off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application
programs. The FMH5RCV field continues to be set to YES as long as an
FMH-5 is waiting to be received by the application program. The
application program must issue APPCCMD CONTROL=RCVFMH5 in
order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LUAFFIN
The field in the RPL extension that indicates the requested (on input) or actual
(on output) ownership of a Generic Resource affinity with the partner LU, if
one exists. A result value is only returned at completion if a requested value is
specified when the macroinstruction is issued.

NONE (B'00')
GR affinity is not applicable or is unknown.

NOTAPPL (B'01')
GR affinity is not application-owned.

APPL (B'10')
GR affinity is application-owned.

PRSISTVP
Indicates that the partner LU accepts requests for persistent verification. This
field is labeled RPL6PV in the RPL extension.

YES (B'1')
The partner LU accepts persistent-verification indicators.

NO (B'0')
The partner LU does not accept persistent-verification indicators.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when
RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL
extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary
return code is returned to the application program. This field has meaning
only when RTNCD=’X’00’ and FDB2=’X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. However, not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. If the APPCCMD failed because an attempt to establish a session failed, this field contains a sense code indicating the cause of the failure. This field is labeled RPL6SNSI in the RPL extension.

**SESSID**
The field in the RPL extension that returns a session instance identifier of the session over which the FMH-5 flows. The FMH-5 is supplied by the application program using the AREA field. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range of 0-8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B’1’)**
The session was established using session-level LU-LU verification.

**NO (B’0’)**
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Local nonce vector (X’13’)
- Partner’s nonce vector (X’14’)
- Send FMH_5 sequence number vector (X’15’)
- Receive FMH_5 sequence number vector (X’16’)
- PCID vector (X’17’)
- Name change vector (X’18’)
- Session information vector (X’19’)
- Partner’s application capabilities vector (X’1A’)

242  z/OS V2R1.0 Communications Server: SNA Programmer’s LU 6.2 Reference
State changes

The conversation state is PENDING_ALLOCATE after successful completion of this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>SESSIONS_WILL_USE_APPL_NAME_GENERIC_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>SESSIONS_WILL_USE_GENERIC_NAME_APPL_NAME_REQUESTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000E'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000F'</td>
<td>DEALLOCATION_REQUESTED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>PARAMETER_ERROR—INVALID_LU_NAME_OR_NETWORK_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>PARAMETER_ERROR—INVALID_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>PARAMETER_ERROR—NETWORK_QUALIFIED_NAME_REQUIRED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0003'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_VARIANT_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0004'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_SUPPLIED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0005'</td>
<td>NAME_RESOLUTION_ERROR—PARTNER_NETWORK_NAME_MISMATCH</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0006'</td>
<td>NAME_RESOLUTION_ERROR—LUNAME_FOUND_IN_UNASSIGNED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0007'</td>
<td>NAME_RESOLUTION_ERROR—NAME_RETURNED_FOUND_IN_UNASSIGNED_NAME_ENTRY</td>
</tr>
<tr>
<td>X'00B0'</td>
<td>X'0008'</td>
<td>NAME_RESOLUTION_ERROR—LU_NAME_FOUND_IN_A_DISASSOCIATED_NAME_ENTRY</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREPRCV, QUALIFY=CONFIRM

Purpose

This macroinstruction is used to change the local conversation state of a half-duplex conversation from SEND to RECEIVE. This macroinstruction flushes the SEND buffer and then sends a confirmation request to the partner application program. When a positive acknowledgment to the confirmation is received, the macroinstruction changes the conversation state from SEND to RECEIVE.

Usage

This macroinstruction synchronizes the communication between the local and remote LUs. It is issued when the application program has finished sending and is ready to receive. This macroinstruction causes VTAM to flush the SEND buffer (in the same way as it does for APPCCMD CONTROL=SEND, QUALIFY=CONFIRM) and send a confirmation request to the partner LU.

If a positive acknowledgment to the confirmation is received (as indicated by an RCPRI of X'0000'), VTAM changes the conversation from SEND to RECEIVE state in preparation to receive data. If a negative confirmation response is received (RCPRI is not X'0000'), the state of the conversation is found in the CONSTATE field.

This macroinstruction corresponds to the PREPARE_TO_RECEIVE (TYPE=CONFIRM) verb described in the LU 6.2 architecture.

Context

This macroinstruction can be issued only from SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

The local application can specify whether this acknowledgment is a response (LOCKS=SHORT) or data received from the partner (LOCKS=LONG). The LOCKS=SHORT specification completes more quickly and the LOCKS=LONG specification uses fewer transmission flows and processing cycles.

Syntax
Chapter 1. LU 6.2 macroinstruction syntax and operands
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFRGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction
programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- **BRANCH=NO**
  
  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

- **BRANCH=YES**
  
  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- **CONMODE=BUFFCA**
  
  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

- **CONMODE=CS**
  
  Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

- **CONMODE=LLCA**
  
  Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

- **CONMODE=SAME**
  
  Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.
CONXMOD
Specifications the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOCKS
Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb, as described in the LU 6.2 architecture. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information on the use of this function.) This field is labeled RPL6LOCK in the RPL extension.

LOCKS=LONG
Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner application. The receipt of data presumes an affirmative reply to the confirmation request. The local application program must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the macroinstruction to complete.
LOCKS=SHORT
Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

Note: The partner cannot determine whether LOCKS=LONG or SHORT was specified. The APPCCMD CONTROL=SEND, QUALIFY=CONFRMD must be specified in either case.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:
- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values:
- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. It is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004' ALLOCATION_ERROR
- X'0014' DEALLOCATE_ABEND_PROGRAM
- X'0018' DEALLOCATE_ABEND_SERVICE
- X'001C' DEALLOCATE_ABEND_TIMER
- X'0030' PROGRAM_ERROR_NO_TRUNC
- X'0034' PROGRAM_ERROR_PURGING
- X'0038' PROGRAM_ERROR_TRUNC
- X'003C' SERVICE_ERROR_NO_TRUNC
- X'0040' SERVICE_ERROR_PURGING
- X'0044' SERVICE_ERROR_TRUNC
- X'005C' USER_ERROR_CODE_RECEIVED

NO (B'0')

Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE

The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also
can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPI values have sense data associated with them. If the RCPI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6NSI in the RPL extension.

**USERFLD**

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

These changes are applicable when RCPI indicates OK.

The conversation state is RECEIVE after successful processing.

See[Chapter 2, “Return codes,” on page 591] for state changes associated with other return codes.

**Return codes**

The following (RCPI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See[Chapter 2, “Return codes,” on page 591] for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
</tbody>
</table>
**APPCCMD CONTROL=PREPRCV, QUALIFY=DATACON**

**Purpose**

This macroinstruction sends data, flushes the SEND buffer, and then sends a confirmation request to the partner application program. If a positive confirmation acknowledgment is received, the local conversation state is changed from SEND to RECEIVE state.

**Usage**

This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=PREPRCV, QUALIFY=CONFIRM. VTAM flushes the SEND buffer and sends the data that is specified on the macroinstruction. A confirmation request follows. The application program must ensure that the data sent completes a logical record.

If a positive acknowledgment to the confirmation request is received, the conversation is placed in RECEIVE state. When this macroinstruction completes without error, the state of the conversation is contained in the CONSTATE field.

This macroinstruction corresponds to the verbs SEND_DATA followed by PREPARE_TO_RECEIVE (TYPE=CONFIRM) described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

Chapter 1. LU 6.2 macroinstruction syntax and operands 253
APPCCMD CONTROL = PREPRCV, QUALIFY = DATACON

RPL RPL_address_field
   RPL_address_register

AAREA AAREA_extension_address_field
   AAREA_extension_address_register

ACB ACB_address_field
   ACB_address_register

AREA AREA_buffer_list_address_field
   AREA_buffer_list_address_register

BRANCH BRANCH
   YES

CONMODE CONMODE
   BUFFCA
      CS
      LLCA
      SAME

CONVID CONVID
   32-bit_resource_id_field
      32-bit_resource_id_register

CONXMOD CONXMOD
   CA
      CS
      SAME

CRYPT CRYPT
   NO

ECB ECB_address_field
   ECB_address_register

EXIT EXIT_address_field
   EXIT_address_register

LOCKS LOCKS
   LONG
   SHORT
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

Following are descriptions of the input parameters:

\[ \text{AREA} = \text{rpl_extension_address_field} \]
\[ \text{AREA} = (\text{rpl_extension_address_register}) \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ \text{ACB} = \text{acb_address_field} \]
\[ \text{ACB} = (\text{acb_address_register}) \]

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY
can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**CRYPT**
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

**CRYPT=NO**
Do not encrypt data before it is sent.
CRYPT=YES

Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOLOPT1 field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

LOCKS

Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb, as described in the LU 6.2 architecture. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information on the use of this function.) This field is labeled RPL6LOCK in the RPL extension.

LOCKS=LONG

Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner application. The receipt of data presumes an affirmative reply to the confirmation request. The local application program must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the macroinstruction to complete.

LOCKS=SHORT

Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

Note: The partner cannot determine whether LOCKS=LONG or SHORT was specified. The APPCCMD CONTROL=SEND, QUALIFY=CONFRMD must be specified in either case.
OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPOPT6 field of the RPL.

OPTCD=XBUFFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:
Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.

Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLEN=data_length
RECLEN=(data_length_register)
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

Following are descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. It is labeled RPL6CCST in the RPL extension.

This field can have the following values:
X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05' RECEIVE_CONFIRM_DEALLOCATE
X'07' PENDING_END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.
**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. It is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue an APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner application program. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004'
  ALLOCATION_ERROR
- X'0014'
  DEALLOCATE_ABEND_PROGRAM
- X'0018'
  DEALLOCATE_ABEND_SERVICE
- X'001C'
  DEALLOCATE_ABEND_TIMER
- X'0030'
  PROGRAM_ERROR_NO_TRUNC
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RPLXSRV**
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/downloads/press/1190662).

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM. This field is labeled RPL6SNSI in the RPL extension.
SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. It is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). This field is labeled RPL6RSIG in the RPL extension.

YES (B’1’)
A SIGNAL RU has been received from the partner application program. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B’0’)
No SIGNAL RU has been received from the partner application program. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STD5 in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

These changes are applicable when RCPRI indicates OK.

The conversation enters RECEIVE state after successful completion of the macroinstruction.

See [Chapter 2, “Return codes,” on page 591](#) for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU

Purpose

This macroinstruction sends data to a partner LU and flushes the SEND buffer. The conversation state for the application program is then changed from SEND to RECEIVE.

Usage

This macroinstruction combines the functions of two macroinstructions, APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=PREPRCV, QUALIFY=FLUSH. VTAM sends any data currently in the SEND buffer. This data is followed by the data specified on the macroinstruction to the partner LU. The application program must ensure that the data sent completes a logical record.

If the data is sent successfully, the conversation is placed in RECEIVE state. The conversation state is found in the CONSTATE field when the macroinstruction completes.

This macroinstruction corresponds to the SEND_DATA followed by PREPARE_TO_RECEIVE (TYPE=FLUSH) verbs described in the LU 6.2 architecture.

Context

This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for synchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ \text{AAREA=\text{rpl\_extension\_address\_field}} \]

\[ \text{AAREA=\text{(rpl\_extension\_address\_register)}} \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ \text{ACB=\text{acb\_address\_field}} \]

\[ \text{ACB=\text{(acb\_address\_register)}} \]

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_or_buffer_list_address_field

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE,
QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.
CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT1 field of the RPL.
OPTCD=BUFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X’002C’, X’0010’ (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

RECLEN=data_length
RECLEN=(data_length_register)
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=rpl_address_field
**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE-ONLY_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.
LOGRCV

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')

An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004' ALLOCATION_ERROR
- X'0014' DEALLOCATE_ABEND_PROGRAM
- X'0018' DEALLOCATE_ABEND_SERVICE
- X'001C' DEALLOCATE_ABEND_TIMER
- X'0030' PROGRAM_ERROR_NO_TRUNC
- X'0034' PROGRAM_ERROR_PURGING
- X'0038' PROGRAM_ERROR_TRUNC
- X'003C' SERVICE_ERROR_NO_TRUNC
- X'0040' SERVICE_ERROR_PURGING
- X'0044' SERVICE_ERROR_TRUNC
- X'005C' USER_ERROR_CODE_RECEIVED

NO (B'0')

Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in
the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RPLXSRV**

A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](http://www.ibm.com/servers/eserver/zseries/zos/bkserv/).

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that VTAM did not recognize.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. It is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B’1’)**

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.
NO (B’0’)
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list entry (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list entry (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.

The conversation state is RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0002’</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0004’</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0005’</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0007’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0008’</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0009’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000A’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000B’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000C’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
</tbody>
</table>
### RCPRI  |  RCSEC  |  Meaning
--- | --- | ---
X'0004'  |  X'000D'  |  ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM
X'0014'  |  X'0000'  |  DEALLOCATE_ABEND_PROGRAM
X'0018'  |  X'0000'  |  DEALLOCATE_ABEND_SERVICE
X'001C'  |  X'0000'  |  DEALLOCATE_ABEND_TIMER
X'0024'  |  X'0000'  |  LOGICAL_RECORD_BOUNDARY_ERROR
X'002C'  |  X'0002'  |  PARAMETER_ERROR—INVALID_CONVERSATION_ID
X'002C'  |  X'0003'  |  PARAMETER_ERROR—INVALID_LL
X'002C'  |  X'000C'  |  PARAMETER_ERROR—ZERO_EXIT_FIELD
X'002C'  |  X'000D'  |  PARAMETER_ERROR—ZERO_ECB_FIELD
X'002C'  |  X'000E'  |  PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE
X'002C'  |  X'000F'  |  PARAMETER_ERROR—CONTROL_BLOCK_INVALID
X'002C'  |  X'0010'  |  PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH
X'002C'  |  X'0011'  |  PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING
X'002C'  |  X'0012'  |  PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID
X'002C'  |  X'001F'  |  PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC
X'002C'  |  X'0024'  |  PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED
X'002C'  |  X'0025'  |  PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT
X'002C'  |  X'0028'  |  PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE
X'002C'  |  X'0032'  |  PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD
X'0034'  |  X'0000'  |  PROGRAM_ERROR_PURGING
X'0040'  |  X'0000'  |  SERVICE_ERROR_PURGING
X'0048'  |  X'0000'  |  RESOURCE_FAILURE_NO_RETRY
X'004C'  |  X'0000'  |  RESOURCE_FAILURE_RETRY
X'0050'  |  X'0000'  |  STATE_ERROR
X'005C'  |  X'0000'  |  USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE
X'0070'  |  X'0000'  |  TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE
X'0078'  |  X'0000'  |  VTAM_INACTIVE_FOR_YOUR_ACB
X'007C'  |  X'0000'  |  REQUEST_ABORTED
X'0088'  |  X'0000'  |  CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND
X'0090'  |  X'0000'  |  APPLICATION_NOT_APPC_CAPABLE
X'0094'  |  X'0000'  |  INVALID_CONDITION_FOR_SENDING_DATA
X'0098'  |  X'0000'  |  STORAGE_SHORTAGE_WHILE_SENDING_DATA
X'00A0'  |  X'0004'  |  CONTROL/QUALIFY_VALUE_NOT_VALID_FOR_FULL-DUPLEX_CONVERSATIONS
X'00A0'  |  X'0006'  |  PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION
X'00A8'  |  X'0000'  |  ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION
X'00A8'  |  X'0001'  |  ENVIRONMENT_ERROR—SUSPEND_FAILURE
X'00A8'  |  X'0002'  |  ENVIRONMENT_ERROR—RESUME_FAILURE
X'00B4'  |  X'0001'  |  CSM_DETECTED_ERROR—NOT_SPECIFIED
X'00B4'  |  X'0002'  |  CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED
X'00B4'  |  X'0003'  |  CSM_DETECTED_ERROR—INVALIDINSTANCE_ID_SPECIFIED

### APPCCMD CONTROL=PREPRCV, QUALIFY=FLUSH

#### Purpose

This macroinstruction flushes the application program's SEND buffer and changes the conversation state from SEND to RECEIVE.
Usage

This macroinstruction executes the function of the APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction. The application program must ensure that the data in the SEND buffer completes a logical record.

If the data is sent successfully, the conversation is put in RECEIVE state. The conversation state is in the CONSTATE field when the macroinstruction completes.

This macroinstruction corresponds to the PREPARE_TO_RECEIVE (TYPE=FLUSH) verb described in the LU 6.2 architecture.

Context

This macroinstruction can be issued from the SEND or PENDING_SEND conversation state. This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD  CONTROL=PREPRCV, QUALIFY=FLUSH
```

1. **name**
2. **RPL**
3. **AAREA**
4. **ACB**
5. **CONMODE**
6. **CONVID**
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPERSB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME

Specifies that the continuation mode of the conversation is to remain unchanged.
**CONVID**=32-bit_resource_id_field

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

 Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

**CONXMOD=CS**

 Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=SAME**

 Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

 Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**

**ECB=(ecb_address_register)**

 Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

 Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

 Specifies that control is to be returned to the application program...
immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. It can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.
YES (B’1’)
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 to receive an FMH-5.

NO (B’0’)
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.

The conversation state is RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. Refer to Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0024’</td>
<td>X’0000’</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY

Purpose

This macroinstruction receives expedited information from any active conversation whose expedited information mode is continue-any. VTAM will wait for expedited information to arrive on a conversation in continue-any mode to satisfy the macroinstruction request.

Usage

This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC for each conversation, the application program can put the conversations in continue-any mode for receiving expedited information and issue a single APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY.

The application program must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI,RCSEC combination of PARAMETER_ERROR_SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. The maximum amount of data that can be received from the partner is 86 bytes.

A Request_To_Send_Received indication is sufficient to complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present, then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on a conversation whose expedited information mode is continue-any, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction. The conversation identifier of the conversation that satisfied the macroinstruction is placed in the CONVID field.
Multiple APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY macroinstructions can be outstanding concurrently. The macroinstruction can be issued when no conversations exist that are in continue-any mode for receiving expedited information. VTAM queues the APPCCMD until one or more conversations are placed in continue-any mode for receiving information and has expedited information available to be received.

**Context**

Input states are not applicable to this macroinstruction. Only expedited information for a conversation that is not in PENDING_DEALLOCATE, END_CONV or FDX_RESET state and whose expedited information mode is continue-any satisfies this type of RCVEXPD.

**Syntax**

```plaintext
APPCCMD — CONTROL — RCVEXPD — QUALIFY — ANY

name

RPL — rpl_address_field
— rpl_address_register

AAREA — rpl_extension_address_field
— rpl_extension_address_register

ACB — acb_address_field
— acb_address_register

AREA — data_area_address_field
— data_area_address_register

AREALEN — data_area_length
— data_area_length_register

BRANCH — NO
— YES

CONMODE — BUFFCA
— CS
— LLCA
— SAME
```

284  z/OS V2R1.0 Communications Server: SNA Programmer's LU 6.2 Reference
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA** = `rpl_extension_address_field`

**AAREA** = `(rpl_extension_address_register)`

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = `acb_address_field`
ACB=(*acb_address_register*)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

AREA=(*data_area_address_field*)
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

AREALEN=(*data_area_length*)
Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.
CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-
any mode. It indicates that APPCCMD CONTROL=RECEIVE,
QUALIFY=ANY | IANY can be used to receive data on this conversation
and that the application program is to receive data in terms of the
logical-record format of the data. LLCA corresponds to FILL=LL on the
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC
macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain
unchanged.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the
APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received by either a specific-type
macroinstruction or an any-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received only by a specific-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD,
QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain
unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:
OPTCD=SYN
Specifications that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVEDCONFIRM_SEND
X'05' RECEIVE_CONFIRM_DEALLOCATE
X'06' PENDING_DEALLOCATE
X'07' PENDING_END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

X'80' FDX_RESET
X'81' SEND/RECEIVE
X'82' SEND_ONLY
X'83' RECEIVE_ONLY
\texttt{\textasciitilde84'}  PENDING\_SEND/RECEIVE\_LOG

\texttt{\textasciitilde85'}  PENDING\_RECEIVE-ONLY\_LOG

\texttt{\textasciitilde86'}  PENDING\_RESET\_LOG

\textbf{CONVID}

Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=ANY. This field is labeled RPL6CNVD in the RPL extension.

\textbf{EXPDLEN}

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

\textbf{EXPDRCV}

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

\textbf{FDB2}

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

\textbf{FMH5LEN}

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

\textbf{FMH5RCV}

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

\textbf{YES (B'1')}

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

\textbf{NO (B'0')}

No FMH-5s are waiting to be received by the application program.

\textbf{RCPRI}

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

\textbf{RCSEC}

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

\textbf{RECLEN}

The field in the RPL that returns to the application program the actual amount
of expedited data the application program received. If the application program receives information other than data, this variable is set to 0. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXPD, QUALIFY=IANY

Purpose

This macroinstruction receives expedited information from any active conversation whose expedited information mode is continue-any. VTAM will not wait for expedited information to arrive on a conversation in continue-any mode to satisfy the macroinstruction.

Usage

This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC for each conversation, the application program can put the conversations in continue-any mode for receiving expedited information and issue a single APPCCMD CONTROL=RCVEXPD, QUALIFY=IANY.

A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on a conversation whose expedited information mode is continue-any, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction. The conversation identifier of the conversation that satisfied the macroinstruction is placed in the CONVID field.

When issued and no conversation exists in a continue-any mode for expedited data or no conversations in continue-any mode have received expedited information, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application program.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'002C', X'0008',
PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application.

**Context**

Input states are not applicable to this macroinstruction. Only expedited information for a conversation that is not in PENDING_DEALLOCATE, END_CONV, or FDX_RESET and whose expedited information mode is continue-any satisfies this type of RCVEXPD.

**Syntax**

```
APPCCMD  name  CONTROL  RCVEXPD  QUALIFY  IANY

RPL  rpl_address_field
     (rpl_address_register)

AAREA  rpl_extension_address_field
       (rpl_extension_address_register)

ACB  acb_address_field
    (acb_address_register)

AREA  data_area_address_field
     (data_area_address_register)

AREALEN  data_area_length
        (data_area_length_register)

BRANCH  YES
        NO

CONMODE  SAME
        LLCA
        CS
        BUFFCA
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:

- **AAREA** = `rpl_extension_address_field`  
  Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- **ACB** = `acb_address_field`
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLARE in the RPL.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.
CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:
**OPTCD=SYN**  
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**  
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**  
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**  
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL**

- **RPL=rpl_address_field**  
  Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

- **RPL=(rpl_address_register)**  
  Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**  
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01'  SEND
- X'02'  RECEIVE
- X'03'  RECEIVE_CONFIRM
- X'04'  RECEIVE_CONFIRM_SEND
- X'05'  RECEIVE_CONFIRM_DEALLOCATE
- X'06'  PENDING_DEALLOCATE
- X'07'  PENDING_END_CONVERSATION_LOG
- X'08'  END_CONVERSATION
- X'09'  PENDING_SEND
- X'0A'  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
CONVID
Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=ANY | IANY. This field is labeled RPL6CNVD in the RPL extension.

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B’1’)
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B’0’)
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLLEN
The field in the RPL that returns to the application program the actual amount of expedited data the application program received up to the maximum. If the
application program receives information other than data, this variable is set to 0. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B’1’)**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B’0’)**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
No state changes are associated with this macroinstruction.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0000’</td>
<td>X’0008’</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0008’</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVEXPD, QUALIFY=ISPEC

Purpose

This macroinstruction receives expedited information immediately available on a specified conversation. VTAM will not wait for expedited information to arrive to satisfy the macroinstruction request.

Usage

A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. The conversation mode (CONXMOD) for expedited data may be either CA or CS. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on the conversation, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction.

If expedited information is not available, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application.

If this macroinstruction is issued while another RCVEXPD macroinstruction is currently outstanding for the specified conversation, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application. The maximum amount of expedited data that can be received is 86 bytes.

If the RECEIVE EXPEDITED queue has been prohibited, then an RCPRI, RCSEC combination of X'00A0', X'0002', REQUEST_NOT_ALLOWED—
REQUEST_BLOCKED is returned to the application. The RECEIVE EXPEDITED queue is prohibited when the conversation is in the process of being deallocated or terminated.

If the macroinstruction is issued for a conversation in PENDING DEALLOCATE state, an RCPRI, RCSEC combination of X'0050', X'0000', STATE_ERROR is returned to the conversation.

If the conversation ends before this macroinstruction can process, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST TERMINATED BY END OF CONVERSATION is returned to the application.

This macroinstruction corresponds to the RECEIVE EXPEDITED_DATA (IMMEDIATE) verb described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from any conversation state except PENDING DEALLOCATE, END_CONV, or FDX_RESET.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL=RCVEXPD QUALIFY=ISPEC

RPL=rpl_address_field (rpl_address_register)

AAREA=rpl_extension_address_field (rpl_extension_address_register)

ACB=acb_address_field (acb_address_register)

AREA=data_area_address_field (data_area_address_register)

AREALEN=data_area_length (data_area_length_register)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:

- **AAREA=rpl_extension_address_field**
  - Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- **ACB=acb_address_field**
  - Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDxCB in the RPL.

- **AREA=data_area_address_field**
  - Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

- **AREALEN=data_area_length**
  - Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

- **BRANCH**
  - Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

  - **BRANCH=NO**
    - Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

  - **BRANCH=YES**
    - Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

- **CONMODE**
  - Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

  - **CONMODE=BUFFCA**
    - Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY I IANY can be used to receive data and that the application program is to receive
data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCMD macroinstruction completes.
**ECB=** `ecb_address_field`

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. `Event_control_block_address` is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=** `exit_routine_address_field`

Valid only if `OPTCD=ASY`. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=** `rpl_address_field`

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X’01’ Send
- X’02’ Receive
- X’03’ Receive_Confirm
- X’04’ Receive_Confirm_Send
X'05' RECV_CONFIRM_DEALLOCATE
X'06' PENDING_DEALLOCATE
X'07' PENDING_END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:
X'80' FDX_RESET
X'81' SEND/RECEIVE
X'82' SEND_ONLY
X'83' RECEIVE_ONLY
X'84' PENDING_SEND/RECEIVE_LOG
X'85' PENDING_RECEIVE-ONLY_LOG
X'86' PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received. The field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES** (B'1')

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO** (B'0')

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual amount of expedited data the application program received. The value returned will always be less than or equal to the value specified for AREALEN. This value will be set to 0 if the macroinstruction is being completed because of a REQUEST_TO_SEND being received.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMI5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
There are no state changes caused by the execution of this macroinstruction.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0009'</td>
<td>X'002C'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'0009'</td>
<td>X'002C'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'000F'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'0010'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'0011'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'0032'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—UNEXPECTED VECTOR PROVIDED ON APPCCMD</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0050'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0070'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0078'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'007C'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0090'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'00A0'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'00A8'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT_</td>
</tr>
<tr>
<td>X'0001'</td>
<td>X'00A8'</td>
<td>Suspend FAILURE</td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'00A8'</td>
<td>RESUME FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC

Purpose

This macroinstruction receives expedited information from the specified conversation. VTAM will wait for expedited information to arrive to satisfy the macroinstruction request. If expedited information is immediately available, then the application receives it without waiting. The expedited information mode may be continue-any or continue-specific.

Usage

A Request_To_Send_Received indication is sufficient to successfully complete this macroinstruction. If a Request_To_Send_Received indication and expedited data are present then both will be returned to the application. The settings of the SIGRCV and SIGDATA returned parameter fields will indicate whether a Request_To_Send_Received indication (Signal Data) was received on the conversation. When expedited data is available on the conversation, VTAM copies the data into the data area that is specified on the AREA parameter and completes the macroinstruction.

The application must receive the entire amount of expedited data available. If the length of the area specified by the application program is not sufficient to receive all the expedited data available, an RCPRI, RCSEC combination of X'002C', X'0008',
PARAMETER_ERROR_SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. The maximum amount of expedited data that can be received is 86 bytes.

If this macroinstruction is issued while another RCVEXPD macroinstruction is currently outstanding for the specified conversation, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application.

If the RECEIVE EXPEDITED queue has been prohibited, then an RCPRI, RCSEC combination of X'00A0', X'0002', REQUEST_NOT_ALLOWED—REQUEST_BLOCKED is returned to the application. The RECEIVE EXPEDITED queue is prohibited when the conversation is in the process of being allocated or terminated.

If the macroinstruction is issued for a half-duplex conversation and a negative response is received from the partner, then an RCPRI, RCSEC combination of X'00A0', X'0003', REQUEST_NOT_ALLOWED—EXECUTION_OF_REQUEST_TERMINATED will be returned to the application.

If the macroinstruction is issued for a conversation in PENDING_DEALLOCATE state, an RCPRI, RCSEC combination of X'0050', X'0000', STATE_ERROR is returned to the application.

If the conversation is terminated before expedited information is received, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.

This macroinstruction corresponds to the RECEIVE_EXPEDITED_DATA (WHEN_EXPEDITED_DATA_RECEIVED) verb described in the LU 6.2 architecture.

**Context**

This macroinstruction can be issued from any conversation state except PENDING_DEALLOCATE, END_CONV, or FDX_RESET.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
Chapter 1. LU 6.2 macroinstruction syntax and operands
Notes:

1  Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2  See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3  Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4  ECB is meaningful only for asynchronous operations.

5  EXIT is meaningful only for asynchronous operations.

6  You can code more than one suboperand on OPTCD, but no more than one from each group.

7  KEEPSRB is meaningful only for synchronous operations.

8  NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction
programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=** data_area_address_field  
**AREA=(** data_area_address_register **)**  
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

**AREALEN=** data_area_length  
**AREALEN=(** data_area_length_register **)**  
Specifies the length value that is the maximum amount of data the application program is to receive. This field is labeled RPLBUFL in the RPL.

**BRANCH**  
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**  
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**  
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**  
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**  
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**  
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**  
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the
logical-record format of the data. LLCA corresponds to FILL=LL on the
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC
macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain
unchanged.

CONVID
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD
in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the
APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received by either a specific-type
macroinstruction or an any-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received only by a specific-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD,
QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain
unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction
The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'  SEND
X'02'  RECEIVE
X'03'  RECEIVE_CONFIRM
X'04'  RECEIVE_CONFIRM_SEND
X'05'  RECEIVE_CONFIRM_DEALLOCATE
X'06'  PENDING_DEALLOCATE
X'07'  PENDING_END_CONVERSATION_LOG
X'08'  END_CONVERSATION
X'09'  PENDING_SEND
X'0A'  PENDING_RECEIVE_LOG

For full duplex conversations, this field can have the following values:

X'80'  FDX_RESET
X'81'  SEND/RECEIVE
X'82'  SEND_ONLY
X'83'  RECEIVE_ONLY
X'84'  PENDING_SEND/RECEIVE_LOG
X'85'  PENDING_RECEIVE-ONLY_LOG
X'86'  PENDING_RESET_LOG

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the actual amount of expedited data the application program received. The value returned will always be less than or equal to the value specified for AREALEN. This value is set to 0 if the macroinstruction completes only due to receipt of a REQUEST_TO_SEND indication.
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

There are no state changes caused by the execution of this macroinstruction.

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0000’</td>
<td>X’0009’</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0008’</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0010’</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE

Purpose

This macroinstruction receives an FMH-5, which begins the application program's participation in a conversation.

This macroinstruction allows the application to specify how expedited information is received.

Usage

When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

If this macroinstruction is issued before an FMH-5 is received, VTAM waits until the FMH-5 is received to complete the macroinstruction. When an FMH-5 is received, VTAM bypasses the ATTN exit. If VTAM receives the FMH-5 before this macroinstruction is issued, VTAM schedules the ATTN exit. In either case, VTAM then moves the FMH-5 to the application's buffer and returns the CONVID and other return parameters.

After performing the operation of the RCVFMH5, VTAM examines the setting of the FILL parameter. If FILL=LL has been specified, this macroinstruction performs the functions of an APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC with a FILL=LL. That is, VTAM receives a single logical record. This would be the first logical record after the FMH-5. However, if FILL=BUFF has been specified, this macroinstruction performs the functions of an APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC with a FILL=BUFF. If there is insufficient information to complete the receive, the macroinstruction is suspended until more information is received from the partner. As is normally done for a receive macroinstruction, if any of the following conditions occurs, the APPCCMD CONTROL=RCVFMH5, QUALIFY=DATAQUE completes:

- The local receive buffer is completely filled.
- A complete logical record is received AND FILL=LL was specified.
- A SEND indication is received.
A CONFIRM indication is received.
A DEALLOCATE indication is received.
An ERROR condition is detected.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5.

**Context**

This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

**Syntax**

```
APPCCMD name
  CONTROL = RCVFMH5, QUALIFY = DATAQUE

  (1) RPL = rpl_address_field
       (rpl_address_register)

  AAREA = rpl_extension_address_field
       (rpl_extension_address_register)

  (2) ACB = acb_address_field
       (acb_address_register)

  AREA = data_area_address_field
       (data_area_address_register)

  AREALEN = data_area_length
       (data_area_length_register)

  (2) BRANCH = NO
       YES

  (3) CD = DEFER
       IMMED
```
Notes:

1. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the FMH-5 and any associated data. This field is labeled RPLAREA in the RPL.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the size of the supplied buffer area. An FMH-5 is, at most, 255 bytes in length. Because the application cannot determine the length of the FMH-5 when the RCVFMH5 request is queued, VTAM fails this macroinstruction if the length of AREALEN is less than 255 with an RCPR, RCSEC combination of X’002C’, X’0008’. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CD
Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data. This parameter is valid only for half-duplex conversations.

CD=DEFER
Specifies that the conversation state will be in PENDSEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.
CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ebc_address_field
ECB=(ebc_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RTSRTRN

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

RTSRTRN=BOTH

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

RTSRTRN=EXPD

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

USERFLD=4_bytes_of_user_data
USERFLD=user_data_register

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

• VTRINA=0
• The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CGID

Specifies the 32-bit conversation group identifier.
It is labeled RPL6CGID in the RPL extension.

**CONVID**
The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONSTATE**
The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST.

For half-duplex conversations, this field can have the following values:
- X'00'  RESET
- X'01'  SEND
- X'02'  RECEIVE
- X'03'  RECEIVE_CONFIRM
- X'04'  RECEIVE_CONFIRM_SEND
- X'05'  RECEIVE_CONFIRM_DEALLOC
- X'07'  PENDING_END_CONVERSATION_LOG
- X'08'  END_CONVERSATION
- X'09'  PENDING_SEND
- X'0A'  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
- X'84'  PENDING_SEND/RECEIVE_LOG
- X'85'  PENDING_RECEIVE_ONLY_LOG
- X'86'  PENDING_RESET_LOG

**CRYPTLVL**
Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  No data is to be encrypted.

- **SELECTIVE (B'01')**
  The application program specifies the data that is to be encrypted.

- **REQUIRED (B'11')**
  All data is to be encrypted.

**EXPDLEN**
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No other FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR

X'0014'
  DEALLOCATE_ABEND_PROGRAM

X'0018'
  DEALLOCATE_ABEND_SERVICE

X'001C'
  DEALLOCATE_ABEND_TIMER

X'0030'
  PROGRAM_ERROR_NO_TRUNC

X'0034'
  PROGRAM_ERROR_PURGING
NO (B'0')

Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**LOGMODE**

The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE.

**LUNAME**

The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

**NETID**

The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**

The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLLEN in the RPL. If the RCPRI field equals X'0000', (OK), RECLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRRTNCD in the RPL.
SENSE
The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

SESSID
The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in the z/OS Communications Server: SNA Programmer's LU 6.2 Guide. This field is labeled RPL6SSID in the RPL extension.

SESSIDL
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. It is labeled RPL6RSIG. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B’1’)
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B’0’)
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

SLS
The field in the RPL extension that indicates whether or not the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B’1’)
The session was established using session-level LU-LU verification.
NO (B'0')

The session was not established using session-level LU-LU verification.

WHATRCV

The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DATA</td>
<td>0</td>
<td>PARTIAL_PS_HEADER</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
<td>1-7</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
<td>6</td>
<td>LOG_DATA</td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

Vectors returned

VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X'10')
- Partner’s DCE capabilities vector (X’12’)
- Local nonce vector (X’13’)
- Partner’s nonce vector (X’14’)
- PCID vector (X’17’)
- Session information vector (X’19’)
- Partner’s application capabilities vector (X’1A’)

State changes

These changes are applicable when RCPRI indicates OK.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPC CMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.
APPCCMD control=RCVFMH5, QUALIFY=NULL

Purpose

This macroinstruction receives an FMH-5, which begins the application program’s participation in a conversation.

This macroinstruction allows the application to specify how expedited information is received.
Usage

When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5. If so, the application program should issue APPCCMD CONTROL=RECEIVE to receive the GDS data.

If no FMH-5 is available for the application to receive, this macroinstruction is rejected with an RCPRI return code of X'0060'.

For information on how the application program is informed that an FMH-5 is ready to be received, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

Syntax
Notes:
1 See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.
Input parameters

The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl\_extension\_address\_field}

\texttt{AAREA=()}\texttt{rpl\_extension\_address\_register}

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}

\texttt{ACB=()}\texttt{acb\_address\_register}

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=\texttt{data\_area\_address\_field}

\texttt{AREA=()}\texttt{data\_area\_address\_register}

Specifies the data area in which the application program is to receive the FMH-5. This field is labeled RPLAREA in the RPL.

AREALEN=\texttt{data\_area\_length}

\texttt{AREALEN=()}\texttt{data\_area\_length\_register}

 Specifies the size of the supplied buffer area. The supplied buffer area must be large enough to contain the entire FMH-5. An FMH-5 is at most 255 bytes in length (it has only 1 byte for a length count). If a 255-byte buffer is used to receive the FMH-5, the RCVFMH5 macroinstruction will never fail for lack of buffer space. This field is labeled RPLBUFL in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

\texttt{BRANCH=NO}

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

\texttt{BRANCH=YES}

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE

Specifies the mode for receiving normal information on completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

\texttt{CONMODE=BUFFCA}

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\mid IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA
corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

 Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

 Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

 Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

 Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**

 Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**

 Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4_bytes_of_user_data**

**USERFLD=(user_data_register)**

 Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**
**VTRINA** = *(vector_address_register)*

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA = 0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL** = *(vector_length_field)*

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CGID**

Specifies the 32-bit conversation group identifier.

It is labeled RPL6CGID in the RPL extension.

**CONSTATE**

The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'00'  RESET
- X'02'  RECEIVE
- X'08'  END_CONVERSATION

For full-duplex conversations, this field can have the following values:

- X'00'  RESET
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE

**CONVID**

The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.

**CRYPTLVL**

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

- **NONE (B'00')**
  No data is to be encrypted.

- **SELECTIVE (B'01')**
  The application program specifies the data that is to be encrypted.

- **REQUIRED (B'11')**
  All data is to be encrypted.
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No other FMH-5s are waiting to be received by the application program.

LOGMODE
The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE in the RPL extension.

LUNAME
The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID
The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RECLLEN**
The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLEN in the RPL. If the RCPRI field equals X'0000', (OK), RECLLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program. If the (RCPRI, RCSEC) fields equal X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT, it indicates the size of the FMH-5. However, in the latter case, because the supplied buffer was not large enough to contain the entire FMH-5, the FMH-5 is not returned to the application program. The application program is informed, through the FMH5LEN, of how large the buffer must be in order to receive the FMH-5.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI in the RPL extension. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

**SESSID**
The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.6/com.ibm.zos.v2r11/iosghs00.htm). This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**
The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**
The field in the RPL extension that indicates whether or not the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

**Vectors returned**
VTAM may return the following vectors in the area supplied by the VTRINA parameter:
- VTAM-to-APPL required information vector (X’10’)
- Partner’s DCE capabilities vector (X’12’)
- Local nonce vector (X’13’)
- Partner’s nonce vector (X’14’)
- PCID vector (X’17’)
- Session information vector (X’19’)
- Partner’s application capabilities vector (X’1A’)
State changes

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is RECEIVE after successful processing.

For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0060'</td>
<td>X'0000'</td>
<td>NO_FMH5_AVAILABLE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=RCVFMH5, QUALIFY=QUEUE

Purpose

This macroinstruction receives an FMH-5, which begins the application program’s participation in a conversation.

This macroinstruction allows the application to specify how expedited information is received.
Usage

When this macroinstruction is issued, VTAM copies the FMH-5, which represents a new conversation, into the area specified on the AREA parameter. When the macroinstruction completes, the new conversation identifier can be found in the CONVID field. The new conversation will be in RECEIVE state for half-duplex conversations and in SEND/RECEIVE state for full-duplex conversations.

If this macroinstruction is issued before an FMH-5 is received, VTAM waits for the FMH-5 to complete the macroinstruction. When an FMH-5 is received, VTAM bypasses the ATTN exit. If VTAM receives the FMH-5 before this macroinstruction is issued, VTAM schedules the ATTN exit. In either case, VTAM then moves the FMH-5 to the application's buffer and returns the CONVID and other return parameters. VTAM retains any data that accompanies the FMH-5.

The application program can use the FMH-5 to perform conversation level security processing. Also, the FMH-5 indicates whether any GDS fields, such as DCE security or program initialization (PIP) data, follows the FMH-5. If so, the application program should issue APPCCMD CONTROL=RECEIVE to receive the PIP data.

For information on how the application program is informed that an FMH-5 is ready to be received, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

This macroinstruction can be issued from the RESET conversation state.

This macroinstruction is not mode-specific and might be issued for a mode that is retained for persistent LU-LU sessions. However, an FMH-5 is not returned for a mode that is being retained for persistent LU-LU sessions when this macroinstruction is issued.

Syntax
Notes:

1. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.
Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
    Specifies the address of the LU 6.2 RPL extension that will be associated with
    this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
    Specifies the address of an access method control block that identifies the
    application program that is issuing the APPCCMD macroinstruction. VTAM
    associates conversations with application programs using the conversation ID
    (CONVID). The application program associates conversations with transaction
    programs. Application programs cannot issue APPCCMD macroinstructions in
    address spaces other than the ACB address space. This field is labeled
    RPLDACB in the RPL.

AREA=data_area_address_field
AREA=(data_area_address_register)
    Specifies the data area in which the application program is to receive the
    FMH-5. This field is labeled RPLAREA in the RPL.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
    Specifies the size of the supplied buffer area. An FMH-5 is, at most, 255 bytes
    in length. Because the application cannot determine the length of the FMH-5
    when the RCVFHM5 request is queued, VTAM fails this macroinstruction if
    the length of AREALEN is less than 255 with an RCPRI, RCSEC combination
    of X'002C', X'0008'. This field is labeled RPLBUFL in the RPL.

BRANCH
    Specifies whether authorized path processing is to be used for application
    programs running in supervisor state under a TCB. Application programs
    running in TCB-mode supervisor state can use BRANCH=YES to obtain
    authorized path services. The indicator resides within the RPLEXTDS field of
    the RPL.

BRANCH=NO
    Authorized path processing is not to be used. For application programs
    running in problem state (non-supervisor state) under a TCB,
    BRANCH=NO is the only option.

BRANCH=YES
    Authorized path processing is to be used. For application programs
    running under an SRB rather than under a TCB, the macroinstruction is
    processed in this manner automatically, regardless of the actual setting of
    the BRANCH field.

CONMODE
    Specifies the mode for receiving normal information on completion of the
    APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
    Specifies that the conversation is to be placed in buffer-continue-any mode.
    It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY
    can be used to receive data and that the application program is to receive
    data independently of the logical-record format of the data. BUFFCA
The CONMODE=CS specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

CONMODE=LLCA specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONXMOD specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL specifies that VTAM is to post an internal ECB when the APPPCMD macroinstruction completes.

ECB= ECB=specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled.
when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RTSRTRN**

Specifies the manner in which the Request_To_Send_Received indication is to be reported to the application on subsequent macroinstructions.

**RTSRTRN=BOTH**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on all APPCCMDs that return these parameters.

**RTSRTRN=EXPD**

Specifies that the Request_To_Send_Received indication can be reported in the SIGRCV and SIGDATA fields on an APPCCMD CONTROL=SENDEXPD or an APPCCMD CONTROL=RCVEXPD.

**USERFLD=4_bytes_of_user_data**

Specifies 4 bytes of user data to be associated with the new conversation. Whenever an APPCCMD macroinstruction completes for this conversation, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**VTRINA=vector_address_field**
**VTRINA**=(vector_address_register)

Specifies the address of the data area where VTAM places vector list information for the application.

This parameter is ignored if one of the following items is true:

- VTRINA=0
- The value for VTRINL is less than the minimum length required to return the APPCCMD vector area header.
- The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

**VTRINL=vector_length_field**

Specifies the length of the data area where VTAM places vector list information for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This field is labeled RPL6VAIL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CGID**

Specifies the 32-bit conversation group identifier. It is labeled RPL6CGID in the RPL extension.

**CONVID**

The field in the RPL extension that returns the resource identifier of the new conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONSTATE**

The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST.

For half-duplex conversations, this field can have the following values:

- X'00'  RESET
- X'02'  RECEIVE
- X'08'  END_CONVERSATION

For full-duplex conversations, this field can have the following values:

- X'00'  RESET
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE

**CRYPTLVL**

Indicates the encryption level for the conversation. This field is labeled RPL6CRYP in the RPL extension.

**NONE (B'00')**

No data is to be encrypted.

**SELECTIVE (B'01')**

The application program specifies the data that is to be encrypted.

**REQUIRED (B'11')**

All data is to be encrypted.
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether another FMH-5, other than the one currently being passed to the application program on this APPCCMD, has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No other FMH-5s are waiting to be received by the application program.

LOGMODE
The field in the RPL extension that returns the logon mode name of the session over which the FMH-5 is being returned on this APPCCMD macroinstruction. It is an 8-byte name, padded on the right with blanks. It is labeled RPL6MODE.

LUNAME
The field in the RPL extension that returns the name of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

NETID
The field in the RPL extension that returns the network identifier of the partner LU that sent the FMH-5 being returned on this APPCCMD macroinstruction. This network identifier is the identifier of the partner LU. It can be up to 8 characters in length. If it is fewer than 8 characters, VTAM pads it on the right with blanks. This field is labeled RPL6NET in the RPL extension.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RECLLEN**

The field in the RPL that returns to the application the size, in bytes, of the FMH-5. This field is labeled RPLRLEN in the RPL. If the RCPRI field equals X'0000' (OK), RECLLEN specifies the number of bytes of the supplied AREA field that were used to return the FMH-5 to the application program.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 4-byte sense code. This sense code has meaning if the RCPRI return code indicates a resource failure problem. It is labeled RPL6SNSI. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated.

**SESSID**

The field in the RPL extension that, when SESSIDL is not equal to 0, returns a session instance identifier for the session over which the FMH-5 was received. The format of the session instance identifier is described in the [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](https://www.ibm.com). This field is labeled RPL6SSID in the RPL extension.

**SESSIDL**

The field in the RPL extension that returns the length of the session instance identifier, which is itself returned in the SESSID field. Values in the range 0–8 are valid. This field is labeled RPL6SIDL in the RPL extension.

**SLS**

The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

**YES (B'1')**

The session was established using session-level LU-LU verification.

**NO (B'0')**

The session was not established using session-level LU-LU verification.

**Vectors returned**

VTAM may return the following vectors in the area supplied by the VTRINA parameter:

- VTAM-to-APPL required information vector (X'10')
- Partner's DCE capabilities vector (X'12')
- Local nonce vector (X'13')
- Partner's nonce vector (X'14')
- PCID vector (X'17')
- Session information vector (X'19')
- Partner's application capabilities vector (X'1A')

**State changes**

These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is RECEIVE after successful processing.
For full-duplex conversations, the conversation state is SEND/RECEIVE after successful processing.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>VECTOR AREA NOT VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002F'</td>
<td>VECTOR AREA LENGTH INSUFFICIENT</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>MODE_MUST_BE_RESTORED_BEFORE_USING</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORTED_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RECEIVE, QUALIFY=ANY**

**Purpose**

This macroinstruction receives normal information on a conversation that is in continue-any mode. Unlike other macroinstructions that are used to receive data, the application program does not specify a partner conversation. Instead, the macroinstruction is associated with the first conversation that is in continue-any mode and that receives data.

**Usage**

This macroinstruction can be used when the application program is maintaining multiple asynchronous conversations. Instead of issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC for each conversation, the application program can put the conversations in continue-any mode and issue a single APPCCMD CONTROL=RECEIVE, QUALIFY=ANY.

When VTAM receives data on a continue-any mode conversation, VTAM copies the data into the data area that is specified on the AREA parameter and if sufficient data has been received, then VTAM completes the macroinstruction. The
conversation identifier of the conversation that is used to complete the macroinstruction is placed in the CONVID field.

This macroinstruction can be used to receive application program data, conversation status information, and confirmation requests. However, it cannot be used to receive error log information. The application program must use APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to receive error log information.

If VTAM receives notification that a conversation fails on a continue-any mode, this macroinstruction completes with a nonzero return code.

Multiple APPCCMD CONTROL=RECEIVE, QUALIFY=ANY macroinstructions can be outstanding concurrently. The order in which these macroinstructions complete is not necessarily the order in which they were issued. This means that if a conversation is left in continue-any mode, data from multiple RECEIVEs could arrive out of order. If the application program cannot detect this and process the data properly, the application program should specify CONMODE=CS on the APPCCMD CONTROL=RECEIVE, QUALIFY=ANY macroinstruction. For more information on specifying CONMODE, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

APPCCMD CONTROL=RECEIVE, QUALIFY=ANY can be issued when no conversations exist that are in continue-any mode and in RECEIVE, SEND/RECEIVE, or RECEIVE-ONLY state. The APPCCMD waits until one or more conversations are placed into continue-any mode and the right state.

An any-mode RECEIVE can lock out a specific-mode RECEIVE. For example, if an application program has issued an any-mode RECEIVE that receives data in terms of buffers, and enough data has not arrived to satisfy the buffer length, VTAM waits until enough data arrives to satisfy the buffer length before honoring a specific mode RECEIVE for the conversation.

This macroinstruction does not directly correspond to any architected verb described in the LU 6.2 architecture.

**Context**

Input states are not applicable to this macroinstruction. Only information for a conversation in RECEIVE, SEND/RECEIVE, or RECEIVE_ONLY state and continue-any mode satisfies this type of RECEIVE.

**Syntax**
APPCCMD CONTROL RECEIVED, QUALIFY ANY

RPL = rpl_address_field (rpl_address_register)

AAREA = rpl_extension_address_field (rpl_extension_address_register)

ACB = acb_address_field (acb_address_register)

AREA = data_area_address_field (data_area_address_register)

AREALEN = data_area_length (data_area_length_register)

BRANCH = NO YES, CD = DEFER IMMED

CONMODE = BUFFCA CS LLCA SAME

CONXMOD = CA CS LLCA SAME

ECB = INTERNAL ecb_address_field (ecb_address_register)

EXIT = exit_routine_address_field (exit_routine_address_register)
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEP SRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA** = rpl_extension_address_field

**AAREA** = (rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = acb_address_field

**ACB** = (acb_address_register)

Specifies the address of an access method control block that identifies the
application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=**

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

**AREALEN=**

Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PENDING_SEND when a change of direction is received with no data.
**CD=DEFER**
Specifies that the conversation state will be in PENDING_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC\|ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC\|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY\|IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.
**CONXMOD=SAME**

Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=** *ecb_address_field*

**ECB=(ecb_address_register)**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event control block address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=** *exit_routine_address_field*

**EXIT=(exit_routine_address_register)**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NBUFFLST**

Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

**OPTCD=XBUFLST**

Specifies that the HPDT interface is to be used. VTAM builds an extended
buffer list in the address specified by the AREA parameter. Each entry in
the buffer list points to a CSM buffer containing the data being received by
the application. The indicator is labeled RPLXBFL and resides within the
RPLOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must
specify BRANCH=YES for HPDT requests.

RPL=\textit{rpl\_address\_field}
\textbf{RPL}=(\textit{rpl\_address\_register})
Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

\textbf{VTROUTA}=vector\_address\_field
\textbf{VTROUTA}=(vector\_address\_register)
Specifies the address of the area where the application places vector list
information for VTAM. If OPTCD=XBUFLST is specified, this field must point
to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL.
(Refer to \textit{z/OS Communications Server: SNA Programmer's LU 6.2 Guide} for
more information.)

This field is labeled RPL6VAOA in the RPL extension.

\textbf{VTROUTL}=vector\_length\_field
\textbf{VTROUTL}=(vector\_length\_register)
Specifies the length of the area where the application places vector list
information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

\textbf{RPL and RPL extension fields modified by macroinstruction}

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}
The field in the RPL6 extension that indicates the state of the conversation.
This field is labeled RPL6CCST in the RPL extension. For half-duplex
conversations, this field can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE\_CONFIRM
X'04' RECEIVE\_CONFIRM\_SEND
X'05' RECEIVE\_CONFIRM\_DEALLOCATE
X'07' PENDING\_END\_CONVERSATION\_LOG
X'08' END\_CONVERSATION
X'09' PENDING\_SEND
X'0A' PENDING\_RECEIVE\_LOG

For full-duplex conversations, this field can have the following values:

X'80' FDX\_RESET
X'81' SEND/RECEIVE
X'82' SEND\_ONLY
X'83' RECEIVE\_ONLY
X'84' PENDING\_SEND/RECEIVE\_LOG
CONVID
Specifies the resource identifier of the conversation on which information was received. This field is labeled RPL6CNVD in the RPL extension.

EXPDLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.
LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:
X'0004'
    ALLOCATION_ERROR
X'0014'
    DEALLOCATE_ABEND_PROGRAM
X'0018'
    DEALLOCATE_ABEND_SERVICE
X'001C'
    DEALLOCATE_ABEND_TIMER
X'0030'
    PROGRAM_ERROR_NO_TRUNC
X'0034'
    PROGRAM_ERROR_PURGING
X'0038'
    PROGRAM_ERROR_TRUNC
X'003C'
    SERVICE_ERROR_NO_TRUNC
X'0040'
    SERVICE_ERROR_PURGING
X'0044'
    SERVICE_ERROR_TRUNC
X'005C'
    USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received
but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when
RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL
extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary
return code is returned to the application program. This field has meaning
only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in
the RPL extension. The combination of the RCPRI and RCSEC fields indicates
the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual amount
of data the application program received up to the maximum. If the
application program receives information other than data, this variable is set to
0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the
extended buffer list that is built in the buffer list area pointed to by the AREA
operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned
to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled
RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field
is set to a nonzero value. The sense code also can be set when the return code
is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for
the conversation was deactivated. Not all RCPRI values have sense data
associated with them. If the RCPRI field indicates
USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense
code that was not recognized by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension
fields of a received SIGNAL RU are returned to the application program. This
field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in
the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received
from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated
the conversation (APPCCMD CONTROL=ALLOC or APPCCMD
CONTROL=RCVFMH5), the application specified RSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an
application program's partner has requested permission to send. This field and
the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED
parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated
the conversation (APPCCMD CONTROL=ALLOC or APPCCMD
CONTROL=RCVFMH5), the application specified RSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled
RPL6RSIG in the RPL extension.

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried
in the signal code and signal extension fields of the SIGNAL RU are
returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When
SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD
Specifies 4 bytes of user data that the application program requests be
associated with a conversation. Whenever an APPCCMD completes, VTAM
places in the USERFLD field of the RPL extension the 4 bytes that were
supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the
conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a
remote application program). This field is labeled RPL6USR in the RPL
extension.

WHATRCV
The field in the RPL extension that returns a mask specifying what the
application program received. It is labeled RPL6WHAT in the RPL extension.
The application program should examine this WHATRCV mask only when
RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.
When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B’1’) to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format:

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>Meaning</th>
<th>RPL6RCV2</th>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DATA</td>
<td></td>
<td>0</td>
<td>PARTIAL_PS_HEADER</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
<td></td>
<td>1–7</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X’8000’.

Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field. However, LOG_DATA cannot be set on this macroinstruction.

State changes

See the description of the WHATRCV mask for a description of the state changes that occur when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0002’</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0004’</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0005’</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0007’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0008’</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0009’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000A’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000B’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000D’</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X’0014’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X’0018’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X’001C’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0008’</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RECEIVE, QUALIFY=IANY

Purpose

This macroinstruction receives normal information that is immediately available on a conversation that is in continue-any mode. VTAM does not wait for data to be received before completing this macroinstruction.

Usage

When this macroinstruction is issued, VTAM copies all data that is immediately available into the supplied data area or control block that is specified by the AREA parameter. VTAM also returns the identification of the conversation that satisfied the macroinstruction in the CONVID parameter.

This macroinstruction can be used to receive application program data, conversation status information, and confirmation requests. However, it cannot be used to receive error log information. The application program must use APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to receive error log information.
This macroinstruction does not directly correspond to any architected verb described in the LU 6.2 architecture.

If no data is immediately available, an RCPRI,RCSEC of (X’0000’, X’0008’) NO_INFORMATION_IMMEDIATELY_AVAILABLE is returned to the application.

Context

Input states are not applicable to this macroinstruction. Only data for a conversation in RECEIVE, SEND/RECEIVE, or RECEIVE ONLY state and continue-any mode satisfies this type of RECEIVE.

Syntax

(1) APPCCMD — CONTROL = RECEIVE, QUALIFY = IANY

(2) RPL = rpl_address_field (rpl_address_register)

(3) AAREA = rpl_extension_address_field (rpl_extension_address_register)

(3) ACB = acb_address_field (acb_address_register)

(3) AREA = data_area_address_field (data_area_address_register)

(3) AREALEN = data_area_length (data_area_length_register)

(3) BRANCH = NO YES

(1) CD = DEFER IMMED
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.
You can code more than one suboperand on OPTCD, but no more than one from each group.

KEEPSRB is meaningful only for synchronous operations.

NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:

- **AAREA** = `rpl_extension_address_field`
  - Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- **ACB** = `acb_address_field`
  - Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

- **AREA** = `data_area_address_field`
  - Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

  When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

- **AREALEN** = `data_area_length`
  - Specifies the length value that is the maximum amount of data the application program is to receive.

  If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

  This field is labeled RPLBUFL in the RPL.

- **BRANCH**
  - Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PENDING_SEND when a change of direction is received with no data.

**CD=DEFER**

Specifies that the conversation state will be in PENDING_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**

Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.
CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NBUFFLST
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05' RECEIVED_CONFIRM_DEALLOCATE
X'07' PENDING_END_CONVERSATION_LOG
X'08' END_CONVERSATION
X'09' PENDING_SEND
X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:
X'80' FDX_RESET
X'81' SEND/RECEIVE
X'82' SEND ONLY
X'83' RECEIVE ONLY
X'84' PENDING_SEND/RECEIVE_LOG
X'85' PENDING_RECEIVE-ONLY_LOG
X'86' PENDING_RESET_LOG

CONVID
Specifies the resource identifier of the conversation on which information was received. A value is placed in this field by VTAM only if QUALIFY=*ANY. This field is labeled RPL6CNVD in the RPL extension.

EXPDLLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.
LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR
X'0014'
  DEALLOCATE_ABEND_PROGRAM
X'0018'
  DEALLOCATE_ABEND_SERVICE
X'001C'
  DEALLOCATE_ABEND_TIMER
X'0030'
  PROGRAM_ERROR_NO_TRUNC
X'0034'
  PROGRAM_ERROR_PURGING
X'0038'
  PROGRAM_ERROR_TRUNC
X'003C'
  SERVICE_ERROR_NO_TRUNC
X'0040'
  SERVICE_ERROR_PURGING
X'0044'
  SERVICE_ERROR_TRUNC
X'005C'
  USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning
only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual amount of data the application program received up to the maximum. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBULST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not recognized by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B'0')**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.
USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV

The field in the RPL extension that indicates what the application program received. It is labeled RPL6WHAT in the RPL extension. The application program should examine the WHATRCV field only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format:

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a discussion of the meaning of this field. However, LOG_DATA cannot be set on this macroinstruction.

State changes

See the description of the WHATRCV mask for a description of the state changes that occur when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.
<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_CORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=RECEIVE, QUALIFY=ISPEC

Purpose

This macroinstruction receives normal information that is immediately available from a specified conversation. The conversation may be in continue-any or continue-specific mode. VTAM does not wait for more data to be received before completing this macroinstruction.

Usage

When this macroinstruction is issued, VTAM copies all data that is immediately available into the supplied data area or control block that is specified by the AREA parameter. The AREALEN parameter specifies the length of the data area. VTAM does not wait to receive any more data before completing the macroinstruction request. If there is no information available, VTAM issues an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION.

When this macroinstruction completes, the RECLEN field indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

If VTAM is processing APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\|IANY for a conversation and the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=ISPEC for the same conversation, the QUALIFY=ISPEC request fails with an RCPRI, RCSEC combination of X'0000', X'0003', RECEIVE_SPECIFIC_REJECTED. VTAM cannot allow a specific-mode RECEIVE while an any-mode RECEIVE is being processed.

This macroinstruction corresponds to the RECEIVE_IMMEDIATE verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- RECEIVE
- PEND_END_CONV_LOG
- PEND_RCV_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
APPCCMD CONTROL = RECEIVE, QUALIFY = ISPEC

RPL = rpl_address_field
  (rpl_address_register)

AAREA = rpl_extension_address_field
  (rpl_extension_address_register)

ACB = acb_address_field
  (acb_address_register)

AREA = data_area_address_field
  (data_area_address_register)

AREALEN = data_area_length
  (data_area_length_register)

BRANCH = NO
  YES

CD = DEFER
  IMMED

CONMODE = BUFFCA
  CS
  LLCA
  SAME

CONVID = 32-bit_resource_id_field
  (32-bit_resource_id_register)

CONXMOD = CA
  CS
  SAME
Notes:

1._operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

3. ECB is meaningful only for asynchronous operations.

4. EXIT is meaningful only for asynchronous operations.

5. You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPESRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**\(^1\) = rpl_extension_address_field
**AAREA**\(^2\) = rpl_extension_address_register
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = acb_address_field
**ACB** = acb_address_register
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLACB in the RPL.

**AREA** = data_area_address_field
**AREA** = data_area_address_register
Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

**AREALEN** = data_area_length
**AREALEN** = data_area_length_register
Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

**BRANCH**
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CD**

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

**CD=DEFER**

Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**

Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
CONVID=(32-bit_resource_id_register)  
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD  
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA  
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS  
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME  
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB  
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL  
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field  
ECB=(ecb_address_register)  
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field  
EXIT=(exit_routine_address_register)  
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

FILL  
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL|BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF  
Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.
FILL=LL  
Specifies the application program is to receive one logical record, or whatever portion of the logical record is available, up to the length specified by the AREALEN field of the RPL. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

OPTCD  
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN  
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY  
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB  
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB  
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NBUFLST  
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFLST  
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field  
RPL=(rpl_address_register)  
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field  
VTROUTA=(vector_address_register)  
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.)
This field is labeled RPL6VAOA in the RPL extension.

\[ \text{VTROUTL} = \text{vector_length_field} \]
\[ \text{VTROUTL} = (\text{vector_length_register}) \]

Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
- X'86' PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received.
by the application program. This field has meaning only when 
FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 
has been received. The indication is either YES or NO (RPL6RMH5 set on or 
off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B’1’)**
One or more FMH-5s have been received from partner application 
programs. The FMH5RCV field continues to be set to YES as long as an 
FMH-5 is waiting to be received by the application program. The 
application program must issue APPCCMD CONTROL=RCVFMH5 in 
order to receive an FMH-5.

**NO (B’0’)**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log 
data is expected. The indication is either YES or NO (RPL6RLOG set on or off). 
This field is labeled RPL6RLOG in the RPL extension.

**YES (B’1’)**
An FMH-7 was received that specified that error log data follows. The 
application program must issue APPCCMD CONTROL=RECEIVE, 
QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the 
responsibility of the application program to perform an optional receive 
check after issuing APPCCMD CONTROL=RECEIVE, 
QUALIFY=SPEC | ISPEC to determine whether the expected log data was 
sent by the partner LU. The data must be error log data and it must be in 
the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of 
the following values:

- X’0004’ ALLOCATION_ERROR
- X’0014’ DEALLOCATE_ABEND_PROGRAM
- X’0018’ DEALLOCATE_ABEND_SERVICE
- X’001C’ DEALLOCATE_ABEND_TIMER
- X’0030’ PROGRAM_ERROR_NO_TRUNC
- X’0034’ PROGRAM_ERROR_PURGING
- X’0038’ PROGRAM_ERROR_TRUNC
- X’003C’ SERVICE_ERROR_NO_TRUNC
- X’0040’ SERVICE_ERROR_PURGING
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual amount of data the application program received. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.
- X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.
SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV
The field in the RPL extension that indicates what the application program received. It is labeled RPL6WHAT in the RPL extension. The application program should examine the WHATRCV field only when RCPRI indicates $'0000'$. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the format shown in Table 1.

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Meaning</td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
</tr>
</tbody>
</table>
For example, a WHATRCV value indicating that DATA has been received would be represented by X’8000’. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a discussion of the meaning of this field.

State changes

See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0003'</td>
<td>RECEIVE_SPECIFIC_REJECTED</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR_INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_ OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUSTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOT_AUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC**

**Purpose**

This macroinstruction receives information on a specified conversation. The conversation may be in any continuation mode.

**Usage**

When this macroinstruction is issued, VTAM copies any available data from the conversation that is specified by the CONVID parameter to the data area that is specified by the AREA parameter. The AREALEN parameter specifies the length of the data area. If no data is ready to be received on the conversation, VTAM queues the macroinstruction until data arrives.

When this macroinstruction completes, the RECLEN field indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

The application program can issue this macroinstruction when the conversation is in SEND state. In this case, VTAM flushes its SEND buffer, sending all buffered information, along with the SEND indicator, to the partner LU. This changes the conversation to RECEIVE state. VTAM then waits for information to arrive. The remote application program can send data to the local application program after it receives the SEND indication.

If VTAM is processing APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY for a conversation and the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC for the same conversation, the QUALIFY=SPEC request fails with an RCPRI, RCSEC return code of X'0000', X'0003'. (VTAM cannot allow a specific-mode RECEIVE while an any-mode RECEIVE is being processed because if a SEND indication was received on the any-mode RECEIVE while the specific-mode RECEIVE was being processed, a SEND indicator would erroneously be sent to the partner LU as a result of the specific-mode RECEIVE.)
This macroinstruction corresponds to the RECEIVE_AND_WAIT verb described in the LU 6.2 architecture.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:
- RECEIVE
- SEND
- PEND_END_CONV_LOG
- PEND_RCV_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:
- SEND/RECEIVE
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL = RECEIVE , QUALIFY = SPEC

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

AREA = data_area_address_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ \text{AAREA} = \text{rpl_extension_address_field} \]
\[ \text{AAREA} = \text{(rpl_extension_address_register)} \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ \text{ACB} = \text{acb_address_field} \]
\[ \text{ACB} = \text{(acb_address_register)} \]

Specifies the address of an access method control block that identifies the
application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

\textbf{AREA=}\textit{data\_area\_address\_field}

\textbf{AREA=}(\textit{data\_area\_address\_register})

Specifies the data area in which the application program is to receive the data. When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST, AREA specifies an address in which VTAM is to build an extended buffer list. The AREALEN field of the RPL specifies a length of this area that is a nonzero multiple of 48 bytes. Each entry in the buffer list points to a CSM buffer. For each list entry, VTAM provides the CSM token, data length and information necessary for the application to address the storage (address and data space ALET). Note that a large buffer list area can help prevent excessive API crossings. The format of the extended buffer list pointed to by the AREA parameter is mapped by the ISTBLXEN mapping DSECT.

\textbf{AREALEN=}\textit{data\_area\_length}

\textbf{AREALEN=}(\textit{data\_area\_length\_register})

Specifies the length value that is the maximum amount of data the application program is to receive.

If OPTCD=XBUFLST, AREALEN specifies the length of the area in which VTAM builds a buffer list. The buffer list in turn points to the data that has been received. The AREALEN parameter specifies an area length that is a nonzero multiple of 48 bytes.

This field is labeled RPLBUFL in the RPL.

\textbf{BRANCH}

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

\textbf{BRANCH=NO}

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

\textbf{BRANCH=YES}

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

\textbf{CD}

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.
CD=DEFER
Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

FILL
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL | BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF
Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

FILL=LL
Specifies the application program is to receive one logical record, or whatever portion of the logical record is available, up to the length specified by the AREALEN field of the RPL. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.
OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NBUFFLST
Specifies that the AREA field contains the address of the area in which the application is to receive the data. The RECLEN field specifies the length of the data area.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. VTAM builds an extended buffer list in the address specified by the AREA parameter. Each entry in the buffer list points to a CSM buffer containing the data being received by the application. The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

Note: Application programs running in TCB-mode supervisor state must specify BRANCH=YES for HPDT requests.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

VTROUTA=vector_address_field
VTROUTA=(vector_address_register)
Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=vector_length_field
VTROUTL=(vector_length_register)
Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:
X'01'    SEND
X'02'    RECEIVE
X'03'    RECEIVE_CONFIRM
X'04'    RECEIVE_CONFIRM_SEND
X'05'    RECEIVE_CONFIRM_DEALLOCATE
X'07'    PENDING_END_CONVERSATION_LOG
X'08'    END_CONVERSATION
X'09'    PENDING_SEND
X'0A'    PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

X'80'    FDX_RESET
X'81'    SEND/RECEIVE
X'82'    SEND_ONLY
X'83'    RECEIVE_ONLY
X'84'    PENDING_SEND/RECEIVE_LOG
X'85'    PENDING_RECEIVE-ONLY_LOG
X'86'    PENDING_RESET_LOG

**EXPDLLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.
NO (B’0’)
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B’1’)
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR

X'0014'
DEALLOCATE_ABEND_PROGRAM

X'0018'
DEALLOCATE_ABEND_SERVICE

X'001C'
DEALLOCATE_ABEND_TIMER

X'0030'
PROGRAM_ERROR_NO_TRUNC

X'0034'
PROGRAM_ERROR_PURGING

X'0038'
PROGRAM_ERROR_TRUNC

X'003C'
SERVICE_ERROR_NO_TRUNC

X'0040'
SERVICE_ERROR_PURGING

X'0044'
SERVICE_ERROR_TRUNC

X'0048'
RESOURCE_FAILURE_NO_RETRY

X'005C'
USER_ERROR_CODE_RECEIVED

NO (B’0’)
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**

The field in the RPL that returns to the application program the actual amount of data the application program received. If the application program receives information other than data, this variable is set to 0. When OPTCD=XBUFLST is specified, VTAM returns the actual length of the extended buffer list that is built in the buffer list area pointed to by the AREA operand. This field is labeled RPLRLLEN in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETR. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.
YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

WHATRCV
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT in the RPL. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.

<table>
<thead>
<tr>
<th>RPL6RCV1</th>
<th>RPL6RCV2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit</strong></td>
<td><strong>Meaning</strong></td>
</tr>
<tr>
<td>0</td>
<td>DATA</td>
</tr>
<tr>
<td>1</td>
<td>DATA_COMPLETE</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
</tr>
<tr>
<td>6</td>
<td>LOG_DATA</td>
</tr>
<tr>
<td>7</td>
<td>PS_HEADER</td>
</tr>
</tbody>
</table>

For example, a WHATRCV value indicating that DATA has been received would be represented by X'8000'. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a discussion of the meaning of this field.

State changes
See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.
### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0003'</td>
<td>RECEIVE_SPECIFIC_REJECTED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0030'</td>
<td>PARAMETER_ERROR—STORAGE_TYPE_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—EXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>PARAMETER_ERROR—A_REQUIREDVECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0038'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'003C'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0044'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>PARTNER_COMMITTED протокол VIOLATION</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=REJECT, QUALIFY=CONV

Purpose

This macroinstruction deallocates a conversation abnormally as well as its underlying session when the application program detects a protocol violation on the conversation.

If the conversation is no longer associated with a session when APPCCMD CONTROL=REJECT, QUALIFY=CONV is issued, VTAM does not unbind the session.

Usage

When the application program detects a protocol violation on the conversation, it issues this macroinstruction and specifies a sense code on the SENSE parameter. VTAM deallocates the conversation first. If the conversation is still associated with a session, VTAM deactivates the session by issuing an UNBIND of type X'FE', which contains the user-specified sense code. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a list of valid UNBIND sense codes.

As an example, suppose the local application program issues an APPCCMD macroinstruction that completes with a return code of PROGRAM_ERROR_NO_TRUNC and LOGRCV=YES, which indicates that an error is detected and that the partner LU is sending error log data. Also, suppose the local application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to receive the error log data and either no data is received or the data that is received is not error-log data. This means that the partner LU committed a protocol violation, and the application program could issue this macroinstruction to end the conversation and session.

APPCCMD CONTROL=REJECT, QUALIFY=CONV can be issued to cancel an APPCCMD macroinstruction that was issued on the conversation previously. However, it cannot cancel an APPCCMD CONTROL=OPRCNTL, APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY, or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY macroinstruction that has not been matched to a conversation. Nor can it cancel an APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction that was issued previously for the same conversation or an APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL | IALL. 

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- RECEIVE_ONLY
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD — CONTROL — REJECT — QUALIFY — CONV

(1) RPL — rpl_address_field — rpl_address_register

(2) AAREA — rpl_extension_address_field — rpl_extension_address_register

(2) ACB — acb_address_field — acb_address_register

(2) BRANCH — NO — YES

(3) CONVID — 32-bit_resource_id_field — 32-bit_resource_id_register
```
**Notes:**

1. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

**Input parameters**

The following information shows descriptions of the input parameters:

**AAREA** = rpl_extension_address_field

**AAREA** = (rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB** = acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the
application program that is issuing the APPCCMD macroinstruction. VTAM
associates conversations with application programs using the conversation ID
(CONVID). The application program associates conversations with transaction
programs. Application programs cannot issue APPCCMD macroinstructions in
address spaces other than the ACB address space. This field is labeled
RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application
programs running in supervisor state under a TCB. Application programs
running in TCB-mode supervisor state can use BRANCH=YES to obtain
authorized path services. The indicator resides within the RPLEXTDS field of
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs
running in problem state (non-supervisor state) under a TCB,
BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is
processed in this manner automatically, regardless of the actual setting of
the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD
in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLEXTDS field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:
**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing the APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource. The application program is allowed to issue APPCCMDs against other conversations.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=NKEEPSRB**
Specify that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

**RPL=rpl_address_field**
**RPL=(rpl_address_register)**
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**SENSE=32-bit_unbind_sense_code**
**SENSE=(32-bit_unbind_sense_code_register)**
Indicates the reason for the APPCCMD CONTROL=REJECT macroinstruction. This field specifies a 32-bit UNBIND (X'FE') sense code. VTAM generates an UNBIND (X'FE') carrying the supplied sense code and ends the conversation. This field is labeled RPL6SNSO in the RPL extension. (Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.wss?uid=swg21092064) for more information on sense codes.)

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following value:

X'08' END_CONVERSATION

For full-duplex conversations, this field can have the following value:

X'80' FDX_RESET

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is END_CONV after successful processing.

For full-duplex conversations, the conversations state is FDX_RESET after successful processing.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0020'</td>
<td>PARAMETER_ERROR—PREVIOUS_REJECT_REQUEST_OUTSTANDING</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL DOES NOT SUPPORT REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP

Purpose

This macroinstruction deactivates the session associated with the conversation group and any conversations matched to the session. The application program specifies, through the deactivation type code, that either a protocol violation has occurred or cleanup is necessary.

Usage

If the application program detects a protocol violation committed by the partner LU or if the architected processing indicates that a cleanup deactivation of the session should occur, the application program issues APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP to terminate the session. This session can have an active conversation associated with it. If so, the conversation fails with an indication of an abnormal termination.

By using the deactivation type (DEACTYP) parameter, the application program can indicate that VTAM should send either an UNBIND_PROTOCOL_VIOLATION (X'0F') or an UNBIND_CLEANUP (X'0F') to deactivate the session. If the deactivation type parameter is omitted, or is equal to a value other than X'0F' or X'FE', VTAM generates an UNBIND (X'0F'). The sense code parameter is valid only if UNBIND (X'FE') is specified.

The application program must specify the conversation group that is to be deactivated. To do this, it uses the CGID parameters to specify the conversation group identifier.

VTAM posts the application program with successful return codes if no session is active with the specified conversation group identifier.
APPCCMD CONTROL=REJECT, QUALIFY=CONVGRP can be issued without knowledge of any conversations associated with the specified session through the CGID parameter. It corresponds to the DEACTIVATE_CONVERSATION_GROUP verb in the LU 6.2 architecture.

**Context**

This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

**Syntax**

```
APPCCMD  CONTROL=REJECT, QUALIFY=CONVGRP

(1) RPL=rpl_address_field
    (  rpl_address_register  )

(2) AAREA=rpl_extension_address_field
    (  rpl_extension_address_register  )

(2) ACB=acb_address_field
    (  acb_address_register  )

(2) BRANCH=NO

(3) CGID=32-bit_conversation_group_id_field
    (  32-bit_conversation_group_id_register  )

(3) DEACTYP=8-bit_unbind_type_code
    (  8-bit_unbind_type_code_register  )
```
Notes:
1. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=`rpl_extension_address_field`

**AAREA**=`(rpl_extension_address_register)`

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=`acb_address_field`
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CGID=32-bit_conversation_group_id_field

CGID=(32-bit_conversation_group_id_register)
Specifies the 32-bit conversation group ID.

This value can be obtained from a previous APPCCMD CONTROL=ALLOC, CONTROL=PREALLOC, or CONTROL=RCVFMH5 macroinstruction. If the CGID operand is not specified, VTAM uses the conversation group ID that is already in the RPL6CGID field on the RPL extension.

The conversation group ID identifies a specific session between two specific LUs. It provides a means by which a VTAM LU 6.2 application program and its partner LU can share serially the same session.

DEACTYP=8-bit_unbind_type_code
DEACTYP=(8-bit_unbind_type_code_register)
The UNBIND type code can be specified as cleanup (X'0F') or as protocol violation (X'FE'). If DEACTYP specifies cleanup, the value specified on the SENSE operand will be ignored. However, if DEACTYP specifies protocol error, the UNBIND will flow with the sense code specified by the SENSE operand. If the DEACTYP operand is omitted or a value other than X'0F' or X'FE' is entered, VTAM will generate an UNBIND of X'0F'. The application program can be posted with a return code of INVALID_DEACTIVATION_TYPE_CODE, but the session may still have been deactivated successfully. This field is labeled RPL6DETP in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.
ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event control block address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SENSE=32-bit_unbind_sense_code
SENSE=(32-bit_unbind_sense_code_register)
Indicates the reason for the APPCCMD CONTROL=REJECT macroinstruction. This field specifies a 32-bit UNBIND (X’FE’) sense code. VTAM generates an UNBIND (X’FE’) carrying the supplied sense code and ends the conversation. This field is labeled RPL6SNSO in the RPL extension. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on sense codes.)
RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**
- The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
- The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
- The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.
  - **YES (B’1’)**
    - One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.
  - **NO (B’0’)**
    - No FMH-5s are waiting to be received by the application program.

**RCPRI**
- The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
- The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
- The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**USERFLD**
- Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

Conversation states do not apply to this macroinstruction.
## Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO ECB FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0020'</td>
<td>PARAMETER_ERROR—PREVIOUS REJECT REQUEST OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0027'</td>
<td>PARAMETER_ERROR—INVALID DEACTIVATION TYPE_CODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002A'</td>
<td>PARAMETER_ERROR—INVALID CGID VALUE SPECIFIED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR.OS_LEVEL DOES NOT SUPPORT REQUESTED FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME FAILURE</td>
</tr>
</tbody>
</table>

## APPCCMD CONTROL=REJECT, QUALIFY=SESSION

### Purpose

This macroinstruction deactivates the session and any conversation matched to this session. The application program specifies, through the deactivation type code, that either a protocol violation has occurred or cleanup is necessary.

### Usage

If the application program detects a protocol violation committed by the partner LU or if the architected processing indicates that a cleanup deactivation of the session should occur, the application program issues APPCCMD CONTROL=REJECT, QUALIFY=SESSION to terminate the session. This session can have an active conversation associated with it. If so, the conversation fails with an indication of an abnormal termination. The application must issue an APPCCMD to receive the conversation failure notification and cause conversation cleanup.

By using the deactivation type (DEACTYP) parameter, the application program can indicate that VTAM should send either an UNBIND PROTOCOL VIOLATION (X'FE') or an UNBIND CLEANUP (X'0F') to deactivate the session. If the deactivation type parameter is omitted, or is equal to a value other than X'0F' or X'FE', VTAM generates an UNBIND (X'0F'). The sense code parameter is ignored unless an UNBIND (X'FE') is specified.

The application program must specify the session that is to be deactivated. To do this, it uses the SESSID and SESSIDL parameters to specify the session instance identifier. These parameters were made available to the conversation at conversation allocation from the APPCCMD CONTROL=RCVFMH5 macroinstruction and the APPCCMD CONTROL=ALLOC macroinstruction.
VTAM posts the application program with successful return codes if no session is active with the specified session identifier and session identifier length.

APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued without knowledge of any conversations associated with the specified session through the SESSID parameter.

**Context**

This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

**Syntax**

```
APPCCMD --CONTROL--REJECT-- --QUALIFY--SESSION

(1)
RPL = rpl_address_field (rpl_address_register)

(2)
AAREA = rpl_extension_address_field (rpl_extension_address_register)

(2)
ACB = acb_address_field (acb_address_register)

(2)
BRANCH = NO

(3)
DEACTYP = 8-bit_unbind_type_code (8-bit_unbind_type_code_register)

(4) (2)
ECB = INTERNAL

(4) (2)
EXIT = exit_routine_address_field (exit_routine_address_register)
```
Notes:
1 See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
**ACB=acb_address_field**  
ACB=(*acb_address_register*)  
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**  
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**  
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**  
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**DEACTYP=8-bit_unbind_type_code**  
**DEACTYP=(8-bit_unbind_type_code_register)**  
The UNBIND type code can be specified as cleanup (X'0F') or as protocol violation (X'FE'). If DEACTYP specifies cleanup, the value specified on the SENSE operand will be ignored. However, if DEACTYP specifies protocol error, the UNBIND will flow with the sense code specified by the SENSE operand. If the DEACTYP operand is omitted or a value other than X'0F' or X'FE' is entered, VTAM will generate an UNBIND of X'0F'. The application program can be posted with a return code of INVALID_DEACTIVATION_TYPE_CODE, but the session may still have been deactivated successfully. This field is labeled RPL6DETP in the RPL extension.

**ECB**  
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**  
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**  
**ECB=(ecb_address_register)**  
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**  
**EXIT=(exit_routine_address_register)**  
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled.
when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTD5 field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**SENSE=32-bit_unbind_sense_code**

**SENSE=(32-bit_unbind_sense_code_register)**

Indicates the reason for the APPCCMD CONTROL=REJECT macroinstruction. This field specifies a 32-bit UNBIND (X'FE') sense code. VTAM generates an UNBIND (X'FE') carrying the supplied sense code and ends the conversation. This field is labeled RPL6SSNSO in the RPL extension. (Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](http://www-01.ibm.com/support/docview.wss?uid=ssg1S1005947) for more information on sense codes.)

**SESSID=session_instance_id_field**

**SESSID=(session_instance_id_register)**

Specifies the session to be deactivated. The session instance identifier must refer to an active session. (A session must be activated before it can be deactivated.) The session instance identifier is passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. A session that is in pending activation state cannot be specified. A conversation that is matched to this session fails with a session outage notification. This field is labeled RPL6SSID in the RPL extension.

**SESSIDL=session_instance_id_length**

**SESSIDL=(session_instance_id_length_register)**

Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD
CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction.
This field is labeled RPL6SIDL in the RPL extension.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES** (B'1')
  
  One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO** (B'0')

  No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

**State changes**

Conversation states do not apply to this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.
APPCCMD CONTROL=RESETRCV

Purpose

This macroinstruction changes the existing continuation modes of a conversation. For example, it can change the conversation from continue-specific (CS) mode to logical-record-continue-any (LLCA) mode for receiving normal information.

This macroinstruction can also change the existing mode for receiving expedited information.

Usage

When this macroinstruction is issued, VTAM changes the continuation mode for receiving normal information of the conversation specified with the CONVID parameter to the continuation mode specified on the CONMODE parameter.

VTAM also changes the expedited information mode of the conversation specified with the CONXMOD parameter to the expedited information mode specified on the CONXMOD parameter.

For a complete discussion of continuation modes and an example of how this macroinstruction can be used, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PEND_END_CONV_LOG
- **PEND_RCV_LOG**

For full-duplex conversation, this macroinstruction can be issued from the following conversations states:
- **SEND/RECEIVE**
- **SEND_ONLY**
- **RECEIVE_ONLY**
- **PENDING_SEND/RECEIVE_LOG**
- **PENDING_RECEIVE-ONLY_LOG**
- **PENDING_RESET_LOG**

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL = RESETRCV name, QUALIFY = NULL

(1) RPL = rpl_address_field
     (rpl_address_register)

(2) AAREA = rpl_extension_address_field
     (rpl_extension_address_register)

(2) ACB =acb_address_field
     (acb_address_register)

(2) BRANCH = NO, YES
(3) CONMODE = BUFFCA, CS, LLCA, SAME

(3) CONVID = 32-bit_resource_id_field
     (32-bit_resource_id_register)
```
Notes:
1 See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
2 Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
3 Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
4 ECB is meaningful only for asynchronous operations.
5 EXIT is meaningful only for asynchronous operations.
6 You can code more than one suboperand on OPTCD, but no more than one from each group.
7 KEEPSRB is meaningful only for synchronous operations.
8 NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
    Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
**ACB**=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.
CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY|IANY.

CONXMOD=CS

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC.

CONXMOD=SAME

Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY

Specifies that control is to be returned to the application program.
immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing this APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex, or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstruction. The application program can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

X'01'  SEND
X'02'  RECEIVE
X'03'  RECEIVE_CONFIRM
X'04'  RECEIVE_CONFIRM_SEND
X'05'  RECEIVE_CONFIRM_DEALLOCATE
X'07'  PENDING_END_CONVERSATION_LOG
X'08'  END_CONVERSATION
X'09'  PENDING_SEND
- PENDING_RECEIVE_LOG

For full-duplex conversations, this field can contain the following values:
- FDX_RESET
- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

- FDB2
  - The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

- FMH5LEN
  - The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

- FMH5RCV
  - The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

    **YES (B'1')**
    - One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

    **NO (B'0')**
    - No FMH-5s are waiting to be received by the application program.

- RCPRI
  - The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

- RCSEC
  - The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

- RTNCD
  - The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

- USERFLD
  - Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM
places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

There are no state changes associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPCC</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=SEND, QUALIFY=CONFIRM

Purpose

This macroinstruction sends a confirmation request on a half-duplex conversation to a remote application program and waits for a confirmation reply (either synchronously or asynchronously).

Usage

This macroinstruction can be used only for half-duplex conversations.

When this macroinstruction is issued, VTAM flushes the SEND buffer of the specified conversation and sends a confirmation request. This macroinstruction completes only after the partner LU receives the confirmation request and issues APPCCMD CONTROL=SEND, QUALIFY=CONFIRMED.
This macroinstruction enables the local and remote application programs to synchronize their processing with one another. The application program can use this APPCCMD for various transaction program-level functions. For example:

- The application program can issue this APPCCMD immediately after an APPCCMD CONTROL=ALLOC macroinstruction in order to determine whether the allocation of the conversation is successful before sending any data.
- The application program can issue this APPCCMD as a request for acknowledgment of data that it sent to the remote program.

The application program must ensure that APPCCMD CONTROL=SEND, QUALIFY=CONFIRM is not issued by a transaction program against a conversation that was allocated with a synchronization level of NONE.

This macroinstruction corresponds to the CONFIRM verb described in the LU 6.2 architecture.

**Context**

For half-duplex conversations, this macroinstruction can be issued from following conversation states:

- SEND
- PENDING_SEND

This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
   Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
   Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH
   Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT1 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
  Specifies the address of the request parameter list that contains information to
  be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
  The field in the RPL6 extension that indicates the state of the conversation.
  This field is labeled RPL6CCST in the RPL extension.

  This field can have the following values:
  
  X'01'  SEND
  X'02'  RECEIVE
  X'03'  RECEIVE_CONFIRM
  X'04'  RECEIVE_CONFIRM_SEND
  X'05'  RECEIVE_CONFIRM_DEALLOCATE
  X'07'  PENDING_END_CONVERSATION_LOG
  X'08'  END_CONVERSATION
  X'09'  PENDING_SEND
  X'0A'  PENDING_RECEIVE_LOG

EXPDLEN
  The field in the RPL6 that shows the length of the expedited data waiting to be
  received. This field has meaning only when EXPDRCV=YES. This field is
  labeled RPL6EXDL in the RPL extension.

EXPDRCV
  The field in the RPL6 that indicates whether expedited data is waiting to be
  received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
  The field in the RPL in which a global VTAM secondary return code is
  returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
  The field in the RPL extension that returns the length of the FMH-5 waiting to
  be received by the application program. If multiple FMH-5s are waiting to be
  received, FMH5LEN specifies the length of the longest FMH-5 to be received
  by the application program. This field has meaning only when
  FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
  The field in the RPL extension that returns an indication of whether an FMH-5
  has been received. The indication is either YES or NO (RPL6RMH5 set on or
  off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
  One or more FMH-5s have been received from partner LUs. The
  FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting
  to be received by the application program. The application program must
  issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO  (B'0')
  No FMH-5s are waiting to be received by the application program.
LOGRCV

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')

An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR

X'0014'
DEALLOCATE_ABEND_PROGRAM

X'0018'
DEALLOCATE_ABEND_SERVICE

X'001C'
DEALLOCATE_ABEND_TIMER

X'0030'
PROGRAM_ERROR_NO_TRUNC

X'0034'
PROGRAM_ERROR_PURGING

X'0038'
PROGRAM_ERROR_TRUNC

X'003C'
SERVICE_ERROR_NO_TRUNC

X'0040'
SERVICE_ERROR_PURGING

X'0044'
SERVICE_ERROR_TRUNC

X'005C'
USER_ERROR_CODE_RECEIVED

NO (B'0')

Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning
only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

- X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG in the RPL extension.

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

There are no state changes associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERRORPURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_INVALID_FOR_FULL-DUPELEX_CONVERSATION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OSCLEVEL_DOESNOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SEND, QUALIFY=CONFIRMED

Purpose

This macroinstruction sends a positive confirmation reply to the remote application program on a half-duplex conversation.

Usage

This macroinstruction can only be used for half-duplex conversations.

When the application program receives a CONFIRM indication in the WHATRCV field after an APPCCMD CONTROL=RECEIVE macroinstruction, the application issues this macroinstruction to indicate that all the data that was sent by the CONFIRM indication has been received and is acceptable. This allows an application program to synchronize processing with its partner application.

If the application program receives a CONFIRM indication and it detects an error in the data it received before the CONFIRM, it can issue APPCCMD CONTROL=SEND, QUALIFY=ERROR to send a negative reply to the CONFIRM.

This macroinstruction corresponds to the CONFIRMED verb described in the LU 6.2 architecture.

Context

This macroinstruction may only be issued from the following conversation states on a half-duplex conversation:

- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE

This macroinstruction is not allowed on a full-duplex conversation.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
Chapter 1. LU 6.2 macroinstruction syntax and operands

433
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=\texttt{rpl\_extension\_address\_field}

\texttt{AAREA=} (\texttt{rpl\_extension\_address\_register})

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=\texttt{acb\_address\_field}

\texttt{ACB=} (\texttt{acb\_address\_register})

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(epl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**
The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

This field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

- **YES (B'1')**
  One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

- **NO (B'0')**
  No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary
return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**
These changes are applicable when RCPRI indicates OK.

**RECEIVE** state is entered when an indicator of CONFIRM, DATA_CONFIRM, or DATA_COMPLETE_CONFIRM was received on the preceding APPCCMD CONTROL=RECEIVE.

**SEND** state is entered when an indicator of CONFIRM_SEND, DATA_CONFIRM_SEND, or DATA_COMPLETE_CONFIRM_SEND was received on the preceding APPCCMD CONTROL=RECEIVE.

**END_CONV** state is entered when an indicator of CONFIRM_DEALLOCATE, DATA_CONFIRM_DEALLOCATE, or DATA_COMPLETE_CONFIRM_DEALLOCATE was received on the preceding APPCCMD CONTROL=RECEIVE.

**Return codes**
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (PARTNER LU REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SEND, QUALIFY=DATA

Purpose

This macroinstruction sends data to a partner LU.

Usage

This macroinstruction transfers data that is specified by the AREA parameter into the SEND buffer of the conversation that is specified by the CONVID parameter. When there is more data in the conversation's SEND buffer than the maximum RU size for the conversation's session, an RU is sent to the partner LU. If the data does not exceed a maximum RU size, the data in the buffer remains there until the application program sends more data or causes the SEND buffer to be flushed.

Note: If OPTCD=XBUFLST is specified on this macroinstruction, all of the data is sent to the partner LU, even if the data does not exceed the maximum RU size.

The AREA parameter can specify a single data area to be sent, or it can specify a buffer list that points to multiple data areas to be sent. The OPTCD parameter specifies which of these methods is used.


This macroinstruction corresponds to the SEND_DATA verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax
APPCCMD CONTROL = SEND, QUALIFY = DATA

RPL = rpl_address_field
(rpl_address_register)

AAREA = rpl_extension_address_field
(rpl_extension_address_register)

ACB = acb_address_field
(acb_address_register)

AREA = data_area_or_buffer_list_address_field
(data_area_or_buffer_list_address_register)

BRANCH = NO, YES

CONMODE = BUFFCA
CS
LLCA
SAME

CONVID = 32-bit_resource_id_field
(32-bit_resource_id_register)

CONXMOD = CA
CS
SAME

CRYPT = NO, YES

ECB = INTERNAL
(ecb_address_field
(ecb_address_register)

EXIT = exit_routine_address_field
(exit_routine_address_register)
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for synchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)
Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the
data to be sent. Unless an HPDT request has proceeded this
macroinstruction on this conversation, VTAM tracks the logical records
supplied by the application program, examining the logical-record length
(LL) field associated with each logical record. (It does not inspect the data
portion of the logical record.)

- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry
in the buffer list points to the data to be sent. Unless an HPDT request has
proceeded this macroinstruction on this conversation, VTAM tracks the
logical records supplied by the application program, examining the
logical-record length (LL) field associated with each logical record. (It does
not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list.
The data to be sent resides in CSM buffers. Once XBUFLST has been
specified on an APPCCMD, VTAM does not track logical records supplied
by the application on this or subsequent requests, for the duration of the
conversation. Each entry in the extended buffer list is 48 bytes. RU
boundaries and logical record boundaries are independent of the buffer
boundaries. Each entry in the buffer list can specify any displacement in a
CSM buffer. VTAM uses the CSM token rather than the storage address to
track a given CSM buffer. Note that a CSM token cannot be repeated in an
extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM
buffer must first be segmented by using the IVTCSM
REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional
tokens for the storage area. The tokens are provided on the extended buffer
list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

When OPTCD=XBUFLST is specified on this macroinstruction, VTAM
performs an internal flush of any data remaining in the send buffer, even if it
does not exceed the maximum RU size.

BRANCH
Specifies whether authorized path processing is to be used for application
programs running in supervisor state under a TCB. Application programs
running in TCB-mode supervisor state can use BRANCH=YES to obtain
authorized path services. The indicator resides within the RPLEXTDS field of
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs
running in problem state (non-supervisor state) under a TCB,
BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is
processed in this manner automatically, regardless of the actual setting of
the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the
APPCCMD. This field is labeled RPL6CMOD in the RPL extension.
CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.
CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
 Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
 Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD.
CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to "z/OS Communications Server: SNA Programmer’s LU 6.2 Guide"

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to "z/OS Communications Server: SNA Programmer’s LU 6.2 Guide" for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to "z/OS Communications Server: SNA Programmer’s LU 6.2 Guide" for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is
rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

**RECLEN**=

**RECLEN**=

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

**RPL**=

**RPL**=

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can contain the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
EXPDLLEN
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

Note: The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
ALLOCATION_ERROR

X'0014'
DEALLOCATE_ABEND_PROGRAM
X'0018'
DEALLOCATE_ABEND_SERVICE
X'001C'
DEALLOCATE_ABEND_TIMER
X'0030'
PROGRAM_ERROR_NO_TRUNC
X'0034'
PROGRAM_ERROR_PURGING
X'0038'
PROGRAM_ERROR_TRUNC
X'003C'
SERVICE_ERROR_NO_TRUNC
X'0040'
SERVICE_ERROR_PURGING
X'0044'
SERVICE_ERROR_TRUNC
X'005C'
USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled
RPL6SNSI in the RPL. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STD$ in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be
associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT.Recognized</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED_FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SEND, QUALIFY=DATACON

Purpose

This macroinstruction sends data that is supplied by the application program and any data that is already in the SEND buffer to a partner application program on a half-duplex conversation. The data is followed by a confirmation request.

Usage

This macroinstruction can only be used on a half-duplex conversation.

VTAM places the data specified by the AREA parameter in the SEND buffer of the conversation specified by the CONVID parameter. VTAM sends all data in the SEND buffer to the partner LU. The data is followed by a confirmation request. This macroinstruction completes only after a confirmation reply is received from the partner LU. The application program must ensure that the data that it sends completes a logical record.

For more information on sending and responding to confirmation requests, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

This macroinstruction corresponds to the SEND_DATA followed by CONFIRM verbs described in the LU 6.2 architecture.

Context

This macroinstruction can be issued on a half-duplex conversation from the following conversation states:

- SEND
• **PENDING_SEND**

This macroinstruction is not allowed on a full-duplex conversation.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD  CONTROL=SEND, QUALIFY=DATACON

(name) APPCCMD  CONTROL=SEND

(rpl_address_field) RPL=rpl_address_field

(rpl_extension_address_field) AAREA=rpl_extension_address_field

(ocb_address_field) ACB=ocb_address_field

(data_area_or_buffer_list_address_field) AREA=data_area_or_buffer_list_address_field

(NONE) BRANCH=NO

(BUFFCA) CONMODE=BUFCA

(32-bit_resource_id_field) CONVID=32-bit_resource_id_field

(CA) CONXMOD=CA

(NO) CRYPT=CRYPT
```


Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. KEEPSRB is meaningful only for synchronous operations.

7. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field

Specifies the address of an access method control block that identifies the...
application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

AREA=data_area_or_buffer_list_address_field
AREA=(data_area_or_buffer_list_address_register)

Specifies the address of a data buffer or buffer list.

- If OPTCD=NBUFFLST, AREA specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=BUFFLST, AREA specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If OPTCD=XBUFLST, AREA specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.
CONMODE

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME

Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field

CONVID=(32-bit_resource_id_register)

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME

Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.
CRYPT
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCRYP in the RPL.

CRYPT=NO
Do not encrypt data before it is sent.

CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=(ecb_address_field)

ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=(exit_routine_address_field)
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

When the application program regains control after issuing the APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the SEND/RECEIVE
queue until the command has completed. The exception to this is the APPCCMD CONTROL=REJECT, QUALIFY=CONV macroinstruction. The application can issue APPCCMDs against the same conversation resource that processes on the EXPEDITED SEND, EXPEDITED RECEIVE and TESTSTAT queues. For more information about conversation queues refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of 'X'002C', 'X'0010' (INVALID DATA ADDRESS OR LENGTH).
The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

\textbf{RELEN=}data\_length  
\textbf{RELEN=}\textit{(data\_length\_register)}  
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

- If OPTCD=NBUFFLST, RELEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RELEN specifies the length of the buffer list that in turn points to the data to be sent. RELEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFFLST, RELEN specifies the length of the extended buffer list that in turn points to the data to be sent. RELEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

\textbf{RPL=}rpl\_address\_field  
\textbf{RPL=}\textit{(rpl\_address\_register)}  
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

\textbf{CONSTATE}  
The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

This field can contain the following values:

- \textbf{X'01'} SEND
- \textbf{X'02'} RECEIVE
- \textbf{X'03'} RECEIVE\_CONFIRM
- \textbf{X'04'} RECEIVE\_CONFIRM\_SEND
- \textbf{X'05'} RECEIVE\_CONFIRM\_DEALLOCATE
- \textbf{X'07'} PENDING\_END\_CONVERSATION\_LOG
- \textbf{X'08'} END\_CONVERSATION
- \textbf{X'09'} PENDING\_SEND
- \textbf{X'0A'} PENDING\_RECEIVE\_LOG

\textbf{EXPDLLEN}  
The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

\textbf{EXPDRCV}  
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

\textbf{FDB2}  
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

\textbf{FMH5LEN}  
The field in the RPL extension that returns the length of the FMH-5 waiting to
be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LU's. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR

X'0014'
  DEALLOCATE_ABEND_PROGRAM

X'0018'
  DEALLOCATE_ABEND_SERVICE

X'001C'
  DEALLOCATE_ABEND_TIMER

X'0030'
  PROGRAM_ERROR_NO_TRUNC

X'0034'
  PROGRAM_ERROR_PURGING

X'0038'
  PROGRAM_ERROR_TRUNC

X'003C'
  SERVICE_ERROR_NO_TRUNC

X'0040'
  SERVICE_ERROR_PURGING
X'0044'
    SERVICE_ERROR_TRUNC

X'005C'
    USER_ERROR_CODE_RECEIVED

NO (B'0')
    Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
    The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
    The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
    A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.
    
    For more information about application recovery options when RPLXSRV is not set, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](http://www.ibm.com). The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
    The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
    The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
    The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.
    
    X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.
Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
No state changes are associated with this macroinstruction.
### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK (REMOTE PROGRAM REPLIED AFFIRMATIVELY)</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BYPROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BYPGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0094'</td>
<td>X'0000'</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>REQUEST_NOT_ALLOWED—CONTROL/QUALIFY_VALUE_INVALID FOR FULL-DUPLEX_CONVERSATION</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=SEND, QUALIFY=DATAFLU

**Purpose**

This macroinstruction sends data supplied by the application program as well as any data that is already in the SEND buffer to the partner application.

**Usage**

This macroinstruction combines the functions of two macroinstructions: APPCCMD CONTROL=SEND, QUALIFY=DATA followed by APPCCMD CONTROL=SEND, QUALIFY=FLUSH. VTAM places the data that is specified by the AREA parameter in the SEND buffer of the conversation that is specified by the CONVID parameter. VTAM sends all data in the SEND buffer to the partner LU.

This macroinstruction corresponds to SEND_DATA followed by FLUSH verbs described in the LU 6.2 architecture.

For a complete discussion of sending data, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
APPCCMD CONTROL = SEND, QUALIFY = DATAFLU

RPL rpl_address_field (rpl_address_register)

AAREA rpl_extension_address_field (rpl_extension_address_register)

ACB acb_address_field (acb_address_register)

AREA data_area_or_buffer_list_address_field (data_area_or_buffer_list_address_register)

BRANCH NO YES

CONMODE BUFFCA CS LLCA SAME

CONVID 32-bit_resource_id_field (32-bit_resource_id_register)

CONXMOD CA CS SAME

CRYPT NO YES

ECB INTERNAL (ecb_address_field (ecb_address_register))

EXIT exit_routine_address_field (exit_routine_address_register)
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AAREA</strong></td>
<td>Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.</td>
</tr>
<tr>
<td><strong>ACB</strong></td>
<td>Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.</td>
</tr>
</tbody>
</table>
\texttt{AREA=\text{data\_area\_or\_buffer\_list\_address\_field}}

\texttt{AREA=(\text{data\_area\_or\_buffer\_list\_address\_register})}

Specifies the address of a data buffer or buffer list.

- If \texttt{OPTCD=NBUFFLST}, \texttt{AREA} specifies the address of an area containing the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If \texttt{OPTCD=BUFFLST}, \texttt{AREA} specifies the address of a buffer list. Each entry in the buffer list points to the data to be sent. Unless an HPDT request has proceeded this macroinstruction on this conversation, VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.)

- If \texttt{OPTCD=XBUFLST}, \texttt{AREA} specifies the address of an extended buffer list. The data to be sent resides in CSM buffers. Once XBUFLST has been specified on an APPCCMD, VTAM does not track logical records supplied by the application on this or subsequent requests, for the duration of the conversation. Each entry in the extended buffer list is 48 bytes. RU boundaries and logical record boundaries are independent of the buffer boundaries. Each entry in the buffer list can specify any displacement in a CSM buffer. VTAM uses the CSM token rather than the storage address to track a given CSM buffer. Note that a CSM token cannot be repeated in an extended buffer list.

If multiple areas of a CSM buffer are to be used on an APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction, which obtains additional tokens for the storage area. The tokens are provided on the extended buffer list and specified on the APPCCMD macroinstruction.

This field is labeled RPLAREA in the RPL.

\textbf{BRANCH}

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use \texttt{BRANCH=YES} to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

- \texttt{BRANCH=NO}

  Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, \texttt{BRANCH=NO} is the only option.

- \texttt{BRANCH=YES}

  Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the \texttt{BRANCH} field.

\textbf{CONMODE}

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

- \texttt{CONMODE=BUFFCA}

  Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY\textls{-ANY}
can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC macroinstruction.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC ISPEC macroinstruction.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**
**CONVID=(32-bit_resource_id_register)**
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY IANY.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC ISPEC.

**CONXMOD=SAME**
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

**CRYPT**
Specifies whether data at the location indicated by the AREA is to be encrypted before it is sent on the conversation. This field is labeled RPLTCTRYP in the RPL.

**CRYPT=NO**
Do not encrypt data before it is sent.
CRYPT=YES
Encrypt the data before it is sent. Specify CRYPT=YES only if encryption is allowed on the mode to which the conversation is allocated. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a description of how VTAM determines the level of cryptography.)

Note: If CRYPT=YES is specified, VTAM does not use HPDT services to transfer data, even if OPTCD=XBUFLST is specified. Instead, the normal send or receive path is used.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED
SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

OPTCD=XBUFLST
Specifies that the HPDT interface is to be used. The AREA field of the RPL points to an extended buffer list containing 48-byte buffer list entries. Each entry in the buffer list points to a CSM buffer to be used for sending data. (Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 48 bytes.

The following requirements apply to APPCCMD macroinstructions used to send data from an application-supplied extended buffer list:

- Applications using HPDT must use authorized path processing. Therefore, BRANCH=NO cannot be specified when OPTCD=XBUFLST.
- Entries in the extended buffer list must not contain any negative values. If a negative value exists in the entry, then the macroinstruction is rejected with an RCPRI, RCSEC combination of X'002C', X'0010' (INVALID DATA ADDRESS OR LENGTH).

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.
RECLEN=data_length

RECLEN=(data_length_register)

Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLN in the RPL.

- If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the data area specified by AREA.
- If OPTCD=BUFFLST, RECLEN specifies the length of the buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)
- If OPTCD=XBUFFLST, RECLEN specifies the length of the extended buffer list that in turn points to the data to be sent. RECLEN must be a nonzero multiple of 48 bytes. (Extended buffer list entries consist of 48 bytes.)

RPL=rpl_address_field

RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01'  SEND
- X'02'  RECEIVE
- X'03'  RECEIVE_CONFIRM
- X'04'  RECEIVE_CONFIRM_SEND
- X'05'  RECEIVE_CONFIRM_DEALLOCATE
- X'07'  PENDING_END_CONVERSATION_LOG
- X'08'  END_CONVERSATION
- X'09'  PENDING_SEND
- X'0A'  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
- X'84'  PENDING_SEND/RECEIVE_LOG
- X'85'  PENDING_RECEIVE-ONLY_LOG
- X'86'  PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.
EXPDRCV
The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

LOGRCV
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

Note: The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.

YES (B'1')
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

X'0004'
  ALLOCATION_ERROR

X'0014'
  DEALLOCATE_ABEND_PROGRAM

X'0018'
  DEALLOCATE_ABEND_SERVICE

X'001C'
  DEALLOCATE_ABEND_TIMER
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RPLXSRV
A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application's responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates
USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B’1’)**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

**NO (B’0’)**
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

**STSHBF**
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

**STSHDS**
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

**USERFLD**
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATAADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—INVALID_EXPEDITED_DATA_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=SEND, QUALIFY=ERROR

#### Purpose

This macroinstruction informs the partner LU that the local application program detects an error.

#### Usage

When this macroinstruction is issued, VTAM builds an FMH-7, based on the TYPE and SENSE parameters, to represent the error that the application program detected.

The application program can specify one of the following types of errors:

- **PROGRAM**—error in an end-user transaction program
- **SERVICE**—error in a service component of a transaction program
- **USER**—user-specified error.

VTAM determines the sense code to place in the FMH-7 for program and service errors. The application program specifies the sense code on the SENSE parameter for user errors. The sense code specified must be appropriate to the error. Otherwise, improper processing of the macroinstruction might result. For a list of valid sense codes for an FMH-7, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/knowledgecenter/SG2479_9.4.0/com.ibm.zos.v9r4.mcpguide/).  

A negative response must be sent to the partner LU before the FMH-7 can be transmitted if the conversation is in one of the following states:

- **RECEIVE**
- **PEND_SEND**
- **RECEIVE_CONFIRM**
- **RECEIVE_CONFIRM_SEND**
- **RECEIVE_CONFIRM_DEALLOCATE**
VTAM flushes the SEND buffer before the FMH-7 is created and a negative response is not sent if the conversation is in one of the following states:

- SEND
- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

For half-duplex conversations, the FMH-7 (and error log data that is supplied) is not sent to the partner LU until the application program issues a macroinstruction such as APPCCMD CONTROL=SEND, QUALIFY=FLUSH that causes the SEND buffer to be flushed. For full-duplex conversations, the FMH-7 is sent immediately to the conversation partner.

This macroinstruction corresponds to the SEND_ERROR verb described in the LU 6.2 architecture.

For more details on error handling, refer to the z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_RECEIVE_LOG

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
**Input parameters**

The following information shows descriptions of the input parameters:

**AAREA**

*AAREA= rpl_extension_address_field*

*AAREA= (rpl_extension_address_register)*

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**

*ACB= acb_address_field*

*ACB= (acb_address_register)*

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**

*AREA= optional_log_data_area_address_field*

*AREA= (optional_log_data_area_address_register)*

Specifies the address of a data area containing a formatted error log GDS variable to be sent to the partner LU. The application program is responsible for placing the error log data into the local system log. VTAM treats the error log GDS variable the same as other conversation data. (Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.wss?uid=swg27016019) for a description of the error log GDS variable.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.
CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC|ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY|IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.
ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

**OPTCD=KEEPSSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM
returns to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPOPT11 field of the RPL.

**RECLEN=data_length**

**RECLEN=(data_length_register)**

Specifies the length of the data area indicated by the AREA field. This field is
labeled RPLRLEN in the RPL. A 0 value in the RECLEN field indicates that the
application program has chosen not to provide optional error log data to
VTAM. If the application program specifies RECLEN=0, VTAM indicates in the
FMH-7 it creates as a result of this APPCCMD that no error log data follows
the FMH-7, and the AREA field in the RPL is ignored.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

**SENSE=user-supplied_32-bit_fmh-7_sense_code**

**SENSE=(user-supplied_32-bit_fmh-7_sense_code_register)**

Specifies the sense code that VTAM places in the FMH-7. This field is
applicable only when TYPE=USER is specified. It is labeled RPL6SNSO in the
RPL extension.

**TYPE**

Specifies the type of error being reported. This field is intended to distinguish
between errors to be reported to end-user transaction programs and errors to
be reported to a service component, such as a mapped conversation
component, of the LU. This field is labeled RPL6TYPE in the RPL extension.

**TYPE=PROGRAM**

Specifies that an end-user transaction program error is being reported.
VTAM determines the appropriate sense code to be placed in the FMH-7
based upon the state of the conversation and of the LU’s SEND buffer.
VTAM also determines whether the FMH-7 should be preceded by a
negative response, based upon the current state of the conversation.

VTAM will place a sense code of either X’08890000’ or X’08890001’ in the
FMH-7 for this type of error.

**TYPE=SERVICE**

Specifies that a service-component error is being reported. VTAM
determines the appropriate sense code to be placed in the FMH-7 based
upon the state of the conversation and of the LU’s SEND buffer. VTAM
also determines whether the FMH-7 should be preceded by a negative
response, based upon the current state of the conversation.

VTAM will place a sense code of either X’08890100’ or X’08890101’ in the
FMH-7 for this type of error.

**TYPE=USER**

Specifies that the application program is providing to VTAM a
user-specified sense code that is to be placed in the FMH-7. The FMH-7
sense code is passed to VTAM through the SENSE field of the RPL
extension. It is the responsibility of the application program to supply a
valid FMH-7 sense code. This user-specified sense code must be
appropriate for the error. Otherwise, improper processing of the
macroinstruction might occur. VTAM determines whether the FMH-7 should be preceded by a negative response, based upon the current state of the conversation. For a list of sense codes that the application program can use, refer to the z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

For more discussion on this type of error, refer to the z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVE_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
- X'86' PENDING_RESET_LOG

**EXPDLEN**

The field in the RPL6 extension that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to
be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**LOGRCV**
The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**Note:** The LOGRCV field is reserved if this macroinstruction is issued on a full-duplex conversation.

**YES (B'1')**
An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC in order to retrieve the log data. The application program must perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data, and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPR field of the RPL extension contains one of the following values:

- X'0004'  ALLOCATION_ERROR
- X'0014'  DEALLOCATE_ABEND_PROGRAM
- X'0018'  DEALLOCATE_ABEND_SERVICE
- X'001C'  DEALLOCATE_ABEND_TIMER
- X'0030'  PROGRAM_ERROR_NO_TRUNC
- X'0034'  PROGRAM_ERROR_PURGING
- X'0038'  PROGRAM_ERROR_TRUNC
- X'003C'  SERVICE_ERROR_NO_TRUNC
SERVICE_ERROR_PURGING
X'0040'

SERVICE_ERROR_TRUNC
X'0044'

USER_ERROR_CODE_RECEIVED
X'005C'

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SENSE
The field in the RPL extension that returns a 32-bit sense code. It is labeled RPL6SNSI in the RPL extension. This field has meaning only if the RCPRI field is set to a nonzero value. If the session for the conversation was deactivated, this code explains why. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM.

SIGDATA
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

- Hex 00010001 indicates a REQUEST_TO_SEND notification has been received from the remote application program.

Note: The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

SIGRCV
The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.
Note: The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFWMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off).

YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied data buffer or buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFWMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
These changes are applicable when RCPRI indicates OK.

For half-duplex conversations, the conversation state is SEND after successful completion.

For full-duplex conversations, no conversation state changes occur.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return
The return codes that can be returned depend on the state of the conversation at the time this APPCCMD is issued.

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in SEND state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0002'</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0003'</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0004'</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0005'</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000C'</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'000D'</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X'0014'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'0018'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0022'</td>
<td>PARAMETER_ERROR—UNEXPECTED VECTOR PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0040'</td>
<td>X'0000'</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USER_ERROR_CODE_RECEIVED—WITHOUT_NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in SEND/RECEIVE, SEND_ONLY, or PENDING_SEND/RECEIVE_log state, the following values can be returned:
### RCPRI RCSEC Meaning

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED VECTOR_PROVIDED_ON_APPCCMD</td>
<td></td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
<td></td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
<td></td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
<td></td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
<td></td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
<td></td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR DEALLOCATE_ABEND</td>
<td></td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
<td></td>
</tr>
<tr>
<td>X'0098'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE_WHILE_SENDING_DATA</td>
<td></td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
<td></td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
<td></td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
<td></td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_PROGRAM</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_SERVICE</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_TIME</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED—ALLOCATE_ERROR</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED—UNKNOWN_ERROR_CODE</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_RETRY</td>
<td></td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_NO_RETRY</td>
<td></td>
</tr>
</tbody>
</table>

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in RECEIVE, PEND_SEND, or PEND_RCV_LOG state, the following values can be returned:

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'0003'</td>
<td>INVALID_LL</td>
<td></td>
</tr>
<tr>
<td>X'0002'</td>
<td>X'000B'</td>
<td>INCOMPLETE_GDS_VARIABLE_SUPPLIED</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
<td></td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
<td></td>
</tr>
</tbody>
</table>
### RCPRI  |  RCSEC  | Meaning
--- | --- | ---
X'002C'  |  X'0011'  | PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING
X'002C'  |  X'001F'  | PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC
X'002C'  |  X'0032'  | PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD
X'0048'  |  X'0000'  | RESOURCE_FAILURE_NO_RETRY
X'004C'  |  X'0000'  | RESOURCE_FAILURE_RETRY
X'0050'  |  X'0000'  | STATE_ERROR
X'0070'  |  X'0000'  | TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE
X'0078'  |  X'0000'  | VTAM_INACTIVE_FOR_YOUR_ACB
X'007C'  |  X'0000'  | REQUEST_ABORTED
X'0080'  |  X'0000'  | DEALLOCATE_NORMAL
X'0088'  |  X'0000'  | CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND
X'0090'  |  X'0000'  | APPLICATION_NOT_APPC_CAPABLE
X'0098'  |  X'0000'  | STORAGE_SHORTAGE_WHILE_SENDING_DATA
X'00A0'  |  X'0002'  | REQUEST_NOT_ALLOWED—REQUEST_BLOCKED
X'00A8'  |  X'0000'  | ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION
X'00A8'  |  X'0001'  | ENVIRONMENT_ERROR—SUSPEND_FAILURE
X'00A8'  |  X'0002'  | ENVIRONMENT_ERROR—RESUME_FAILURE

If APPCCMD CONTROL=SEND, QUALIFY=ERROR is issued in RECEIVE_CONFIRM, RECEIVE_CONFIRM_SEND, or RECEIVE_CONFIRM_DEALLOCATE state, the following values can be returned:

### RCPRI  |  RCSEC  | Meaning
--- | --- | ---
X'0000'  |  X'0000'  | OK
X'002C'  |  X'0002'  | PARAMETER_ERROR—INVALID_CONVERSATION_ID
X'002C'  |  X'0003'  | INVALID_LL
X'002C'  |  X'000B'  | INCOMPLETE_GDS_VARIABLE_SUPPLIED
X'002C'  |  X'000C'  | PARAMETER_ERROR—ZERO_EXIT_FIELD
X'002C'  |  X'000D'  | PARAMETER_ERROR—ZERO_ECB_FIELD
X'002C'  |  X'000E'  | PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE
X'002C'  |  X'000F'  | PARAMETER_ERROR—CONTROL_BLOCK_INVALID
X'002C'  |  X'0010'  | PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH
X'002C'  |  X'0011'  | PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING
X'002C'  |  X'001F'  | PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC
X'0048'  |  X'0000'  | RESOURCE_FAILURE_NO_RETRY
X'004C'  |  X'0000'  | RESOURCE_FAILURE_RETRY
X'0050'  |  X'0000'  | STATE_ERROR
X'0070'  |  X'0000'  | TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE
X'0078'  |  X'0000'  | VTAM_INACTIVE_FOR_YOUR_ACB
X'007C'  |  X'0000'  | REQUEST_ABORTED
X'0088'  |  X'0000'  | CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND
X'0090'  |  X'0000'  | APPLICATION_NOT_APPC_CAPABLE
X'0098'  |  X'0000'  | STORAGE_SHORTAGE_WHILE_SENDING_DATA
X'00A0'  |  X'0002'  | REQUEST_NOT_ALLOWED—REQUEST_BLOCKED
X'00A8'  |  X'0000'  | ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION
X'00A8'  |  X'0001'  | ENVIRONMENT_ERROR—SUSPEND_FAILURE
APPCCMD CONTROL=SEND, QUALIFY=FLUSH

Purpose

This macroinstruction flushes the VTAM SEND buffer associated with the specified conversation.

Usage

This macroinstruction is useful for optimizing processing between the application program and its partner LU. VTAM normally buffers the data from consecutive SEND macroinstructions until it has enough data for transmission. With this macroinstruction, the application program causes VTAM to transmit the data immediately.

VTAM flushes the buffer only when there is something in it. Issuing this macroinstruction when the SEND buffer is empty does not cause anything to flow to the partner LU.

For half-duplex conversations, VTAM buffers function management headers (FMH-5 and FMH-7). The FLUSH macroinstruction may be used to ensure that the headers are sent to the partner LU immediately.

Issuing an APPCCMD CONTROL=SEND, QUALIFY=FLUSH on a full-duplex conversation may cause the early completion of an APPCCMD CONTROL=RECEIVE, FILL=BUFF for the partner transaction program.

This macroinstruction corresponds to the FLUSH verb described in the LU 6.2 architecture.

Context

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- PENDING_SEND

For full-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND/RECEIVE
- SEND_ONLY
- PENDING_SEND/RECEIVE_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

Chapter 1. LU 6.2 macroinstruction syntax and operands 489
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\( AAREA = rpl\_extension\_address\_field \)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\( ACB = acb\_address\_field \)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

\( BRANCH \)

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY I ANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I SPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I SPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I SPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY I ANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC I SPEC macroinstruction.

**CONMODE=SAME**

Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID=32-bit_resource_id_field**

**CONVID=(32-bit_resource_id_register)**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**CONXMOD**

Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

**CONXMOD=CA**

Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXP, QUALIFY=SPEC I SPEC or APPCCMD CONTROL=RCVEXP, QUALIFY=ANY I ANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource (that processes on the SEND/RECEIVE queue if the conversation is half-duplex or on the SEND queue if the conversation is full-duplex) until the command has completed. The exceptions to this are the APPCCMD CONTROL=REJECT, QUALIFY=CONV and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. The application can issue APPCCMDs against the same conversation resource that processes on the RECEIVE (if the conversation is full-duplex), EXPEDITED
SEND, EXPEDITED RECEIVE, and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing the APPCCMD on a full-duplex conversation.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

\[\text{RPL=rlpl_address_field}\]

\[\text{RPL=(rlpl_address_register)}\]

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01' SEND
- X'02' RECEIVE
- X'03' RECEIVE_CONFIRM
- X'04' RECEIVE_CONFIRM_SEND
- X'05' RECEIVED_CONFIRM_DEALLOCATE
- X'07' PENDING_END_CONVERSATION_LOG
- X'08' END_CONVERSATION
- X'09' PENDING_SEND
- X'0A' PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80' FDX_RESET
- X'81' SEND/RECEIVE
- X'82' SEND_ONLY
- X'83' RECEIVE_ONLY
- X'84' PENDING_SEND/RECEIVE_LOG
- X'85' PENDING_RECEIVE-ONLY_LOG
- X'86' PENDING_RESET_LOG
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
No state changes are associated with this macroinstruction.
Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>ERROR_INDICATION_RECEIVED—DEALLOCATE_ABEND_TIME</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>ERROR_INDICATION_RECEIVED—ALLOCATION_ERROR</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>ERROR_INDICATION_RECEIVED—UNKNOWN_ERROR_CODE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>ERROR_INDICATION_RECEIVED—RESOURCE FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_RequestedFunction</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SEND, QUALIFY=RQSEND**

**Purpose**

This macroinstruction notifies the partner LU on a half-duplex conversation that the local application program is requesting to enter SEND state. The conversation is changed to SEND state when the local application program subsequently receives a SEND indication from the remote application program.

**Usage**

This macroinstruction can only be used on half-duplex conversations.

When this macroinstruction is issued, VTAM sends a SIGNAL RU to the partner application program to indicate that the local LU is requesting to enter SEND state. The signal code and signal extension fields of the SIGNAL RU carry X'00010001'.

When the partner application receives the REQUEST_TO_SEND notification, it can enter RECEIVE state. When an APPCCMD CONTROL=RECEIVE macroinstruction
completes and the SEND indicator is on in the WHATRCV field, the local application program is informed that it is in SEND state.

This macroinstruction can be issued while other macroinstructions are outstanding. However, if this macroinstruction is issued while an APPCCMD CONTROL=SEND, QUALIFY=RQSEND or an APPCCMD CONTROL=SENDEXPD macroinstruction is outstanding, it completes with return codes that indicate PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING.

This macroinstruction corresponds to the REQUEST_TO_SEND verb described in the LU 6.2 architecture.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- SEND
- PENDING_SEND

This macroinstruction is not allowed on full-duplex conversations.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
APPCCMD

| CONTROL | SEND | QUALIFY | RQSEND |

| RPL | rpl_address_field | rpl_address_register |

| AAREA | rpl_extension_address_field | rpl_extension_address_register |

| ACB | acb_address_field | acb_address_register |

| BRANCH | NO | YES |

| CONMODE | BUFFCA | CS | LLCA | SAME |

| CONVID | 32-bit_resource_id_field | 32-bit_resource_id_register |

| CONXMOD | CA | CS | SAME |

| ECB | INTERNAL | ecb_address_field | ecb_address_register |

| EXIT | exit_routine_address_field | exit_routine_address_register |
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**

```plaintext
AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)
```

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**

```plaintext
ACB=acb_address_field
ACB=(acb_address_register)
```

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs
running in problem state (non-supervisor state) under a TCB,
BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is
processed in this manner automatically, regardless of the actual setting of
the BRANCH field.

CONMODE
Specifies the mode for receiving normal information upon completion of the
APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode.
It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY
can be used to receive data and that the application program is to receive
data independently of the logical-record format of the data. BUFFCA
corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It
indicates that only APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC can be used to receive data on this conversation.
When the application program issues APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be
received in terms of the logical-record format of the data or independently
of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-
any mode. It indicates that APPCCMD CONTROL=RECEIVE,
QUALIFY=ANY | IANY can be used to receive data on this conversation
and that the application program is to receive data in terms of the
logical-record format of the data. LLCA corresponds to FILL=LL on the
APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC
macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain
unchanged.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD
in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the
APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received by either a specific-type
macroinstruction or an any-type macroinstruction, such as, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD
CONTROL=RCVEXPD, QUALIFY=ANY | IANY.
CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a
state that expedited information can be received only by a specific-type
macroinstruction, such as, APPCCMD CONTROL=RCVEXPD,
QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
Specifies that the conversation mode for expedited information is to remain
unchanged at the completion of this macroinstruction.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPPCCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program
immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the
scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

When the application program regains control after issuing an APPCCMD
asynchronously, it is prevented from issuing another APPCCMD against
the same conversation resource that processes on the EXPEDITED SEND
queue until the command has completed. The application can issue
APPCCMDs against the same conversation resource that processes on the
SEND/RECEIVE if the conversation is half-duplex, or the SEND and
RECEIVE queues if the conversation is full-duplex, and the EXPEDITED
RECEIVE and TESTSTAT queues. For more information about conversation
queues, refer to z/OS Communications Server: SNA Programmer’s LU 6.2
Guide
The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**RPL=**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- X'01'  SEND
- X'02'  RECEIVE
- X'03'  RECEIVE_CONFIRM
- X'04'  RECEIVE_CONFIRM_SEND
- X'05'  RECEIVE_CONFIRM_DEALLOCATE
- X'06'  PENDING_DEALLOCATE
- X'07'  PENDING_END_CONVERSATION_LOG
- X'08'  END_CONVERSATION
- X'09'  PENDING_SEND
- X'0A'  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can have the following values:

- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
- X'82'  SEND_ONLY
- X'83'  RECEIVE_ONLY
- X'84'  PENDING_SEND/RECEIVE_LOG
- X'85'  PENDING_RECEIVE-ONLY_LOG
- X'86'  PENDING_RESET_LOG

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.
FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes
No state changes are associated with this macroinstruction.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.
### APPCCMD CONTROL=SENDEXPD, QUALIFY=DATA

#### Purpose
This macroinstruction sends expedited data to a partner LU over a full-duplex or a half-duplex conversation established on a full-duplex-capable session. If the session is not full-duplex capable, an RCPRI, RCSEC combination of X'00A0', X'0001', REQUEST_NOT_ALLOWED—LU_PAIR_DOES_NOT_SUPPORT_SENDING_EXPEDITED_DATA is returned to the application.

#### Usage
The amount of expedited data specified by the application should be in the range of 1–86 bytes. If the length of the expedited data is outside of this range, an RCPRI, RCSEC combination of X'002C', X'0002', PARAMETER_ERROR—INVALID_EXPEDITED_DATA_LENGTH is returned to the application.

This macroinstruction will be posted complete immediately without waiting for a response from the partner LU. A response will not be sent by the partner until the expedited data has been received by the partner application.

If the conversation ends before the macroinstruction has a chance to process, RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.
If this macroinstruction is issued while another APPCCMD CONTROL=SENDEXPD macroinstruction or an APPCCMD CONTROL=SEND, QUALIFY=RQSEND macroinstruction is currently outstanding for the specified conversation, an RCRPI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application.

If the EXPEDITED SEND queue has been prohibited, then an RCRPI, RCSEC combination of X'00A0', X'0002', REQUEST_NOT_ALLOWED—REQUEST_BLOCKED, is returned to the application.

An RCRPI, RCSEC combination of X'0050', X'0000', STATE_ERROR, will be returned when the macroinstruction is issued in PENDING_DEALLOCATE state.

If the macroinstruction is issued and the response to a previously issued SENDEXPD request has not been received, then an RCPRI, RCSEC combination of X'00A0', X'0005', REQUEST_NOT_ALLOWED—RSP_HAS_NOT_BEEN_RECEIVED_ FOR_A_PREVIOUS_SENDEXPD_REQUEST is returned to the application.

This macroinstruction will always cause a flow.

This macroinstruction corresponds to the SEND_EXPEDITED_DATA verb described in the LU 6.2 architecture.

**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- SEND
- RECEIVE
- RECEIVE_CONFIRM
- RECEIVE_CONFIRM_SEND
- RECEIVE_CONFIRM_DEALLOCATE
- PENDING_END_CONV_LOG
- PENDING_SEND
- PENDING_RECEIVE_LOG

For full-duplex conversation, this macroinstruction can be issued form the following conversations states:

- SEND/RECEIVE
- SEND_ONLY
- RECEIVE_ONLY
- PENDING_SEND/RECEIVE_LOG
- PENDING_RECEIVE-ONLY_LOG
- PENDING_RESET_LOG

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**
APPCCMD CONTROL = SENDEXPD , QUALIFY = DATA

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

AREA = data_area_or_buffer_list_address_field

BRANCH = NO YES

CONMODE = BUFFCA CS LLCA SAME

CONVID = 32-bit_resource_id_field

CONXMOD = CA CS SAME

ECB = INTERNAL

EXIT = exit_routine_address_field
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=

**AAREA**=

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=

**ACB**=

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
**AREA**=

Specifies the address of a data buffer or buffer list. If OPTCD=NBUFLST, AREA specifies the address of a data area containing the data to be sent. If OPTCD=BUFLST, AREA specifies the address of a buffer list that in turn points to the data to be sent. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONMODE**

Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**

Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=CS**

Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

**CONMODE=LLCA**

 Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC macroinstruction.

**CONMODE=SAME**

 Specifies that the continuation mode of the conversation is to remain unchanged.

**CONVID**=

32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
    Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
    Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
    Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
    Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

CONXMOD=SAME
    Specifies that the conversation mode for expedited information is to remain unchanged at the completion of this macroinstruction.

ECB
    Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
    Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
    Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
    Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
    Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
    Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
    Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the EXPEDITED SEND queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex, and the EXPEDITED RECEIVE and TESTSTAT queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=BUFFLST
Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. The indicator resides within the RPLOPT6 field of the RPL.

If OPTCD=BUFFLST is chosen, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The RECLEN field of the RPL specifies a buffer list length that is a nonzero multiple of 16 bytes. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries.

OPTCD=NBUFFLST
Specifies that the data supplied by the application program is contained within a single buffer area. The AREA field specifies the address of the buffer and the RECLEN field specifies the length of the buffer. The indicator resides within the RPLOPT6 field of the RPL.

RECLEN=data_length
RECLEN=(data_length_register)
Specifies the length of the data to be sent or the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

• If OPTCD=NBUFFLST, RECLEN specifies the number of bytes of data to be sent from the buffer area specified by AREA.
• If OPTCD=BUFLST, RECLEN specifies the length of the buffer list that in
turn points to the data to be sent. The RECLEN specifies a buffer list length
that is a nonzero multiple of 16 bytes. (Buffer list entries consist of 16 bytes.)

\[ \text{RPL} = \text{rpl\_address\_field} \]
\[ \text{RPL} = \{ \text{rpl\_address\_register} \} \]

Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation.
This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

\[ \text{X'01'} \quad \text{SEND} \]
\[ \text{X'02'} \quad \text{RECEIVE} \]
\[ \text{X'03'} \quad \text{RECEIVE\_CONFIRM} \]
\[ \text{X'04'} \quad \text{RECEIVE\_CONFIRM\_SEND} \]
\[ \text{X'05'} \quad \text{RECEIVE\_CONFIRM\_DEALLOCATE} \]
\[ \text{X'06'} \quad \text{PENDING\_DEALLOCATE} \]
\[ \text{X'07'} \quad \text{PENDING\_END\_CONVERSATION\_LOG} \]
\[ \text{X'08'} \quad \text{END\_CONVERSATION} \]
\[ \text{X'09'} \quad \text{PENDING\_SEND} \]
\[ \text{X'0A'} \quad \text{PENDING\_RECEIVE\_LOG} \]

For full-duplex conversations, this field can have the following values:

\[ \text{X'80'} \quad \text{FDX\_RESET} \]
\[ \text{X'81'} \quad \text{SEND/RECEIVE} \]
\[ \text{X'82'} \quad \text{SEND\_ONLY} \]
\[ \text{X'83'} \quad \text{RECEIVE\_ONLY} \]
\[ \text{X'84'} \quad \text{PENDING\_SEND/RECEIVE\_LOG} \]
\[ \text{X'85'} \quad \text{PENDING\_RECEIVE\_ONLY\_LOG} \]
\[ \text{X'86'} \quad \text{PENDING\_RESET\_LOG} \]

**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be
received. This field has meaning only when EXPDRCV=YES. This field is
labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be
received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is
returned to the application program. It is labeled RPLFDB2 in the RPL.
**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SIGDATA**
The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X'00010001' indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**SIGRCV**
The field in the RPL extension that returns an indication of whether an application program's partner has requested permission to send. This field is labeled RPL6RSIG in the RPL extension. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture. The indication is either YES or NO (RPL6RSIG bit set on or off).

**YES (B'1')**
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.
NO (B'0')

No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

USERFLD

Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_For_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002C'</td>
<td>PARAMETER_ERROR—INVALID_EXPEDITED_DATA_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0001'</td>
<td>REQUEST_NOT_ALLOWED—LU_PAIR_DOES_NOT_SUPPORT_SENDING_EXPEDITED_DATA</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0005'</td>
<td>REQUEST_NOT_ALLOWED—RSP_TO_PREVIOUS_REQUEST_NOT_RECEIVED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SENDFMH5, QUALIFY=NULL

Purpose

This macroinstruction accepts and sends an FMH-5 for a conversation reserved by the APPCCMD CONTROL=PREALLOC macroinstruction.

Usage

This macroinstruction completes the allocation of a conversation begun by a previous APPCCMD CONTROL=PREALLOC. VTAM does not activate any additional session between the application program and its partner LU as a result of this command.

The APPCCMD CONTROL=SEND FMH5 macroinstruction does not return any vectors to the application in the vector area. For conversations on half-duplex-capable sessions, the FMH-5 is stored in the SEND buffer. For conversations on full-duplex-capable sessions, the FMH-5 is flushed immediately.

Context

This macroinstruction can only be issued from the PENDING_ALLOCATE conversation state.

Syntax

```
APPCCMD CONTROL=SENDFMH5 QUALIFY=NULL

RPL=rpl_address_field

AREA=rpl_extension_address_field

ACB=qcb_address_field

AREA=fmh-5_and_optional_pip_gds_variable_address_field

BRANCH=NO

CONVID=32-bit_resource_id_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field
AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field
ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction
programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=**

Specifies the address of a data buffer or buffer list. If OPTCD=NBUFLST, AREA specifies the address of a data area containing the data to be sent. If OPTCD=BUFLST, AREA specifies the address of a buffer list that in turn points to the data to be sent. In either case, the data consists of logical records. VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.) This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**

Specifies that VTAM is to post an internal ECB when the APPPCCMD macroinstruction completes.

**ECB=ecb_address_field**

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=**

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled.
when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOPT11 field of the RPL.

**RECLEN=fmh-5_and_optional_gds_field_length**

**RECLEN=(fhm-5_and_optional_gds_field_length_register)**

Specifies the length of the data within the data area indicated by the AREA field. This field is labeled RPLRLEN in the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of the RPL and RPL extension fields:

**CONSTATE**

The field in the RPL extension that indicates what state the conversation is in. It is labeled RPL6CCST in the RPL extension.

This field can have the following values for half-duplex conversations:

- X'00'  RESET
- X'01'  SEND
- X'08'  END_CONVERSATION

This field can have the following values for full-duplex conversations:

- X'00'  RESET
- X'80'  FDX_RESET
- X'81'  SEND/RECEIVE
FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

SLS
The field in the RPL extension that indicates whether the session was established using session-level LU-LU verification. This field is labeled RPL6SLS in the RPL extension.

YES (B'1')
The session was established using session-level LU-LU verification.

NO (B'0')
The session was not established using session-level LU-LU verification.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD
CONTROL=RCVFHM5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

State changes

After successful completion of this macroinstruction, the conversation state is SEND if issued over a half-duplex session or SEND/RECEIVE if issued over a full-duplex session.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0006'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY LU</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>ALLOCATION_ERROR—ALLOCATION_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0010'</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_VALID_FOR_FULL-DUPLEX</td>
</tr>
<tr>
<td>X'0004'</td>
<td>X'0011'</td>
<td>ALLOCATION_ERROR—LU_PAIR_NOT_SUPPORTING_FDX_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>PARAMETER_ERROR—INCOMPLETE_FMH5_SUPPLIED</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>PARAMETER_ERROR—INVALID_TPN</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0022'</td>
<td>PARAMETER_ERROR—INVALID_CONTROL_OR_QUALIFY_VALUE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0048'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE,_NO_RETRY</td>
</tr>
<tr>
<td>X'004C'</td>
<td>X'0000'</td>
<td>RESOURCE_FAILURE,_RETRY</td>
</tr>
<tr>
<td>X'0050'</td>
<td>X'0000'</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE_OR_RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>HALT_ISSUED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
</tbody>
</table>

APPCCMD CONTROL=SENDRCV, QUALIFY=DATAFLU

Purpose

This macroinstruction provides a dual function; it performs the function of a send, and when the send is complete it automatically performs the function of a receive.

The send portion of this macroinstruction sends data supplied by the application program and any data that is already in the SEND buffer to the partner application. After the send portion of this macroinstruction is successfully completed, the conversation is placed in receive state and the macroinstruction waits for data from the partner.
This macroinstruction can only be issued for half-duplex conversations.

**Usage**

This macroinstruction combines the functions of two macroinstructions: APPCCMD CONTROL=SEND, QUALIFY=DATAFLU followed by APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC. A buffer list format must be used to allow the application program to specify areas and lengths separately for both the send and receive portions of this macroinstruction. For a description of how to use both non-extended buffer list entries and extended buffer list entries refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.wss?uid=swg27043446).

When this macroinstruction is issued, VTAM places data in the SEND buffer of the conversation that is specified by the CONVID parameter. VTAM determines the location of the data to be sent from the buffer list entries specified by the AREA parameter. VTAM sends all data in the SEND buffer to the partner LU.

When the send portion of this macroinstruction completes, there is no data ready to be received on the conversation; therefore, VTAM queues the macroinstruction until data arrives. This macroinstruction has just turned the flow around and the SEND indication is still enroute to the partner. After the partner receives the data just sent and also the SEND indication, it may then send data back to the local application. When enough of this data is received by VTAM to satisfy the receive portion of this macroinstruction the macroinstruction will be completed.

After data is received, VTAM copies any received data from the conversation that is specified by the CONVID parameter to the data area that is specified by the last entry in the buffer list.

When this macroinstruction completes, the BLERECLN field of the last buffer list entry indicates how much data was written to the data area. The WHATRCV field indicates what type of data was received.

The application program can issue this macroinstruction when the conversation is in SEND or PENDING_SEND state. VTAM flushes its SEND buffer, sending all buffered information, along with the SEND indicator, to the partner LU. This changes the conversation to RECEIVE state. VTAM then waits for information to arrive. The remote application program can send data to the local application program after it receives the SEND indication.


**Context**

For half-duplex conversations, this macroinstruction can be issued from the following conversation states:

- **SEND**
- **PENDING_SEND**

For full-duplex conversations, this macroinstruction is not allowed.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.
Syntax

```plaintext
APPCCMD, CONTROL=SENRcv, QUALIFY=DATAFLU

RPL=rpl_address_field (rpl_address_register)

AAREA=rpl_extension_address_field (rpl_extension_address_register)

ACB=acb_address_field (acb_address_register)

AREA=buffer_list_address_field (buffer_list_address_register)

BRANCH=NO, YES

CD=DEFER, IMMED

CONMODE=BUFFCA, CS, LLCA, SAME

CONVID=32-bit_resource_id_field (32-bit_resource_id_register)

CONXMOD=CA, CS
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. Refer to "Coding Default Values" in z/OS Communications Server: SNA Programmer's LU 6.2 Guide for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[\text{AREA} = \text{rpl\_extension\_address\_field}\]
**AAREA=rpl_extension_address_register**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=acb_address_field**

**ACB=acb_address_register**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=buffer_list_address_field**

**AREA=buffer_list_address_register**

Specifies the address of a list of buffer entries.

- If OPTCD=BUFFLST, all entries in the buffer list except the last specify the address and length of data to be sent. The data consists of logical records. VTAM tracks the logical records supplied by the application program, examining the logical-record length (LL) field associated with each logical record. (It does not inspect the data portion of the logical record.) The last entry specifies the address and length of an area in which data is to be received. When this macroinstruction completes, another field in this last entry contains the number of bytes placed in this receive buffer by VTAM.

Both the send and receive buffers are described using the ISTBLENT DSECT. For a more detailed description of how to use buffer list entries refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

- If OPTCD=XBUFLST, all entries in the buffer list except the last specify the address and length of data to be sent. The send data resides in CSM buffers. VTAM does not track logical records supplied by the application. Like OPTCD=BUFFLST, the last entry specifies the address and length of an area in which data is to be received. When this macroinstruction completes, another field in this last entry contains the number of bytes placed in this receive buffer by VTAM. This receive buffer is not a CSM buffer.

The send buffers are described using the ISTBLXEN DSECT and the receive buffer is described using the ISTBLENT DSECT. For a more detailed description of how to use extended buffer list entries, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide.

When the application program receives information other than data, as indicated by the WHATRCV field of the RPL extension, nothing is placed in this data area. This field is labeled RPLAREA in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

**BRANCH=NO**

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.
BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CD
This parameter controls subsequent actions if a SEND indication is received in the WHATRCV field on the receive portion of this macroinstruction. For this to happen, the send portion of this macroinstruction transmitted the SEND indication to the partner, as is normally done on this macroinstruction, which in turn returned it. The SEND indication is being reported back to the local application on the receive portion of this macroinstruction. In particular CD specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.

CD=DEFER
Specifies that the conversation state will be in PEND_SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CD=IMMED
Specifies that the conversation state will be in SEND state when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

CONMODE
Specifies the mode for receiving normal information upon completion of the APPCCMD. This field is labeled RPL6CMOD in the RPL extension.

CONMODE=BUFFCA
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=CS
Specifies that the conversation is to be placed in continue-specific mode. It indicates that only APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC can be used to receive data on this conversation. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data or independently of the logical-record format of the data.

CONMODE=LLCA
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that APPCCMD CONTROL=RECEIVE, QUALIFY=ANY | IANY can be used to receive data on this conversation and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC | ISPEC macroinstruction.

CONMODE=SAME
Specifies that the continuation mode of the conversation is to remain unchanged.
CONVID=32-bit_resource_id_field
CONVID=32-bit_resource_id_register
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

CONXMOD
Specifies the mode for receiving expedited information upon completion of the APPCCMD. This field is labeled RPL6CXMD in the RPL extension.

CONXMOD=CA
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received by either a specific-type macroinstruction or an any-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC | ISPEC or APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY | IANY.

CONXMOD=CS
Specifies that the mode for expedited information is to be put in such a state that expedited information can be received only by a specific-type macroinstruction, such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEOPTI field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=exit_routine_address_register
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

FILL
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter applies only to the receive portion of this macroinstruction and corresponds to FILL=LL | BUFFER described in the LU 6.2 architecture. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF
Specifies the application program is to receive data independently of its logical-record format. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

FILL=LL
Specifies the application program is to receive one logical record, or
whatever portion of the logical record is available. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

In general, when the application program regains control after issuing an asynchronous APPCCMD, it is prevented from issuing another APPCCMD against the same conversation resource until the prior asynchronous command has completed. The exceptions to this are the APPCCMD CONTROL=SEND, QUALIFY=RQSEND; APPCCMD CONTROL=REJECT; and the abnormal termination APPCCMD CONTROL=DEALLOC|DEALLOCQ macroinstructions. (For more information, refer to the descriptions of the particular macroinstructions). The application program is allowed to issue APPCCMDs against other conversations.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=BUFFLST**

Specifies that the data supplied by the application program is contained within multiple buffers. This option allows the application program to provide data from discontiguous buffer areas. RU boundaries are independent of the buffer boundaries. VTAM creates RUs based upon the maximum SEND RU size regardless of whether the data is taken from one buffer, part of a buffer, or multiple buffers. Logical records are also independent of the buffer boundaries. This field is labeled RPLBUFL in the RPL. When OPTCD=BUFFLST, the AREA field of the RPL points to a buffer list that is a contiguous set of 16-byte control blocks, called buffer list entries. (Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for a detailed description of these buffer list entries.) The buffer list created by the application must have at least two entries. One or more entries must be send buffer list entries. This specifies the layout of the send buffers. The last entry must be a special receive entry that points to the receive buffer and indicates the area length. Both the send buffer(s) and the receive buffer are described by the ISTBLENT macroinstruction. The following list explains the layout of the receive entry:
The first 4 bytes are reserved and should be set to 0 when the macroinstruction is issued. This field will be used to return the amount of data received to the application.

The second 4 bytes contain the length of the receive buffer. This is similar to the AREALEN field of an RPL that accompanies a receive type macroinstruction.

The third 4 bytes contain the address of a receive buffer. This is similar to the AREA field that accompanies a receive type macroinstruction.

The fourth 4 bytes must contain zeros (the send length field).

**OPTCD=XBUFLST**

Specifies that the data supplied by the application program is contained within an extended buffer list. The AREA field of the RPL points to an extended buffer list that contains a contiguous set of 48-byte send extended buffer list entries followed immediately by a 16-byte receive buffer entry. Once OPTCD=XBUFLST has been issued, VTAM no longer tracks logical records for the duration of the conversation.

The indicator is labeled RPLXBFL and resides within the RPLOPT6 field of the RPL.

Each send entry in the extended buffer list can point to any displacement into a CSM buffer and is described by ISTBLXEN. VTAM uses the CSM token rather than the storage address to track a CSM buffer. A CSM token cannot be repeated in an extended buffer list. If multiple areas of a CSM buffer are to be used on one APPCCMD, the CSM buffer must first be segmented by using the IVTCSM REQUEST=ASSIGN_BUFFER macroinstruction. This macroinstruction returns a new token for each CSM buffer segment. The new tokens should then be used on the APPCCMD. VTAM treats the CSM storage associated with the new CSM tokens as separate CSM buffers.

The last entry describes the receive buffer. This buffer is not a CSM buffer. It is described using the ISTBLENT DSECT.

**RECLEN**

Specifies the length of the buffer list containing the data to be sent. This field is labeled RPLRLEN in the RPL.

• If OPTCD=BUFLST, the length of the buffer list is determined by the product of 16 and the number of entries, both send and receive. (Each buffer list entry consists of 16 bytes.)
• If OPTCD=XBUFLST, the length of the buffer list is determined by the product of 48 and the number of send entries plus 16 bytes for the receive buffer entry. (Each CSM buffer list entry consists of 48 bytes.)

**RPL**

- **RPL=rpl_address_field**
- **RPL=rpl_address_register**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

The following conversation states are possible:
**EXPDLEN**

The field in the RPL6 that shows the length of the expedited data waiting to be received. This field has meaning only when EXPDRCV=YES. This field is labeled RPL6EXDL in the RPL extension.

**EXPDRCV**

The field in the RPL6 that indicates whether expedited data is waiting to be received. This field is labeled RPL6EXDR in the RPL extension.

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner application programs. The FMH5RCV field continues to be set to YES as long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**LOGRCV**

The field in the RPL extension that returns an indication of whether error log data is expected. The indication is either YES or NO (RPL6RLOG set on or off). This field is labeled RPL6RLOG in the RPL extension.

**YES (B'1')**

An FMH-7 was received that specified that error log data follows. The application program must issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC in order to retrieve the log data. It is the responsibility of the application program to perform an optional receive check after issuing APPCCMD CONTROL=RECEIVE,
QUALIFY=SPEC | ISPEC to determine whether the expected log data was sent by the partner LU. The data must be error log data and it must be in the form of a GDS variable.

LOGRCV=YES only if the RCPRI field of the RPL extension contains one of the following values:

- X'0004' ALLOCATION_ERROR
- X'0014' DEALLOCATE_ABEND_PROGRAM
- X'0018' DEALLOCATE_ABEND_SERVICE
- X'001C' DEALLOCATE_ABEND_TIMER
- X'0030' PROGRAM_ERROR_NO_TRUNC
- X'0034' PROGRAM_ERROR_PURGING
- X'0038' PROGRAM_ERROR_TRUNC
- X'003C' SERVICE_ERROR_NO_TRUNC
- X'0040' SERVICE_ERROR_PURGING
- X'0044' SERVICE_ERROR_TRUNC
- X'0048' RESOURCE_FAILURE_NO_RETRY
- X'005C' USER_ERROR_CODE_RECEIVED

NO (B'0')
Either no error indicator was received or an error indicator was received but indicated that no log data follows.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field used on the SEND portion of this macroinstruction, similar to a send with a buffer list. It is used to calculate the number of entries in the buffer list.
For the receive portion of this macroinstruction, VTAM calculates a RECLEN value but does not overlay the RECLEN provided by the application in the RPL. Instead, VTAM returns the receive RECLEN in the first 4 bytes of the last entry in the buffer list (BFERECLN), which is the entry used to describe the receive area.

**RPLXSRV**

A field in the RPL that is set if VTAM accepts all the CSM buffers from the application on an HPDT request. If the APPCCMD completes unsuccessfully and the completion status is stored in the RPL, the application must examine RPLXSRV. Some TPEND exits are driven where the RPL is canceled and not posted complete. It is the application’s responsibility to examine the RPLXSRV bit and determine if CSM storage needs to be freed.

For more information about application recovery options when RPLXSRV is not set, refer to [z/OS Communications Server: SNA Programmer’s LU 6.2 Guide](#).

The RPLXSRV indicator is contained in the RPLEXTDS field in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**SENSE**

The field in the RPL extension that returns a 32-bit sense code. This field has meaning only if the RCPRI field is set to a nonzero value. The sense code also can be set when the return code is RESOURCE_FAILURE_NO_RETRY. This code indicates why the session for the conversation was deactivated. Not all RCPRI values have sense data associated with them. If the RCPRI field indicates USER_ERROR_CODE_RECEIVED, the SENSE field contains an FMH-7 sense code that was not interpreted by VTAM. It is labeled RPL6NSI in the RPL extension.

**SIGDATA**

The field in the RPL extension in which the signal code and signal extension fields of a received SIGNAL RU are returned to the application program. This field has meaning only when SIGRCV=YES. This field is labeled RPL6SGNL in the RPL extension.

X’00010001’ indicates a REQUEST_TO_SEND notification has been received from the remote application program.

**Note:** The SIGDATA field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

**SIGRCV**

The field in the RPL extension that returns an indication of whether an application program’s partner has requested permission to send. This field and the SIGDATA field correspond to the REQUEST_TO_SEND_RECEIVED parameter described in the LU 6.2 architecture.

**Note:** The SIGRCV field is reserved if, on the macroinstruction that initiated the conversation (APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5), the application specified RTSRTRN=EXPD.

The indication is either YES or NO (RPL6RSIG bit set on or off). It is labeled RPL6RSIG.
YES (B'1')
A SIGNAL RU has been received from the partner LU. The values carried in the signal code and signal extension fields of the SIGNAL RU are returned to the application program in the SIGDATA field.

NO (B'0')
No SIGNAL RU has been received from the partner LU. When SIGRCV=NO, the SIGDATA field contains no meaningful information.

STSHBF
The field in the RPL extension that returns the address of the current buffer. It is used with STSHDS to give the current position (address and displacement) in the application-supplied buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STBF in the RPL extension.

STSHDS
The field in the RPL extension that returns the displacement into the current buffer. It is used with STSHBF to give the current position (address and displacement) in the application-supplied buffer list (the area pointed to by the AREA field of the RPL) when a temporary storage shortage occurs while data is being sent. All data prior to this buffer has been sent. This field is labeled RPL6STDS in the RPL extension.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by a remote application program). This field is labeled RPL6USR in the RPL extension.

Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for more information.

WHATRCV
The field in the RPL extension that returns a mask specifying what the application program received. It is labeled RPL6WHAT. The application program should examine this WHATRCV mask only when RCPRI indicates X'0000'. Otherwise, WHATRCV has no meaning.

When RCPRI indicates OK, one or more bits in the mask can be turned on (contain a value of B'1') to indicate the type of information that has been received. For instance, if the application program is being passed both conversation data and a request for confirmation, both the DATA and CONFIRM bits will be set on; the other bits will be set off.

The 2-byte WHATRCV mask has the following format.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
<th>RPL6RCV2 Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DATA</td>
<td>0</td>
<td>PARTIAL_PS_HEADER</td>
</tr>
<tr>
<td>1</td>
<td>DATACOMPLETE</td>
<td>1–7</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>DATA_INCOMPLETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SEND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CONFIRM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DEALLOCATE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RPL6RCV1 | RPL6RCV2
---|---
Bit | Meaning | Bit | Meaning
6 | LOG_DATA | | |
7 | PS_HEADER | | |

For example, a WHATRCV value indicating that DATA has been received would be represented by X’8000’. Refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide for a discussion of the meaning of this field.

State changes

See the description of the WHATRCV mask for state changes when RCPRI indicates OK.

See Chapter 2, “Return codes,” on page 591 for state changes associated with other return codes.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0002’</td>
<td>ALLOCATION_ERROR—CONVERSATION_TYPE_MISMATCH</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0003’</td>
<td>ALLOCATION_ERROR—PIP_NOT_ALLOWED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0004’</td>
<td>ALLOCATION_ERROR—PIP_NOT_SPECIFIED_CORRECTLY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0005’</td>
<td>ALLOCATION_ERROR—SECURITY_NOT_VALID</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0007’</td>
<td>ALLOCATION_ERROR—SYNC_LEVEL_NOT_SUPPORTED_BY_PROGRAM</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0008’</td>
<td>ALLOCATION_ERROR—TPN_NOT_RECOGNIZED</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’0009’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000A’</td>
<td>ALLOCATION_ERROR—TRANS_PGM_NOT_AVAIL_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000B’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_NO_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000C’</td>
<td>ALLOCATION_ERROR—CANNOT_RECONNECT_TRANS_PGM_RETRY</td>
</tr>
<tr>
<td>X’0004’</td>
<td>X’000D’</td>
<td>ALLOCATION_ERROR—RECONNECT_NOT_SUPPORTED_BY_PGM</td>
</tr>
<tr>
<td>X’0014’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_PROGRAM</td>
</tr>
<tr>
<td>X’0018’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_SERVICE</td>
</tr>
<tr>
<td>X’001C’</td>
<td>X’0000’</td>
<td>DEALLOCATE_ABEND_TIMER</td>
</tr>
<tr>
<td>X’0024’</td>
<td>X’0000’</td>
<td>LOGICAL_RECORD_BOUNDARY_ERROR</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0002’</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0003’</td>
<td>PARAMETER_ERROR—INVALID_LL</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000D’</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000E’</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000F’</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0010’</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0011’</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0012’</td>
<td>PARAMETER_ERROR—BUFFER_LIST_LENGTH_INVALID</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’001F’</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON_APPC</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0024’</td>
<td>PARAMETER_ERROR—PS_HEADER_NOT_SUPPLIED</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0025’</td>
<td>PARAMETER_ERROR—PS_HEADER_LENGTH_IS_INSUFFICIENT</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0028’</td>
<td>PARAMETER_ERROR—CRYPTOGRAPHY_NOT_ALLOWED_ON_MODE</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’0031’</td>
<td>PARAMETER_ERROR_SENDRCV_SPECIFIED_WITHOUT_OPTCD=BUFFLST</td>
</tr>
<tr>
<td>RCPRI</td>
<td>RCSEC</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>X’002C′</td>
<td>X’0032′</td>
<td>PARAMETER_ERROR— UNEXPECTED VECTOR PROVIDED ON APPCCMD</td>
</tr>
<tr>
<td>X’0030′</td>
<td>X’0000′</td>
<td>PROGRAM_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X’0034′</td>
<td>X’0000′</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
<tr>
<td>X’0038′</td>
<td>X’0000′</td>
<td>PROGRAM_ERROR_TRUNC</td>
</tr>
<tr>
<td>X’003C′</td>
<td>X’0000′</td>
<td>SERVICE_ERROR_NO_TRUNC</td>
</tr>
<tr>
<td>X’0040′</td>
<td>X’0000′</td>
<td>SERVICE_ERROR_PURGING</td>
</tr>
<tr>
<td>X’0044′</td>
<td>X’0000′</td>
<td>SERVICE_ERROR_TRUNC</td>
</tr>
<tr>
<td>X’0048′</td>
<td>X’0000′</td>
<td>RESOURCE_FAILURE_NO_RETRY</td>
</tr>
<tr>
<td>X’004C′</td>
<td>X’0000′</td>
<td>RESOURCE_FAILURE_RETRY</td>
</tr>
<tr>
<td>X’0050′</td>
<td>X’0000′</td>
<td>STATE_ERROR</td>
</tr>
<tr>
<td>X’005C′</td>
<td>X’0000′</td>
<td>USER_ERROR_CODE_RECEIVED—FOLLOWING_ NEGATIVE_RESPONSE</td>
</tr>
<tr>
<td>X’0070′</td>
<td>X’0000′</td>
<td>TEMPORARY_STORAGE_SHORTAGE OR RESOURCE_SHORTAGE</td>
</tr>
<tr>
<td>X’0078′</td>
<td>X’0000′</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X’007C′</td>
<td>X’0000′</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X’0084′</td>
<td>X’0000′</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X’0088′</td>
<td>X’0000′</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOCATE_ABEND</td>
</tr>
<tr>
<td>X’008C′</td>
<td>X’0000′</td>
<td>PARTNER_COMMMITTED_PROTOCOL_VIOLATION</td>
</tr>
<tr>
<td>X’0090′</td>
<td>X’0000′</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X’0094′</td>
<td>X’0000′</td>
<td>INVALID_CONDITION_FOR_SENDING_DATA</td>
</tr>
<tr>
<td>X’0098′</td>
<td>X’0000′</td>
<td>STORAGE_SHORTAGE WHILE_SENDING_DATA</td>
</tr>
<tr>
<td>X’00A0′</td>
<td>X’0004′</td>
<td>CONTROL/QUALIFY_VALUE_INVALID_FOR_FULL-DUPLICITY_CONVERSATION</td>
</tr>
<tr>
<td>X’00A0′</td>
<td>X’0006′</td>
<td>REQUEST_NOT_ALLOWED—PROGRAM_NOT_AUTHORIZED__FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X’00A8′</td>
<td>X’0000′</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT__FOR_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X’00A8′</td>
<td>X’0001′</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X’00A8′</td>
<td>X’0002′</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
<tr>
<td>X’00B4′</td>
<td>X’0001′</td>
<td>CSM_DETECTED_ERROR— NOT_SPECIFIED</td>
</tr>
<tr>
<td>X’00B4′</td>
<td>X’0002′</td>
<td>CSM_DETECTED_ERROR— INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
<tr>
<td>X’00B4′</td>
<td>X’0003′</td>
<td>CSM_DETECTED_ERROR— INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SETSESS, QUALIFY=RESUME**

**Purpose**

This macroinstruction resumes sending any outgoing normal data that was held because APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND was issued previously on the specified session.

**Usage**

This macroinstruction should be issued to notify VTAM to allow any outbound normal data to flow to the partner if any has been held due to a previously issued APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND command. APPCCMD CONTROL=SETSESS, QUALIFY=RESUME also enables the following items to resume:

- Normal data flow from any conversations matched to the session
- Normal session deactivation
- Session bidding

This macroinstruction indicates to VTAM that the application program (which is supporting a sync point manager) has completed its synchronization processing successfully.
APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued if the application program's synchronization processing was unsuccessful and the application program does not wish to imply by the normal data flow that the sync point completed successfully.

If this macroinstruction is issued and the session has not been suspended, a return code of 0 is received, but no changes are made.

**Context**

This macroinstruction is not conversation-specific and therefore is not conversation-state-driven.

**Syntax**

```plaintext
APPCCMD --CONTROL=SETSESS, QUALIFY=RESUME

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

BRANCH = NO

ECB = INTERNAL

EXIT = exit_routine_address_field

OPTCD = (ASY, SYN, KEEPSRB, NKEEPSRB)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

AAREA=rpl_extension_address_field

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

ACB=acb_address_field

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

BRANCH

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
**BRANCH=NO**
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

**BRANCH=YES**
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**ECB**
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

**ECB=INTERNAL**
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

**ECB=ecb_address_field**
**ECB=(ecb_address_register)**
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. *Event_control_block_address* is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**
**EXIT=(exit_routine_address_register)**
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**
Specifies the following processing options that can be selected for the macroinstruction request:

**OPTCD=SYN**
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=ASY**
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
RPL=rpl_address_field
RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)

Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction, indicates the session to be released. This field is labeled RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)

Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

Following are descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND

Purpose

This macroinstruction specifies that the application program wants VTAM to suspend any outgoing normal data flow on the specified session after the current conversation has been deallocated. APPCCMD CONTROL=SETSESS, QUALIFY=RESUME resumes the outgoing normal flow.

Usage

This macroinstruction should be issued to notify VTAM to not allow outbound flow on the session. It should be issued if the application program (which is supporting a sync point manager) has not completed the synchronization processing needed before the partner can continue its synchronization processing. The application program must issue this command before the conversation supporting the sync point exchange is deallocated to ensure the flow is stopped on the free session.

Suspending the session gives the application program with the sync point manager control of the outbound flow whose subsequent receipt at the partner implies a successful sync point has completed. The partner application program can then continue synchronization cleanup. Further information on the sync point services function is described in the *SNA Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2*.

APPCCMD CONTROL=SETSESS, QUALIFY=RESUME indicates that the application program is ready to resume normal flow because its sync point processing completed successfully. APPCCMD CONTROL=REJECT, QUALIFY=SESSION can be issued if the sync point processing is unsuccessful.

If an application program is executing under persistent LU-LU session support and the application program fails after APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND has been issued and APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has not been issued, VTAM UNBINDs the session and deallocates the conversation on which the synchronization is taking place. In the same situation, VTAM also UNBINDs sync point sessions for which APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEBEG has been issued but neither APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND nor APPCCMD
CONTROL=SETSESS, QUALIFY=RESUME has been issued at the time of the failure.

**Context**

This macroinstruction is not conversation-specific and, therefore, is not driven by the conversation state.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

**Syntax**

```
APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND

RPL=rpl_address_field (rpl_address_register)

AAREA=rpl_extension_address_field (rpl_extension_address_register)

ACB=acb_address_field (acb_address_register)

BRANCH=NO YES

CONVID=32-bit_resource_id_field (32-bit_resource_id_register)

ECB=INTERNAL ecb_address_field (ecb_address_register)

EXIT=exit_routine_address_field (exit_routine_address_register)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

- AAREA=rpl_extension_address_field
- AAREA=(rpl_extension_address_register)
  Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

- ACB=acb_address_field
- ACB=(acb_address_register)
  Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction
programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO

Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES

Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

**CONVID=32-bit_resource_id_field**

Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

**ECB**

Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

ECB=INTERNAL

Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field

ECB=(ecb_address_register)

Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

**EXIT=exit_routine_address_field**

EXIT=(exit_routine_address_register)

Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

**OPTCD**

Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN

Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLEXT1 field of the RPL.
OPTCD=ASY
 Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=KEEPSRB
 Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

OPTCD=NKEEPSRB
 Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
 Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
 Specifies the session to which this macroinstruction applies. The session instance identifier, which was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction, indicates the session to be held. This field is labeled RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
 Specifies the length of the session instance ID. The value specified must be greater than 0 and less than or equal to 8. The session instance ID length was passed to the application program on a previous APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction. This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2
 The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
 The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
 The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**State changes**

There are no state changes associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>PARAMETER_ERROR—SESSION_INSTANCE_IDENTIFIER_AND_CONVERSATION_ID_ARE_MISMATCHED</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>CANCELLED_BY_REJECT_OR_DEALLOC_ABEND</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SETSESS, QUALIFY=SYN CBEG**

**Purpose**

This macroinstruction notifies VTAM that a sync point exchange is beginning.

If an application program is executing under persistent LU-LU session support, persistence must be overridden for a session during the time that a sync point exchange takes place. If the application program fails after APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND has been issued and APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has not been issued, VTAM UNBINDs the session and deallocates the conversation on which the synchronization is taking place. In the same situation, VTAM also UNBINDs sync point sessions for which APPCCMD CONTROL=SETSESS, QUALIFY=SYN CBEG has been issued, but neither APPCCMD CONTROL=SETSESS, QUALIFY=SYN CEND nor APPCCMD CONTROL=SETSESS, QUALIFY=RESUME has been issued at the time of the failure.
Usage

This macroinstruction is issued to notify VTAM that the sync point manager is beginning a synchronization exchange because a SYNCPT is being issued or a TAKE-SYNCPT is being received. To ensure that synchronization protocols are followed, VTAM UNBINDs this session when the application program fails, even though the application program has enabled persistence. The UNBIND permits the LUs to make consistent decisions and ensures continued synchronization between the two LUs. If the data is critical enough to use a synchronization exchange, APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG and APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND should be used. For circumstances for use, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

Context

This macroinstruction is not conversation-specific and, therefore, is not driven by the conversation state. It performs a useful function only for application programs that are using persistent LU-LU sessions. If application programs that have not enabled persistence issue this macroinstruction, a good return code is sent but no action is taken.

This macroinstruction is not allowed for conversations pending deallocation for persistent LU-LU sessions.

Syntax

```
APPCCMD  CONTROL=SETSESS, QUALIFY=SYNCBEG

RPL = rpl_address_field

AAREA = rpl_extension_address_field

ACB = acb_address_field

CONVID = 32-bit_resource_id_field
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

- **AAREA=rpl_extension_address_field**
  - Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.
\textbf{ACB} = \texttt{acb\_address\_field}  \\
\textbf{ACB} = (\texttt{acb\_address\_register})  \\
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDacb in the RPL.

\textbf{BRANCH}  \\
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLExTDS field of the RPL.

\textbf{BRANCH=NO}  \\
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

\textbf{BRANCH=YES}  \\
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

\textbf{CONVID} = \texttt{32-bit\_resource\_id\_field}  \\
\textbf{CONVID} = (\texttt{32-bit\_resource\_id\_register})  \\
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

\textbf{ECB}  \\
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

\textbf{ECB=INTERNAL}  \\
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

\textbf{ECB} = \texttt{ecb\_address\_field}  \\
\textbf{ECB} = (\texttt{ecb\_address\_register})  \\
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. \texttt{Event\_control\_block\_address} is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

\textbf{EXIT} = \texttt{exit\_routine\_address\_field}  \\
\textbf{EXIT} = (\texttt{exit\_routine\_address\_register})  \\
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

\textbf{OPTCD}  \\
Specifies the following processing options that can be selected for the macroinstruction request:
OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program
immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the
scheduling of an exit. The indicator resides within the RPLOPT1 field of
the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM
returns to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does
not return to the application under the same SRB in which VTAM was
invoked. The indicator resides within the RPLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to
be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
Specifies the session to which this macroinstruction applies. The session
instance identifier, which was passed to the application program on a previous
APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5
macroinstruction, indicates the session to be released. This field is labeled
RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
Specifies the length of the session instance ID. The value specified must be
greater than 0 and less than or equal to 8. The session instance ID length was
passed to the application program on a previous APPCCMD
CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction.
This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2
The field in the RPL in which a global VTAM secondary return code is
returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return
code is returned to the application program. This field has meaning only when
RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL
extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary
return code is returned to the application program. This field has meaning...
only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**State changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>INVALID_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>PARAMETER_ERROR—APPCCMD_ISSUED_FOR_NON-APPC</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>PARAMETER_ERROR—INVALID_SESSION_INSTANCE_IDENTIFIER</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>SESSION_INSTANCE_IDENTIFIER_AND_CONVERSATION_IDENTIFIER_MISMATCH</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=SETSESS, QUALIFY=SYNCEND**

**Purpose**

This macroinstruction indicates to VTAM that the sync point exchange has completed.

**Note:** This macroinstruction only has meaning for MVS and VSE applications using persistent sessions. VTAM ignores this macroinstruction if issued from a VM application.

**Usage**

This macroinstruction is issued to notify VTAM that the sync point exchange has completed, whether successful or not, and that VTAM no longer needs to UNBIND sync point sessions during a failure after persistence has been enabled. It is used with APPCCMD CONTROL=SETSESS, QUALIFY=SYNCBEG. For circumstances
for use, refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide

Context

This macroinstruction is not conversation-specific and, therefore, is not driven by conversation state. It performs a useful function only for application programs that are using persistent LU-LU sessions. If application programs that have not enabled persistence issue this macroinstruction, a good return code is sent but no action is taken.

Syntax

APPCCMD — CONTROL — SETSESS, — QUALIFY — SYNCEND

name

APPCCMD — CONTROL — SETSESS, — QUALIFY — SYNCEND

name

RPL — rpl_address_field

rpl_address_register

RPL — rpl_address_field

rpl_address_register

AAREA — rpl_extension_address_field

rpl_extension_address_register

ACB — acb_address_field

acb_address_register

BRANCH — NO

YES

ECB — INTERNAL

ecb_address_field

ecb_address_register

EXIT — exit_routine_address_field

exit_routine_address_register

OPTCD — (ASY)

SYN

KEEPSRB

NKEEPSRB

Chapter 1. LU 6.2 macroinstruction syntax and operands 549
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See  "Coding default values" on page 3 for information on coding operands on the RPL or APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPERSB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl_extension_address_field

**AAREA=(rpl_extension_address_register)**

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field

**ACB=(acb_address_register)**

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPOOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOOPT11 field of the RPL.
RPL=rpl_address_field
RPL=(rpl_address_register)
    Specifies the address of the request parameter list that contains information to
    be used during the processing of the APPCCMD macroinstruction.

SESSID=session_instance_id_field
SESSID=(session_instance_id_register)
    Specifies the session to which this macroinstruction applies. The session
    instance identifier, which was passed to the application program on a previous
    APPCCMD CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5
    macroinstruction, indicates the session to be released. This field is labeled
    RPL6SSID in the RPL extension.

SESSIDL=session_instance_id_length
SESSIDL=(session_instance_id_length_register)
    Specifies the length of the session instance ID. The value specified must be
    greater than 0 and less than or equal to 8. The session instance ID length was
    passed to the application program on a previous APPCCMD
    CONTROL=ALLOC or APPCCMD CONTROL=RCVFMH5 macroinstruction.
    This field is labeled RPL6SIDL in the RPL extension.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

FDB2
    The field in the RPL in which a global VTAM secondary return code is
    returned to the application program. It is labeled RPLFDB2 in the RPL.

RCPRI
    The field in the RPL extension in which an APPCCMD-specific primary return
    code is returned to the application program. This field has meaning only when
    RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCPR in the RPL
    extension.

RCSEC
    The field in the RPL extension in which an APPCCMD-specific secondary
    return code is returned to the application program. This field has meaning
    only when RTNCD=X’00’ and FDB2=X’0B’. This field is labeled RPL6RCSC in
    the RPL extension. The combination of the RCPRI and RCSEC fields indicates
    the result of the macroinstruction processing.

RTNCD
    The field in the RPL in which a global VTAM primary return code is returned
    to the application program. This field is labeled RPLRTNCD in the RPL.

State changes

No state changes are associated with this macroinstruction.

Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application
program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’0000’</td>
<td>X’0000’</td>
<td>OK</td>
</tr>
<tr>
<td>X’002C’</td>
<td>X’000C’</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
</tbody>
</table>
### APPCCMD CONTROL=TESTSTAT, QUALIFY=ALL

#### Purpose

This macroinstruction obtains status on information from any active conversation. VTAM will wait for information to arrive on a conversation to satisfy the macroinstruction request. If information is available to be received, the application will receive status on the information without waiting.

#### Usage

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field which can be set with the AREA keyword. See "Status data structure (ISTSTATD)" on page 656 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be at least 48 bytes) an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

If this macroinstruction is issued while another TESTSTAT ALL|IALL is currently outstanding, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application program.

This macroinstruction will not alter the conversation.

#### Context

Input states are not applicable to this macroinstruction.
Syntax

```
APPCCMD CONTROL = TESTSTAT , QUALIFY = ALL

RPL = rpl_address_field
     ( rpl_address_register )

AAREA = rpl_extension_address_field
        ( rpl_extension_address_register )

ACB = acb_address_field
     ( acb_address_register )

AREA = data_area_address_field
      ( data_area_address_register )

AREALEN = data_area_length
        ( data_area_length_register )

BRANCH = NO
      YES

ECB = INTERNAL
     ( ecb_address_field
       ( ecb_address_register )

EXIT = exit_routine_address_field
      ( exit_routine_address_register )
```
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters
The following information shows descriptions of the input parameters:

\[ \text{AAREA} = \text{rpl\_extension\_address\_field} \]
\[ \text{AAREA} = (\text{rpl\_extension\_address\_register}) \]
Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ \text{ACB} = \text{acb\_address\_field} \]
\[ \text{ACB} = (\text{acb\_address\_register}) \]
Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

\[ \text{AREA} = \text{data\_area\_address\_field} \]
\[ \text{AREA} = (\text{data\_area\_address\_register}) \]
Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.

\[ \text{AREALEN} = \text{data\_area\_length} \]
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of
the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RECLLEN**

The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X'0000', X'0000'. If the RCPRI,RCSEC fields equal X'002C', X'0008' RECLLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLEN in the RPL.

**RTNCD**

The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD.

**State Changes**

No state changes are associated with this macroinstruction.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0012'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_NOT_VALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'002E'</td>
<td>PARAMETER_ERROR—VECTOR_AREA_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_ REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

**APPCCMD CONTROL=TESTSTAT, Qualifiable**

**Purpose**

This macroinstruction obtains status on information immediately available from any active conversation. VTAM will not wait for information to arrive on a conversation to satisfy the macroinstruction request.

**Usage**

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be
specified in the RPLAREA field, which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 656 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be a least 48 bytes), an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

If this macroinstruction is issued and information is not available on any conversation, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application.

Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

• Normal information
• Expedited information (data and/or Request_To_Send Received)

If this macroinstruction is issued while another TESTSTAT ALL|IALL is currently outstanding, an RCPRI, RCSEC combination of X'002C', X'0011', PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING is returned to the application program.

This macroinstruction will not alter the conversation.

Context

Input states are not applicable to this macroinstruction.

Syntax
APPCCMD CONTROL = TESTSTAT, QUALIFY = IALL

RPL = rpl_address_field
    (rpl_address_register)

AAREA = rpl_extension_address_field
    (rpl_extension_address_register)

ACB = acb_address_field
    (acb_address_register)

AREA = data_area_address_field
    (data_area_address_register)

AREALEN = data_area_length
    (data_area_length_register)

BRANCH = NO
    YES

ECB = INTERNAL
    (ecb_address_field)
    (ecb_address_register)

EXIT = exit_routine_address_field
    (exit_routine_address_register)

OPTCD = (ASY)
    SYN
    KEEPSRB
    NKEEPSRB
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA=** rpl_extension_address_field

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB=** acb_address_field

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA=** data_area_address_field

Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.

**AREALEN=** data_area_length

Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

**BRANCH**

Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.
BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.
RPL=rpl_address_field
RPL=(rpl_address_register)

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

**RPL and RPL extension fields modified by macroinstruction**

The following information shows descriptions of RPL and RPL extension fields:

**FDB2**
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

**RCPRI**
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

**RECLEN**
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X'0000', X'0000'. If the RCPRI,RCSEC fields equal X'002C', X'0008' RECLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLLEN in the RPL.

**RTNCD**
The field in the RPL in which a global VTAM primary return code is returned to the application program. It is labeled RPLRTNCD.
### Return codes

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See [Chapter 2, “Return codes,” on page 591](#) for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

### APPCCMD CONTROL=TESTSTAT, QUALIFY=ISPEC

#### Purpose

This macroinstruction obtains status on information immediately available on a specified conversation. VTAM will not wait for information to arrive to satisfy the macroinstruction request.

#### Usage

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field which can be set with the AREA keyword. See [“Status data structure (ISTSTATD)” on page 656](#) for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be at least 48 bytes) an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

If information is not available, an RCPRI, RCSEC combination of X'0000', X'0008', NO_IMMEDIATELY_AVAILABLE_INFORMATION is returned to the application program.

If the conversation ends before this macroinstruction can query the information received, if any, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application.
Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

- Normal information
- Expedited information (data and/or Request_To_Send Received)

This macroinstruction will not alter the conversation.

**Context**

This macroinstruction can be issued in any conversation state while the conversation is active so long as another APPCCMD CONTROL=TESTSTAT, QUALIFY=SPEC|ISPEC macroinstruction is not currently outstanding for the specified conversation.

**Syntax**

```plaintext
APPCCMD  CONTROL=TESTSTAT, QUALIFY=ISPEC

RPL=rpl_address_field
    (rpl_address_register)

AAREA=rpl_extension_address_field
    (rpl_extension_address_register)

ACB=acb_address_field
    (acb_address_register)

AREA=data_area_address_field
    (data_area_address_register)

AREALEN=data_area_length
    (data_area_length_register)

BRANCH=NO
    YES

CONVID=32-bit_resource_id_field
    (32-bit_resource_id_register)
```
Notes:

1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.

3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.

4. ECB is meaningful only for asynchronous operations.

5. EXIT is meaningful only for asynchronous operations.

6. You can code more than one suboperand on OPTCD, but no more than one from each group.

7. KEEPSRB is meaningful only for synchronous operations.

8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

\[ \text{AAREA} = \text{rpl_extension_address_field} \]

\[ \text{AAREA} = (\text{rpl_extension_address_register}) \]

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

\[ \text{ACB} = \text{acb_address_field} \]

\[ \text{ACB} = (\text{acb_address_register}) \]

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.
AREA=data_area_address_field
AREA=(data_area_address_register)
Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, CITY-STATE. This field is labeled RPLAREA in the RPL.

AREALEN=data_area_length
AREALEN=(data_area_length_register)
Specifies the length value that is the maximum amount of data the application program is to receive. The application program must receive at least 48 bytes of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application programs running in supervisor state under a TCB. Application programs running in TCB-mode supervisor state can use BRANCH=YES to obtain authorized path services. The indicator resides within the RPLEXTDS field of the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs running in problem state (non-supervisor state) under a TCB, BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs running under an SRB rather than under a TCB, the macroinstruction is processed in this manner automatically, regardless of the actual setting of the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to be informed of the completion of the APPCCMD macroinstruction. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an asynchronous APPCCMD completes. Event_control_block_address is the location of the ECB to be posted. The ECB can be any fullword of storage aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled when the APPCCMD completes. You cannot specify both ECB and EXIT on a single APPCCMD macroinstruction. The indicator resides within the RPLEXTDS field of the RPL.
OPTCD
Specifies the following processing options that can be selected for the macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application program when the function of the APPCCMD has completed. The indicator resides within the RPOLOPT1 field of the RPL.

OPTCD=ASY
Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPOLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the TESTSTAT queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE queue if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex, and the EXPEDITED RECEIVE and EXPEDITED SEND queues. For more information about conversation queues, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

OPTCD=KEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

OPTCD=NKEEPSRB
Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPOLOPT11 field of the RPL.

RPL=rpl_address_field
RPL=(rpl_address_register)
Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

CONSTATE
The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension. For half-duplex conversations, this field can have the following values:

X'01' SEND
X'02' RECEIVE
X'03' RECEIVE_CONFIRM
X'04' RECEIVE_CONFIRM_SEND
X'05'  RECEIVE_CONFIRM_DEALLOCATE
X'06'  PENDING_DEALLOCATE
X'07'  PENDING_END_CONVERSATION_LOG
X'08'  END_CONVERSATION
X'09'  PENDING_SEND
X'0A'  PENDING_RECEIVE_LOG

For full-duplex conversations, this field can contain the following values:
X'80'  FDX_RESET
X'81'  SEND/RECEIVE
X'82'  SEND_ONLY
X'83'  RECEIVE_ONLY
X'84'  PENDING_SEND/RECEIVE_LOG
X'85'  PENDING_RECEIVE-ONLY_LOG
X'86'  PENDING_RESET_LOG

**FDB2**

The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

**FMH5LEN**

The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

**FMH5RCV**

The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

**YES (B'1')**

One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

**NO (B'0')**

No FMH-5s are waiting to be received by the application program.

**RCPRI**

The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

**RCSEC**

The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.
**RELEN**  
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X'0000', X'0000'. If the RCPRI,RCSEC fields equal X'002C', X'0008' RELEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLEN in the RPL.

**RTNCD**  
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

**USERFLD**  
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

**Return codes**

The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>
APPCCMD CONTROL=TESTSTAT, QUALIFY=SPEC

Purpose

This macroinstruction obtains status on information available on a specified conversation. VTAM will wait for information to arrive to satisfy the macroinstruction request. If information is already available, the application program receives status on it without waiting.

Usage

The information returned from this macroinstruction is contained in the status data structure control block, CITY-STATE. The address of the control block must be specified in the RPLAREA field, which can be set with the AREA keyword. See “Status data structure (ISTSTATD)” on page 656 for a description of the control block.

If the length of the area specified by the application is not sufficient to receive the entire status data structure (AREALEN should be a least 48 bytes), an RCPRI, RCSEC combination of X'002C', X'0008', PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT is returned to the application. RECLEN will contain the length of the data structure.

Upon successful completion, this macroinstruction will return status on one or more of the following types of information:

• Normal information
• Expedited information (data and/or Request_To_Send Received)

If the conversation is terminated before information is received, an RCPRI, RCSEC combination of X'0000', X'0009', REQUEST_TERMINATED_BY_END_OF_CONVERSATION is returned to the application program.

This macroinstruction will not alter the conversation.

Context

This macroinstruction can be issued in any conversation state while the conversation is active so long as another APPCCMD CONTROL=TESTSTAT QUALIFY=SPEC | ISPEC macroinstruction is not currently outstanding for the specified conversation.

Syntax
APPCCMD CONTROL = TESTSTAT, QUALIFY = SPEC

RPL = rpl_address_field (rpl_address_register)

AAREA = rpl_extension_address_field (rpl_extension_address_register)

ACB = acb_address_field (acb_address_register)

AREA = data_area_address_field (data_area_address_register)

AREALEN = data_area_length (data_area_length_register)

BRANCH = NO

CONVID = 32-bit_resource_id_field (32-bit_resource_id_register)

ECB = INTERNAL ecb_address_field (ecb_address_register)

EXIT = exit_routine_address_field (exit_routine_address_register)
Notes:
1. Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.
2. See “Coding default values” on page 3 for information on coding operands on the RPL or the APPCCMD macroinstruction.
3. Operand value might be placed in its RPL field either by specification on an RPL macroinstruction operand or by explicitly setting the field using the IFGRPL DSECT.
4. ECB is meaningful only for asynchronous operations.
5. EXIT is meaningful only for asynchronous operations.
6. You can code more than one suboperand on OPTCD, but no more than one from each group.
7. KEEPSRB is meaningful only for synchronous operations.
8. NKEEPSRB is meaningful only for synchronous operations.

Input parameters

The following information shows descriptions of the input parameters:

**AAREA**=rpl_extension_address_field

AAREA=(rpl_extension_address_register)

Specifies the address of the LU 6.2 RPL extension that will be associated with this APPCCMD macroinstruction. This field is labeled RPLAAREA in the RPL.

**ACB**=acb_address_field

ACB=(acb_address_register)

Specifies the address of an access method control block that identifies the application program that is issuing the APPCCMD macroinstruction. VTAM associates conversations with application programs using the conversation ID (CONVID). The application program associates conversations with transaction programs. Application programs cannot issue APPCCMD macroinstructions in address spaces other than the ACB address space. This field is labeled RPLDACB in the RPL.

**AREA**=data_area_address_field

AREA=(data_area_address_register)

Specifies the data area in which the application program is to receive the data. The data returned should be mapped using the status data structure, ISTSTATD. This field is labeled RPLAREA in the RPL.

**AREALEN**=data_area_length
AREALEN=(data_area_length_register)
Specifications the length value that is the maximum amount of data the application
program is to receive. The application program must receive at least 48 bytes
of data, or it will be rejected. This field is labeled RPLBUFL in the RPL.

BRANCH
Specifies whether authorized path processing is to be used for application
programs running in supervisor state under a TCB. Application programs
running in TCB-mode supervisor state can use BRANCH=YES to obtain
authorized path services. The indicator resides within the RPLEXTDS field of
the RPL.

BRANCH=NO
Authorized path processing is not to be used. For application programs
running in problem state (non-supervisor state) under a TCB,
BRANCH=NO is the only option.

BRANCH=YES
Authorized path processing is to be used. For application programs
running under an SRB rather than under a TCB, the macroinstruction is
processed in this manner automatically, regardless of the actual setting of
the BRANCH field.

CONVID=32-bit_resource_id_field
CONVID=(32-bit_resource_id_register)
Specifies the resource ID of the conversation. This field is labeled RPL6CNVD
in the RPL extension.

ECB
Valid only if OPTCD=ASY. Specifies how the application program requests to
be informed of the completion of the APPCCMD macroinstruction. You cannot
specify both ECB and EXIT on a single APPCCMD macroinstruction. The
indicator resides within the RPOPT1 field of the RPL.

ECB=INTERNAL
Specifies that VTAM is to post an internal ECB when the APPCCMD
macroinstruction completes.

ECB=ecb_address_field
ECB=(ecb_address_register)
Specifies that VTAM is to post an event control block (ECB) when an
asynchronous APPCCMD completes. Event_control_block_address is the
location of the ECB to be posted. The ECB can be any fullword of storage
aligned on a fullword boundary.

EXIT=exit_routine_address_field
EXIT=(exit_routine_address_register)
Valid only if OPTCD=ASY. It indicates the address of a routine to be scheduled
when the APPCCMD completes. You cannot specify both ECB and EXIT on a
single APPCCMD macroinstruction. The indicator resides within the
RPLEXTDS field of the RPL.

OPTCD
Specifies the following processing options that can be selected for the
macroinstruction request:

OPTCD=SYN
Specifies that control is to be returned synchronously to the application
program when the function of the APPCCMD has completed. The indicator
resides within the RPOPT1 field of the RPL.
**OPTCD=ASY**

Specifies that control is to be returned to the application program immediately and that the application program is to be informed later of the completion of the macroinstruction by the posting of an ECB or the scheduling of an exit. The indicator resides within the RPLOPT1 field of the RPL.

When the application program regains control after issuing an APPCCMD asynchronously, it is prevented from issuing another APPCCMD against the same conversation resource that processes on the TESTSTAT queue until the command has completed. The application can issue APPCCMDs against the same conversation resource that processes on the SEND/RECEIVE queue if the conversation is half-duplex, or the SEND and RECEIVE queues if the conversation is full-duplex, and the EXPEDITED RECEIVE and EXPEDITED SEND queues. For more information about conversation queues, refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](#).

The application program is allowed to issue APPCCMDs against other conversations. OPTCD=ASY is recommended when issuing this APPCCMD.

**OPTCD=KEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM returns to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**OPTCD=NKEEPSRB**

Specifies that for a synchronous request issued in SRB mode, VTAM does not return to the application under the same SRB in which VTAM was invoked. The indicator resides within the RPLOPT11 field of the RPL.

**RPL=rpl_address_field**

**RPL=(rpl_address_register)**

Specifies the address of the request parameter list that contains information to be used during the processing of the APPCCMD macroinstruction.

### RPL and RPL extension fields modified by macroinstruction

The following information shows descriptions of RPL and RPL extension fields:

**CONSTATE**

The field in the RPL6 extension that indicates the state of the conversation. This field is labeled RPL6CCST in the RPL extension.

For half-duplex conversations, this field can have the following values:

- **X'01'**  SEND
- **X'02'**  RECEIVE
- **X'03'**  RECEIVE_CONFIRM
- **X'04'**  RECEIVE_CONFIRM_SEND
- **X'05'**  RECEIVE_CONFIRM_DEALLOCATE
- **X'06'**  PENDING_DEALLOCATE
- **X'07'**  PENDING_END_CONVERSATION_LOG
- **X'08'**  END_CONVERSATION
- **X'09'**  PENDING_SEND
X'0A' PENDING_RECEIVE_LOG
For full-duplex conversations, this field can have the following values:
X'80' FDX_RESET
X'81' SEND/RECEIVE
X'82' SEND_ONLY
X'83' RECEIVE_ONLY
X'84' PENDING_SEND/RECEIVE_LOG
X'85' PENDING_RECEIVE-ONLY_LOG
X'86' PENDING_RESET_LOG

FDB2
The field in the RPL in which a global VTAM secondary return code is returned to the application program. It is labeled RPLFDB2 in the RPL.

FMH5LEN
The field in the RPL extension that returns the length of the FMH-5 waiting to be received by the application program. If multiple FMH-5s are waiting to be received, FMH5LEN specifies the length of the longest FMH-5 to be received by the application program. This field has meaning only when FMH5RCV=YES. This field is labeled RPL6MH5L in the RPL extension.

FMH5RCV
The field in the RPL extension that returns an indication of whether an FMH-5 has been received. The indication is either YES or NO (RPL6RMH5 set on or off). This field is labeled RPL6RMH5 in the RPL extension.

YES (B'1')
One or more FMH-5s have been received from partner LUs. The FMH5RCV field continues to be set to YES so long as an FMH-5 is waiting to be received by the application program. The application program must issue APPCCMD CONTROL=RCVFMH5 in order to receive an FMH-5.

NO (B'0')
No FMH-5s are waiting to be received by the application program.

RCPRI
The field in the RPL extension in which an APPCCMD-specific primary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCPR in the RPL extension.

RCSEC
The field in the RPL extension in which an APPCCMD-specific secondary return code is returned to the application program. This field has meaning only when RTNCD=X'00' and FDB2=X'0B'. This field is labeled RPL6RCSC in the RPL extension. The combination of the RCPRI and RCSEC fields indicates the result of the macroinstruction processing.

RECLEN
The field in the RPL that returns to the application program the actual size of the structure containing the status information VTAM placed in the AREA if the RCPRI,RCSEC fields equal X'0000', X'0000'. If the RCPRI,RCSEC fields equal X'002C', X'0008' RECLEN indicates the length of the status data structure, but because the receive buffer is not sufficient to contain the entire structure, none of the status data structure is returned to the application program. This field is labeled RPLRLLEN in the RPL.
RTNCD
The field in the RPL in which a global VTAM primary return code is returned to the application program. This field is labeled RPLRTNCD in the RPL.

USERFLD
Specifies 4 bytes of user data that the application program requests be associated with a conversation. Whenever an APPCCMD completes, VTAM places in the USERFLD field of the RPL extension the 4 bytes that were supplied on the APPCCMD CONTROL=ALLOC macroinstruction (if the conversation was initiated by the local application program) or the APPCCMD CONTROL=RCVFMH5 macroinstruction (if the conversation was initiated by the remote application program). This field is labeled RPL6USR in the RPL extension.

Return codes
The following (RCPRI, RCSEC) combinations can be returned to the application program when it issues this APPCCMD macroinstruction. See Chapter 2, “Return codes,” on page 591 for a description of these return codes.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>OK</td>
</tr>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>REQUEST_TERMINATED_BY_END_OF_CONVERSATION</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>PARAMETER_ERROR—INVALID_CONVERSATION_ID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0008'</td>
<td>PARAMETER_ERROR—SUPPLIED_LENGTH_INSUFFICIENT</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>PARAMETER_ERROR—ZERO_EXIT_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>PARAMETER_ERROR—ZERO_ECB_FIELD</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000E'</td>
<td>PARAMETER_ERROR—REQUEST_INVALID_FOR_ADDRESS_SPACE</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'000F'</td>
<td>PARAMETER_ERROR—CONTROL_BLOCK_INVALID</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0010'</td>
<td>PARAMETER_ERROR—INVALID_DATA_ADDRESS_OR_LENGTH</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0011'</td>
<td>PARAMETER_ERROR—PREVIOUS_MACROINSTRUCTION_OUTSTANDING</td>
</tr>
<tr>
<td>X'002C'</td>
<td>X'0032'</td>
<td>PARAMETER_ERROR—UNEXPECTED_VECTOR_PROVIDED_ON_APPCCMD</td>
</tr>
<tr>
<td>X'0070'</td>
<td>X'0000'</td>
<td>TEMPORARY_STORAGE_SHORTAGE</td>
</tr>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>VTAM_INACTIVE_FOR_YOUR_ACB</td>
</tr>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>REQUEST_ABORTED</td>
</tr>
<tr>
<td>X'0090'</td>
<td>X'0000'</td>
<td>APPLICATION_NOT_APPC_CAPABLE</td>
</tr>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>REQUEST_NOT_ALLOWED—REQUEST_BLOCKED</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>ENVIRONMENT_ERROR_OS_LEVEL_DOES_NOT_SUPPORT_REQUESTED_FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>ENVIRONMENT_ERROR—SUSPEND_FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>ENVIRONMENT_ERROR—RESUME_FAILURE</td>
</tr>
</tbody>
</table>

ISTGAPPC

Purpose
This macroinstruction declares and sets a list of global variables to indicate which LU 6.2 options are supported by the installed release of VTAM.

Usage
ISTGAPPC can be invoked directly, or by either IFGRPL or IFGACB as an inner macroinstruction call. The global variables defined for ISTGAPPC are shown in Table 2 on page 579.
To use the ISTGAPPC macroinstruction, the programmer must be familiar with the
GBLA and SETA assembler language instructions, which are described in the
assembler language publication for your operating system.

The use of ISTGAPPC is similar to the use of the ISTGLBAL macroinstruction. For
details, refer to the description of ISTGLBAL in [z/OS Communications Server:
SNA Programming]

The variables defined by ISTGAPPC are available to the application program at
assembly time. If you want the application program to check these values at
execution time, you can use the function-list vector described in the [z/OS
Communications Server: SNA Programmer's LU 6.2 Guide]

Each global variable is an arithmetic symbol that can be set to 0, 1, or 2. The
following information shows the meanings for the global variables and the
corresponding levels of support.

Global Variable
   Support Level
X'00'   No (Option is not supported.)
X'01'   Yes (Option is supported.)
X'02'   Pass-through (VTAM offers support for this function, but the application
        program must implement the function.)

Context

Input states are not applicable to this macroinstruction.

Syntax
Comments

ISTGAPPC sets the following global variables:

**Table 2. LU 6.2 global macro variables set by ISTGAPPC**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Function Indicated</th>
<th>Support Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ISTGA01</td>
<td>Conversations between transaction programs at the same LU</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA02</td>
<td>Delayed session allocation</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA03</td>
<td>Immediate session allocation</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA04</td>
<td>Sync point services</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA05</td>
<td>Program reconnect</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA06</td>
<td>Reserved</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA07</td>
<td>Session-level LU-LU verification</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA08</td>
<td>User identifier verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA09</td>
<td>Program-supplied user identifier and password</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA10</td>
<td>User identifier authorization</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA11</td>
<td>Profile verification and authorization</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA12</td>
<td>Reserved</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA13</td>
<td>Profile pass-through</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA14</td>
<td>Program-supplied profile</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA15</td>
<td>Send persistent verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA16</td>
<td>Receive persistent verification</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA17</td>
<td>PIP data</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA18</td>
<td>Logging of data in system log</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA19</td>
<td>Flush LU's SEND buffer</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA20</td>
<td>LUW identifier</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA21</td>
<td>Prepare to receive</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA22</td>
<td>Long locks</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA23</td>
<td>Post on receipt with wait</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA24</td>
<td>Post on receipt with test for posting</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA25</td>
<td>Receive immediate</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA26</td>
<td>Test for request-to-send received</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA27</td>
<td>Data mapping</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA28</td>
<td>FMH application program data</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA29</td>
<td>Get attributes</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA30</td>
<td>Get conversation type</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA31</td>
<td>Mapped conversation LU services component</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA32</td>
<td>CHANGE_SESSION_LIMIT verb</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA33</td>
<td>MIN_CONTENTION WINNERS_TARGET parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA34</td>
<td>RESPONSIBLE(TARGET) parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA35</td>
<td>DRAIN_TARGET(NO) parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA36</td>
<td>FORCE parameter</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA37</td>
<td>ACTIVATE_SESSION verb</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA38</td>
<td>DEACTIVATE_SESSION verb</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA39</td>
<td>LU parameter verbs</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA40</td>
<td>LU-LU session limit</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA41</td>
<td>Locally-known LU names</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA42</td>
<td>Uninterpreted LU names</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Table 2. LU 6.2 global macro variables set by ISTGAPPC (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Function Indicated</th>
<th>Support Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;ISTGA43</td>
<td>Single-session reinitiation</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA44</td>
<td>Alternate code processing</td>
<td>No</td>
</tr>
<tr>
<td>&amp;ISTGA45</td>
<td>Maximum RU size bounds</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA46</td>
<td>Session-level mandatory cryptography</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA47</td>
<td>Contention-winner automatic-activation limit</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA48</td>
<td>Queued allocation of a contention-winner session</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA49</td>
<td>Enhanced security (SAME)</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA50</td>
<td>Session-level selective cryptography</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA51</td>
<td>Conversation group support</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA52</td>
<td>ALLOCATE WHEN_SESSION_FREE verb</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA53</td>
<td>LU 6.2 full-duplex protocols</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA54</td>
<td>VTAM-to-application vector list</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA55</td>
<td>Queued RCVFMH5</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA56</td>
<td>High performance data transfer</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA57</td>
<td>APPCCMD SENDRCV</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA58</td>
<td>Intra-LU conversations</td>
<td>Yes</td>
</tr>
<tr>
<td>&amp;ISTGA59</td>
<td>Password substitution</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA60</td>
<td>Extended security sense codes</td>
<td>Pass-through</td>
</tr>
<tr>
<td>&amp;ISTGA61</td>
<td>DCE security services</td>
<td>Pass-through</td>
</tr>
</tbody>
</table>

ISTRPL6

Purpose

This macroinstruction obtains storage for the RPL extension at assembly time and initializes any fields included as parameters on the macroinstruction. Any fields without a default value and not explicitly included on the macroinstruction are set to 0.

Context

Input states are not applicable to this macroinstruction.

Syntax
Chapter 1. LU 6.2 macroinstruction syntax and operands 581
Notes:

1  In this macroinstruction, all operands except the first must be preceded by a comma. For example, you would code ISTRPL6 CD=DEFER,FILL=LL,LOCKS=LONG.

2  Operand value might be placed in its RPL extension field either by specification on an ISTRPL6 macroinstruction operand or by explicitly setting the field using the ISTRPL6X DSECT.

Input parameters

CD

Specifies whether the LU immediately goes to SEND or whether the LU defers the SEND transition by going into PEND_SEND when a change of direction is received with no data.
**CD=DEFER**
Specifies that the conversation state will be PEND_SEND when the SEND indicator of the WHATRCV field is set and none of the data indicators are set.

**CD=IMMED**
Specifies that the conversation state will be SEND when the SEND indicator of the WHATRCV field is set and none of the data indicators are set. IMMED is the default.

**CONMODE**
Specifies that upon completion of the APPCCMD, the conversation is to be placed in logical-record-continue-any, buffer-continue-any, or continue-specific mode. This field is labeled RPL6CMOD in the RPL extension.

**CONMODE=BUFFCA**
Specifies that the conversation is to be placed in buffer-continue-any mode. It indicates that this conversation is to apply when APPCCMD CONTROL=RECEIVE, QUALIFY=ANY is issued and that the application program is to receive data independently of the logical-record format of the data. BUFFCA corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**CONMODE=CS**
Specifies that the conversation is to be placed in continue-specific mode. It indicates that data is to be received from this conversation by the application program only if the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC. When the application program issues APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC, it must indicate whether the data is to be received in terms of the logical-record format of the data, or independently of the logical-record format of the data.

**CONMODE=LLCA**
Specifies that the conversation is to be placed in logical-record-continue-any mode. It indicates that this conversation is to apply when APPCCMD CONTROL=RECEIVE, QUALIFY=ANY is issued and that the application program is to receive data in terms of the logical-record format of the data. LLCA corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

**CONMODE=SAME**
Specifies that the continuation mode of the conversation should remain unchanged after the completion of the APPCCMD macroinstruction using this RPL.

**CONXMOD**
Specifies the mode for receiving expedited information upon completion of the APPCCMD.

**CONXMOD=CS**
Specifies that the mode for expedited information is to be put in such a state that expedited information can only be received by a specific-type of macroinstruction for such as, APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC.

**CONXMOD=CA**
Specifies that the mode for expedited information is to be put in such a state that expedited information can only be received by either a specific-type of macroinstruction, for example, APPCCMD
CONTROL=RCVEXPD, QUALIFY=SPEC or ISPEC, or by any type of macroinstruction, for example, APPCCMD CONTROL=RCVEXPD, QUALIFY=ANY or IANY.

CONXMOD=SAME
Specifies that the conversation mode for expedited data is to remain unchanged at the completion of this macroinstruction.

FILL
Specifies whether the application program is to receive data in terms of the logical-record format of the data. This parameter corresponds to FILL=LL | BUFFER described in the LU 6.2 architecture. The field is ignored unless QUALIFY=SPEC. This field is labeled RPL6FILL in the RPL extension.

FILL=BUFF
Specifies the application program is to receive data independently of its logical-record format, up to the length specified by the AREALEN field of the RPL. FILL=BUFF corresponds to FILL=BUFFER on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

FILL=LL
Specifies the application program is to receive one logical record, or a portion of the logical record up to the length specified by the AREALEN field of the RPL. If only a portion of the logical record is received, the DATA_INCOMPLETE bit in the what-received field is set on. The remainder of the logical record is buffered by VTAM, and will be used to satisfy the next RECEIVE request. FILL=LL corresponds to FILL=LL on the RECEIVE_AND_WAIT verb, as described in the LU 6.2 architecture.

LIST
Specifies the amount of detail to be provided about LUs, modes, and sessions. The requested information is provided in a RESTORE structure and describes the LUs, modes, and sessions that have been restored. This field is labeled RPL6LIST in the RPL extension.

LIST=ALL
Specifies that all LU, mode, and session information is included in the RESTORE structure.

LIST=NONE
Specifies that no RESTORE structure is returned.

LIST=NOSESS
Specifies that all LU and mode information is included in the RESTORE structure; session information is not included.

LOCKS
Specifies when the execution of the macroinstruction is complete following execution of the CONFIRM function. This field corresponds to the LOCKS parameter on the PREPARE_TO_RECEIVE verb as described in the LU 6.2 architecture. This field is labeled RPL6LOCK in the RPL extension.

LOCKS=SHORT
Specifies that the function of this macroinstruction is complete when a positive response is received to the confirmation request.

LOCKS=LONG
Specifies that the function of this macroinstruction is complete when information, such as data, is received from the partner LU after an affirmative reply to the confirmation request. The application program
must issue an APPCCMD CONTROL=RECEIVE in order to get the information that caused the prior macroinstruction to complete.

**LOGMODE=8-byte_logon_mode_name**

The field that holds the logon mode name of the session over which an FMH-5 flows. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6MODE in the RPL extension.

**LUAFFIN**

Specifies whether the application program or VTAM will be the owner of the Generic Resource affinity for this specific LU partner.

**LUAFFIN=APPL**

The application program will own the GR affinity for this LU.

**LUAFFIN=NOTAPPL**

VTAM will own the GR affinity for this LU.

The LUAFFIN keyword is only meaningful when the issuing application is acting as a generic resource. If the application does not support a generic name, LUAFFIN is ignored.

The LUAFFIN value is honored if no sessions currently exist with the partner LU. If any active or pending sessions exist, the LUAFFIN value is ignored, and the previously established ownership is used for new sessions. If LUAFFIN is not specified and no sessions currently exist with the partner LU, the generic resource affinity ownership will be based on the type of LU 6.2 session or the owner will be the application if SETLOGON OPTCD=GNAMEADD, AFFIN=APPL was issued.

For more information about affinity ownership between an LU and a generic resource member, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

**LUNAME=8-byte_lu_name**

The field that holds the name of a partner LU. This LU name is the network name of the partner LU. It is an 8-byte name, padded on the right with blanks. This field is labeled RPL6LU in the RPL extension.

**NETID=8-byte_network_identifier**

The field that holds the network identifier of the partner LU. This identifier is the network identifier of the partner LU. If NQNAMES=YES, LUNAME and NETID are used together to form the network-qualified of the target LU. (If NETID is specified, LUNAME is specified.)

The network identifier is an 8-byte name, padded on the right with blanks. This field is labeled RPL6NET in the RPL extension.

**QUALIFY=one_of_the_qualify_values_listed_below**

Specifies the exact function of an APPCCMD macroinstruction. The general function of the macroinstruction is determined by the CONTROL keyword, required on each APPCCCMD macroinstruction.

See the individual macroinstruction descriptions for details.

**ABNDPROG**

Specifies abnormal termination of a conversation because of a transaction program error.

**ABNDSEVR**

Specifies abnormal termination of a conversation because of an LU services component error.
ABNDTIME
Specifies abnormal termination of a conversation because of excessive elapsed time.

ABNDUSER
Specifies abnormal termination of a conversation because of a user-specified condition.

ACTSESS
Responds positively to a session-initiation request being processed in the LOGON or SCIP exit.

ALL
Specifies a TESTSTAT that can return status on information that is available on any conversation.

ALLOC
Allocates a session for use by a conversation.

ANY
Used to specify a RECEIVE or RCVEXPD that will accept normal or expedited information, respectively, for more than one conversation.

CNOS
Regulates session limits with another application program.

CONFIRM
Sends a confirmation request to another application program.

CONFIRM
Sends a reply to a confirmation request.

CONV
Deallocates the conversation and its underlying session.

CONVGRP
Associates a session having a specified conversation group identifier with a conversation for allocation of a conversation or deactivation of the session.

CONWIN
Allocates a conversation to a contention-winner session.

DACTSESS
Responds negatively to a session-initiation request in the LOGON or SCIP exit.

DATA
Sends data to a partner LU.

DATACON
Sends data and a confirmation request to a partner LU.

DATAFLU
Sends data to a partner LU and forces flushing of the SEND buffer.

DATAQUE
Specifies that the macroinstruction be queued pending receipt of the FMH-5 from the partner LU and that the FMH-5 as well as any data should be received to the application's buffer when received by VTAM.

DEFINE
Alters information in the LU-mode table.

DISPLAY
Displays information in the LU-mode table.
ERROR
Sends an error indication to a partner LU.

FLUSH
Forces flushing of the SEND buffer.

IALL
Specifies a TESTSTAT that can return status on information that is immediately available on any conversation.

IANY
Specifies a RECEIVE or RCVEXPD that can receive normal or expedited information, respectively, that is immediately available from a conversation in continue-any mode.

IMMED
Allocates a contention-winner session for immediate use by a conversation.

ISPEC
Specifies a RECEIVE that will accept normal information that is immediately available from a user-specified conversation.

NULL
Optional value that can be used when no other QUALIFY value applies

QUEUE
Specifies that the macroinstruction be queued pending receipt of the FMH-5 from the partner LU and that the FMH-5 should be received to the application's buffer when received by VTAM.

RESTORE
Restores a mode (or modes) that has been retained pending recovery of one or more persistent LU-LU sessions.

RESUME
Releases a session that has been suspended.

RQSEND
Requests that an application program be placed in SEND state.

SESSION
Deactivates the session and deallocates any conversation associated with it.

SPEC
Satisfies a RECEIVE using data for a particular conversation.

SUSPEND
Suspends a subsequent conversation.

SYNCBEG
Indicates the beginning of a synchronization exchange.

SYNCEND
Indicates the end of a synchronization exchange.

WHENFREE
Specifies to allocate a session for the conversation if a session is available or pending or one can be activated.

RTSRTRN
Specifies, upon completion of the APPCCMD, the manner in which Request_To_Send_Received indication is to be received.

RTSRTRN=BOTH
Specifies that Request_To_Send_Received indication can be received either
by an APPCCMD CONTROL=SENDEXPD or an APPCCMD
CONTROL=RCVEXPD or reported in the SIGRCV and SIGDATA fields
returned with other APPCCMDs.

RTSRTRN=EXPD
Specifies that Request_To_Send_Received indication can be received only
by an APPCCMD CONTROL=SENDEXPD or an APPCCMD
CONTROL=RCVEXPD.

SENSE=32-bit_unbind_sense_code
SENSE=(32-bit_unbind_sense_code_register)
The field that holds a 32-bit sense code. This field is labeled RPL6SNSO in the
RPL extension.

TYPE
Specifies the level of error being reported on an APPCCMD CONTROL=SEND,
QUALIFY=ERROR macroinstruction. This field is intended to distinguish
between errors to be reported to end-user transaction programs and errors to
be reported to a service component, such as a mapped conversation
component, of the LU. This field is labeled RPL6TYPE in the RPL extension.
See “APPCCMD CONTROL=SEND, QUALIFY=ERROR” on page 475 for more
details.

TYPE=PROGRAM
Specifies an end-user transaction program error is being reported.

TYPE=SERVICE
Specifies a service-component error is being reported.

TYPE=USER
Specifies that the application program is providing to VTAM a user-specific
sense code that it requests be placed in the FMH-7 that VTAM creates as a
result of this APPCCMD macroinstruction.

USERFLD=4_bytes_of_user_data
USERFLD=(user_data_register)
Specifies 4 bytes of user data that the application program requests be
associated with a conversation. This field is labeled RPL6USR in the RPL
extension.

VTRINA=vector_address_field
VTRINA=(vector_address_register)
Specifies the address of the data area where VTAM places vector list
information for the application.

This parameter is ignored if one of the following items is true:
• VTRINA=0
• The value for VTRINL is less than the minimum length required to return
the APPCCMD vector area header.
• The value for VTRINL is not specified.

This field is labeled RPL6VAIA in the RPL extension.

VTRINL=vector_length_field
VTRINL=(vector_length_register)
Specifies the length of the data area where VTAM places vector list information
for the application.

This parameter is ignored if the value for VTRINA is 0 or is not specified. This
field is labeled RPL6VAIL in the RPL extension.

VTROUTA=vector_address_field
VTROUTA=(*vector_address_register*)

Specifies the address of the area where the application places vector list information for VTAM. If OPTCD=XBUFLST is specified, this field must point to the XBUFLST-receive vector (ISTAPC82), which is mapped by ISTAPCVL. (Refer to [z/OS Communications Server: SNA Programmer's LU 6.2 Guide](https://www.ibm.com/support/docview.wss?uid=swg21355704) for more information.)

This field is labeled RPL6VAOA in the RPL extension.

VTROUTL=*vector_length_field*

VTROUTL=(*vector_length_register*)

Specifies the length of the area where the application places vector list information for VTAM. This field is labeled RPL6VAOL in the RPL extension.
Chapter 2. Return codes

VTAM passes feedback return codes to the LU 6.2 application program in a variety of ways. The principal feedback mechanism is the RCPRI and RCSEC return code fields in the RPL extension. These fields have meaning only when register 15 is set to X'00' and register 0 is set to X'0B'. These values are also the values of the RPL's RTNCD and FDB2 fields, respectively.

For a general discussion of how register contents relate to RPL feedback fields, refer to z/OS Communications Server: SNA Programmer's LU 6.2 Guide.

RCPRI and RCSEC codes

The RPL extension contains two fields in which return code information is passed to the application program at the completion of an APPCCMD macroinstruction execution. The two fields are RPL6RCPRI and RPL6RCSEC, and together they indicate the result of the macroinstruction execution, including any state changes to the specified conversation. The RCPRI field returns a primary return code to the application; the RCSEC field returns a secondary return code to the application. Some RCPRI codes do not have associated RCSEC subcodes. For these RCPRI codes, the RCSEC field is set to X'0000'.

Some of the (RCPRI, RCSEC) return codes indicate the results of the local VTAM's processing of the macroinstruction; these return codes are returned on the APPCCMD that invoked the local processing. Other (RCPRI, RCSEC) return codes indicate the results of processing invoked at the remote end of the conversation and, depending upon the CONTROL and QUALIFY settings of the APPCCMD, can be returned on the APPCCMD that invoked the remote processing or on a subsequent APPCCMD. Still other return codes report events that originate at the remote end of the conversation.

The following information describes the RCPRI and RCSEC codes. Each description includes the meaning of the code, the reason for the condition indicated by the code, when the code can be reported to the application program, and the state of the conversation (if applicable) when the function of the APPCCMD completes. Actions taken by the local application program are discussed in the return code descriptions in terms of APPCCMD macroinstructions; actions taken by the remote LU or transaction program are described more generically using the architected protocol boundary verbs documented in the LU 6.2 architecture.

Note: Some application programs change the hexadecimal values from the RCPRI, RCSEC fields to decimal values. You may need to convert these back to hexadecimal values for problem determination.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000' (all)</td>
<td></td>
<td>USF6OK</td>
<td>OK</td>
</tr>
</tbody>
</table>

The local application program issued an APPCCMD macroinstruction that executed without error. The function defined for the APPCCMD was performed as specified.
The OK RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the application; the RCSEC subcode provides additional information.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0000'</td>
<td>USF6OKSC</td>
<td>OK</td>
</tr>
</tbody>
</table>

The APPCCMD completed successfully and no additional information is defined for the APPCCMD. If a conversation-related macroinstruction is issued, the conversation state can be found in the CONSTATE field. Whenever this RCPRI,RCSEC combination is present, registers 15 and 0 are also set to 0.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0001'</td>
<td>USF6ASSP</td>
<td>AS SPECIFIED</td>
</tr>
</tbody>
</table>

The CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction were accepted by the partner LU as specified, without negotiation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0002'</td>
<td>USF6ASNG</td>
<td>AS NEGOTIATED</td>
</tr>
</tbody>
</table>

One or more of the CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was changed by negotiation with the partner LU. The values are returned to the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. (The macroinstruction description defines which values can be negotiated.)

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0003'</td>
<td>USF6RCVR</td>
<td>RECEIVE SPECIFIC REJECTED</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC or APPCCMD CONTROL=RECEIVE, QUALIFY=ISPEC macroinstruction was rejected because an APPCCMD CONTROL=RECEIVE, QUALIFY=ANY or APPCCMD CONTROL=RECEIVE, QUALIFY=IANY macroinstruction is currently being processed on this conversation. There is no state change. See z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for more information on the APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC|ISPEC and APPCCMD CONTROL=RECEIVE, QUALIFY=ANY|IANY macroinstructions.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0004'</td>
<td>USF6SNGL</td>
<td>PARTNER LU SUPPORTS SINGLE SESSION</td>
</tr>
</tbody>
</table>
VTAM has determined that the partner LU supports only single sessions. If the session limit you specified was greater than 1, or if you did not specify a session limit, then the default values of 1, 0, 0 were used for your CNOS request.

If the partner LU indicated single-session capability using a negative BIND response, the partner LU's name will be missing from the Userdata subfield of the BIND. When the application program issues an APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY macroinstruction, it should verify the presence of the partner LU's fully qualified name. If the FQNLEN field is 0, the partner LU's name is not available. Check the FQNLEN field before checking the FQNAME field.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0005'</td>
<td>USF6INER</td>
<td>INTERNAL VTAM ERROR</td>
<td></td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=REJECT, QUALIFY=SESSION macroinstruction because of an internal error other than a storage shortage condition.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0006'</td>
<td>USF6RSUN</td>
<td>RESTORE_UNNECESSARY—NO_MODES_TO_RESTORE</td>
<td></td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE macroinstruction is unnecessary. The associated mode (or modes) has been restored already, or nothing existed to restore.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0007'</td>
<td>USF6RSIN</td>
<td>RESTORE_INCOMPLETE—INPUT_WORK_AREA_TOO_SMALL</td>
<td></td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE macroinstruction is incomplete. The AREA supplied is too small to hold all the information that needs to be returned. Reissue the macroinstruction one or more times to obtain all the restore information and to complete the restore.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0008'</td>
<td>USF6NINA</td>
<td>NO_IMMEDIATELY_AVAILABLE_INFORMATION</td>
<td></td>
</tr>
</tbody>
</table>

An APPCCMD that requested the immediate return of available information was issued. However, no information that could satisfy the request was available.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'0009'</td>
<td>USF6RTEC</td>
<td>REQUEST TERMINATED BY END OF CONVERSATION</td>
<td></td>
</tr>
</tbody>
</table>
An APPCCMD was awaiting processing or awaiting the arrival of information or a response on a specific conversation. The command has terminated because the conversation ended before the requested information became available or before it could be processed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000A'</td>
<td>USF6ANMS</td>
<td>SESSIONS WILL USE APPL NAME, GENERIC NAME REQUESTED</td>
</tr>
</tbody>
</table>

Use of the generic resource name was requested but the application network name is required.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000B'</td>
<td>USF6GNMS</td>
<td>SESSIONS WILL USE GENERIC NAME, APPL NAME WAS REQUESTED</td>
</tr>
</tbody>
</table>

Use of the application network name was requested but the generic resource name is required.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000C'</td>
<td>USF6NAM1</td>
<td>AS SPECIFIED, PARTNER LU KNOWN BY DIFFERENT NAME</td>
</tr>
</tbody>
</table>

The CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction were acceptable by the partner LU as specified, without negotiation. Furthermore, the CNOS operation caused an LU entry of type RCVD_NAME to be changed to a VARIANT_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000'</td>
<td>X'000D'</td>
<td>USF6NAM2</td>
<td>AS NEGOTIATED, PARTNER LU KNOWN BY DIFFERENT NAME</td>
</tr>
</tbody>
</table>

One or more of the CNOS values supplied by the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was changed by negotiation with the partner LU. The values are returned to the application program on the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction. (The macroinstruction description defines which values can be negotiated.) Furthermore, the CNOS operation caused an LU entry of type RCVD_NAME to be changed to a VARIANT_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>(all)</td>
<td>USF6ALLC</td>
<td>ALLOCATION ERROR</td>
</tr>
</tbody>
</table>

The application program issued APPCCMD CONTROL=ALLOC and allocation of the specified conversation could not be completed. When the ALLOCATION_ERROR RCPRI code is used with one of the RCSEC subcodes
(X'0000'–X'000F'), they form the complete return code that is returned to the program. The RCSEC subcode identifies the specific error. (The partner LU and remote transaction program referred to in the RCSEC definitions are the LU named in the LUNAME field of the APPCCMD, and the transaction program named in the FMH-5 supplied through the AREA field of the APPCCMD, respectively.)

If the partner LU detects the error that causes an ALLOCATION_ERROR RCPRI code to be returned to the application, the error indicator sent by the partner LU can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If an ALLOCATION_ERROR RCPRI code is returned to the application along with LOGRCV=YES, the conversation should issue APPCCMD CONTROL=RECEIVE, QUALIFY=SPEC to receive the error log data. When the error log data is received, the conversation is over.

If an ALLOCATION_ERROR RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0000'</td>
<td>USF6ALNR</td>
<td>ALLOCATION FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

The conversation cannot be allocated on a session because of a permanent condition. For example, the session to be used for the conversation cannot be activated for one of the reasons:

- The mode is closed; the current session limit is 0.
  - CNOS has not been negotiated and no entry has been created for the mode.
  - A previous CNOS request has set limits to 0.
- A system definition error.
- A session-activation protocol error.

The session also might be deactivated because of a session protocol error before the conversation could be allocated. The application program should not try the allocation request again until the condition is corrected. The application should check the returned SENSE field in the RPL extension for an indication of the exact error.

If this code occurs when issuing a DISPLAY APING operator command, the session may have been deactivated as a result of processing a received APING request for the same mode. Reissue the operator command.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0001'</td>
<td>USF6ALR</td>
<td>ALLOCATION FAILURE RETRY</td>
</tr>
</tbody>
</table>

The conversation cannot be allocated on a session because of a temporary condition. For example, the session to be used for the conversation cannot be activated because of a temporary lack of resources at the remote LU; or the session was deactivated because of session outage before the conversation could be allocated. The condition is temporary, and the program can try the allocation request again.
The partner LU rejected the allocation request because the remote transaction program does not support the respective mapped or basic protocol boundary. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application program provided program initialization parameter (PIP) data (along with the FMH-5) and either the partner LU does not support PIP data, or the remote transaction program has no PIP variables defined. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the remote transaction program has one or more PIP variables defined and the local application program provided no program initialization parameters, or the local application program specified program initialization parameters (along with the FMH-5) that do not correspond in number to those defined for the remote transaction program. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the access security information supplied by the local application (in the FMH-5) is not valid. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the synchronization level specified in the allocation request is not supported by both the local and partner LU. The local LU specifies its level of synchronization support on its APPL statement. The partner LU has returned the negotiated level between the two LUs in the BIND response. This return code is returned on the APPCCMD CONTROL=ALLOC macroinstruction for the local LU.
The partner LU rejected the allocation request because the local application program specified a synchronization level (in the FMH-5) that the remote transaction program does not support. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0007'</td>
<td>USF6ALSL</td>
<td>SYNC LEVEL NOT SUPPORTED BY PROGRAM</td>
</tr>
</tbody>
</table>

The partner LU rejected the allocation request because the local application program specified a remote transaction program name (TPN) that the partner LU does not recognize. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0008'</td>
<td>USF6ALTP</td>
<td>TPN NOT RECOGNIZED</td>
</tr>
</tbody>
</table>

The partner LU rejected the allocation request because the local application program specified a remote transaction program that the partner LU recognizes but cannot start. The condition is not temporary, and the application should not try the allocation request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'0009'</td>
<td>USF6ALTN</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY</td>
</tr>
</tbody>
</table>

The partner LU rejected the allocation request because the local application specified a remote program that the remote LU recognizes but currently cannot start. The condition is temporary, and the application can try the allocation request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'000A'</td>
<td>USF6ALTR</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, RETRY</td>
</tr>
</tbody>
</table>

The partner LU rejected the reconnection request because it does not recognize the conversation correlator. The condition is not temporary, and the application should not try the reconnection request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0004'</td>
<td>X'000B'</td>
<td>USF6ALRN</td>
<td>CANNOT RECONNECT TRANSACTION PROGRAM, NO RETRY</td>
</tr>
</tbody>
</table>
The partner LU rejected the reconnection request because it currently cannot reconnect the remote transaction program implied by the conversation correlator. The condition is temporary, however, and the application can try the reconnection request again. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The partner LU rejected the allocation request because the local application program specified a recovery level of program reconnect (in the FMH-5) and the remote transaction program does not support program reconnect. This return code is returned on an APPCCMD subsequent to APPCCMD CONTROL=ALLOC.

The APPCCMD CONTROL=ALLOC macroinstruction is rejected because the specified mode name is pending recovery for persistent LU-LU sessions. Restore the mode by issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

The allocation request has been canceled before its normal processing could be completed. The local application program issued a request for abnormal deallocation of the pending conversation.

The allocation request has been rejected because it specifies a full-duplex conversation with a sync point level not allowed for a full-duplex conversation.
The allocation request has been rejected because it specifies a full-duplex conversation and the negotiated level of support between the local application and the partner LU does not allow full-duplex conversations.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>(all)</td>
<td>USF6CNSA</td>
<td>CNOS FAILURE</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not process successfully. The CNOS_ALLOCATION_ERROR RCPRI code together with one of the RCSEC subcodes (X'0000'–X'0006') form the complete return code that is returned to the transaction program. The RCSEC subcode identifies the specific error. The local and partner LUs’ CNOS parameters are not changed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0000'</td>
<td>USF6CANR</td>
<td>ALLOCATION FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

The control operator conversation cannot be allocated because of a condition that is not temporary. For example, the session to be used for the control operator conversation cannot be activated because the session limit for the specified partner LU and SNASVCMG mode name is currently 0 at either the local LU or partner LU; or because of a system definition error or a session-activation protocol error; or because a session protocol error caused the session to be deactivated before the conversation could be allocated. The CNOS will not be able to complete successfully until the condition is corrected. This code can also be returned if a partner LU rejects a SNASVCMG mode name BIND.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0001'</td>
<td>USF6CAR</td>
<td>ALLOCATION FAILURE, RETRY</td>
</tr>
</tbody>
</table>

The control operator conversation cannot be allocated because of a temporary condition. For example, the session to be used for the control operator conversation cannot be activated because of a temporary lack of resources at the local LU or partner LU, or the session was deactivated because of session outage before the conversation could be allocated. The condition is temporary, and the control operator can try the transaction again later.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0002'</td>
<td>USF6CATR</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, RETRY</td>
</tr>
</tbody>
</table>

The partner LU is currently unable to start the transaction program identified as hex 06F1, which is the SNA service transaction program for the control operator. For example, there can be a temporary lack of resources the partner LU needs to start the transaction program. The condition is temporary, and the control operator can try the transaction again later.
The partner LU is unable to start the transaction program identified as X'06F1', which is the SNA service transaction program for the control operator. The condition is not temporary, and the application should not try again the CNOS request.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0003'</td>
<td>USF6CATN</td>
<td>TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY</td>
</tr>
</tbody>
</table>

The partner LU rejected the CNOS conversation allocation request because the remote transaction program does not support the respective mapped or basic protocol boundary.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0004'</td>
<td>USF6CACM</td>
<td>CONVERSATION TYPE MISMATCH</td>
</tr>
</tbody>
</table>

The partner LU rejected the CNOS conversation allocation request because the access security information supplied by VTAM (in the FMH-5) is not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0005'</td>
<td>USF6CASC</td>
<td>SECURITY NOT VALID</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction is rejected because the specified mode name is pending recovery for persistent LU-LU sessions. Restore the mode by issuing APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE. New modes can be added once the SNASVCMG mode for an LU has been restored, but any mode that exists when the failure (or takeover) occurs cannot be used until that mode has been restored.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0006'</td>
<td>USF6SPMC</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
</tbody>
</table>

The name on an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was an ACB name. The ACB name is not identical to the network resource name. ACB names cannot be used in cross-domain, cross-network, or network qualified. For information on coding the ACBNAME operand, see the z/OS Communications Server: SNA Resource Definition Reference.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0008'</td>
<td>X'0007'</td>
<td>USF6NQNM</td>
<td>NETWORK QUALIFIED NAME MISMATCH</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'000C'</td>
<td>X'0000'</td>
<td>USF6CNSN</td>
<td>CNOS RESOURCE FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>
The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because of a failure that caused the control operator conversation to be deallocated prematurely. For example, the session being used for the control operator conversation was deactivated for one of the reasons:

- A session protocol error
- A session outage from which the control operator component of the LU could not recover

The conversation also might be deallocated because of a protocol error between the control operator components of the LUs. The condition is not temporary, and the control operator should not try the transaction again until the condition is corrected. The CNOS parameters remain unchanged at the local LU, or both the local and partner LUs, depending on when the failure occurred.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>(all)</td>
<td>USF6CRRJ</td>
<td>COMMAND RACE REJECT</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because two CNOS operations caused contention for the needed resources.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0000'</td>
<td>USF6CRPR</td>
<td>PARTNER GRANTED RETRY</td>
</tr>
</tbody>
</table>

Both LUs initiated a CNOS negotiation for the same mode at the same time. The partner LU will try the CNOS request again. VTAM fails the CNOS request from the local LU.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0001'</td>
<td>USF6CRLR</td>
<td>CONTROL OPERATOR FOR LOCAL LU RETRIED</td>
</tr>
</tbody>
</table>

Both LUs initiated CNOS processing for the same mode at the same time. VTAM failed the partner’s CNOS attempt, and the local LU was given permission to try the CNOS request again. VTAM attempted CNOS processing again but the subsequent CNOS negotiation failed as well. VTAM was forced to fail the local LU’s CNOS request.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0010'</td>
<td>X'0002'</td>
<td>USF6PCIP</td>
<td>PARTNER CNOS IN PROGRESS</td>
</tr>
</tbody>
</table>

The partner LU has already begun processing a CNOS for the same mode name, and its processing will continue uninterrupted. The application program must reissue this APPCCMD for it to be processed.
The CNOS negotiation cannot be attempted at this time because the partner LU has initiated a CNOS request for the same mode. The partner LU might be a single-session-capable LU. The local LU cannot issue a CNOS request until the CNOS request initiated by the partner LU completes.

A partner LU that provides only single-session support is currently initiating a session. Because only one session can be active at a time, the application program's CNOS request is rejected. The application program can try the CNOS command again later.

The remote transaction program issued a DEALLOCATE verb, as defined in the LU 6.2 architecture, specifying the TYPE(ABEND_PROG) parameter, or the remote LU did so because of a remote transaction program abend condition. If the conversation for the remote transaction program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application and not yet received by the remote transaction program was purged. This return code can be reported to the local application on any APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_PROGRAM condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If a DEALLOCATE_ABEND_PROGRAM RCPRI code is returned to the application along with LOGRCV=YES, the conversation is then ended. If a DEALLOCATE_ABEND_PROGRAM RCPRI code is returned to the application along with LOGRCV=NO, the conversation is ended.

The remote transaction program issued a DEALLOCATE verb, as described in the LU 6.2 architecture, specifying the TYPE(ABEND_SVC) parameter. If the conversation for the remote transaction program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application and not yet received by the remote transaction program was purged. This return code can be reported to the local application on any
APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_SERVICE condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If a DEALLOCATE_ABEND_SERVICE RCPRI code is returned to the application along with LOGRCV=YES, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state. If a DEALLOCATE_ABEND_SERVICE RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'001C'</td>
<td>X'0000'</td>
<td>USF6DABT</td>
<td>DEALLOCATE ABEND TIMER</td>
</tr>
</tbody>
</table>

The remote transaction program issued a DEALLOCATE verb, as described in the LU 6.2 architecture, specifying the TYPE(ABEND_TIMER) parameter. If the conversation for the remote program was in a state in which information can be received when the DEALLOCATE was issued, information sent by the local application program and not yet received by the remote transaction program was purged. This return code can be reported to the local program on any APPCCMD macroinstruction that can process the error notification on a half-duplex conversation. This return code can only be reported on an APPCCMD CONTROL=RECEIVE on a full-duplex conversation. The error indicator sent by the partner LU to specify the DEALLOCATE_ABEND_TIMER condition can specify that error log data follows the error indicator. The error log data indicator is returned to the application program in the LOGRCV field of the completed macroinstruction. If a DEALLOCATE_ABEND_TIMER RCPRI code is returned to the application along with LOGRCV=YES, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state. If a DEALLOCATE_ABEND_TIMER RCPRI code is returned to the application along with LOGRCV=NO, the conversation is in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0020'</td>
<td>X'0000'</td>
<td>USF6CNSR</td>
<td>CNOS FAILURE, RETRY</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and a conversation was begun with the partner LU. However, a failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session outage, such as a line failure or a modem failure. The condition is temporary, and the application can try the transaction again.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0024'</td>
<td>X'0000'</td>
<td>USF6LRBE</td>
<td>LOGICAL RECORD BOUNDARY ERROR</td>
</tr>
</tbody>
</table>

The application program began sending a logical record before the previous logical record was sent in its entirety. The conversation state does not change.
For macroinstructions that use the QUALIFY=DATACON keyword, the data that was to be sent with the confirmation request is held. The application program must either furnish more data to finish the logical record, or truncate the incomplete record. The application cannot immediately send more data to complete the logical record, but must explicitly flush the send buffer and then send data to complete the logical record.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0028'</td>
<td>X'0000'</td>
<td>USF6SLCL</td>
<td>LU MODE SESSION LIMIT CLOSED</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because the partner LU currently will not allow the session limit for the specified mode name to be raised above 0. The session limit remains at 0. This condition is not necessarily permanent; the control operator can try the CNOS transaction again later.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>(all)</td>
<td>USF6PARM</td>
<td>PARAMETER ERROR</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD because one of the RPL, RPL extension, or session limits structure fields specified in the APPCCMD contained a value that was not valid. The PARAMETER_ERROR RCPRI code together with the RCSEC subcodes (X'0000'–X'002D') form the complete return code that is returned to the application. The subcode identifies the specific error. This RCPRI code is returned on the APPCCMD that contained the parameter that was not valid. When this RCPRI code is returned on a conversation APPCCMD macroinstruction (that is, a macroinstruction that does not specify CONTROL=OPRCNTL), the state of the conversation remains unchanged. When this RCPRI code is returned on an APPCCMD CONTROL=OPRCNTL macroinstruction, the local and partner LUs' CNOS parameters are not changed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0000'</td>
<td>USF6IVLU</td>
<td>INVALID LU NAME OR NETWORK IDENTIFIER</td>
</tr>
</tbody>
</table>

The APPCCMD specified an unrecognized partner LU name or network identifier.

This combination of return codes might result if VTAM does not find the LU name for a partner in the LU-mode table. The partner LU name and the logon mode name are added to the dynamically built LU-mode table during CNOS negotiation. To initiate CNOS negotiation, the application program issues the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction and specifies the LU name and logon mode (LOGMODE) name to be used during communication.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0001'</td>
<td>USF6IVMD</td>
<td>INVALID MODE</td>
</tr>
</tbody>
</table>
The APPCCMD specified an unrecognized logmode name, or the logmode name is not allowed for the LU-LU pair.

This combination of return codes might occur if the LU name specified for a conversation allocation request is present in the LU-mode table but the logon mode name is not present. The partner LU name and the logon mode name are added to the dynamically built LU-mode table during CNOS negotiation. To initiate CNOS negotiation, the application program issues the APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction and specifies the LU name and logon mode (LOGMODE) name to be used during communication.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0002'</td>
<td>USF6IVCI</td>
<td>INVALID CONVERSATION</td>
</tr>
</tbody>
</table>

The APPCCMD specified an unassigned conversation ID, or the RPL used for the request specified an ACB other than the one associated with the conversation assigned that CONVID. The value specified might have been a valid CONVID, but the conversation might not be active.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0003'</td>
<td>USF6IVLL</td>
<td>INVALID LL</td>
</tr>
</tbody>
</table>

The data provided by the application program on an APPCCMD CONTROL=SEND, an APPCCMD CONTROL=PREPRCV, or an APPCCMD CONTROL=DEALLOC macroinstruction was not valid. It contained a logical record length (LL) value of X'0000', X'0001', X'8000', or X'8001'. An LL value of hex 0001, which indicates that the data contains a presentation services (PS) header for sync point, is allowed only on conversations with a synchronization level of sync point.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0004'</td>
<td>USF6IVSV</td>
<td>INVALID VALUES FOR SNASVCMG MODE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and the values specified for the SESSLIM, MINWINL, and MINWINR do not specify (2,1,1) or (0,0,0), respectively.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0005'</td>
<td>USF6IVDL</td>
<td>INVALID DRAINL CHANGE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued, NBRMODE=ONE and DRAINL=YES were specified, the session limit in effect when the APPCCMD was issued was 0, and DRAINL=NO was in effect when the APPCCMD was issued. (The application program attempted to change DRAINL from NO to YES on an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction when session limits were 0.)
An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction is issued, the SNASVCMG mode name is specified, and either one or more session limits for the mode name group for the partner LU is not 0; or one or more session limits for the mode name group for the partner LU are 0, but draining is enabled.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS or QUALIFY=DEFINE macroinstruction was issued and either the sum of MINWINL plus MINWINR is greater than the SESSLIM value specified, or the sum of DMINWNL plus DMINWNR is greater than the DSESLIM value specified.

The application issued one of the macroinstructions:

- APPCCMD CONTROL=RCVEXP
- APPCCMD CONTROL=RCVF
- APPCCMD CONTROL=RECEIVE,OPTCD=XBUFLST
- APPCCMD CONTROL=OPRCNTL,QUALIFY=ACTSESS
- APPCCMD CONTROL=OPRCNTL,QUALIFY=DISPLAY
- APPCCMD CONTROL=OPRCNTL,QUALIFY=RESTORE
- APPCCMD CONTROL=TESTSTAT.

The data area or data length was not suitable as indicated in the items:

**RECEIVE,OPTCD=XBUFLST**

The area specified is not large enough to hold one extended buffer list entry.

**RCVEXP**

Data area is too small to contain all the expedited data.

**RCVF**

Data area is too small to contain the next available FMH-5.

**QUALIFY=ACTSESS**

Data length indicated in the supplied session parameters was larger than the amount of data provided or exceeds the maximum size allowed.

**QUALIFY=DISPLAY**

Data area is too small to contain the DEFINE/DISPLAY (ISTSLD) structure.

**QUALIFY=RESTORE**

Data area is too small to contain the RESTORE (ISTREST) structure.
Data area is too small to contain the status data structure (ISTSTATD).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0009'</td>
<td>USF6INSL</td>
<td>INCOMPLETE STRUCTURE SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:
- APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS
- APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS
- APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE.

The data length was not suitable as indicated in the:

**QUALIFY=ACTSESS**
Data length provided was less than the minimum size for the session parameters.

**QUALIFY=CNOS**
Data length provided was less than the minimum size for the session limits structure (ISTSLCNS).

**QUALIFY=DEFINE**
Data length provided was less than the minimum size for the DEFINE/DISPLAY (ISTSLD) structure.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000A'</td>
<td>USF6INFM</td>
<td>INCOMPLETE FMH5 SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued APPCCMD CONTROL=ALLOC, but did not supply an entire FMH-5.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000B'</td>
<td>USF6INGD</td>
<td>INCOMPLETE GDS VARIABLE SUPPLIED</td>
</tr>
</tbody>
</table>

The application program issued an abnormal termination APPCCMD deallocation macroinstruction, but did not supply an entire GDS variable.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000C'</td>
<td>USF60EXT</td>
<td>ZERO EXIT FIELD</td>
</tr>
</tbody>
</table>

The RPL specified that the ECB-EXIT field is being used as an EXIT field, but the RPL exit routine address in the field is 0. No RPL exit routine has been scheduled.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'000D'</td>
<td>USF60ECB</td>
<td>ZERO ECB FIELD</td>
</tr>
</tbody>
</table>
The RPL specified that the ECB-EXIT field is being used to point to an external ECB, but the address in the field is 0. No ECB has been posted.

RCPRI  RCSEC  ISTUSFBC   EQU Label   Meaning
X'002C'  X'000E'  USF6RIAS  REQUEST INVALID FOR ADDRESS SPACE

An internal error occurred.

RCPRI  RCSEC  ISTUSFBC   EQU Label   Meaning
X'002C'  X'000F'  USF6CBIN  CONTROL BLOCK INVALID

The RPL’s ACB field does not contain the address of a valid ACB or the ACB is closed.

RCPRI  RCSEC  ISTUSFBC   EQU Label   Meaning
X'002C'  X'0010'  USF6INDL  INVALID DATA ADDRESS OR LENGTH

An APPCCMD was issued that specified a work area address that is beyond the addressable range of the application program.

If using a buffer list or extended buffer list to send data, check entries to ensure that the length field does not contain any negative values.

RCPRI  RCSEC  ISTUSFBC   EQU Label   Meaning
X'002C'  X'0011'  USF6PRVO  PREVIOUS MACROINSTRUCTION OUTSTANDING

An APPCCMD is issued that specifies a conversation resource while an outstanding macroinstruction that targets the same conversation and processes on the same conversation queue is pending completion, or an APPCCMD CONTROL=OPRCNTRL is issued while an outstanding operator control APPCCMD that targets the same LU is pending completion. Wait until the first macroinstruction completes or coordinate this request with the one that is outstanding.

RCPRI  RCSEC  ISTUSFBC   EQU Label   Meaning
X'002C'  X'0012'  USF6BLIV  BUFFER LIST LENGTH INVALID

The RECLEN field of the RPL was not valid.

For the macroinstructions, the RECLEN field must be a nonzero multiple of 16:
- APPCCMD CONTROL=DEALLOC, OPTCD=BUFLST
- APPCCMD CONTROL=PREPRCV, OPTCD=BUFLST
- APPCCMD CONTROL=SEND, OPTCD=BUFLST
- APPCCMD CONTROL=SENDEXPD, OPTCD=BUFLST
• APPCCMD CONTROL=SENDRCV, OPTCD=BUFFLST.

For the macroinstructions, the RECLLEN field must be a nonzero multiple of 48:
• APPCCMD CONTROL=DEALLOC, OPTCD=XBUFLST
• APPCCMD CONTROL=PREPRCV, OPTCD=XBUFLST
• APPCCMD CONTROL=SEND, OPTCD=XBUFLST

For the APPCCMD CONTROL=SENDRCV, OPTCD=XBUFLST macroinstruction, the value for RECLLEN minus 16 must be a nonzero multiple of 48.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0013'</td>
<td>USF6NOMD</td>
<td>NO CORRESPONDING MODE IN LM TABLE</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:
• APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY
• APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

The application program also specified a mode name for which no corresponding entry exists in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0014'</td>
<td>USF6IVBP</td>
<td>INVALID BIND PARAMETERS</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS and specified a set of BIND parameters that were not valid, or the parameters in the BIND that was received were not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0015'</td>
<td>USF6IVTP</td>
<td>INVALID TPN</td>
</tr>
</tbody>
</table>

The application program issued an APPCCMD CONTROL=ALLOC with an FMH-5 that contained a transaction program name that was reserved or not valid, such as X'06F1', which is the SNA service transaction program for the control operator.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0016'</td>
<td>USF6NOLU</td>
<td>NO CORRESPONDING LU IN LM TABLE</td>
</tr>
</tbody>
</table>

The application program issued one of the macroinstructions:
• APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY
• APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE.

The application program also specified an LU name for which no corresponding entry exists in the LU-mode table.
The application program issued an APPCCMD CONTROL=OPRCNTL, QUALIFY=DEFINE macroinstruction and specified mode name SNASVCMG.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued and one of the session limit fields was an incorrect value.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued in order to initialize the SNASVCMG mode. However, it was already initialized, and no action was taken.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued against all the mode names of the LU specified. However, the partner LU is single-session capable. Therefore, an APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction must be issued against a specific mode name.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued for the SNASVCMG or CPSVCMG mode name. However, the partner LU is single-session capable, and the SNASVCMG or CPSVCMG is not allowed.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction was issued for a partner LU that is single-session capable. However, another of the LU’s
mode names is already initialized to nonzero session limits, and only one mode name can have nonzero session limits at a time.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'001E'</td>
<td>USF6CIDI</td>
<td>CID INVALID</td>
</tr>
</tbody>
</table>

The RPL’s ARG field does not contain a valid session identifier (CID). You might have inadvertently modified the field or failed to set it in the first place, or you might have used the CID of a session that no longer exists.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'001F'</td>
<td>USF6APNA</td>
<td>APPCCMD ISSUED FOR NON-APPC</td>
</tr>
</tbody>
</table>

The application issued an APPCCMD against a non-LU 6.2 session or resource. The APPCCMD is rejected.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0020'</td>
<td>USF6PRRO</td>
<td>PREVIOUS REJECT REQUEST OUTSTANDING</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=REJECT request was issued. However, a previous APPCCMD CONTROL=REJECT request has already been issued for the same resource. The later APPCCMD CONTROL=REJECT was rejected.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0021'</td>
<td>USF6DARJ</td>
<td>ABNORMAL DEALLOCATE REJECTED, RETRY</td>
</tr>
</tbody>
</table>

One of the macroinstructions was issued:

- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER.

However, a prior macroinstruction that cannot be canceled is outstanding. The command is not allowed in this case and is rejected. This command also is not allowed to be issued when the conversation is in RECEIVE state and no data has been received for the conversation. APPCCMD CONTROL=REJECT, QUALIFY=CONV can be issued to terminate the conversation and session in this case.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0022'</td>
<td>USF6IVCQ</td>
<td>INVALID CONTROL OR QUALIFY VALUE</td>
</tr>
</tbody>
</table>
An undefined value for the CONTROL or QUALIFY keyword was specified, or a QUALIFY value is not valid to use with the specified CONTROL value. For CONTROL types that do not use a QUALIFY value, RPL6QUAL must be set to 0.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0023'</td>
<td>USF6INSI</td>
<td>INVALID SESSION INSTANCE IDENTIFIER</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=REJECT, QUALIFY=SESSION request or an APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND request or an APPCCMD CONTROL=SETSESS, QUALIFY=RESUME request because the local application specified:
- A session instance identifier for a session that was not active at the time of the request.
- A session ID length that was not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0024'</td>
<td>USF6PSHI</td>
<td>PS HEADER NOT SUPPLIED</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=SEND request because the local application did not supply a complete PS header. (For example, the PS header length and data are missing.)

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0025'</td>
<td>USF6PSLI</td>
<td>PS HEADER LENGTH IS INSUFFICIENT</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=SEND request because the local application specified an insufficient PS header length (the length equals 0).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0026'</td>
<td>USF6NMSC</td>
<td>SESSION INSTANCE IDENTIFIER AND CONVERSATION IDENTIFIER MISMATCH</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND request because the application program requested a session with APPCCMD CONTROL=SETSESS, QUALIFY=SUSPEND, but the conversation identified by CONVID was not currently assigned to the session identified by SESSID. VTAM rejected the request and nothing was suspended.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0027'</td>
<td>USF6IDET</td>
<td>INVALID DEACTIVATION TYPE CODE</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=REJECT, QUALIFY=SESSION request because the local application program omitted the DEACTYP parameter or
specified an UNBIND deactivation type code value other than cleanup (X'0F') or protocol violation (X'FE'). The session has been successfully deactivated with UNBIND (X'0F').

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0028'</td>
<td>USF6NCRY</td>
<td>CRYPTOGRAPHY NOT ALLOWED ON MODE</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=SEND, an APPCCMD CONTROL=PREPRCV, or an APPCCMD CONTROL=DEALLOC macroinstruction is rejected because CRYPT=YES is specified, and the mode does not support encryption.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0029'</td>
<td>USF6INLI</td>
<td>INVALID LIST VALUE SPECIFIED ON APPCCMD FOR RESTORE</td>
</tr>
</tbody>
</table>

The value for the LIST field in the RPL is not equal to NONE, ALL, or NOSESS. The keyword LIST=ALL, LIST=NONE, or LIST=NOSESS can be specified on the APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002A'</td>
<td>USF6INC</td>
<td>INVALID CGID VALUE SPECIFIED</td>
</tr>
</tbody>
</table>

A macroinstruction was issued specifying CONVGRP, but the conversation group ID (CGID) was not valid. You might have unintentionally modified the field, failed to set it correctly, or used a CGID that corresponds to a session that no longer exists.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002B'</td>
<td>USF6NONI</td>
<td>NETWORK-QUALIFIED NAME REQUIRED</td>
</tr>
</tbody>
</table>

NETID was not coded on the APPCCMD although PARMS=(NQNAMES=YES) was coded on the ACB macroinstruction.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'002C'</td>
<td>USF6INEL</td>
<td>PARAMETER ERROR - INVALID EXPEDITED DATA LENGTH</td>
</tr>
</tbody>
</table>

An APPCCMD CONTROL=SENDEXPD was issued that specified an expedited data length of 0 or an expedited data length greater than the allowed maximum. The largest expedited data size that can be sent with one macroinstruction invocation is 86 bytes.
An APPCCMD CONTROL=DEALLOC | DEALLOCQ,QUALIFY=ABNDUSER was specified with a sense code that was not an allocation or abnormal deallocation sense code value.

The application supplied VTAM with a vector area address that is not valid or is write-protected.

The application supplied VTAM with a vector area that is smaller than the minimum required size.

A storage type indication was not supplied or is not valid. Storage type is required to be specified via the ISTAPC82 mapping DSECT that is mapped within the ISTAPCVL mapping DSECT.

The APPCCMD CONTROL=SENDRCV was issued without specifying a buffer. OPTCD=BUFLST | XBUFLST is required for this macroinstruction.

An unexpected vector was provided on an APPCCMD request. An input vector is not defined for the APPCCMD.
### Label Meaning

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0033'</td>
<td>USF6VNPV</td>
<td>PARAMETER_ERROR—A_REQUIRED_VECTOR_WAS_NOT_PROVIDED_OR_SPECIFIED_INCORRECTLY</td>
</tr>
</tbody>
</table>

A required input vector was either not provided or specified incorrectly on an APPCCMD request.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'002C'</td>
<td>X'0034'</td>
<td>USF6LNSP</td>
<td>PASSWORD_SUBSTITUTION_VALUE_SET_IN_ERROR</td>
</tr>
</tbody>
</table>

The FMH-5 received from the application indicated password substitution in byte 4, bit 3. The session established with the partner does not support password substitution. Reissue the macroinstruction with this bit setting off.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0030'</td>
<td>X'0000'</td>
<td>USF6PENT</td>
<td>PROGRAM_ERROR_NO_TRUNCATION</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb specifying the TYPE(PROG) parameter; the conversation for the remote program was in a sending state; and the LU 6.2 SEND_ERROR verb did not truncate a logical record. No truncation occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in RECEIVE state before receiving any logical records or after sending a complete logical record. This return code is reported to the local application program when it issues an APPCCMD CONTROL=RECEIVE macroinstruction prior to receiving any logical records or after receiving one or more complete logical records.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0034'</td>
<td>X'0000'</td>
<td>USF6PEPU</td>
<td>PROGRAM_ERROR_PURGING</td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(PROG) parameter, and the conversation for the remote transaction program was in RECEIVE state. The LU 6.2 SEND_ERROR verb might have caused information to be purged. Purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in RECEIVE state before receiving all the information sent by the local application, that is, all the information sent prior to the reporting of the PROGRAM_ERROR_PURGING return code to the local application. The purging can occur at the local LU, the remote LU, or both. No purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb in a CONFIRM state, or in RECEIVE state after receiving all the information sent by the local application. This RCPRI code is normally reported to the local application on an APPCCMD it issues after receiving some information to the remote transaction program. However, the RCPRI code can be reported on an APPCCMD the application issues prior to sending any information, depending on the CONTROL and QUALIFY fields of the APPCCMD and when it is issued. The conversation is in RECEIVE state.
The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying
the TYPE(PROG) parameter; the conversation for the remote transaction program
was in a sending state; and the LU 6.2 SEND_ERROR verb truncated a logical
record. Truncation occurs when a transaction program begins sending a logical
record and then issues the LU 6.2 SEND_ERROR verb before sending the complete
logical record. This return code is reported to the local application on an
APPCCMD CONTROL=RECEIVE macroinstruction issued after receiving the
truncated logical record. The conversation state is unchanged.

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying
the TYPE(SVC) parameter; the conversation for the remote transaction program
was in a sending state; and the LU 6.2 SEND_ERROR verb did not truncate a
logical record. No truncation occurs when a transaction program issues the LU 6.2
SEND_ERROR verb before sending any logical records or after sending a complete
logical record. This return code is reported to the local application on an
APPCCMD CONTROL=RECEIVE macroinstruction it issues prior to receiving any
logical records or after receiving one or more complete logical records. The
conversation state is unchanged.

The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying
the TYPE(SVC) parameter, and the conversation for the remote transaction
program was in RECEIVE state. The LU 6.2 SEND_ERROR verb might have
caused information to be purged. Purging occurs when a transaction program
issues the LU 6.2 SEND_ERROR verb in RECEIVE state before receiving all the
information sent by the local application, that is, all the information sent prior to
the reporting of the SERVICE_ERROR_PURGING return code to the local
application. The purging can occur at the local LU, the remote LU, or both. No
purging occurs when a transaction program issues the LU 6.2 SEND_ERROR verb
in a CONFIRM state, or in RECEIVE state after receiving all the information sent
by the local application. This return code is normally reported to the local
application on an APPCCMD it issues after sending some information to the
remote transaction program. However, the return code can be reported on an
APPCCMD the application issues prior to sending any information, depending on
the CONTROL and QUALIFY fields of the APPCCMD and when it is issued. The
conversation is in RECEIVE state.

Note: This code is never reported on an APPCCMD issued on a full-duplex
conversation.
The remote transaction program issued an LU 6.2 SEND_ERROR verb, specifying the TYPE(SVC) parameter; the conversation for the remote transaction program was in a sending state; and the LU 6.2 SEND_ERROR verb truncated a logical record. Truncation occurs when a program begins sending a logical record and then issues the LU 6.2 SEND_ERROR verb before sending the complete logical record. This return code is reported to the local application on an APPCCMD CONTROL=RECEIVE macroinstruction issued after receiving the truncated logical record. The conversation state is unchanged.

A failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session protocol error. The condition is not temporary, and the application should not try the transaction again until the condition is corrected. The conversation is in END_CONV or FDX_RESET state if no log data is present. If log data is present, the conversation is in PEND_END_CONV_LOG or PEND_RESET_LOG state.

Two common failures are:

- Local LU sends unexpected control information.
  For example, the conversation can be in PENDING_DEALLOCATE state, but something other than a deallocate is received, or an FMH-7 is not received when it is expected.
- Local LU sends unexpected data on the conversation.
  For example, a logical record that is not valid, PS header or FMH-7, might have been received, or a logical record is truncated by something other than an FMH-7.

A failure occurred that caused the conversation to be prematurely terminated. For example, the session being used for the conversation was deactivated because of a session outage, such as a line failure or a modem failure. The application can try the transaction again when the error that caused the session outage has been corrected. The conversation is in END_CONV or FDX_RESET state.

The specified conversation was not in an appropriate state to issue the specified APPCCMD. For example, the application program issued APPCCMD.
CONTROL=SEND, QUALIFY=DATA, but the conversation was in RECEIVE state. The state of the conversation remains unchanged.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0054'</td>
<td>X'0000'</td>
<td>USF6URMD</td>
<td>UNRECOGNIZED MODE NAME</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS macroinstruction did not execute successfully because the partner LU does not recognize the specified mode name. The local and partner LUs’ CNOS parameters are not changed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0058'</td>
<td>X'0000'</td>
<td>USF6UNSC</td>
<td>UNSUCCESSFUL, SESSION NOT AVAILABLE</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=ALLOC, QUALIFY=IMMED macroinstruction issued by the local application program did not execute successfully because there was not a contention-winner session available for use by a new conversation request. This RCPRI code is returned on the unsuccessful APPCCMD.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'005C'</td>
<td>(all)</td>
<td>USF6UECR</td>
<td>USER ERROR CODE RECEIVED</td>
</tr>
</tbody>
</table>

An FMH-7 was received that contained a sense code not interpreted by VTAM. The unrecognized sense code is passed to the application program through the SENSE field in the RPL extension. The application program must determine whether the sense code is a valid user-supplied sense code or a code that is not valid. The USER_ERROR_CODE_RECEIVED RCPRI code together with the RCSEC subcodes (X'0000' X'0001') form the complete return code that is returned to the application. The subcode specifies whether a negative response preceded the FMH-7 containing the unrecognized sense code. The conversation is in a receiving state.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'005C'</td>
<td>X'0000'</td>
<td>USF6FNGR</td>
<td>NEGATIVE RESPONSE</td>
</tr>
</tbody>
</table>

The FMH-7 containing the unrecognized sense code was received by VTAM after the receipt of a negative response.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'005C'</td>
<td>X'0001'</td>
<td>USF6WNGR</td>
<td>WITHOUT NEGATIVE RESPONSE</td>
</tr>
</tbody>
</table>

The FMH-7 containing the unrecognized sense code was not preceded by a negative response.
The application issued an APPCCMD CONTROL=RCVFMH5, but there is currently no FMH-5 waiting to be received by the application program.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS macroinstruction did not execute successfully because activation for the pending active session failed. For example, the path between the application and the other LU could have been lost.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS macroinstruction did not execute successfully because activating the pending active session would have caused the session limits for the mode name group to be exceeded.

An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS or QUALIFY=DACTSESS macroinstruction was issued for a session that is no longer pending. The CID for the session is valid but a BIND or CINIT is no longer queued, or the session is being deactivated due to a previous error or request.

VTAM is unable to process the request because of a temporary storage shortage, a resource shortage, or other shortage.

- If a sense code is not provided, a temporary storage shortage has occurred.
- If a sense code is provided indicating insufficient resources, then a storage shortage or other resource shortage has occurred. In either of these cases, the request can be reissued (with EXECRPL, for example.) There is no state change. This return code is reported to the application program to allow time for the problem to diminish or disappear. If VTAM attempts to try the request again, the additional storage might not be available immediately, and the problem might occur again.
If a sense code is provided other than one for insufficient resources, examine the sense code explanation to determine the action required. In this situation, whether the request can be reissued depends on the information contained in the sense code.

If this return code is received at the completion of an APPCCMD with CONTROL=RECEIVE, OPTCD=(XBUFLST), then a CSM buffer that meets the storage type specified in the XBUFLST-receive vector could not be obtained to receive the data, or other VTAM internal resources required to receive the data could not be obtained. The system is storage constrained. No data is received. The application can take several possible actions:

- Reissue the APPCCMD several times as a temporary try recovery action again.
- Issue a receive without the XBUFLST specification so the data can be copied into application private storage.
- Explicitly deallocate the conversation via APPCCMD services.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0074'</td>
<td>X'0000'</td>
<td>USF6HALT</td>
<td>HALT ISSUED</td>
<td></td>
</tr>
</tbody>
</table>

The operator has issued a HALT command. Depending on the type of HALT, the application program can no longer issue certain macroinstructions.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0078'</td>
<td>X'0000'</td>
<td>USF6VIYA</td>
<td>VTAM INACTIVE FOR YOUR ACB</td>
<td></td>
</tr>
</tbody>
</table>

The association between VTAM and the application program (ACB) that was established with the OPEN macroinstruction has been broken (the ACB is in the process of being closed). This might have occurred because:

- The application program has elsewhere issued a CLOSE that has not yet completed
- VTAM has become inactive
- A VARY NET,INACT command was issued for the application program.

Any active conversations are placed in END_CONV or FDX_RESET state.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'007C'</td>
<td>X'0000'</td>
<td>USF6RQAB</td>
<td>REQUEST ABORTED</td>
<td></td>
</tr>
</tbody>
</table>

VTAM has rejected a request because of an error detected while processing the request or because of an error in the associated session, task, or address space. For example, an abend. An abend might or might not be retried.

<table>
<thead>
<tr>
<th>RCPI</th>
<th>RCSEC</th>
<th>ISTUSFBC</th>
<th>Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0080'</td>
<td>X'0000'</td>
<td>USF6DLNR</td>
<td>DEALLOCATE NORMAL</td>
<td></td>
</tr>
</tbody>
</table>

The remote transaction program issued an LU 6.2 DEALLOCATE TYPE(FLUSH) verb. This return code is reported to the application program on an APPCCMD
CONTROL=SEND, QUALIFY=ERROR macroinstruction issued when the
conversation is in RECEIVE state. The conversation is in END_CONV state. The
conversation can be in RECEIVE state or in PEND_RCV_LOG state. This return
code applies only to half-duplex conversations.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>ISTUSFBCEQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0084'</td>
<td>X'0000'</td>
<td>USF6STSH</td>
<td>STORAGE SHORTAGE</td>
</tr>
</tbody>
</table>

Indicates VTAM has encountered a storage shortage when attempting to satisfy an
APPCCMD CONTROL=RECEIVE or an APPCCMD CONTROL=RCVFMH5, either
while storing incoming data or sending a pacing response. There is no state
change.

This return code can also be issued when a storage failure occurs while processing
an internal DEALLOC FLUSH request. VTAM does internal DEALLOC FLUSH
processing when it receives an indication that the partner has issued an abnormal
deallocation request on the full-duplex conversation.

The application should issue one of the abnormal termination APPCCMD
CONTROL=DEALLOC | DEALLOCQ macroinstructions to end the conversation.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>ISTUSFBCEQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0088'</td>
<td>X'0000'</td>
<td>USF6CREJ</td>
<td>CANCELED BY REJECT OR ABNORMAL DEALLOCATE</td>
</tr>
</tbody>
</table>

The request, while in progress, was canceled by the issuance of an APPCCMD
CONTROL=REJECT or abnormal deallocation APPCCMD, which has requested the
termination of the current conversation and, possibly, the session.

<table>
<thead>
<tr>
<th>RCPR</th>
<th>RCSEC</th>
<th>ISTUSFBCEQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'008C'</td>
<td>X'0000'</td>
<td>USF6PROE</td>
<td>PARTNER COMMITTED PROTOCOL VIOLATION</td>
</tr>
</tbody>
</table>

The partner LU has violated conversation protocols during the execution of this
command. Notification of conversation failure will be received on a subsequent
APPCCMD command. There is no state change.

Two common protocol violations are:

- Partner LU sends unexpected control information.
  For example, the conversation can be in PENDING_DEALLOCATE state, but
  something other than a deallocate is received, or an FMH-7 is not received when
  it is expected.

- Partner LU sends unexpected data on the conversation.
  For example, a logical record that is not valid, PS header or FMH-7, might have
  been received, or a logical record is truncated by something other than an
  FMH-7.
The application program issued an APPCCMD, but the application program has APPC=NO coded on its APPL definition statement. The APPL definition statement must have APPC=YES coded before the application program can issue APPCCMD macroinstructions.

This indicates that the application program issued an APPCCMD that provided data to be sent on a previous QUALIFY=DATAFLU or QUALIFY=DATACON type of send (either CONTROL=SEND, CONTROL=PREPRCV or CONTROL=DEALLOC). However, data remains, held by VTAM, from the error on the previous DATAFLU or DATACON macroinstruction.

Before sending more data, issue a macroinstruction that flushes VTAM's buffers. An APPCCMD CONTROL=SEND, QUALIFY=FLUSH macroinstruction, an APPCCMD CONTROL=SEND, QUALIFY=ERROR macroinstruction, or one of the abnormal termination CONTROL=DEALLOC macroinstructions will flush the send data queue so that processing can continue.

This indicates a temporary storage shortage has occurred while sending data. This RCPRI, RCSEC combination might be returned for one of the macroinstructions:

- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOC, QUALIFY=ABNDUSER
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDPROG
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDSERV
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDTIME
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=ABNDUSER
- APPCCMD CONTROL=DEALLOC, QUALIFY=DATAFLU
- APPCCMD CONTROL=DEALLOC, QUALIFY=DATACON
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=DATAFLU
- APPCCMD CONTROL=DEALLOCQ, QUALIFY=DATACON
- APPCCMD CONTROL=PREPRCV, QUALIFY=DATAFLU
- APPCCMD CONTROL=PREPRCV, QUALIFY=DATACON
- APPCCMD CONTROL=SEND, QUALIFY=DATAFLU
- APPCCMD CONTROL=SEND, QUALIFY=DATACON
- APPCCMD CONTROL=SEND, QUALIFY=ERROR
• APPCCMD CONTROL=SENDRCV, QUALIFY=DATAFLU.

The current position in the application-supplied data buffer (the area pointed to by the AREA field of the RPL) is returned in RPL6STBF (the current buffer) and RPL6STDS (displacement in the data). All data prior to this buffer or buffer list entry has been sent.

The user has two alternatives when this return code is received.

• Attempt to continue sending data on the conversation by issuing an APPCCMD macroinstruction with the data pointers and length set to reflect the values returned in RPL6STBF and RPL6STDS. The subsequent macroinstruction must be issued with the AREA field set with the RPL6STBF value plus the RPL6STDS value to avoid duplicating any data already sent. The data length (the RECLEN field in the RPL) must also be adjusted to indicate the amount of remaining data. Once the subsequent macroinstruction with the updated data location completes successfully, the conversation can be continued as if the storage shortage did not occur.

• Deactivate the conversation by issuing one of the abnormal termination CONTROL=DEALLOC macroinstructions, or APPCCMD CONTROL=REJECT macroinstructions. Note that REJECT must be issued to deactivate a conversation if the abnormal termination CONTROL=DEALLOC macroinstructions are unsuccessful.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'009C'</td>
<td>X'0001'</td>
<td>USF6RSTF</td>
<td>RESTORE REJECTED—RESTORE ISSUED BEFORE SETLOGON START</td>
</tr>
</tbody>
</table>

The APPCCMD CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction is issued before the SETLOGON START macroinstruction is issued.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>(all)</td>
<td>USF6RNAL</td>
<td>REQUEST NOT ALLOWED</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD because the macroinstruction request conflicts in some way with the capabilities of the session or conversation to which it applies. The REQUEST_NOT_ALLOWED RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0001'</td>
<td>USF6LNSE</td>
<td>LU PAIR DOES NOT SUPPORT SENDING EXPEDITED DATA</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD CONTROL=SENDEXPD because the negotiated support level of the current session does not support protocols needed to transmit expedited data.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0002'</td>
<td>USF6RQBL</td>
<td>REQUEST BLOCKED</td>
</tr>
</tbody>
</table>
VTAM rejected the APPCCMD because the conversation with which it is associated is in the process of being deallocated or terminated.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0003'</td>
<td>USF6RNEX</td>
<td>EXECUTION OF REQUEST TERMINATED</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=RCVEXPD, QUALIFY=SPEC on a half-duplex conversation because the partner LU is awaiting a change-direction or end-of-chain indicator before sending error information. No expedited information was available to be received.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0004'</td>
<td>USF6VNVF</td>
<td>CONTROL/QUALIFY VALUE INVALID FOR FULL-DUPLEX CONVERSATION</td>
</tr>
</tbody>
</table>

VTAM rejected the APPCCMD because the CONTROL= and QUALIFY= value combination specified is not allowed for a full-duplex conversation.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0005'</td>
<td>USF6EXRO</td>
<td>RSP HAS NOT BEEN RECEIVED FOR A PREVIOUS SENDEXPD REQUEST</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA or an APPCCMD CONTROL=SEND, QUALIFY=RQSEND because the response to a previously issued APPCCMD CONTROL=SENDEXPD,QUALIFY=DATA had not been received from the partner LU.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0006'</td>
<td>USF6NAUT</td>
<td>PROGRAM_NOTAUTHORIZED_FOR_REQUESTEDFUNCTION</td>
</tr>
</tbody>
</table>

An application not using VTAM authorized path attempted to use the HPDT interface. The request is disallowed.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A0'</td>
<td>X'0008'</td>
<td>USF6ENEL</td>
<td>NAMED RESOURCE NOT ELIGIBLE FOR REQUESTED ALTERATION</td>
</tr>
</tbody>
</table>

A MODIFY DEFINE command with DELETE=UNUSE was issued for an entry in the LU-mode table, but the entry type is not UNUSABLE.
An APPC CMD macroinstruction is issued with a mode name that is pending recovery for persistent LU-LU sessions. Issue the APPC Macrocontrol CONTROL=OPRCNTL, QUALIFY=RESTORE macroinstruction to restore the mode.

A macroinstruction has failed for some reason related to the system environment in which the request was processed. The RCSEC subcode identifies the specific error.

A macroinstruction request required the use of an operating system service which is not supported by the active operating system level.

VTAM attempted to suspend processing of an APPCCMD macroinstruction issued in either cross-memory mode or in synchronous SRB-mode with OPTCD=KEEPSRB specified. The attempt failed, probably due to conditions in the operation system environment. The application may reissue the request.

VTAM attempted to resume processing of an APPCCMD macroinstruction issued in either cross-memory mode or in synchronous SRB-mode with OPTCD=KEEPSRB specified. The attempt failed. VTAM is unable to post the request complete. If the application has a LOSTERM exit, it will be scheduled with a reason code of 44. For more information about the LOSTERM exit, see z/OS Communications Server: SNA Programming. The RPL is now available for reuse.

| RCPRI | RCSEC | ISTUSFBC EQU Label | Meaning |}
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00A4'</td>
<td>X'0000'</td>
<td>USF6SPMD</td>
<td>MODE MUST BE RESTORED BEFORE USING</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>(all)</td>
<td>USF6ENVE</td>
<td>ENVIRONMENT ERROR</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0000'</td>
<td>USF6OSLV</td>
<td>OS LEVEL DOES NOT SUPPORT REQUESTED FUNCTION</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0001'</td>
<td>USF6XMES</td>
<td>SUSPEND FAILURE</td>
</tr>
<tr>
<td>X'00A8'</td>
<td>X'0002'</td>
<td>USF6XMER</td>
<td>RESUME FAILURE</td>
</tr>
<tr>
<td>X'00AC'</td>
<td>(all)</td>
<td>USF6ERIN</td>
<td>ERROR INDICATION RECEIVED</td>
</tr>
</tbody>
</table>
VTAM's processing of an APPCCMD request stored on the SEND queue of a full-duplex conversation was ended because the remote transaction program or LU issued an LU 6.2 architecture verb that canceled further processing of the request. An associated Secondary Return Code value indicates the type of operation that caused the request to be ended.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0001'</td>
<td>USF6EIAS</td>
<td>DEALLOCATE ABEND PROGRAM</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_PROG).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0002'</td>
<td>USF6ERAS</td>
<td>DEALLOCATE ABEND SERVICE</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_SVC).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0003'</td>
<td>USF6EAT</td>
<td>DEALLOCATE ABEND TIME</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that the remote transaction program issued a DEALLOCATE verb with TYPE(ABEND_TIMER).

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0004'</td>
<td>USF6EIAT</td>
<td>ALLOCATION ERROR</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code indicating that an allocation request was rejected by the remote transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0005'</td>
<td>USF6EIUN</td>
<td>UNKNOWN ERROR CODE</td>
</tr>
</tbody>
</table>
An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because an abnormal deallocation request was issued by the remote transaction program. The FMH-7 received from the partner LU carried a sense code other than the Deallocate ABEND, Allocation Error, or Resource Failure codes. The application program must determine whether the sense code is a valid user-supplied sense code or is a code that is not valid.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0006'</td>
<td>USF6EIRR</td>
<td>RESOURCE FAILURE, RETRY</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because a failure occurred that caused the conversation to be prematurely terminated. The application can try the transaction again when the error that caused the session outage has been corrected.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00AC'</td>
<td>X'0007'</td>
<td>USF6EIRN</td>
<td>RESOURCE FAILURE, NO RETRY</td>
</tr>
</tbody>
</table>

An APPCCMD that processes on the SEND queue of a full-duplex conversation was terminated because a failure occurred that caused the conversation to be prematurely terminated. The condition is not temporary, and the application should not try the transaction again until the condition is corrected.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'(all)'</td>
<td>USF6NRER</td>
<td>NAME RESOLUTION ERROR</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because there was an inappropriate name translation. The NAME_RESOLUTION_ERROR RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0001'</td>
<td>USF6NRRE</td>
<td>LUNAME FOUND IN A VARIANT_NAME ENTRY</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because the LUNAME specified on the macroinstruction was found in a VARIANT_NAME entry in the LU-mode table.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B0'</td>
<td>X'0002'</td>
<td>USF6NRRD</td>
<td>NAME RETURNED DIFFERS FROM ASSOCIATED NAME</td>
</tr>
</tbody>
</table>

VTAM rejected an APPCCMD because the BIND RSP contained an LUNAME that is different from the associated name in the SUPPLIED_NAME entry in the LU-mode table. The association of names for the partner LU had previously occurred.
VTAM rejected an APPCCMD because the LUNAME returned in the BIND RSP was found in a VARIANT_NAME entry in the LU-mode table. The association of names for the partner LU has not occurred.

VTAM rejected an APPCCMD because the LUNAME contained in the BIND RSP was found in a SUPPLIED_NAME entry in the LU-mode table. The SUPPLIED_NAME entry was different than the entry used in the session initiation.

VTAM rejected an APPCCMD because the NETID contained in the BIND RSP was different than that previously saved in the LU-mode table for that LUNAME.

VTAM rejected an APPCCMD because the LUNAME specified on the macroinstruction was found in an UNUSABLE_NAME entry in the LU-mode table.

VTAM rejected an APPCCMD because the partner LU returned an LUNAME in the BIND response that was found in an UNUSABLE_NAME entry in the LU-mode table.

VTAM rejected an APPCCMD macroinstruction request or an operator command because the LU name specified is a DISASSOCIATED_NAME entry. This type of
entry has no mode values and thus has no sessions. The LU name was previously a VARIANT_NAME entry but is no longer associated with a SUPPLIED_NAME entry.

If the request or operator command was to display information about the LU, reissue the request with LOGMODE=0 and any LU-specific information will be returned.

If the request was for an allocate, a CNOS must be issued to establish mode information before the allocate can be retried.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>(all)</td>
<td>USF6CSME</td>
<td>CSM_DETECTED_ERROR</td>
</tr>
</tbody>
</table>

CSM detected an error. The CSM_DETECTED_ERROR RCPRI code together with one of the RCSEC subcodes form the complete return code that is returned to the transaction program.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0001'</td>
<td>USF6NSPC</td>
<td>CSM_DETECTED_ERROR—NOT_SPECIFIED</td>
</tr>
</tbody>
</table>

CSM detected a problem during APPCCMD processing of the request. The specific reason for the error is not passed back to the APPCCMD application.

Upon receipt of this return code the application can:

- Optionally consider the error temporary and try the request again several times. Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.
- Consider the error permanent and terminate the conversation.

Refer to z/OS Communications Server: CSM Guide for more information about these CSM errors.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0002'</td>
<td>USF6IBTK</td>
<td>CSM_DETECTED_ERROR—INVALID_BUFFER_TOKEN_SPECIFIED</td>
</tr>
</tbody>
</table>

The communications storage manager (CSM) detected a problem during APPCCMD processing of the request. The specific reason for the error is that CSM detected that the CSM buffer token being used for the APPCCMD is not a valid CSM buffer token.

Upon receipt of this return code the application can:

- Check the current buffer pointer (RPL6STBF) in the RPL extension to determine the address of the buffer list entry that was processed when the error occurred.
- Optionally consider the error temporary and try the request again several times.
Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.

- Consider the error permanent and terminate the conversation.
- Continue using the conversation with a different CSM buffer.

Refer to [z/OS Communications Server: CSM Guide](#) for more information about these CSM errors.

<table>
<thead>
<tr>
<th>RCPRI</th>
<th>RCSEC</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00B4'</td>
<td>X'0003'</td>
<td>USF6IID</td>
<td>CSM_DETECTED_ERROR—INVALID_INSTANCE_ID_SPECIFIED</td>
</tr>
</tbody>
</table>

The communications storage manager (CSM) detected a problem during APPCCMD processing of the request. The specific reason for the error is that CSM detected that the instance ID portion of the CSM buffer token being used for the APPCCMD is not a valid CSM instance ID. Because the instance ID is not valid, it is possible that the CSM buffer being specified on the APPCCMD has been previously freed and a new instance ID has been assigned to the storage by CSM.

Upon receipt of this return code the application can:
- Check the current buffer pointer (RPL6STBF) in the RPL extension to determine the address of the buffer list entry that was processed when the error occurred.
- Optionally consider the error temporary and try the request again several times.
  Note that it is possible that the error may not recur. This temporary error condition could occur in the case where a VTAM-built parameter list to CSM is randomly corrupted on a particular request, but not on a subsequent request.
- Consider the error permanent and terminate the conversation.
- Continue using the conversation with a different CSM buffer.

Refer to [z/OS Communications Server: CSM Guide](#) for more information about these CSM errors.

**RTNCD, FDB2 information for LU 6.2**

While most of the LU 6.2 feedback information from errors is found in the RCPRI and RCSEC fields, some error return codes in the RPL RTNCD and FDB2 fields are meaningful for LU 6.2 applications. The X'00', X'0B' combination in the RPL indicates some problem might have occurred while the macroinstruction was executing, RCPRI and RCSEC should be used for further diagnosis. The other RTNCD, FDB2 combinations refer to attempts to start an LU 6.2 session independent of VTAM or attempts to use non-APPCCMD macroinstructions for APPCCMD functions. The following information shows the relevant codes.

<table>
<thead>
<tr>
<th>RTNCD</th>
<th>FDB2</th>
<th>ISTUSFBC EQU Label</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>X'0B'</td>
<td>USF6APPC</td>
<td>CONDITIONAL COMPLETION FOR APPCCMD</td>
</tr>
</tbody>
</table>

Some type of error might have occurred on an APPCCMD macroinstruction. For further problem determination, refer to the primary and secondary return codes in the RPL extension. These fields are RPL6RCPR and RPL6RCSC.
A real-to-symbolic translation request is made, and NIBNET is filled in with a network identifier, but VTAM cannot provide a symbolic name. VTAM knows this resource only by its network-qualified name. No symbolic name represents this resource. Do one of the following actions:

- Use the network-qualified name
- Define a symbolic name to represent this resource

An LU 6.2 application program has tried to start an LU 6.2 session independent of VTAM. No pending sessions have been disturbed. This occurs when an OPNDST is issued with an LU 6.2 user-specified BIND.

An LU 6.2 application program has tried to start an LU 6.2 session independent of VTAM. The pending session has been terminated. This occurs when the LOGMODE specified on an OPNDST resolves to an LU 6.2 BIND or when OPNSEC is issued for an LU 6.2 BIND.

An OPNDST or CLSDST has been issued for a pending LU 6.2 session. An APPCCMD CONTROL=OPRCNTL, QUALIFY=ACTSESS or QUALIFY=DACTSESS macroinstruction must be issued for this session.

An application program issued a non-APPCCMD macroinstruction to establish an LU 6.2 session, or issued a non-APPCCMD macroinstruction against a current LU 6.2 session.
Chapter 3. DSECTs

This chapter contains the LU 6.2 DSECTs. For general information on the use and purpose of DSECTs, refer to z/OS Communications Server: SNA Programming.

The DSECTs are shown as an abbreviated form of an assembler listing. The first column indicates the offsets within the DSECT and is the “LOC” column of an assembler listing. The object code, address columns and statement number columns of the listing, however, are not included. The source statements and comments are found next to the offset column. All numbers in the offset column are in hexadecimal.

BIND image (ISTDBIND)

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTDBIND DSECT</td>
</tr>
<tr>
<td>000000</td>
<td>BINFMTY DS C BIND FORMAT AND TYPE</td>
</tr>
<tr>
<td></td>
<td>BINFMT EQU X’F0’ BIND FORMAT</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINFMT (FORMAT)</td>
</tr>
<tr>
<td></td>
<td>BINFMT0 EQU X’00’ FORMAT 0</td>
</tr>
<tr>
<td></td>
<td>BINTYPE EQU X’0F’ BIND TYPE</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINTYPE (TYPE)</td>
</tr>
<tr>
<td></td>
<td>BINNEG0 EQU X’00’ NEGOTIABLE</td>
</tr>
<tr>
<td></td>
<td>BINONEGO EQU X’01’ NON NEGOTIABLE</td>
</tr>
<tr>
<td></td>
<td>BINCOLD EQU X’01’ NON NEGOTIABLE</td>
</tr>
<tr>
<td>000001</td>
<td>BINFM DS C FUNCTION MANAGEMENT</td>
</tr>
<tr>
<td></td>
<td>* PROFILE</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINFM - FUNCTION MANAGEMENT PROFILE</td>
</tr>
<tr>
<td></td>
<td>BINFM19 EQU X’13’ FM PROFILE 19</td>
</tr>
<tr>
<td>000002</td>
<td>BINTS DS C TRANSMISSION SERVICES</td>
</tr>
<tr>
<td></td>
<td>* PROFILE</td>
</tr>
<tr>
<td></td>
<td>* VALUES FOR BINTS (TRANSMISSION SERVICES PROFILE)</td>
</tr>
<tr>
<td></td>
<td>BINTS7 EQU X’07’ SEQ NOS - NO RESET STATE</td>
</tr>
<tr>
<td></td>
<td>BINTS4 EQU X’04’ SEQ NOS - RESET STATE</td>
</tr>
<tr>
<td></td>
<td>BINTS3 EQU X’03’ SEQ NOS - RESET STATE</td>
</tr>
<tr>
<td></td>
<td>BINTS2 EQU X’02’ SEQ NOS - NO RESET STATE</td>
</tr>
<tr>
<td></td>
<td>BINTS1 EQU X’01’ NOT VALID ON LU-LU SESSION</td>
</tr>
<tr>
<td></td>
<td>BINTS0 EQU X’00’ NOT VALID ON LU-LU SESSION</td>
</tr>
<tr>
<td>000003</td>
<td>BINPRIP DS C PRIMARY LU PROTOCOLS FOR SENDING FM DATA</td>
</tr>
<tr>
<td></td>
<td>* SENDING FM DATA</td>
</tr>
<tr>
<td></td>
<td>BINPCHN EQU X’80’ 1 = MULTIPLE RU CHAINS</td>
</tr>
<tr>
<td></td>
<td>0 = SINGLE RU CHAINS</td>
</tr>
<tr>
<td></td>
<td>BINPMCH EQU X’40’ 1 = MULTIPLE OUTSTANDING</td>
</tr>
<tr>
<td></td>
<td>CHAINS (DELAYED)</td>
</tr>
<tr>
<td></td>
<td>* REQUEST MODE</td>
</tr>
<tr>
<td></td>
<td>0 = SINGLE OUTSTANDING</td>
</tr>
<tr>
<td></td>
<td>CHAIN (IMMEDIATE REQUEST MODE)</td>
</tr>
<tr>
<td></td>
<td>BINPCHRNR EQU X’30’ CHAIN RESPONSE PROTOCOL (SEE BINSCHRNR BELOW FOR VALUES)</td>
</tr>
<tr>
<td></td>
<td>BINRSVOI EQU X’0C’ RESERVED</td>
</tr>
<tr>
<td></td>
<td>BINCMP EQU X’02’ 1 = COMPRESSION MAY BE USED</td>
</tr>
<tr>
<td></td>
<td>0 = COMPRESSION MUST NOT BE USED</td>
</tr>
<tr>
<td></td>
<td>BINPSEB EQU X’01’ 1 = PRIMARY MAY SEND EB</td>
</tr>
<tr>
<td></td>
<td>0 = PRIMARY WILL NOT SEND EB</td>
</tr>
<tr>
<td>000004</td>
<td>BINSECP DS C SECONDARY LU PROTOCOLS FOR SENDING FM DATA</td>
</tr>
<tr>
<td></td>
<td>* SENDING FM DATA</td>
</tr>
<tr>
<td></td>
<td>BINSCHRNR EQU X’80’ 1 = MULTIPLE RU CHAINS</td>
</tr>
</tbody>
</table>
* 0 = SINGLE RU CHAIN
BINSCH EQU X'40'
* 1 = MULTIPLE OUTSTANDING
* CHAINS (DELAYED REQUEST MODE)
* 0 = SINGLE OUTSTANDING
* CHAIN (IMMEDIATE REQUEST MODE)
BINSCHNR EQU X'30'
*********************************************************************
* VALUES FOR BINPCHNR/BINSCHNR (TYPE OF RESPONSES ASKED FOR BY REQUESTS FROM PRIMARY/SECONDARY)
*********************************************************************
BINNYRSP EQU X'30' DEFINITE OR EXCEPTION RESPONSE
* BINDFRSP EQU X'20' DEFINITE RESPONSE
BINEXRSP EQU X'10' EXCEPTION RESPONSE
BINNORSP EQU X'00' NO RESPONSE
BINRSV02 EQU X'0C' RESERVED
BINSCH RP X'02' 1 = COMPRESSION MAY BE USED
* 0 = COMPRESSION MUST NOT BE USED
* BINSSEB EQU X'01' 1 = SECONDARY MAY SEND EB
* 0 = SECONDARY WILL NOT SEND EB
* 000005 BINCMPN DS C COMMON LU PROTOCOLS
BINWBREQ EQU X'80' WHOLE-BINS-REQUIRED INDICATOR
* BINFMD EQU X'40' 1 = FM HEADERS MAY BE USED
* 0 = FM HEADERS MUST NOT BE USED
* BINBKRAK EQU X'20' 1 = BRACKETS WILL BE USED AND RESET STATE IS BETWEEN-BRACKETS
* 0 = BRACKETS WILL NOT BE USED OR, IF USED, RESET STATE IS IN-BRACKETS
* BINKTR EQU X'10' 1 = CONDITIONAL BRACKETS TERMINATION
* 0 = UNCONDITIONAL BRACKETS TERMINATION
* BINALT EQU X'08' 1 = ALTERNATE CODE MAY BE USED
* 0 = ALTERNATE CODE MUST NOT BE USED
* BINRSV04 EQU X'06' RESERVED
BINGUE EQU X'01' BIND-QUEUEING INDICATOR
* 000006 BINCMPN2 DS C COMMON LU PROTOCOLS
BINFMTM EQU X'C0' SEND/RECEIVE MODE
* VALUES FOR BINFMTM
BNMSTSL EQU X'C0' RESERVED
BINHDFX EQU X'80' HDX FLIP FLOP
BINHDCF EQU X'80' HDX CONTENTION
BINFDLPX EQU X'00' FULL DUPLEX
BINRCAV EQU X'20' 1 = SYMMETRIC RESPONSIBILITY FOR RECOVERY
* 0 = CONTENTION LOSER (SEE BINBKFS BELOW) RESPONSIBLE FOR RECOVERY
* BINKF EQU X'10' 1 = PRIMARY IS BRACKETS
* FIRST SPEAKER AND CONTENTION WINNER; SECONDARY IS BRACKETS BIDDER AND CONTENTION LOSER
* 0 = SECONDARY IS BRACKETS
* FIRST SPEAKER AND CONTENTION
* WINNER; PRIMARY IS BRACKETS
* BIDDER AND CONTENTION LOSER
BINASCC EQU X'0C' ALTERNATE CODE PROCESSING IDENTIFIER
*
* 00=ASCII7
* 01=ASCII8
BINCTLV EQU X'02' CONTROL VECTORS ARE INCLUDED
* AFTER THE SLU NAME
BINCONR EQU X'01' RESET STATE FOR HDX FLIP-FLOP
* (E.G. AT START OF SESSION)
* 1 = PRIMARY SENDS FIRST
* WHEN DATA TRAFFIC
* 0 = SECONDARY SENDS FIRST
* RESET STATE IS LEFT

000007 BINTSU DS CL6 TS USAGE
00000D BINPRSVC DS CL12 PRESENTATION SERVICES
000019 BINCRTL DS CL1 CRYPTOGRAPHY CONTROL BYTE
* VALUES FOR BINCRTL
BINNOCRY EQU X'00' NO CRYPTOGRAPHY
BINCRCYCA EQU X'09' CAPABLE OF CRYPTOGRAPHY
BINCRCYSL EQU X'19' SELECTIVE CRYPTOGRAPHY
BINCRCYRQ EQU X'39' REQUIRED CRYPTOGRAPHY
*
* VALUES FOR BINCEUMP
BINCSESS EQU X'30' SESSION LEVEL CRYPTOGRAPHY FLAGS
* VALUES FOR BINCSESS
BINCSENP EQU X'00' NO CRYPTOGRAPHY
BINCSESPE EQU X'10' SELECTIVE CRYPTOGRAPHY
BINCSESR EQU X'30' REQUIRED CRYPTOGRAPHY
*
* LENGTH OF CRYPTOGRAPHY FIELD
BINCLEN EQU X'0F'
*
*
00001A BINPRIML DS C PRIMARY LU NAME LENGTH
00001B BINPRIMN DS CL8 PRIMARY LU NAME
* INCLUDE FOR COMPATIBILITY
*
000023 ORG BINPRIMN
00001B BINPRIM DS 8C PRIMARY LU NAME
*
000023 BINUSEL DS C USER DATA LENGTH
BINUSE EQU * USER DATA
BINUSERD EQU X'00' USER DATA LENGTH DEFAULT
*
* OVERLAY FOR 'BINTSU' (TS USAGE)
000024 ORG BINTSU
000007 BINPSPACE DS C SLU SEND PACING
BINSPTST EQU X'80' NUMBER OF PACING STAGES FROM
* SLU TO PLU ( NOTE-REVERSE OF
* BINPS1ST)
* 1 = TWO STAGES
* 0 = ONE STAGE
BINRSV43 EQU X'40' RESERVED
BINAPACMEQ X'3F' SLU SEND PACING COUNT
000008 BINPSPACE DS C SLU RECEIVE PACING
BINASPI EQU X'80' ADAPTIVE SESSION PACING INDICATOR
*
BINRSV07 EQU X'40' RESERVED
BINRPACMEQ X'3F' SLU RECEIVE PACING COUNT
000009 BINRUSZ DS OCL2 RU SIZES
000009 BINRUSZ DS C SLU MAXIMUM SEND RU SIZE
00000A BINRUSZ EQU X'80' RU SIZE IS SPECIFIED
00000A BINPRUSZ DS C PLU MAXIMUM SEND RU SIZE
00000A BINPRUSZ EQU X'80' RU SIZE IS SPECIFIED
* VALUES FOR BINRUSZ AND BINPRUSZ (RU SIZES) EXCEPT RU SIZE
* SPECIFIED
00000A BINRU256 EQU X'85' 256 BYTE RU (8*2**5)
00000A BINR4096 EQU X'89' 4096 BYTE RU (8*2**9)
00000A BIN61440 EQU X'FC' 61440 BYTE RU (15*2**12)
00000A BINRU1K EQU X'87' 1024 BYTE RU (8*2**7)
00000A BINRUSZM EQU X'F0' MANTISSA (M)
00000A BINRUSZE EQU X'0F' EXPONENT (E) SIZE=M*2**E

00000B BINSPACE DS C PLU SEND PACING
00000B BINPS1ST EQU X'80' NUMBER OF PACING STAGES FROM
* PLU TO SLU (NOTE-REVERSE OF
* BINSP2ST)
* 1 = ONE STAGE
* 0 = TWO STAGE
00000B BINR44 EQU X'40' RESERVED
00000B BINSPACM EQU X'3F' PLU SEND PACING COUNT
00000C BINBPACE DS C PLU RECEIVE PACING
00000C BINPSV10 EQU X'F0' RESERVED
00000C BINBPACM EQU X'3F' PLU RECEIVE PACING COUNT
*
*********************************************************************
* OVERLAY FOR 'BINPRSVC' (PRESENTATION SERVICES)
*********************************************************************
00000D ORG BINPRSVC
00000D BINLUP DS C PS PROFILE
* VALUES FOR BINLUP (PS PROFILE)
00000D BINPSFMT EQU X'80' PS USAGE FIELD FORMAT
00000D BINLUTYP EQU X'7F' LU TYPE
00000D BINLUP6C EQU X'06' LU TYPE 6
00000D BINLUP4C EQU X'04' LU TYPE 4
00000D BINLUP3C EQU X'03' LU TYPE 3
00000D BINLUP2C EQU X'02' LU TYPE 2
00000D BINLUP1C EQU X'01' LU TYPE 1
00000D BINLUP0C EQU X'00' LU TYPE 0
*
00000E BINPSCHR DS CL11 PS PROFILE DEPENDENT
* PRESENTATION SERVICES
*
*********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 1)
*********************************************************************
000019 ORG BINPSCHR
00000F BINLUP1 DS X PS PROFILE 1 FMHS AND DSP
00000F BINFMHS1 EQU X'FO' FM HEADER SUBSET
* VALUES FOR BINFMHS1
00000F BINFM3C EQU X'30' DATA MANAGEMENT SUBSET
00000F BINFMSC EQU X'20' TYPE 1 HEADERS
00000F BINFM1C EQU X'10' TYPE 1 HEADERS WITH
* RESTRICTIONS
00000F BINFM0C EQU X'00' NO FM HEADERS ALLOWED
00000F BINDSPI EQU X'OF' DATA STREAM PROFILE
* VALUES FOR BINDSPI (DATA STREAM PROFILE)
00000F BINDSPI1 EQU X'01' BASIC CONTROLS, CARDS MAY
* SPAN RUS
00000F BINDSP0C EQU X'00' BASIC CONTROLS
00000F BINPLUS1 DS OXL5 PLU USAGE
00000F BINFMF01 DS OXL2 FMH SUBSET DEPENDENT
* FLAGS
00000F BINPFMB1 DS X FIRST BYTE
00000F BINPFMB2 DS X SECOND BYTE
00000F BINPDSPI DS OXL2 DATA STREAM FLAGS FOR
Chapter 3. DSECTs

000011 DSP0 AND DSP1
000012 DSP0 AND DSP1
000013 DSP0 AND DSP1
000014 DSP0 AND DSP1
000015 DSP0 AND DSP1
000016 DSP0 AND DSP1
000017 DSP0 AND DSP1
000018 DSP0 AND DSP1

* DSP0 AND DSP1

000011 BINPDSB1 DS X FIRST BYTE
000012 BINPDSB2 DS X SECOND BYTE
000013 BINPDSB2 DS X SECOND BYTE
000014 BINPDSB2 DS X SECOND BYTE
000015 BINPDSB2 DS X SECOND BYTE
000016 BINPDSB2 DS X SECOND BYTE (RESERVED)
000017 BINPDSB2 DS X SECOND BYTE (RESERVED)
000018 BINPDSB2 DS X SECOND BYTE (RESERVED)

* FLAGS FOR LU PROFILE 1

* FLAGS FOR BINPFMB1 AND BINSFMB1 (FIRST BYTE OF FM
* HEADER FLAGS)
BINDESTS EQU X'80' 0 = TWO DESTINATIONS MAY
* BE OUTSTANDING
* 1 = THREE DESTINATIONS MAY
BE OUTSTANDING

BINCMPCT EQU X'40' 0 = WILL NOT SEND COMPACTION
TABLE/WILL NOT BE QUERIED
FOR COMPACTION TABLES
* 1 = MAY SEND COMPACTION
TABLE/MAY BE QUERIED FOR
COMPACTION TABLES

BINPDIR EQU X'20' 0 = PDIR WILL NOT BE SENT
* 1 = PDIR MAY BE SENT

BINRSV09 EQU X'1F' RESERVED FOR FMHS1
* ADDITIONAL FLAGS FOR FMHS3
BINKDDS1 EQU X'10' 0 = KEYED DIRECT DATA SET
* WILL NOT BE SENT
* 1 = KEYED DIRECT DATA SET
MAY BE SENT

BINSDS1 EQU X'08' 0 = SEQUENTIAL DATA SETS
* WILL NOT BE SENT
* 1 = SEQUENTIAL DATA SETS
MAY BE SENT

BINSASI EQU X'04' 0 = SEQUENTIAL ACCESS TO
ADDRESS DIRECT DATA
SET WILL NOT BE SENT
* 1 = SEQUENTIAL ACCESS TO
ADDRESS DIRECT DATA
SET MAY BE SENT

BINSIDS EQU X'02' 0 = SERIES ID NOT
SUPPORTED (WITH STATUS
IN REPLY)
* 1 = SERIES ID SUPPORTED
(WITH STATUS IN REPLY)

BINARRR EQU X'01' 0 = ADD REPLICATE,
REPLACE REPLICATE NOT
SUPPORTED
* 1 = ADD REPLICATE,
REPLACE REPLICATE
SUPPORTED

* FLAGS FOR BINPFMB2 AND BINSFMB2 (SECOND BYTE OF FM HEADER FLAGS)
BINRSV17 EQU X'FF' RESERVED FOR FMHS1
* ADDITIONAL FLAGS FOR FMHS3
BINRSV16 EQU X'80' RESERVED
BINQDSI EQU X'40' 0 = QUERY FOR DESTINATION
* SELECTION NOT SUPPORTED
* 1 = QUERY FOR DESTINATION
SELECTION SUPPORTED

BINCSDS EQU X'20' 0 = CREATE / SCRATCH /
* SCRATCH ALL DATA
* SET NOT ALLOWED
* 1 = CREATE / SCRATCH /
* SCRATCH ALL DATA
* SET NOT ALLOWED
* BINXFPD EQU X'10'
* 0 = EXECUTE PROGRAM OFFLINE
* NOT ALLOWED
* 1 = EXECUTE PROGRAM OFFLINE
* ALLOWED
* BINRSV11 EQU X'0F'
* RESERVED FOR FMHS3
* * FLAGS FOR 'BINPDSB1 AND BINSDSB1' (PLU/SLU DATA STREAM
* FLAGS FOR DSP0 AND DSP1)
* NL AND FF MAY BE SENT IN ANY SUBSET. EACH SUBSET BELOW CONTAINS
* EVERY PRECEDING SUBSET (E.G. IF AN LU CAN SEND THE HORIZONTAL
* FORMAT SUBSET, IT CAN ALSO SEND THE FULL BASE SET)
* BININTR EQU X'80'
* 0 = FULL BASE SET DATA
* STREAM (BS,CR,LF,ENP, INP,HT,VT) WILL NOT
* BE SENT
* 1 = FULL BASE SET DATA
* STREAM (BS,CR,LF,ENP, INP,HT,VT) MAY BE
* SENT
* BINHFDS EQU X'40'
* 0 = HORIZONTAL FORMAT,
* DATA STREAM(SHF) WILL
* NOT BE SENT
* 1 = HORIZONTAL FORMAT,
* DATA STREAM(SHF) MAY
* BE SENT
* BINVTDS EQU X'20'
* 0 = VERTICAL FORMAT
* DATA STREAM (SVF)
* WILL NOT BE SENT
* 1 = VERTICAL FORMAT
* DATA STREAM (SVF)
* MAY BE SENT
* BINVSDS EQU X'10'
* 0 = VERTICAL CHANNEL DATA
* STREAM (SVF(CHANNELS),SCF, SEL) WILL NOT BE
* SENT
* 1 = VERTICAL CHANNEL DATA
* STREAM (SVF(CHANNELS),SCF, SEL) MAY BE SENT
* BINSLD EQU X'08'
* 0 = SLD WILL NOT BE SENT
* 1 = SLD MAY BE SENT
* BINRSV40 EQU X'06'
* BINTRNDS EQU X'01'
* 0 = TRANSPARENCY DATA
* STREAM (TRN,IRS) WILL
* NOT BE SENT
* 1 = TRANSPARENCY DATA
* STREAM (TRN,IRS)
* MAY BE SENT
* * FLAGS FOR BINPDSB2
* BINUAINT EQU X'80'
* 0 = SLU WILL INITIATE
* ATTENDED
* 1 = SLU WILL INITIATE
* UNATTENDED
* BINUAALT EQU X'40'
* 0 = DURING SESSION SLU
* WILL NOT ALTERNATE
* BETWEEN ATTENDED AND
* UNATTENDED
* 1 = DURING SESSION SLU
* WILL ALTERNATE
* BETWEEN ATTENDED AND
* UNATTENDED
* BINRSV41 EQU X'3F'
* RESERVED
FLAGS FOR BINPMED1 AND BINSMED1 (PLU/SLU MEDIA FLAGS)

BINDOCMT EQU X'80' 0 = DOCUMENT FORMAT WILL NOT BE SENT
1 = DOCUMENT FORMAT MAY BE SENT

BINCARD EQU X'40' 0 = CARD FORMAT WILL NOT BE SENT
1 = CARD FORMAT MAY BE SENT

BINXCHNG EQU X'20' 0 = EXCHANGE MEDIA FORMAT WILL NOT BE SENT
1 = EXCHANGE MEDIA FORMAT MAY BE SENT

BINDISK EQU X'10' 0 = DISK FORMAT WILL NOT BE SENT
1 = DISK FORMAT MAY BE SENT

BINXCDF EQU X'08' 0 = EXTENDED CARD FORMAT WILL NOT BE SENT
1 = EXTENDED CARD FORMAT MAY BE SENT

BINDOCF EQU X'04' 0 = EXTENDED DOCUMENT FORMAT WILL NOT BE SENT
1 = EXTENDED DOCUMENT FORMAT MAY BE SENT

BINCDEDS EQU X'02' 0 = SLU MAY SEND CD EVERY EDS
1 = SLU MUST SEND CD EVERY EDS

BIN1CMP1 EQU X'02' APPLIES ONLY to BINSMEDI
BIN1CMP2 EQU X'01' (SEE BINCMP1 AND BINCMP2)

*********************************************************************
OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS FOR PS PROFILE 2)
*********************************************************************

000009 ORG BINPSCHR
00000E BINDFLAG DS XL1 DEVICE FLAG
00000F BINSEDS EQU X'80' EXTENDED 3270 DATA STREAM
00000F BINRSV14 DS XL4 RESERVED
000013 BINSCRSZ DS XL5 PRESENTATION SPACE SIZE
000013 BINSPRIR DS FL1 PRIMARY (DEFAULT) NUMBER OF ROWS
000014 BINSALTR DS FL1 ALTERNATE NUMBER OF ROWS
000015 BINSALTC DS FL1 ALTERNATE NUMBER OF COLUMNS
000017 BINPRESZ DS FL1 PRESENTATION SPACE SIZE

VALUES FOR BINPRESZ (PRESENTATION SPACE SIZE)
BINPSZRC EQU X'7F' PRESENTATION SPACE HAS DEFAULT AND ALTERNATE SIZES AS DEFINED IN DEFAULT, ALTERNATE ROW/COL FIELDS
BINPSFX EQU X'7E' PRESENTATION SPACE IS FIXED SIZE AS DEFINED BY ROW/COL VALUES IN DEFAULT ROW/COL FIELDS
BINPSZ3 EQU X'03' 24X80 DEFAULT UNDEFINED ALTERNATE
0 DO WRITE STRUCTURED FIELD QUERY
0 TO IDENTIFY ALTERNATE
BINPSZ2 EQU X'02' 24X80 ROW X COLUMN

Chapter 3. DSECTs 639
BINPSZ1 EQU X'01'  12X40 ROW X COLUMN
BINPSZ0 EQU X'00'  UNDEFINED ROW X COLUMN
000018 BINCZMP DS X  COMPRESSION FLAGS
BINCZMP1 EQU X'02'  APPLIES only to BINSMED1
  (SEE BINCZMP1 AND BINCZMP2)
BINCZMP2 EQU X'01'  (SEE BINCZMP1 AND BINCZMP2)

*********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 3)
*********************************************************************

000019 ORG BINPSCHR

000000E BINPVZ6 DS XL5  RESERVED
0000013 BINPVRSZ DS OXL4  PRESENTATION SPACE SIZE
0000013 BINPVFRDS DS FL1  PRIMARY (DEFAULT) NUMBER OF ROWS
0000014 BINPVFCRS DS FL1  PRIMARY (DEFAULT) NUMBER OF
  COLUMNS
0000015 BINPVFRRS DS FL1  ALTERNATE NUMBER OF ROWS
0000016 BINPVFRCS DS FL1  ALTERNATE NUMBER OF COLUMNS
0000017 BINPVFESC DS FL1  PRESENTATION SPACE
  SIZE SPECIFICATION:
  * 0 = MAXIMUM
  * 1 = 480 CHAR
  * 2 = 1920 CHAR
  * X'7E' = FIXED SIZE (SEE
    BINPVFRDS AND
    BINPVFCRS)
  * X'7F' = VARIABLE SIZE AS
    DEFINED BY
    BINPVRSZ
BINPVFSIZ EQU X'7F'  SEE ABOVE
BINPVFSZF EQU X'7E'  SEE ABOVE

000018 BINCZMP DS X  COMPRESSION FLAGS
BINCZMP1 EQU X'02'  APPLIES only to BINSMED1
  (SEE BINCZMP1 AND BINCZMP2)
BINCZMP2 EQU X'01'  (SEE BINCZMP1 AND BINCZMP2)

*********************************************************************
* OVERLAY FOR 'BINPSCHR' (PRESENTATION SERVICES CHARACTERISTICS
* FOR PS PROFILE 4)
*********************************************************************

000019 ORG BINPSCHR

000000E BINPSNDO DS OXL4  PLU SEND CAPABILITY
000000E BINPSDSSP DS X  PRINTER DATA STREAM
  PROFILE
BINPBSDSP EQU X'80'  BASE DATA STREAM PROFILE
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED
BINPSV46 EQU X'40'  RESERVED
BINPSJOB EQU X'20'  JOB SCS SUBSET
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED
BINPSV47 EQU X'10'  RESERVED
BINPRAW EQU X'08'  WORD PROCESSING RAW FORM
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED
BINPSV48 EQU X'07'  RESERVED

000000F BINPSDSSP DS X  ADDITIONAL DATA STREAM
  PROFILE
BINPSV49 EQU X'80'  RESERVED
BINPSDSCD EQU X'40'  0 = CARD NOT SUPPORTED
  * 1 = CARD SUPPORTED
BINPSV29 EQU X'3F'  RESERVED

000010 BINCZLPS DS X  CONSOLE
BINCZLSP EQU X'80'  BASE DATA STREAM PROFILE
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED
BINRSV50 EQU X'40' RESERVED
BINJOB EQU X'20' JOB SCS SUBSET
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED
BINRSV51 EQU X'1F' RESERVED
000011 BINFMHUP DS X FM/FMH USAGE
BINSSDAT EQU X'80' RESERVED
BINDSSTO EQU X'60' 00= 1 LEVEL DESTINATION
  * SELECTION SUSPENSION
  * STACK
  *
  * 01= 2 LEVEL DESTINATION
  * SELECTION SUSPENSION
  * STACK
  *
  * 10= RESERVED
  * 11= 3 LEVEL DESTINATION
  * SELECTION SUSPENSION
  * STACK
BINRSV52 EQU X'1E' RESERVED
BINKIXS EQU X'01' 0 = SLU NEED NOT RECEIVE
  * CD ON EVERY EDS
  * 1 = SLU MUST RECEIVE CD
  * ON EVERY EDS
000012 BINSSNDO DS 0XL4 SLU SEND CAPABILITY
000012 BINPDSPS DS X PRINTER DATA STREAM
  * PROFILE (SEE BINPDSPP)
000013 BINADSPS DS X ADDITIONAL DATA STREAM
  * PROFILE (SEE BINADSPPP)
000014 BINCSELS DS X CONSOLE (SEE BINCSELP)
000015 BINFMHUS DS X FM/FMH USAGE (SEE
  * BINFMHUP; MEANING FOR
  * BINKIXS IS: 0 = PLU NEED
  * NOT RECEIVE CD ON EVERY
  * EDS, 1 = PLU MUST RECEIVE
  * CD ON EVERY EDS)
000016 BINCSE DS X CODE SELECTION
BINCSOR EQU X'F0' REPERTOIRE
BINCSOE EQU X'80' EBCDIC
BINCSOA1 EQU X'40' ASCII / ISCII / ITA#5
BINRSV30 EQU X'20' RESERVED
BINRSV31 EQU X'10' RESERVED
BINCSOC1 EQU X'0C' 00= CODE 0 (MAIN CODE)
  * SELECTION IS EBCDIC
  * 01= CODE 0 (MAIN CODE)
  * SELECTION IS ASCII /
  * ISCII / ITA#5
BINCSOC2 EQU X'03' 00= CODE 1 (ALTERNATE
  * CODE SELECTION IS
  * EBCDIC
  * 01= CODE 1 (ALTERNATE
  * CODE SELECTION IS
  * ASCII / ISCII /
  * ITA#5
000017 BINGENCO DS X GENERAL CHARACTERISTICS
BINRSV32 EQU X'CO' RESERVED
BINWSDF EQU X'20' 0 = PLU MAY SEND DATA
  * FIRST
  *
  * 1 = SLU MUST SEND DATA
  * FIRST
BINRSV33 EQU X'10' RESERVED
BINAIO EQU X'08' 0 = SLU WILL INITIATE
  * ATTENDED
  * 1 = SLU WILL INITIATE
  * UNATTENDED
  *
  * 0 = SLU WILL NOT
  * ALTERNATE BETWEEN
  * ATTENDED AND
  * UNATTENDED

Chapter 3. DSECTs  641
1 = SLU MAY ALTERNATE BETWEEN ATTENDED AND UNATTENDED

BINRSV34 EQU X'03' RESERVED
000018 BINRSV35 DS X RESERVED

*********************************************************************
* OVERLAY FOR 'BINPCHR' (PRESENTATION SERVICES CHARACTERISTICS FOR PS PROFILE 6)
*********************************************************************

000019 ORG BINPCHR
00000E BINLULEV DS X LU-6 LEVEL
BINLV02 EQU X'02' LEVEL 2
00000F BINRSV36 DS XL6 RESERVED
000015 BINFLG0 DS X FLAGS
BINDSSSP EQU X'80' DISTRIBUTED SYSTEMS SECURITY SUPPORT
* 0=EXTENDED SECURITY MECHANISMS ARE NOT SUPPORTED
* 1=AT LEAST ONE SECURITY MECHANISM IS SUPPORTED
BINDESS EQU X'40' Extended Security Sense Codes
* 0= Extended security sense codes will not be accepted on incoming FMH-7s
* 1= Extended security sense codes will be accepted on incoming FMH-7s

000016 BINFLG1 DS X LU 6.2 FLAGS
BINCLSS EQU X'10' ACCESS SECURITY SUBFIELD SUPPORT:
* 0= ACCESS SECURITY INFORMATION FIELD WILL NOT BE ACCEPTED ON INCOMING FMH-5S
* 1= ACCESS SECURITY INFORMATION FIELD WILL BE ACCEPTED ON INCOMING FMH-5S

* BINSYNCH EQU X'60' SYNCHRONIZATION LEVEL:

* BINSLAPS EQU X'08' SESSION LEVEL SECURITY PROTOCOL

* BINDPWS EQU X'04' Password Substitution Support:
* 0= Substituted passwords will not be accepted on incoming FMH-5s
* 1= Substituted passwords will be accepted on incoming FMH-5s

* BINAVFS EQU X'02' ALREADY - VERIFIED FUNCTION SUPPORT
* 0= ALREADY - VERIFIED FUNCTION WILL NOT BE ACCEPTED ON INCOMING FMH 5
* 1= ALREADY - VERIFIED FUNCTION WILL BE ACCEPTED ON INCOMING FMH 5

* BINPV EQU X'01' PERSISTENT VERIFICATION FUNCTION SUPPORT
* 0= PERSISTENT VERIFICATION FUNCTION WILL NOT BE ACCEPTED ON INCOMING FMH 5
* 1= PERSISTENT VERIFICATION FUNCTION WILL BE ACCEPTED ON INCOMING FMH 5

000017 BINFLG2 DS X MORE LU 6.2 FLAGS
BINSYNCH EQU X'60' SYNCHRONIZATION LEVEL:
BINCONF EQU X'20' CONFIRM SUPPORTED
BINCSBK EQU X'40' CONFIRM, SYNC POINT, AND BACKOUT SUPPORTED
BINRS EQU X'10' RECONNECT SUPPORT:
  * 0 = RECONNECT NOT SUPPORTED
  * 1 = RECONNECT SUPPORTED
BINRSR EQU X'0C' RESPONSIBILITY FOR SESSION REINITIATION:
  * NOTE: BINRSR IS RESERVED WHEN PARALLEL SESSIONS ARE SUPPORTED
  (I.E. WHEN BINPSS IS SET)
  * VALUES FOR BINRSR
BINOPRC EQU X'00' OPERATOR CONTROLLED
BINPRIMH EQU X'04' PRIMARY WILL REINITIATE
BINSECNH EQU X'08' SECONDARY WILL REINITIATE
BINTHR EQU X'0C' EITHER MAY REINITIATE

BINPSS EQU X'02' PARALLEL SESSION SUPPORT FOR LU-LU PAIR:
  * 0 = PSS NOT SUPPORTED
  * 1 = PSS SUPPORTED
BINGDSVF EQU X'01' CHANGE NUMBER OF SESSIONS GDS VARIABLE FLOW SUPPORT:
  * 0 = NOT SUPPORTED
  * 1 = SUPPORTED

000018 BINFLG3 DS X
000019 ORG BINPSCHR

FMH-5 (ISTFM5)

Chapter 3. DSECTs 643
FM5PWS EQU X'10' PASSWORD SUBSTITUTION
* IF THIS BIT IS 0 AND A PASSWORD
* IS PRESENT IT IS IN THE CLEAR
* IF THIS BIT IS 1 AND A PASSWORD
* IS PRESENT IT IS A SUBSTITUTED
* PASSWORD
FM5PIPPR EQU X'08' PIP PRESENT AFTER FMH5
FM5DSSPR EQU X'04' DISTRIBUTED SYSTEMS SECURITY
* AUTHENTICATION TOKEN GDS
* PRESENT AFTER FMH-5 (AND PIP
* GDS IF PRESENT). IF THIS BIT
* IS ON, FM5UIDAV, FM5PV1, AND
* FM5PV2 MUST BE ZERO AS WELL AS
* THE SECURITY ACCESS SUBFIELDS.

000005 FM5LNFLP DS X LENGTH OF FIXED LENGTH PARAMETERS
000006 FM5FXLEN DS OXL3 FIXED LENGTH PARAMETERS
000006 FM5RSCTP DS X RESOURCE TYPE
FM5BASIC EQU X'D0' BASIC CONVERSATION
FM5MAPED EQU X'D1' MAPPED CONVERSATION
FM5FDBAS EQU X'D2' FULL-DUPLEX BASIC CONVERSATION
FM5FDMAP EQU X'D3' FULL-DUPLEX MAPPED
* CONVERSATION
* RESERVED
000007 DS C RESERVED
000008 FM5FLAG3 DS X FLAGS FOR FIXED LENGTH PARMS
FM5SYNCH EQU X'C0' SYNCHRONIZATION LEVEL MASK
FM5NONE EQU X'00' NONE
FM5CONF M EQU X'40' CONFIRM, SYNC POINT, BACKOUT
FM5CSB EQU X'80' CONFIRM, SYNC POINT, BACKOUT
FM5RESUP EQU X'20' RECONNECTION SUPPORT
000009 FM5LNTPN DS X LENGTH OF TRANSACTION PROGRAM NAME
* (NOT INCLUDING THIS BYTE)
00000A FM5TPNAM DS OX TRANSACTION PROGRAM NAME
* ACCESS SECURITY INFORMATION
* SUBFIELDS
000000 FM5ASI DSECT
000000 FM5LANASI DS X LENGTH OF ASI SUBFIELDS
* (NOT INCLUDING THIS BYTE)
000001 FM5ASEC DS OX CONTAINS ALL ACCESS SECURITY
* SUBFIELDS. THESE SUBFIELDS ARE
* MAPPED BY THE FM5ACCSE DSECT.
* THERE MAY BE ZERO OR MORE OF
* THESE SUBFIELDS, AND EACH MUST
* BE SEPARATELY MAPPED BY THE
* FM5ACCSE DSECT.

000000 FM5LUOW1 DSECT
* LOGICAL UNIT OF WORK IDENTIFIER
* FIELD
000000 FM5LNLUIW DS X LENGTH OF LUOW ID
* (NOT INCLUDING THIS BYTE)
000001 FM5LUIW DS OX LUOW ID
000001 FM5LNFQN DS X LENGTH OF FULLY QUALIFIED LU NAME
* (NOT INCLUDING THIS BYTE)
000002 FM5FQNAM DS OX FULLY QUALIFIED LU NAME
* LUOW
000000 FM5LUOW2 DSECT
000000 FM5LUWIN DS XL6 LUOW INSTANCE NUMBER
000006 FM5LUWSN DS XL2 LUOW SEQUENCE NUMBER
* CONVERSATION CORRELATOR
000000 FM5CVCOR DSECT
000000 FM5LNCCS DS X LENGTH OF CONVERSATION CORRELATOR
* OF SENDER
* (NOT INCLUDING THIS BYTE)
* CONVERSATION CORRELATOR OF SENDING
* TRANSACTION
000001 FM5CCS DS OX
000000 FM5SEQNM DSECT SEQUENCE NUMBER MAP
**ACCESS SECURITY SUBFIELD**

* THIS DSECT IS USED TO MAP EACH ACCESS SECURITY SUBFIELD. THESE
  SUBFIELDS ARE ALL CONTAINED IN THE FIELD 'FM5ASEC'. YOU MUST
  DETERMINE HOW MANY SUBFIELDS ARE SPECIFIED, AND DETERMINE THE
  LENGTH OF EACH OF THE SUBFIELDS.

**********************************************************************

**PROGRAM INITIALIZATION PARAMETER (PIP).**

* THE PIP, IF IT EXISTS (INDICATED BY FM5PIPPR), FOLLOWS
  THE FMH5.

* ADDRESSABILITY: IF PIP EXISTS, PIP LOCATED AFTER FMH5.

**********************************************************************

**RPL extension (ISTRPL6X)**

LOC SOURCE STATEMENT
000000 ISTRPL6X DSECT

000000 RPL6AREA DS  OCL112  START OF APPC EXTENSION
000000 RPL6CID DS  CL4   CONTROL BLOCK IDENTIFIER
000004 RPL6REQ DS  XL1   TYPE OF APPCCMD
000005 RPL6QUAL DS  XL1   SUBTYPE OF APPCCMD
000006 DS  XL2   RESERVED
000008 RPL6CNDV DS  XL4   CONVERSATION ID
00000C RPL6USR DS  XL4   USER FIELD
000010 RPL6SNOS DS  XL4   SENSE DATA SPECIFIED ON APPCCMD
000014 RPL6SN1 DS  XL4   SENSE DATA RETURNED BY APPCCMD
000018 RPL6SGNL DS  XL4   SIGNAL DATA RETURNED
00001C RPL6SID1 DS  XL1   LENGTH OF SESSION ID
00001D DS  XL3   RESERVED
000020 RPL6SSID DS  XL8   SESSION IDENTIFICATION
000028 RPL6RC DS  OXL4   RPL6 RETURN CODE
000028 RPL6CPR DS  XL2   PRIMARY RETURN CODE
00002A RPL6CRCSC DS  XL2   SECONDARY RETURN CODE
00002C RPL6FLGS DS  OXL4   INDICATORS SPECIFIC TO VTAM'S
  APPCCMD MACRO
00002C RPL6FLGI DS  XL1   FIRST INDICATORS BYTE

Chapter 3. DSECTs 645
RPL6FILL EQU X'80' FILL INDICATOR
RPL6CD EQU X'40' CD KEYWORD INDICATION
* EQU X'20' RESERVED
RPL6LS EQU X'10' PARTNER LU VERIFICATION
* INDICATOR
RPL6CFTX EQU X'08' CONFTXT INDICATOR
RPL6LIST EQU X'06' SCOPE OF INFORMATION TO BE
* RETURNED IN THE RESTORE COMMAND
* EQU X'01' RESERVED
00002D RPL6FLG2 DS XL1 SECOND INDICATORS BYTE
* EQU X'80' RESERVED
RPL6RTSRX EQU X'40' RTS_RCVD RETURN 1=EXPD,0=BOTH
RPL6CXMD EQU X'30' CONXMOD INDICATORS
RPL6TYPE EQU X'0C' TYPE INDICATOR
RPL6NAMU EQU X'03' NAME USE REQUESTED WHEN APPLICATION
* IS ACTING AS A GENERIC RESOURCE
00002E RPL6FLG3 DS XL1 THIRD INDICATORS BYTE
RPL6L0CK EQU X'80' LOCKS INDICATOR
RPL6DERC EQU X'60' DEACTIVATION REASON CODE
RPL6EXDR EQU X'40' EXPEDITED DATA RECEIVED
RPL6CMOD EQU X'30' COMMODE INDICATOR
RPL6LAST EQU X'03' LAST INDICATOR
00002F RPL6FLG4 DS XL1 FOURTH INDICATORS BYTE
RPL6AFFN EQU X'C0' GENERIC RESOURCE AFFINITY OWNER
* EQU X'3F' RESERVED
000030 RPL6LU DS CL8 NAME OF LU
000038 RPL6MODE DS CL8 MODE NAME
000040 RPL6WHAT DS OXL2
000040 RPL6RCV1 DS XL1 WHAT RECEIVED INDICATOR
RPL6WDAT EQU X'80' WHATRCV=DATA
RPL6WDCAC EQU X'40' WHATRCV=DATA_COMPLETE
RPL6WDAAI EQU X'20' WHATRCV=DATA_INCOMPLETE
RPL6WSND EQU X'10' WHATRCV=SEND
RPL6WCFM EQU X'08' WHATRCV=CONFIRM
RPL6WDLAL EQU X'04' WHATRCV=DEALLOCATE
RPL6WLOG EQU X'02' WHATRCV=LOG_DATA
RPL6WPSH EQU X'01' WHATRCV=PS_HEADER
000041 RPL6RCV2 DS XL1 RESERVED FOR BIT MASK FOR THE
* EQU X'7F' NOT USED
000042 RPL6RTUN DS XL1 RETURNED INDICATORS AS
* A RESULT OF APPCCMD
RPL6FH5 EQU X'80' FMH5RCV INDICATOR
RPL6L6LOG EQU X'40' LOGRCV INDICATOR
RPL6RSIG EQU X'20' SIGRCV INDICATOR
RPL6CLSA EQU X'10' PARTNER LU ACCEPTS SECURITY
* SUBFIELDS ON FMH5
RPL6AYFA EQU X'08' PARTNER LU ACCEPTS REQUESTS FOR
* ALREADY VERIFIED
RPL6PV EQU X'04' PARTNER LU ACCEPTS REQUESTS FOR
* PERSISTENT VERIFICATION
RPL6CRYP EQU X'03' ENCRYPTION LEVEL
000043 RPL6MHL5 DS XL1 LENGTH OF THE FMH 5 RECEIVED
000044 RPL6CCST DS XL1 CURRENT CONVERSATION STATE
000045 RPL6ACTV DS XL1 RPL6 ACTIVE INDICATOR
* FF=ACTIVE / 00=INACTIVE
000046 RPL6DETP DS XL1 DEACTIVATION TYPE CODE
000047 RPL6EXDL DS XL1 LENGTH OF EXPEDITED DATA
000048 RPL6TID DS OA TASK ID
000048 RPL6MID DS XL2 MACHINE ID
00004A RPL6TIX DS XL2 TASK INDEX OF CURRENTLY
* EXECUTING TASK
00004C RPL6RPL DS A POINTER BACK TO THE RPL
000050 RPL6STBF DS A POINTER TO CURRENT BUFFER
* AT STORAGE SHORTAGE
000054 RPL6STDS DS A DISPLACEMENT IN CURRENT
* BUFFER AT STORAGE SHORTAGE
Chapter 3. DSECTs
RPL6CFRM EQU X'07'  CONFIRM
RPL6CFMD EQU X'08'  CONFRMD
RPL6DATA EQU X'09'  DATA
RPL6DCON EQU X'0A'  DATACON
RPL6DFLU EQU X'0B'  DATAFLU
RPL6DFIN EQU X'0C'  DEFINE
RPL6DSPY EQU X'0D'  DISPLAY
RPL6ERR EQU X'0E'  ERROR
RPL6FLSH EQU X'0F'  FLUSH
RPL6RQSD EQU X'10'  ROSEND
RPL6SPEC EQU X'11'  SPEC
RPL6ACT EQU X'12'  ACTSESS
RPL6ACTIONS EQU X'13'  DACTSESS
RPL6ALCD EQU X'14'  ALLOCD
RPL6IMED EQU X'15'  IMMED
RPL6WIN EQU X'16'  CONWIN
RPL6SENSE EQU X'17'  SESSION
RPL6CONV EQU X'18'  CONV
RPL6SUSP EQU X'19'  SUSPEND
RPL6RESM EQU X'1A'  RESUME
RPL6REST EQU X'1B'  RESTORE
RPL6SYNB EQU X'1C'  SYNCEBG
RPL6SYNE EQU X'1D'  SYNCCEND
RPL6CNGP EQU X'1E'  CONVGRP
RPL6SYNF EQU X'1F'  WHENFREE
RPL6IANY EQU X'20'  IANY
RPL6ISPEC EQU X'21'  ISPEC
RPL6QALL EQU X'22'  ALL
RPL6QALL EQU X'23'  IALL

***********************************************************************
**
** THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6FILL.  *
** THEY REPRESENT THE "FILL=" VALUE.                           *
**
***********************************************************************
RPL6BUFF EQU X'00'  BUFF
RPL6LL EQU X'00'  LL

***********************************************************************
**
** THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CD  *
** THEY REPRESENT THE "CD=" VALUE  *
**
***********************************************************************
RPL6CDIM EQU X'00'  "CD=IMMED"
RPL6CDE EQU X'40'  "CD=DEFER"

***********************************************************************
**
** THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CFX.  *
** THEY REPRESENT THE "CONFTXT=" VALUE.                        *
**
***********************************************************************
RPL6CF EQU X'08'  YES
RPL6NCF EQU X'00'  NO

***********************************************************************
**
** THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6TYPE.  *
** THEY REPRESENT THE "TYPE=" VALUE.                           *
**
***********************************************************************
RPL6TB EQU X'0C'  TYPE BITS POSITION
RPL6USER EQU X'0C'  USER
RPL6PRM EQU X'04'  PROGRAM
RPL6SVC EQU X'08'  SERVICE

***********************************************************************
**
** THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6NAME.  *
** THEY REPRESENT THE "NAMEUSE=" VALUE.                         *
**
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6LOCK.**
* THEY REPRESENT THE "LOCKS=" VALUE.
*
******************************************************************************
RPL6LONG EQU X'00' LONG
RPL6SHRT EQU X'80' SHORT
******************************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6DERC.**
**
******************************************************************************
RPL6RNRM EQU X'00' NORMAL
RPL6RABN EQU X'40' ABNORMAL
RPL6RANR EQU X'60' ABNORMAL, NORETRY
******************************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CMOD.**
* THEY REPRESENT THE "CONMODE=" VALUE.
*
******************************************************************************
RPL6CBIT EQU X'0C' CONMODE BITS POSITION
RPL6LLCA EQU X'00' LLCA
RPL6BFCA EQU X'04' BUFFCA
RPL6CS EQU X'08' CS
RPL6SAME EQU X'0C' SAME
******************************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6XMD.**
* THEY REPRESENT THE "CONXMOD=" VALUE.
*
******************************************************************************
RPL6CXBT EQU X'30' CONXMOD BIT POSITIONS
RPL6CSCX EQU X'00' CS
RPL6CACX EQU X'10' CA
RPL6SAMX EQU X'30' SAME
******************************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6LAST.**
*
******************************************************************************
RPL6NLST EQU X'00' SESSIONS EXIST FOR THE SPECIFIED MODE
RPL6LMD EQU X'01' LAST SESSION DEACTIVATED FOR THE SPECIFIED MODE
RPL6NCTL EQU X'02' LAST SESSION DEACTIVATED FOR NON-CONTROL MODES
RPL6ALL EQU X'03' ALL SESSIONS FOR THIS LU HAVE BEEN DEACTIVATED
******************************************************************************
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6LIST.**
*
******************************************************************************
RPL6LINO EQU X'00' NO INFORMATION RETURNED
**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CCSL.**
**THEY REPRESENT THE CURRENT CONVERSATION STATE.**

RPL6RSET EQU X'00'  RESET (INITIAL STATE)

**HALF-DUPLEX CONVERSATION STATES**

RPL6SND EQU X'01'  SEND
RPL6RCV EQU X'02'  RECEIVE
RPL6RVCF EQU X'03'  RECEIVE CONFIRM
RPL6RVCS EQU X'04'  RECEIVE CONFIRM SEND
RPL6RVCD EQU X'05'  RECEIVE CONFIRM DEALLOCATE
RPL6PMD D EQU X'06'  PEND DEALLOCATE
RPL6PECL EQU X'07'  PEND END CONVERSATION LOG
RPL6ENDC EQU X'08'  END CONVERSATION (FINAL)
RPL6PND S EQU X'09'  PENDING SEND
RPL6PRVL EQU X'0A'  PENDING RCV LOG

**FULL-DUPLEX CONVERSATION STATES**

RPL6FDRS EQU X'80'  FDX RESET (FINAL)
RPL6FDSR EQU X'81'  FDX SEND/RECEIVE
RPL6FDSO EQU X'82'  FDX SEND-ONLY
RPL6FPDO EQU X'83'  FDX RECEIVE-ONLY
RPL6FSRL EQU X'84'  FDX PENDING SEND/RCV LOG
RPL6FRSL EQU X'85'  FDX PENDING RCV-ONLY LOG
RPL6FROL EQU X'86'  FDX PENDING RESET LOG

**PENDING CONVERSATION ALLOCATION**

RPL6PALC EQU X'FF'  PENDING ALLOCATE

**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6DETP.**
**THEY REPRESENT THE "DEACTYP=" VALUE.**

RPL6TCLP EQU X'0F'  CLEANUP
RPL6TPVL EQU X'FE'  PROTOCOL VIOLATION

**THE FOLLOWING CONSTANT VALUES WILL BE RECORDED IN RPL6CRYP.**
**THEY REPRESENT THE ENCRYPTION LEVEL.**

RPL6CNON EQU X'00'  NONE
RPL6CSEL EQU X'01'  SELECTIVE DATA ENCRYPTION
RPL6CMAN EQU X'03'  MANDATORY DATA ENCRYPTION
<table>
<thead>
<tr>
<th>Address</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>User data - RPL6USR (USERFLD)</td>
</tr>
<tr>
<td>4</td>
<td>Request type&lt;RPL6REQ (CONTROL)</td>
</tr>
<tr>
<td>8</td>
<td>Conversation ID - RPL6CNVD (CONVID)</td>
</tr>
<tr>
<td>10</td>
<td>Sense data specified on APPCCMD - RPL6SNS0 (SENSE)</td>
</tr>
<tr>
<td>14</td>
<td>Sense data returned on APPCCMD - RPL6SNSI</td>
</tr>
<tr>
<td>18</td>
<td>Signal data returned - RPL6SGNL (SIGDATA)</td>
</tr>
<tr>
<td>20</td>
<td>Session ID length - RPL6SIDL (SESSID)</td>
</tr>
<tr>
<td>24</td>
<td>Session identifier - RPL6SSID (SESSID)</td>
</tr>
</tbody>
</table>

*Figure 2. Layout of the RPL extension (part 1 of 3)*
<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Primary return code - RPL6RCPR (RCPRI)</td>
</tr>
<tr>
<td>2C</td>
<td>Secondary return code - RPL6RCSC (RCSEC)</td>
</tr>
<tr>
<td>2C</td>
<td>Flag byte - RPL6FLG1 (FILL, CD SLS, CONFTXT, LIST)</td>
</tr>
<tr>
<td>30</td>
<td>Flag byte - RPL6FLG2 (RTSRTRN CONXMOD, TYPE, NAMEUSE)</td>
</tr>
<tr>
<td>34</td>
<td>Flag byte - RPL6FLG3 (LOCKS, DERCIEXDR, CONMODE LAST)</td>
</tr>
<tr>
<td>38</td>
<td>Flag byte - RPL6FLG4 (LUAFFIN)</td>
</tr>
<tr>
<td>3C</td>
<td>LU name - RPL6LU (LUNAME)</td>
</tr>
<tr>
<td>40</td>
<td>Mode name - RPL6MODE (LOGMODE)</td>
</tr>
<tr>
<td>40</td>
<td>What-received field - RPL6RCV1 (WHATRCV)</td>
</tr>
<tr>
<td>40</td>
<td>What-received field - RPL6RCV2 (WHATRCV)</td>
</tr>
<tr>
<td>42</td>
<td>Returned bits - RPL6RTUN (FMH5RCV, LOGCRV, SIGCRV, CONVSECP, AFVA PRSISTVP, CRYPTLVL)</td>
</tr>
<tr>
<td>42</td>
<td>Received (FMH5 length - RPL6MHSL (FMH5LEN))</td>
</tr>
<tr>
<td>44</td>
<td>Current conversation state RPL6CCST (CONSTATE)</td>
</tr>
<tr>
<td>44</td>
<td>RPL in use - RPL6ACTV</td>
</tr>
<tr>
<td>44</td>
<td>Session deactivation - type code - RPL6DEPT (DEACTYP)</td>
</tr>
<tr>
<td>48</td>
<td>Task ID (The sublevel names are referenced by the VM system) - RPL6TID</td>
</tr>
</tbody>
</table>

*Figure 3. Layout of the RPL extension (part 2 of 3)*
**CNOS session limits data structure (ISTSLCNS)**

<table>
<thead>
<tr>
<th>LOC</th>
<th>SOURCE STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>ISTSLCNS DSECT SESSION LIMITS FOR CNOS</td>
</tr>
<tr>
<td>000000</td>
<td>SLCSESSL DS H SESSION LIMIT</td>
</tr>
<tr>
<td>000002</td>
<td>SLCMCWL DS H MINIMUM NUMBER OF CONTENTION WINNER SESSIONS - LOCAL LU</td>
</tr>
<tr>
<td>000004</td>
<td>SLCMCPW DS H MINIMUM NUMBER OF CONTENTION WINNER SESSIONS - PARTNER LU</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>000006</td>
<td>SLCPARMS DS XL1 CNOS PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>SLCDRAL EQU X'80' DRAINING OF LOCAL LU REQUESTED</td>
</tr>
<tr>
<td></td>
<td>SLCDRAP EQU X'40' DRAINING OF PARTNER LU REQUESTED</td>
</tr>
<tr>
<td></td>
<td>SLCPRSPL EQU X'20' RESPONSIBLE FOR DEACTIVATION</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>

*Figure 4. Layout of the RPL extension (part 3 of 3)*
SLCALL EQU X'10'  INDICATES IF CNOS IS FOR ONE MODE
* OR ALL MODES
* IF SET, CNOS IS FOR ALL MODES
* IF NOT SET, CNOS IS FOR ONE MODE
SLCSSLU EQU X'08'  INDICATE IF THE PARTNER LU IS ONLY
* SINGLE SESSION CAPABLE
* SLCLLSI EQU X'07'  LOCAL LU SECURITY SUBFIELD
* ACCEPTANCE INFORMATION
* SLCLNONE EQU X'00'  NONE ACCEPTED
* SLCLCONV EQU X'04'  SECURITY SUBFIELDS ACCEPTED
* ON FMH5
* SLCLAVFA EQU X'02'  ALREADY VERIFIED REQUESTS ACCEPTED
* SLCLPV EQU X'01'  PERSISTENT VERIFICATION REQUESTS ACCEPTED
* 000007 DS XL1  RESERVED
000008 SLCDSESL DS H  DEFINED SESSION LIMIT
00000A SLCDMCWL DS H  DEFINED MINIMUM NUMBER OF
* CONTENTION WINNER SESSIONS
* LOCAL LU
00000C SLCDMCWP DS H  DEFINED MINIMUM NUMBER OF
* CONTENTION WINNER SESSIONS
* PARTNER LU
00000E SLPARMSL DS XL1  INDICATORS
SLCPPAR EQU X'C0' LU'S SESSION CAPABILITY MASK
SLDCALL EQU X'20'  CONFIRM, SYNC POINT, BACKOUT
* SUPPORTED
SLDCONF EQU X'10'  CONFIRM SUPPORTED
SLDSYNRT EQU X'00'  SYNCHRONIZATION LEVEL NOT SET
* EQU X'0F' RESERVED
00000F DS XL1  RESERVED
* 000010 SLCEND DS 0X  END OF ISTSLCNS

DEFINE/DISPLAY session limits data structure (ISTSLD)

LOC  SOURCE STATEMENT
000000 ISTSLD DSECT  SESSION LIMITS - DEFINE/DISPLAY
*  BEGINING OF LU SPECIFIC FIELDS
*
000000 SLDLUPAR DS 0XL40  LU SPECIFIC FIELDS
000000 SLDLULI DS XL1  LU SPECIFIC FIELDS - BYTE 1
SLDSCAP EQU X'C0'  LU'S SESSION CAPABILITY MASK
SLDPARR EQU X'C0'  PARALLEL SESSION CAPABLE
SLDPMIGE EQU X'80'  PENDING PARALLEL STATE
SLDPMIGD EQU X'40'  PENDING SINGLE STATE
SLDSINGL EQU X'00'  SINGLE SESSION CAPABLE
SLDSYNCH EQU X'30'  NEGOTIATED LEVEL OF SYNCHRONIZATION
* SLDCSBK EQU X'20'  CONFIRM, SYNC POINT, BACKOUT
* SUPPORTED
SLDCONF EQU X'10'  CONFIRM SUPPORTED
SLDSYNRT EQU X'00'  SYNCHRONIZATION LEVEL NOT SET
* EQU X'0F' RESERVED
000001 SLDLUL2 DS XL1  LU SPECIFIC FIELDS - BYTE 2
SLDLSV EQU X'80'  SECURITY SUBFIELD ACCEPTANCE
* INFORMATION IS VALID
* SLDCSA EQU X'40'  PARTNER LU ACCEPTS SECURITY
SUBFIELDS ON FMH5
* SLDPVFA EQU X'20'  PARTNER LU ACCEPTS REQUESTS FOR
* ALREADY VERIFIED FUNCTION
* SLDPV EQU X'20'  PARTNER LU ACCEPTS REQUESTS FOR
* PERSISTENT VERIFICATION
* SLDCLESA EQU X'08'  LOCAL LU ACCEPTS SECURITY

654  z/OS V2R1.0 Communications Server: SNA Programmer's LU 6.2 Reference
* SUBFIELDS ON FMH5 FROM THIS
* PARTNER LU
SLDLAYFA EQU X'04' LOCAL LU ACCEPTS REQUESTS FOR
* ALREADY VERIFIED FUNCTION
SLDLPV EQU X'02' LOCAL LU ACCEPTS REQUESTS FOR
* PERSISTENT VERIFICATION
* EQU X'01' RESERVED
000002 SLDFQNLN DS XL1 LENGTH OF FULLY QUALIFIED
* PARTNER LU NAME
000003 SLDFQNAM DS XL17 FULLY QUALIFIED PARTNER LU
* NAME - USE SLDFQNLN FOR
* ACTUAL LENGTH OF NAME
000014 SLDLU3 DS XL1 LU-SPECIFIC INDICATORS
SLDCNVCP EQU X'C0' CONVERSATION CAPABILITY FIELD
SLDCNVFD EQU X'80' FULL-DUPLEX OR HALF-DUPLEX,
* EXPEDITED DATA ALLOWED
SLDCNVHD EQU X'40' HALF-DUPLEX CONVERSATIONS ONLY
SLDCNVUN EQU X'00' CAPABILITY IS UNKNOWN
* EQU X'20' - X'01' RESERVED
000015 SLDMUSE DS XL1 APPLICATION NAME USE FIELD,
* INDICATES FORM OF LU NAME USED BY
* LOCAL LU FOR SESSIONS WITH THIS
* PARTNER LU
SLDNMUN EQU X'00' NAME USE NOT YET KNOWN
SLDNUMUV EQU X'01' USERSVAR NAME IS BEING USED
SLDNMUAU EQU X'02' APPL NETWORK NAME BEING USED
SLDNUMUG EQU X'03' GENERIC NAME IS BEING USED
000016 SDLTYPE DS XL1 TYPE OF LU ENTRY
SLDSUPNM EQU X'00' SUPPLIED_NAME ENTRY
SLDRCVNM EQU X'01' RCVD_NAME ENTRY
SLDVARNM EQU X'02' VARIANT_NAME ENTRY
SLDUNUMN EQU X'03' UNUSABLE_NAME ENTRY
SLDDISNM EQU X'04' DISASSOC_NAME ENTRY
000017 DS XL17 RESERVED

* * * END OF LU SPECIFIC FIELDS
* * BEGINING OF MODE SPECIFIC FIELDS
* *
000028 SLDDSESL DS H DEFINED SESSION LIMIT
00002A SLDMCWL DS H DEFINED NUMBER OF CONTENTION WINNER
* SESSIONS -- LOCAL LU
00002C SLDMCPW DS H DEFINED NUMBER OF CONTENTION WINNER
* SESSIONS -- PARTNER LU
00002E SLDDEFPA DS XL1 DEFINED PARAMETERS
SLDRSPL EQU X'80' DEFINED ACCEPTANCE OF DEACTIVATION
* RESPONSIBILITY, IF SET THEN THE
* LOCAL LU WILL ACCEPT RESPONSIBILITY
SLDDDRAAL EQU X'40' DEFINED ACCEPTANCE OF REQUEST TO
* DRAIN QUEUED ALLOCS, IF SET THEN
* LOCAL LU WILL ACCEPT THE REQUEST
* MODE DELETION INDICATOR, IF SET
* APPL WILL ALLOW DELETION OF MODE
SLDELET EQU X'20'
SLDAUTOS EQU X'10' AUTOSAS SPECIFIED AS ON DEFINE
SLDMSUS EQU X'08' MODE PENDING RECOVERY
* EQU X'04'-X'01' RESERVED
00002F SLDCNSPA DS XL1 CNOS PARAMETERS
SLDORAL EQU X'80' DRAINING OF LOCAL LU
SLDORAP EQU X'40' DRAINING OF PARTNER LU
* EQU X'20'-X'01' RESERVED
000030 SLDSESSL DS H SESSION LIMIT
000032 SLDMCWL DS H MINIMUM NUMBER OF CONTENTION WINNER
* SESSIONS -- LOCAL LU
000034 SLDMCPW DS H MINIMUM NUMBER OF CONTENTION WINNER
* SESSIONS -- PARTNER LU
000036 SLDAUTO DS H AUTO ACTIVATE LIMIT
Restore data structure (ISTSREST)

LOC SOURCE STATEMENT
000000 ISTSREST DSECT RESTORE STRUCTURE

000000 SRENAME DS CL8 LU NAME
000008 SREMODE DS CL8 LU MODE
000010 SRENXTAD DS A NEXT RESTORE STRUCTURE ADDRESS
000014 SRESLDAD DS A SLD STRUCTURE ADDRESS
000018 SRENSESAD DS A ADDRESS OF FIRST SRESESS
000020 SREFLGS DS XL2 MODE LEVEL FLAGS
SREMDRS EQU X'80' 1=MODE HAS BEEN RESTORED
00001E SRESESCT DS H NUMBER OF SRESESS STRUCTURES
000020 SRENETID DS CL8 NETID OF LU
000028 SREEND DS 0X END OF ISTSREST STRUCTURE

000000 SRESESS DSECT SESSION INFORMATION
000000 SRESNXTA DS A NEXT SESSION STRUCTURE ADDRESS
000004 SRESFLGS DS XL3 SESSION LEVEL FLAGS
SRECONV EQU X'80' 1=CONVERSATION PENDING DEALLOCATION
SRESND EQU X'40' 1=SESSION PENDING DEACTIVATION
000007 SRESID DS XL1 SESSION INSTANCE IDENTIFIER LENGTH
000008 SRESESID DS XL8 SESSION INSTANCE IDENTIFIER
000010 SRESEND DS 0X END OF SESSION INFORMATION

Status data structure (ISTSTATD)

LOC SOURCE STATEMENT
000000 ISTSTATD DSECT TESTSTAT INFORMATION ENTRY

000000 STATENTL DS XL2 LENGTH OF THIS ENTRY
000002 STATENTT DS X ENTRY TYPE
STATNAME EQU X'01' NORMAL DATA INFORMATION ENTRY
STATXPD EQU X'02' EXPEDITED DATA INFORMATION ENTRY
STATRTSE EQU X'03' REQUEST-TO-SEND INFORMATION ENTRY
000003 STAFLAG1 DS X STATUS ENTRY FLAGS
STACMVA EQU X'80' DATA IS IN CA MODE
000004 STACMVID DS XL4 CONVID OF CONVERSATION
000008 STATOTAV DS XL4 TOTAL DATA AVAILABLE (NORM & EXPD)
00000C STACURLL DS XL2 CURRENT ACTIVE LL FIELD (NORM), RESERVED (EXPED & RTS_RCVD)
00000E STACURLR DS XL2 CURRENT LL REMAINDER (NORM), RESERVED (EXPED & RTS_RCVD)
000010 STATENTE DS 0X END OF STATUS ENTRY
Feedback code data structure (ISTUSFBC)

LOC SOURCE STATEMENT
000000 ISTUSFBC DSECT *

******************************************************************************
** THE FOLLOWING CODES ARE STORED IN EITHER 'RPLRTNCD', OR 'RPLFDB2' OR 'RPLFDB3'. SEE THE INTRODUCTORY COMMENTS **
** FOR EACH GROUP FOR FURTHER INFORMATION. **
**
** RPL FIELD NAME OPERAND OF MANIPULATIVE MACRO *
**
** RPLRTNCD RTNCD (FEEDBACK CODE) *
** RPLFDB2 FDBK2 (REASON CODE) *
** RPLFDB3 FDBK (DATA FLAGS) *
**
** IF THE RPLRTNCD IS SET TO X'00' AND THE RPLFDB2 IS SET TO X'1A' THEN THE USER SHOULD REFER TO THE FOLLOWING FIELDS *
** IN THE RPL6. THIS IS ADDED FOR APPC/VTAM. *
**
** RPL6 FIELD NAME *
**
** RPL6RCPR PRIMARY RETURN CODE *
** RPL6RCSC SECONDARY RETURN CODE *
**
******************************************************************************

***** RPLRTNCD CONTAINS A FEEDBACK CODE. IF THE RPL REQUEST IS UNSUCCESSFUL THEN REGISTER *
***** ZEROWILL ALSO CONTAIN THIS CODE. FOR A CERTAIN GROUP OF ERRORS, ONLY REGISTER *
***** ZERO WILL CONTAIN THE FEEDBACK CODE AND NO FEEDBACK INFORMATION WILL BE PLACED IN *
***** THE RPL. *

USFAOK EQU X'00' NORMAL COMPLETION/CONDITIONAL COMPLETION
USFXORDC EQU X'04' EXTRAORDINARY COMPLETION
USFRESSU EQU X'08' REISSUE THIS REQUEST
USFDAMGE EQU X'0C' DAMAGE - INTEGRITY OF REQUEST/DEVICE
USFENVER EQU X'10' ENVIRONMENT ERROR
USFLLOGIC EQU X'14' USER LOGIC ERROR
USFRLLOGIC EQU X'18' USER LOGIC ERROR - SETONLY IN REG ZERO
USF6CHEK EQU X'20' RPL/RPL6 IN WRONG STATE - SET ONLY IN REG00
USF6WRCK EQU X'24' WRONG CHECK MACRO ISSUED - SET ONLY IN REG00

******************************************************************************

***** RPLFDB2 CONTAINS A REASON CODE. THIS REASON CODE **
***** INDICATES ADDITIONAL INFORMATION ABOUT THE **
***** FEEDBACK CODE. **

***** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'00' *****

USFAOOK EQU X'00' OPERATION SUCCESSFULLY COMPLETED
USFRCWNP EQU X'01' RESET CONDITIONAL WAS NO-OPED
USFRCDPR EQU X'02' RESET CONDITIONAL SUCCESSFUL - READ-AHEAD DATA PRESENT
USFYTCNT EQU X'03' YIELDED TO CONTENTION
USFYTCNTL EQU X'04' YIELDED TO CONTENTION, ERROR LOCK SET
USFATSFI EQU X'05' AREA TOO SMALL FOR INQUIRE/INTERPRET
USFNNOIN EQU X'06' NO INPUT AVAILABLE
USFIIINA EQU X'07'  INQUIRE INFORMATION NOT AVAILABLE
USFDSTIU EQU X'08'  DESTINATION IN USE
USFNLGFA EQU X'09'  NO LOGON FOUND FOR ACCEPT MATCH
USFANC EQU X'0A'
USF6APPC EQU X'0B'  INDICATES THAT AN ERROR OCCURRED RUNNING
*          APPC, AND REFER TO THE RPL6 PRIMARY AND
*          SECONDARY RETURN CODES
USFINQPS EQU X'0D'  MORE SESSIONS PENDING RECOVERY ON
*          WHICH TO INQUIRE
  *
*          IF, FOLLOWING A SYNCHRONOUS RPL REQUEST MACRO OR CHECK *
*          MACRO, REGISTER 15 CONTAINS X'00' THEN REGISTER ZERO WILL *
*          CONTAIN ONE OF THE ABOVE REASON CODE VALUES *
*
*
***** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'04' *****

USFRVIRC EQU X'00'  RVI RECEIVED, ERROR LOCK SET
USFATNRC EQU X'01'  ATTENTION RECEIVED, ERROR LOCK SET
USFBSCSM EQU X'02'  BSC STATUS MSG RECEIVED
USFEXRQ EQU X'03'  EXCEPTION REQUEST RECEIVED
USFEXRS EQU X'04'  EXCEPTION RESPONSE RECEIVED
USFNQN EQU X'05'  RESOURCE KNOWN AS NQN ONLY
*
*
***** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'08' *****

USFSTALF EQU X'00'  TEMPORARY OUT OF STORAGE SITUATION EXISTS
*          RPL ECB/EXIT NOT POSTED/INVOKED *
*
*
***** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'0C' *****

USFIOEDU EQU X'00'  I/O ERROR, DEVICE STILL USABLE ER LK SET
USFDVUNS EQU X'01'  I/O ERROR, DEVICE NOT USABLE ER LK SET
USFUNTRM EQU X'02'  REQUEST RESET BY TEST REQUEST MESSAGE
USFBTHEX EQU X'03'  BUFFER THRESHOLD EXCEEDED
USFBEOR EQU X'04'  BUF THRESHOLD EXCEEDED, ONLY READS ALLOW
USFNCPAO EQU X'05'  NCP ABENDED, RESTART O.K.
USFLIORP EQU X'06'  LAST I/O REQUEST PURGED
USFRECIP EQU X'07'  RECOVERY IN PROGRESS
USFTRAF EQU X'08'  RECOVER TERMINAL RESTARTED AFTER FAILURE
USFQOPDC EQU X'09'  QUEUED OPNDST CANCELLED BY CLSDST
USFUSRES EQU X'0A'  REQUEST RESET BY THE USER
USFCLOC EQU X'0B'  CLSDST OR TERMESS ISSUED OR UNBIND SENT
*          IN LIEU OF NEGATIVE BIND RESPONSE *
*          REQUEST WAS CLEAR'ED
USFPOQLE EQU X'0E'  SEND CANCELLED DUE POA QUEUE LIMIT
*
*
***** REASON CODE EQUATES FOR RPLFDB2 IF RPLRTNCD EQUALS X'10' *****

USFTANAV EQU X'00'  TERMINAL OR APPLICATION NOT AVAILABLE
USFSBLF AL EQU X'01'  SESSION BIND FAILED
USFTAPUA EQU X'02'  TARGET APPLICATION UNACCEPTABLE
USFVTHAL EQU X'03'  VTAM IS HALTING
USFLRIS EQU X'04'  INCOMPATIBLE DEFINITION
USFPCF EQU X'05'  PERMANENT FAILURE IN PATH
USFANS EQU X'06'  AUTO NETWORK SHUTDOWN
USFVFOFC EQU X'07'  VARY DEACTIVATE IMMEDIATE OCCURRED
USFVDOFC EQU X'08'  DISCONNECT OCCURRED
USFUTSCR EQU X'09'  UNCONDITIONAL TERMINATE SELF CMD RECEIVED
USFSYERR EQU X'0A'  APPARENT VTAM ERROR
USFDIDOL EQU X'0B'  DISCONNECT ON DIAL-OUT LINE
USFDIDIL EQU X'0C'  
   DISCONNECT ON DIAL-IN LINE
*  NOTE - X'0D' AND X'0E' - RPL ECB/EXIT NOT POSTED/INVOKED  *
USFVTMNA EQU X'0D'  
   VTAM INACTIVE FOR THAT APPLICATION
USFABNDO EQU X'0E'  
   ABEND CONDITION HAS OCCURRED
*  *
USFVTBFO EQU X'0F'  
   VTAM BUFFER OVERFLOW
USFCTERM EQU X'10'  
   CONDITIONAL TERM SELF
USFSDFTF EQU X'11'  
   SDT FAILURE ON OPSEND
USFMFE EQU X'12'  
   MACRO FUNCTION FAILED, SENSE INCLUDED
USF6APRJ EQU X'13'  
   ATTEMPT TO START 6.2 SESSION: REQUEST
   REJECTED
USF6APST EQU X'14'  
   ATTEMPT TO START 6.2 SESSION: PENDING
   SESSION TERMINATED
USF6APIS EQU X'15'  
   MUST ISSUE APPCCMD
USFNONSW EQU X'16'  
   SWITCHED OPERATION ATTEMPTED ON
   NONSWITCHED DEVICE
USFABEOS EQU X'17'  
   XES IS NOT ACCESSIBLE
   SUPPORT THE REQUESTED FUNCTION
   ALLOCATION NOT RESIDENT IN A SYSPLEX
*  *
USFXMEMS EQU X'18'  
   SUSPEND FAILURE
USFXMEMR EQU X'19'  
   RESUME FAILURE
USFSLVNL EQU X'1A'  
   OPERATING SYSTEM LEVEL INSTALLED DOES NOT
   SUPPORT THE REQUESTED FUNCTION
*  *
USFRTCME EQU X'1B'  
   SECURITY MANAGER ERROR
*  *

***** REASON CODE EQUATES FOR RPLFDB2 IF RPLTNCD EQUALS X'14' *****
*  *
USFSNORV EQU X'00'  
   RPL CONTAINS A NON-VTAM REQUEST CODE
*  
USFNOTAS EQU X'01'  
   NOT ASSIGNED
*  RPL ECB/EXIT NOT POSTED/INVOKED
*  *
USFEXTAZ EQU X'02'  
   RPL INDICATES EXIT, EXIT ADDR IS ZERO
*  RPL ECB/EXIT NOT POSTED/INVOKED
*  *
USFEXTEZ EQU X'03'  
   RPL IND EXTERNAL ECB, ECB ADDR IS ZERO
*  RPL ECB/EXIT NOT POSTED/INVOKED
*  *
USFCRPLN EQU X'04'  
   CHECKED RPL IS NOT ACTIVE
   ONLY OCCURS FOLLOWING A CHECK MACRO REQUEST
*  *
USFCBERR EQU X'10'  
   RPL POINTS TO INVALID ACB
USFRNORT EQU X'11'  
   NO RTYPE SPECIFIED
USFLCSIP EQU X'12'  
   CLSDST IN PROGRESS
USFCIODG EQU X'13'  
   CID IS INVALID
USFLDOE EQU X'14'  
   LDO COMMAND FIELD IS INVALID
USFWANCR EQU X'15'  
   READ NOT CHAINED
USFFTOOD EQU X'16'  
   SOLICIT SPECIFIC TO OUTPUT ONLY DEVICE
USFTTOOD EQU X'17'  
   READ TO OUTPUT ONLY DEVICE
USFWTOI EQU X'18'  
   WRITE TO INPUT ONLY DEVICE
USFEWNS EQU X'19'  
   ERASE TO INVALID DEVICE
USFEWAO EQU X'1A'  
   WRITE EAU TO NON-3270
USFCWOO EQU X'1B'  
   WRITE CONV TO OUTPUT ONLY
USFCWO EQU X'1C'  
   WRITE WITH ERASE AND CONV SPECIFIED
USFCOPY EQU X'1D'  
   CHAINED COPY LDO
USFIDA EQU X'1E'  
   INVALID DATA AREA OR LENGTH
USFILOOA EQU X'1F'  
   INVALID LDO ADDRESS
USFJTOJ EQU X'20'  
   JUMP LDO TO JUMP
USFT100E EQU X'21'  
   MORE THAN 100 LDOS
USFRILCP EQU X'22'  
   RESET LDO IS NOT ALONE
USFRCRRT EQU X'23'  
   INVALID MACRO REQUEST TYPE
USFASIDE EQU X'24'  
   ASID MISMATCH
USFEBWKL EQU X'25'  
   WRITE ERASE BLOCK
USFICRSG EQU X'26'  
   SOLICIT LDO WITH DATA CHAINING
USFIREST EQU X'27'  
   RESET OPTION CODE INVALID
USFMT323 EQU X'28'  
   WRITE BLOCK TO 3270 DEVICE
USFRMODS EQU X'29'  
   READ MODIFIED TO NON-3270 DEVICE
USFCTN32 EQU X'2A'  
   COPY TO NON-3270 DEVICE
USFETCNVR EQU X'2B'  
   WRITE CONV ISSUED WHEN DATA EXPECTED
USFRNT3 EQU X'2C' OUTPUT NOT PRECEDED BY INPUT
USFRINV EQU X'2D' RESET CONDITIONAL ILLEGAL
USFINVRM EQU X'2E' INVALID READ MODE
USFLOGNT EQU X'2F' EXCESSIVE LEADING GRAPHICS, ERROR LK SET
USFCPCNT EQU X'30' COPY COUNT ERROR
USFIDAE EQU X'31' INVALID DATA AREA OR LENGTH, ERROR LK SET
USFUSELE EQU X'32' REQUEST INVALID FOR DEVICE, ERROR LK SET
USFCONP EQU X'33' CONV. REPIL NOT POSSIBLE, ERROR LOCK SET
USFORD EQU X'34' NO READ WHERE REQUIRED, ERROR LOCK SET
USFCPE2 EQU X'35' COPY WRONG CLUSTER, ERROR LOCK SET
USFRELNP EQU X'36' REQUEST LOCK NOT ALLOWED, ERROR LOCK SET
USFCPE1 EQU X'37' COPY UNOPENED DEVICE, ERROR LOCK SET
USFDIFIB EQU X'38' FIRST I/O FAILED INVALID BHSET, ER LK SET
USFDIFPO EQU X'39' FIRST I/O FAILED INVALID PROC, ER LK SET
USFIDLGCNT EQU X'3A' EXCESSIVE LEADING GRAPHICS, ERROR LK SET
USFCPCNT EQU X'3B' COPY COUNT ERROR
USFIDAEL EQU X'3C' INVALID DATA AREA OR LENGTH, ERROR LK SET
USFIDAE EQU X'3D' INVALID DATA AREA OR LENGTH, ERROR LK SET
USFSCE EQU X'3E' CHAINING ERROR: FIRST OR ONLY REQUIRED
USFSCFB EQU X'3F' CHAINING ERROR: MIDDLE OR LAST REQUIRED
USFSCEM EQU X'40' CHAINING ERROR: MIDDLE OR LAST REQUIRED
USFSCEF EQU X'41' CHAINING ERROR: FIRST OR ONLY REQUIRED
USFSNQC EQU X'42' QUIESCE COMPLETE RESPONSE NOT REQUESTED
USFSNOUT EQU X'43' SEND RESPONSE NOT REQUESTED
USFSNOUT EQU X'44' NIB RESPLIM EXCEEDED
USFSNOUT EQU X'45' SEQUENCE NUMBER ERROR
USFSNOUT EQU X'46' RESPOND = OPTION MISMATCH
USFSNOUT EQU X'47' RESPOND = OPTION MISMATCH
USFSNOUT EQU X'48' PROTOCOL VIOLATION
USFSNOUT EQU X'49' INVALID ACTION TYPE
USFSNOUT EQU X'4A' INSTALLATION EXIT ROUTINE N/A
USFSNOUT EQU X'4B' INVALID LOGON SEQUENCE
USFSNOUT EQU X'4C' LU NOT SESSION CAPABLE
USFSNOUT EQU X'4D' NO INTERPRET TABLE
USFSNOUT EQU X'4E' ILLEGAL USE OF NIB LIST
USFSNOUT EQU X'4F' INVALID OPNDST TYPE
USFSNOUT EQU X'50' INVALID AQUIRE PARAMETER
USFSNOUT EQU X'51' APPLICATION NEVER ACCEPTS
USFSNOUT EQU X'52' INVALID NIB
USFSNOUT EQU X'53' Symbolic NAME UNKNOWN
USFSNOUT EQU X'54' DESTINATION UNOPENABLE
USFSNOUT EQU X'55' NO OPNDST AUTHORIZATION
USFSNOUT EQU X'56' MODE - DEVICE INCOMPAT
USFSNOUT EQU X'57' INVALID MODE
USFSNOUT EQU X'58' BHSET NAME UNKNOWN
USFSNOUT EQU X'59' MODE NAME AUTHORIZED
USFSNOUT EQU X'5A' MULTIPLE BHSETS SPECIFIED
USFSNOUT EQU X'5B' INVALID LOGON DATA AREA
USFSNOUT EQU X'5C' DUPLICATE NODES
USFSNOUT EQU X'5D' DESTINATION NOT OPENED
USFSNOUT EQU X'5E' NO PASS AUTHORIZATION
USFSNOUT EQU X'5F' RESOURCE NOT OWNED
USFSNOUT EQU X'60' RESOURCE NOT CLOSEABLE
USFSNOUT EQU X'61' INVALID SETLOGON
USFSNOUT EQU X'62' MACRO NOT VALID FOR SPECIFIED DEVICE
USFSNOUT EQU X'63' PROGRAM OPERATOR APPLICATION EXCEEDED
USFSNOUT EQU X'64' LIMIT OF OUTSTANDING RCVCMDs
USFSNOUT EQU X'65' APPLICATION NOT AUTHORIZED
USFSNOUT EQU X'66' REPLY, SENT BY PROGRAM OPERATOR,
USFSNOUT EQU X'67' REJECTED DUE TO SYNTAX ERROR
USFSNOUT EQU X'68' PROGRAM OPERATOR INTERFACE INACTIVE
USFSNOUT EQU X'69' RCVCMD REJECTED BECAUSE PROGRAM
USFSNOUT EQU X'6A' OPERATOR APPLICATION IS CLOSING
USFSNOUT EQU X'6B' V,D,F, SENT BY PROGRAM OPERATOR
USFSNOUT EQU X'6C' REJECTED DUE TO SYNTAX ERROR
USFSNOUT EQU X'6D' REJECTED DUE TO SYNTAX ERROR
USFSNOUT EQU X'6E' LOGICAL ERROR, PRIMARY CANNOT ISSUE
USFSNOUT EQU X'6F' TERMSESS
USFINVSD EQU X'73'  INVALID OPTIONS ON SEND  
*  
USFNRNB0 EQU X'74'  NEGOTIABLE RESPONSE TO NON-NEGOTIABLE BIND  
*  
USFINBRP EQU X'75'  INVALID NEGOTIABLE BIND RESPONSE  
*  
USFINBSZ EQU X'76'  INVALID NEGOTIABLE BIND RESPONSE SIZE  
*  
USFMDQ EQU X'77'  FM DATA REQUEST UNIT REQUIRED  
*  
USFCHINV EQU X'78'  INVALID CHAIN SPECIFICATION  
*  
USFLINV EQU X'79'  INVALID BUFFER LIST LENGTH  
*  
USFINVRH EQU X'7B'  INVALID USER RH  
*  
USFSCINV EQU X'7C'  OPTCD=USERRH INVALID FOR SESSION  
*  
USFPINV EQU X'7D'  XRF PROTOCOL VIOLATION  
*  
USFCONMR EQU X'7E'  CONFLICTING OPTCD ON A MACRO REQUEST  
*  
USF6PENA EQU X'7F'  POLICING ERROR - NON-APPCC MACRO  
*  
USFPRINV EQU X'80'  PERSISTENT LU-LU SESSION SUPPORT REQUESTED FOR APPLICATION THAT IS NOT PERSISTENT SESSION CAPABLE  
*  
USFSTSND EQU X'81'  TERMSSESS WITHOUT UNBIND WITH SESSION IN PENDING ACTIVE STATE  
*  
USFSAML EQU X'82'  PARAMETER LENGTH INVALID  
*  
USFSFERR EQU X'83'  SUBFIELD NOT SUPPORTED, INVALID COMBINATION OF SUBFIELDS, OR SUBFIELD FORMAT ERROR  
*  
USFASDAZ EQU X'84'  ZERO NIBASDPA FIELD  
*  
USFMMRS EQU X'85'  SESSION IS IN RECOVERY STATE AND MUST BE RESTORED  
*  
USFSSESSA EQU X'86'  SESSIONS OR AFFINITIES EXIST  
*  
USFSNAME EQU X'87'  RESOURCE NAME AND GENERIC NAME EQUAL  
*  
USFNSPTE EQU X'88'  NO SPT EXISTS  
*  
USFNSFEC EQU X'89'  NO SECURITY AUTHORIZATION FOR GENERIC RESOURCE  
*  
USFIFNM EQU X'8A'  ALREADY REGISTERED WITH A DIFFERENT GENERIC NAME  
*  
USFNOMAP EQU X'8B'  NOT REGISTERED AS A GENERIC RESOURCE  
*  
USFNRTED EQU X'8C'  ALREADY REGISTERED WITH A DIFFERENT NETWORK ID  
*  
USFCPNM EQU X'8D'  MAPPING ALREADY EXISTS ON A DIFFERENT SYSPLEX NODE  
*  
USFCNFAC EQU X'8E'  CONFLICTING APPC CAPABILITY  
*  
USFVTAMO EQU X'8F'  SPTE IS OWNED BY VTAM  
*  
USFUSVAR EQU X'90'  GENERIC NAME CONFLICTS WITH AN EXISTING USERVAR  
*  
USFGNMN A EQU X'91'  TSO GENERIC NAME CONFLICT  
*  
USFGNMMA EQU X'92'  SETLOGON GNAMESUB FAILURE  
*  
USFSTK EQU X'93'  STOKEN NOT VALID   
*  
****** NO REASON CODE EQUATES EXIST FOR RPLRTNCD EQUALS X'18' ******  
*  
****** EQUATES FOR RPLFDB3 ON RETURN FROM INQUIRE IF RPLRTNCD IS X'00' ******  
*  
USFIACT EQU X'00'  APPLICATION IS ACTIVE  
*  
USFIIA EQU X'04'  APPLICATION IS INACTIVE  
*  
USFIIA EQU X'08'  APPLICATION WILL NOT ACCEPT LOGONS  
*  
USFIIA EQU X'0C'  APPLICATION IS TEMPORARILY NOT ACCEPTING LOGONS  
*
USFIQUIE EQU X'10'                  APPLICATION IS QUIESCING
USFILACT EQU X'80'                  RESOURCE IS ACTIVE
USFILINA EQU X'84'                  RESOURCE IS NOT ACTIVE
*
**********************************************************************
***
*** THE FOLLOWING ARE ALL THE RPL6RCPR (PRIMARY RETURN CODE) VALUES FOR APPC/VTAM.
***
USF6OK EQU X'0000'                  OK
USF6ALLC EQU X'0004'                ALLOCATION ERROR
USF6CNSA EQU X'0008'                CNOS ALLOCATION ERROR
USF6CNSN EQU X'000C'                CNOS RESOURCE FAILURE, NO RETRY
USF6CRJR EQU X'0010'                COMMAND RACE REJECT
USF6DABP EQU X'0014'                DEALLOCATE ABEND PROGRAM
USF6DABS EQU X'0018'                DEALLOCATE ABEND SERVICE
USF6DABT EQU X'001C'                DEALLOCATE ABEND TIMER
USF6CNSR EQU X'0020'                CNOS FAILURE, RETRY
USF6LUBE EQU X'0024'                LOGICAL RECORD BOUNDARY ERROR
USF6LSECl EQU X'0028'               LU MODE SESSION LIMIT CLOSED
USF6SLEM EQU X'002C'                PARAMETER ERROR
USF6PENT EQU X'0030'                PROGRAM ERROR NO TRUNCATION
USF6PSEP EQU X'0034'                PROGRAM ERROR PURGING
USF6PETR EQU X'0038'                PROGRAM ERROR TRUNCATING
USF6SENT EQU X'003C'                SERVICE ERROR NO TRUNCATION
USF6SEP EQU X'0040'                 SERVICE ERROR PURGING
USF6STET EQU X'0044'                SERVICE ERROR TRUNCATING
USF6FRF EQU X'0048'                 RESOURCE FAILURE, NO RETRY
USF6FRF EQU X'004C'                 RESOURCE FAILURE, RETRY
USF6STER EQU X'0050'                STATE ERROR
USF6RMD EQU X'0054'                 UNRECOGNIZED MODE NAME
USF6UNSC EQU X'0058'                UNSUCCESSFUL, SESSION NOT AVAILABLE
*
USF6UECR EQU X'005C'                USER ERROR CODE RECEIVED
USF6NOF EQU X'0060'                 NO FMX5 AVAILABLE
USF6LDFL EQU X'0064'                ACTIVATION FAILURE
USF6LSEG EQU X'0068'                LU MODE SESSION LIMIT EXCEEDED
USF6SACT EQU X'006C'                SESSION NOT PENDING
USF6STOR EQU X'0070'                TEMPORARY STORAGE SHORTAGE OR RESOURCE
   SHORTAGE
USF6HALT EQU X'0074'                HALT ISSUED
USF6VIYI EQU X'0078'                VTAM INACTIVE FOR YOUR ACB
USF6ROAB EQU X'007C'                REQUEST ABORTED
USF6DLN EQU X'0080'                 DEALLOCATE NORMAL
USF6STSH EQU X'0084'                STORAGE SHORTAGE
USF6CREJ EQU X'0088'                CANCELLED BY REJECT OR DEALLOCATE ABND*
*
USF6PR0E EQU X'008C'                PARTNER COMMITTED PROTOCOL VIOLATION
*
USF6NOT EQU X'0090'                 APPLICATION NOT APPC CAPABLE
USF6SNOJ EQU X'0094'                SEND DATA REJECTED INVALID STATE
USF6SSTS EQU X'0098'                STORAGE SHORTAGE WHILE SENDING
*
USF6RSTF EQU X'009C'                RESTORE REJECTED
USF6RNL EQU X'00A0'                 REQUEST NOT ALLOWED
USF6SPMD EQU X'00A4'                MODE MUST BE RESTORED BEFORE USING
USF6ENV EQU X'00A8'                 ENVIRONMENT ERROR
USF6ERN EQU X'00AC'                 ERROR INDICATION WAS RECEIVED
USF6GNER EQU X'00B0'                NAME RESOLUTION ERROR
USF6CSME EQU X'00B4'                CSM DETECTED ERROR
*
***
*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE
*** PRIMARY RETURN CODE IS SET TO X'0000' (USF6OK).
***
USF6OKSC EQU X'0000'  OK
USF6ASSP EQU X'0001'  AS SPECIFIED
USF6ASNG EQU X'0002'  AS NEGOTIATED
USF6RCVR EQU X'0003'  RECEIVE SPECIFIC REJECTED
USF6SNGL EQU X'0004'  PARTNER LU SUPPORTS SINGLE SESSION
* 
USF6INER EQU X'0005'  INTERNAL VTAM ERROR
USF6RSUN EQU X'0006'  RESTORE UNNECESSARY - NO SESSIONS TO RESTORE
* 
USF6RSIN EQU X'0007'  RESTORE INCOMPLETE - INPUT WORK AREA TOO SMALL
* 
USF6NINA EQU X'0008'  NO IMMEDIATELY AVAILABLE INFORMATION FOR REQUEST
* 
USF6NINER EQU X'0009'  REQUEST TERMINATED BY END OF CONVERSATION
* 
USF6INMS EQU X'000A'  SESSIONS WILL USE APPL NETWORK NAME, GENERIC NAME WAS REQUESTED
* 
USF6GNMS EQU X'000B'  SESSIONS WILL USE GENERIC NAME, APPL NETWORK NAME WAS REQUESTED
* 
USF6NAM1 EQU X'000C'  AS SPECIFIED, PARTNER LU KNOWN BY DIFFERENT NAME
* 
USF6NAM2 EQU X'000D'  AS NEGOTIATED, PARTNER LU KNOWN BY DIFFERENT NAME
* 
* *** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'0004' (USF6ALLC).
*** 
USF6ALNR EQU X'0000'  ALLOCATION FAILURE, NO RETRY
USF6ALR EQU X'0001'  ALLOCATION FAILURE, RETRY
USF6ALCM EQU X'0002'  CONVERSATION TYPE MISMATCH
USF6ALPI EQU X'0003'  PIP NOT ALLOWED
USF6ALPP EQU X'0004'  PIP NOT SPECIFIED CORRECTLY
USF6ALSC EQU X'0005'  SECURITY NOT VALID
USF6ALSY EQU X'0006'  SYNC LEVEL NOT SUPPORTED BY LU
USF6ALSLE EQU X'0007'  SYNC LEVEL NOT SUPPORTED BY PROGRAM
* 
USF6ALTP EQU X'0008'  TPN NOT RECOGNIZED
USF6ALTN EQU X'0009'  TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY
* 
USF6ALTR EQU X'000A'  TRANSACTION PROGRAM NOT AVAILABLE, RETRY
* 
USF6ALRN EQU X'000B'  CANNOT RECONNECT TRANSACTION PROGRAM, NO RETRY
* 
USF6ALRR EQU X'000C'  CANNOT RECONNECT TRANSACTION PROGRAM, RETRY
* 
USF6ALNS EQU X'000D'  RECONNECT NOT SUPPORTED BY PROGRAM
* 
USF6SPMA EQU X'000E'  MODE MUST BE RESTORED BEFORE USING
USF6SARQ EQU X'000F'  DEALLOCATION REQUESTED
USF6ALS F EQU X'0010'  REQUESTED SYNCH LEVEL NOT ALLOWED FOR FULL-DUPLEX CONVERSATION
USF6LSF EQU X'0011'  LU PAIR NOT SUPPORTING FULL-DUPLEX CONVERSATIONS
* 
*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X'0008' (USF6CNSA).
*** 
USF6CNAR EQU X'0000'  ALLOCATION FAILURE, NO RETRY
USF6CAR EQU X'0001'  ALLOCATION FAILURE, RETRY
USF6CCTR EQU X'0002'  TRANSACTION PROGRAM NOT AVAILABLE, RETRY
* 
USF6CATN EQU X'0003'  TRANSACTION PROGRAM NOT AVAILABLE, NO RETRY
* 
USF6CAM EQU X'0004'  CONVERSATION TYPE MISMATCH
USF6CASC EQU X’0005’ SECURITY NOT VALID
USF6SPMC EQU X’0006’ MODE MUST BE RESTORED BEFORE USING
USF6QNM EQU X’0007’ NETWORK QUALIFIED NAME MISMATCH

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X’10’ (USF6CRRJ).
***
USF6CRPR EQU X’0000’ PARTNER GRANTED RETRY
USF6CRLR EQU X’0001’ CONTROL OPERATOR OF LOCAL LU RETRIED
USF6CPIC EQU X’0002’ PARTNER CNOS IN PROGRESS
USF6LPSK EQU X’0003’ LU IS IN PENDING SINGLE STATE
USF6PLSS EQU X’0004’ PARTNER LU STARTING SESSION

*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE PRIMARY RETURN CODE IS SET TO X’002C’ (USF6PARM).
***
USF6IVLU EQU X’0000’ INVALID LU NAME OR NETID
USF6IVMD EQU X’0001’ INVALID MODE
USF6ICII EQU X’0002’ INVALID CONVERSATION ID
USF6IVLL EQU X’0003’ INVALID LL
USF6IVSV EQU X’0004’ INVALID VALUES FOR SNASVCMG MODE
USF6IVDOL EQU X’0005’ INVALID DRAIN CHANGE
USF6SNAE EQU X’0006’ SNASVCMG MODE CANNOT CURRENTLY BE RESET
USF6NMEM EQU X’0007’ MINWINL PLUS MINWINR EXCEEDS SESSLIM
USF6LNNI EQU X’0008’ SUPPLIED LENGTH INSUFFICIENT
USF6SNL EQU X’0009’ INCOMPLETE SESSION LIMITS STRUCTURE
USF6INFN EQU X’000A’ INCOMPLETE FMH5 SUPPLIED
USF6NGD EQU X’000B’ INCOMPLETE GDS VARIABLE SUPPLIED
USF6OEXT EQU X’000C’ ZERO EXIT FIELD
USF6OECB EQU X’000D’ ZERO ECB FIELD
USF6RIAS EQU X’000E’ REQUEST INVALID FOR ADDRESS SPACE
USF6CBIN EQU X’000F’ CONTROL BLOCK INVALID
USF6INDL EQU X’0010’ INVALID DATA ADDRESS OR LENGTH
USF6PRVO EQU X’0011’ PREVIOUS MACRO INSTRUCTION OUTSTANDING
USF6BLIV EQU X’0012’ BUFFER LIST LENGTH INVALID
USF6NMDM EQU X’0013’ NO CORRESPONDING MODE IN LM TABLE
USF6IVBP EQU X’0014’ INVALID BIND PARAMETERS
USF6IVTP EQU X’0015’ INVALID TPN
USF6NOLL EQU X’0016’ NO CORRESPONDING LU IN LM TABLE
USF6IMDF EQU X’0017’ INVALID MODE SPECIFIED
USF6ILSP EQU X’0018’ INVALID LIMIT SPECIFIED
USF6SNAI EQU X’0019’ SNASVCMG MODE ALREADY INITIALIZED
USF6ALLS EQU X’001A’ ALL MODES SPECIFIED ON SINGLE SESSION LU
USF6SMSS EQU X’001B’ SNASVCMG MODE FOR SINGLE SESSION LU
USF6SSMI EQU X’001C’ SINGLE SESSION, MODE ALREADY INITIALIZED
USF6CIDI EQU X’001D’ CID INVALID
USF6APNA EQU X’001E’ APPCCMD ISSUED FOR NON-APPCC
USF6PRRO EQU X’001F’ PREVIOUS REJECT REQUEST OUTSTANDING
USF6DARJ EQU X’0020’ DEALLOCATE ABND* REJECTED, RETRY
USF6IVCQ EQU X’0021’ INVALID CONTROL OR QUALIFY VALUE
USF6INSI EQU X’0022’ INVALID SESSION INSTANCE IDENTIFIER
USF6PSHI EQU X’0023’ PS HEADER NOT SUPPLIED
USF6PSLI EQU X’0024’ PS HEADER LENGTH INSUFFICIENT
**USF6NMSC EQU X'0026'**  
SESSION INSTANCE IDENTIFIER AND CONVERSATION ID MISMATCH

**USF6IDET EQU X'0027'**  
INVALID DEACTIVATION TYPE CODE

**USF6CRY EQU X'0028'**  
CRYPTOGRAPHY NOT ALLOWED ON MODE

**USF6LINL EQU X'0029'**  
INVALID LIST VALUE SPECIFIED ON APPCCMD FOR RESTORE

**USF6INCG EQU X'002A'**  
INVALID CGID VALUE ON ALLOCATE

**USF6INEL EQU X'002B'**  
INVALID EXPEDITED DATA LENGTH SPECIFIED

**USF6INCI EQU X'002C'**  
INVALID SENSE CODE SPECIFIED

**USF6VANV EQU X'002D'**  
VECTOR AREA NOT VALID

**USF6VAVL EQU X'002E'**  
VECTOR AREA NOT VALID

**USF6STNV EQU X'002F'**  
VECTOR AREA LENGTH INSUFFICIENT

**USF6STNL EQU X'0030'**  
SENDRCV SPECIFIED WITHOUT OPTCD=BUFLST|XBUFLST

**USF6UNXV EQU X'0031'**  
UNEXPECTED VECTOR PROVIDED ON APPCCMD

**USF6INSP EQU X'0032'**  
A REQUIRED VECTOR WAS NOT PROVIDED OR SPECIFIED INCORRECTLY

**USF6INSL EQU X'0033'**  
PASSWORD SUBSTITUTION VALUE SET IN ERROR

***

**USF6FNGR EQU X'0000'**  
FOLLOWING NEGATIVE RESPONSE

**USF6WNGR EQU X'0001'**  
WITHOUT NEGATIVE RESPONSE

***

**USF6SLSR EQU X'0001'**  
RESTORE ISSUED BEFORE SETLOGON START

***

**USF6LNSE EQU X'0002'**  
LU PAIR NOT SUPPORTING EXPEDITED DATA REQUESTS

**USF6RQBL EQU X'0003'**  
REQUEST BLOCKED DUE TO PENDING CONVERSATION TERMINATION

**USF6RUNE EQU X'0004'**  
EXECUTION OF REQUEST TERMINATED CONVERSATION TERMINATION

**USF6DVFL EQU X'0005'**  
CONTROL/QUALIFY VALUE INVALID ON FULL-DUPLEX CONVERSATION

**USF6XNRC EQU X'0006'**  
CONTROL/QUALIFY VALUE INVALID ON FULL-DUPLEX CONVERSATION

**USF6VYFV EQU X'0007'**  
CONTROL/QUALIFY VALUE INVALID ON FULL-DUPLEX CONVERSATION

**USF6ESEN EQU X'0008'**  
NAMED RESOURCE NOT ELIGIBLE FOR REQUESTED ALTERATION

***

**USF6OSLV EQU X'0000'**  
OPERATING SYSTEM LEVEL DOES NOT SUPPORT REQUESTED FUNCTION

**USF6XMER EQU X'0001'**  
SUSPEND FAILURE

**USF6XMBL EQU X'0002'**  
RESUME FAILURE

***

**USF6EIAP EQU X'0001'**  
DEALLOCATE ABEND PROGRAM

**USF6EIAS EQU X'0002'**  
DEALLOCATE ABEND SERVICE

**USF6EITAP EQU X'0003'**  
DEALLOCATE ABEND TIMER

---

Chapter 3. DSECTs
USF6EIAE EQU X'0004' ALLOCATION ERROR
USF6EINU EQU X'0005' UNKNOWN TERMINATION TYPE RECEIVED
USF6EIRRE EQU X'0006' RESOURCE FAILURE, RETRY
USF6EIRRN EQU X'0007' RESOURCE FAILURE, NO RETRY
*
***
*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE
*** PRIMARY RETURN CODE IS SET TO X'00B0' (USF6NRER).
***
USF6NRE EQU X'0001' LUNAME FOUND IN A VARIANT_NAME ENTRY
USF6NRRD EQU X'0002' NAME RETURNED DIFFERS FROM
* ASSOCIATED NAME
USF6NRRN EQU X'0003' NAME RETURNED FOUND IN A
* VARIANT NAME ENTRY
USF6NRRAP EQU X'0004' NAME RETURNED FOUND IN A
* SUPPLIED NAME ENTRY
USF6NRNM EQU X'0005' NETWORK NAME MISMATCH
USF6NRRA EQU X'0006' LUNAME FOUND IN AN UNUSABLE_NAME
* ENTRY
USF6NRIV EQU X'0007' NAME RETURNED FOUND IN AN
* UNUSABLE_NAME ENTRY
USF6NRDN EQU X'0008' LUNAME FOUND IN A DISASSOCIATED_NAME
* ENTRY
*
***
*** THE FOLLOWING ARE SECONDARY RETURN CODES WHEN THE
*** PRIMARY RETURN CODE IS SET TO X'00B4' (USF6CSME).
***
USF6NSPC EQU X'0001' NOT SPECIFIED
USF6IBTK EQU X'0002' INVALID BUFFER TOKEN SPECIFIED
USF6IID EQU X'0003' INVALID INSTANCE ID SPECIFIED
*
***********************************************************************
***********************************************************************
APPCCMD vector lists (ISTAPCVL)
LOC SOURCE STATEMENT
*** MAPPING FOR VECTORLIST HEADER (LENGTH FIELD) **
000000 ISTAPCVA DSECT VECTOR LIST
*  
000000 APCVALEN DS HL2 LENGTH OF VECTOR LIST
* (INCLUDES LENGTH FIELD & VECTORS)
000002 APCVADTA DS 0X VECTORS
*  
***********************************************************************
*****
***** VECTORS PASSED FROM VTAM TO APPLICATION AT APPCCMD COMPLETION
*****
***** Note: Highorder bit in vector key is off for all vectors sent
***** from VTAM to application.
*****
***********************************************************************
*** ISTAPC10 - maps the VTAM-to-APPL Required INFORMATION vector. **
- Returned on all APPCCMD macros if a vector area is provided.
- Indicates whether VTAM was able to return vector information successfully and length needed.
- NOTE: Application-provided vector area must be large enough to accept at least this vector.

***********************************************************************
000000 ISTAPC10 DSECT INFORMATION VECTOR
* 000000 APC10LEN DS HL2 VECTOR LENGTH
000002 APC10KEY DS X VECTOR KEY
APC10KYC EQU X'10' VECTOR KEY X'10'
000003 APC10DTA DS 0X VECTOR DATA FIELDS
000003 APC10FLG DS X FLAGS
APC10IVL EQU X'80' INSUFFICIENT VECTOR AREA LENGTH
000004 DS X RESERVED
000005 DS X RESERVED
000006 APC10VLN DS HL2 VECTOR AREA LENGTH NEEDED
*
*
***********************************************************************
*** ISTAPC12 - Maps the Partner's DCE Capability vector. ***
*** - Returned on these APPCCMD completions if DCE is active: ***
*** APPCCMD CONTROL=PREALLOC ***
*** APPCCMD CONTROL=RCVFMH5 ***
*** APPCCMD CONTROL=OPRCNTL QUALIFY=CNOS ***
*** APPCCMD CONTROL=OPRCNTL QUALIFY=DISPLAY ***
*** - Also returned on ATTN(CNOS) if DCE is active. ***
*** - Contains the Security Mechanisms Data subfield exchanged during BIND processing if DCE is active. ***
***********************************************************************
000000 ISTAPC12 DSECT PARTNER'S DCE CAPABILITY VECTOR
* MAPPING
000000 APC12LEN DS HL2 LENGTH OF VECTOR (INCLUDING LENGTH OF THIS FIELD)
000002 APC12KEY DS X VECTOR KEY
APC12KYC EQU X'12' KEY IS X'12'
000003 APC12DTA DS 0X SECURITY MECHANISMS DATA
APC12DCE EQU X'01' DCE AUTHENTICATION
APC12KRY EQU X'02' KRYPTOKNIGHT
APC12KER EQU X'03' KERBEROS V5
APC12DCP EQU X'04' DCE PERFORMANCE MECHANISM
*
*
***********************************************************************
*** ISTAPC13 - maps the LOCAL NONCE vector. ***
*** - Returned for these APPCCMD completions if password substitution is supported on session: ***
*** APPCCMD CONTROL=PREALLOC ***
*** APPCCMD CONTROL=RCVFMH5 ***
*** - Contains random data used for password substitution. ***
***********************************************************************
000000 ISTAPC13 DSECT MAPPING FOR LOCAL NONCE VECTOR
000000 APC13LEN DS HL2 LENGTH OF VECTOR
000002 APC13KEY DS X VECTOR KEY
APC13KYC EQU X'13' KEY IS X'13'
000003 APC13DTA DS 0X NONCE DATA
000003 DS XL1 RESERVED
000004 APC13NOF DS CL8 NONCE FIELD
*
*
***********************************************************************
*** ISTAPC14 - maps the PARTNER'S NONCE vector. ***
*** - Returned for these APPCCMD completions if password substitution is supported on session: ***

Chapter 3. DSECTs 667
**APPCCMD CONTROL=PREALLOC**

**APPCCMD CONTROL=RCVFMH5**

- Contains random data used for password substitution.

***********************************************************************

000000 ISTAPC14 DSECT MAPPING FOR PARTNER NONCE
000000 APC14LEN DS HL2 LENGTH OF VECTOR
000002 APC14KEY DS X VECTOR KEY
APC14KYC EQU X'14' KEY IS X'14'
000003 APC14DTA DS 0X NONCE DATA
000003 DS XL1 RESERVED
000004 APC14NOF DS CL8 NONCE FIELD
*

***********************************************************************

**APPCCMD CONTROL=PREALLOC**

- Returned for these APPCCMD completions if password substitution is supported on session:

**APPCCMD CONTROL=PREALLOC**

- Contains the number of FMH_5s which have flowed on this session from the partner LU.

***********************************************************************

000000 ISTAPC15 DSECT MAPPING FOR SEND FMH_5 SEQUENCE NUMBER VECTOR
000000 APC15LEN DS HL2 LENGTH OF VECTOR
000002 APC15KEY DS X VECTOR KEY
APC15KYC EQU X'15' KEY IS X'15'
000003 DS XL1 RESERVED
000004 APC15SNF DS 0X SEQUENCE NUMBER FIELD
000004 APC15SNH DS XL4 SEQUENCE NUMBER FIELD - HIGH-ORDER BITS
000008 APC15SNL DS XL4 SEQUENCE NUMBER FIELD - LOW-ORDER BITS
*

***********************************************************************

**APPCCMD CONTROL=PREALLOC**

- Returned for these APPCCMD completions if password substitution is supported on session:

**APPCCMD CONTROL=RCVFMH5**

- Contains the number of FMH_5s which have flowed on this session from the partner LU.

***********************************************************************

000000 ISTAPC16 DSECT MAPPING FOR RECEIVE FMH_5 SEQUENCE NUMBER VECTOR
000000 APC16LEN DS HL2 LENGTH OF VECTOR
000002 APC16KEY DS X VECTOR KEY
APC16KYC EQU X'16' KEY IS X'16'
000003 DS XL1 RESERVED
000004 APC16SNF DS 0X SEQUENCE NUMBER FIELD
000004 APC16SNH DS XL4 SEQUENCE NUMBER FIELD - HIGH-ORDER BITS
000008 APC16SNL DS XL4 SEQUENCE NUMBER FIELD - LOW-ORDER BITS
*

***********************************************************************

**APPCCMD CONTROL=ALLOC**

- Returned for these APPCCMD completions:

**APPCCMD CONTROL=PREALLOC**

- Contains the PCID for the session being used by the conversation.

***********************************************************************

000000 ISTAPC17 DSECT MAPPING FOR PCID VECTOR
000000 APC17LEN DS HL2 LENGTH OF VECTOR
000002 APC17KEY DS X VECTOR KEY

668  z/OS V2R1.0 Communications Server: SNA Programmer's LU 6.2 Reference
APC17KYC EQU X'17'
000003 APC17DTA DS 0X VECTOR DATA FIELDS
000003 APC17PCF DS CL8 SESSION PCID FIELD

***********************************************************************
*** ISTAPC18 - maps the NAME CHANGE vector. **
*** - Returned for these APPCCMD completions and exits: **
*** APPCCMD CONTROL=ALLOC **
*** APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS **
*** APPCCMD CONTROL=PREALLOC **
*** ATTN(CNOS) exit **
*** ...when a RCVD_NAME LU entry has been changed to **
*** a VARIANT_NAME LU entry in the LU-Mode Table. **
***********************************************************************

000000 ISTAPC18 DSECT MAPPING FOR NAME CHANGE VECTOR
000000 APC18LEN DS HL2 LENGTH OF VECTOR
000002 APC18KEY DS X VECTOR KEY
APC18KYC EQU X'18' KEY IS X'18'
000003 APC18DTA DS 0X VECTOR DATA FIELDS
000003 APC18NET DS CL8 NETWORK IDENTIFIER OF THE LU
00000B APC18RCV DS CL8 LUNAME IN RCVD_NAME LU ENTRY
000013 APC18SUP DS CL8 LUNAME IN SUPPLIED_NAME ENTRY

***********************************************************************
*** ISTAPC19 - maps the Session Information vector. **
*** - Returned for these APPCCMD completions: **
*** APPCCMD CONTROL=ALLOC **
*** APPCCMD CONTROL=PREALLOC **
*** APPCCMD CONTROL=RCVFMH5 **
*** ...to provide session characteristics information **
*** for the conversation. **
***********************************************************************

000000 ISTAPC19 DSECT MAPPING FOR SESSION INFORMATION
  * VECTOR
000000 APC19LEN DS HL2 LENGTH OF VECTOR
000002 APC19KEY DS X VECTOR KEY
APC19KYC EQU X'19' KEY IS X'19'
000003 APC19DTA DS 0X VECTOR DATA FIELDS
000003 APC19CSU DS X COMMUNICATION STORAGE USAGE INDICATORS
  * APC19NOF EQU X'80' NOT AN HPDT-ENABLED SESSION.
  * CSM STORAGE USERS, DUE TO
  * PERFORMANCE CONSTRAINTS, SHOULD
  * EITHER USE CSM PAGEABLE DATA
  * SPACE OR NON_CSM STORAGE
APC19SMB EQU X'40' SMALLER BUFFERS RECOMMENDED FOR
  * CSM STORAGE USERS BECAUSE OF
  * RU SIZE LIMITATIONS.
APC19PGP EQU X'20' PAGEABLE BUFFERS RECOMMENDED. HPDT
  * NO ADDITIONAL PERFORMANCE CAN BE
  * GAINED USING FIXED BUFFERS.
APC19FXP EQU X'10' FIXED BUFFERS RECOMMENDED. HPDT
  * ADDITIONAL PERFORMANCE CAN BE
  * GAINED USING FIXED BUFFERS.
000004 DS XL2 RESERVED
000006 APC19RUO DS FL4 MAXIMUM RU SIZE OUTBOUND
00000A APC19RUI DS FL4 MAXIMUM RU SIZE INBOUND

***********************************************************************
*** ISTAPC1A - maps the Partner Application Capabilities vector **
*** - Returned for these APPCCMD completions: **
*** APPCCMD CONTROL=ALLOC **
*** APPCCMD CONTROL=OPRCNTL, QUALIFY=CNOS **

Chapter 3. DSECTs 669
*** APPCCMD CONTROL=OPRCNTL, QUALIFY=DISPLAY **
*** APPCCMD CONTROL=PREALLOC **
*** APPCCMD CONTROL=RCVFMH5 **
*** - Returned for this exit: **
*** ATTN(CNOS) **
*****
*** ...to provide partner capabilities information **
*** for the conversation. **
**********************************************************************
000000 ISTAPC1A DSECT MAPPING FOR PARTNER APPLICATION
* CAPABILITIES VECTOR
000000 APCIALEN DS HL2 LENGTH OF VECTOR
000002 APCIAKEY DS X VECTOR KEY
APCIAKYC EQU X'1A' KEY IS X'1A'
000003 APCIADTA DS 0X VECTOR DATA FIELDS
000003 APCIAFL1 DS X PARTNER APPLICATION CAPABILITY
* INDICATORS
APCIAPAR EQU X'C0' NEGOTIATED PARALLEL SESSION
* CAPABILITY
APCIASSC EQU X'00' SINGLE SESSION CAPABLE
APCIAASSP EQU X'40' PENDING SINGLE STATE
APCIAAPSP EQU X'CO' PARALLEL SESSION CAPABLE
APCIAAPWS EQU X'30' NEGOTIATED LEVEL OF
* PASSWORD SUBSTITUTION
APCIAPSS EQU X'20' PASSWORD SUBSTITUTION
* SUPPORTED
APCIAPSN EQU X'10' PASSWORD SUBSTITUTION
* NOT SUPPORTED
APCIAPSU EQU X'00' PASSWORD SUBSTITUTION
* LEVEL NOT SET
APCIAESS EQU X'0C' PARTNER SUPPORT FOR
* EXTENDED SECURITY SENSE
* CODES
APCIAASS EQU X'08' EXTENDED SECURITY SENSE CODES
* SUPPORTED
APCIAASN EQU X'04' EXTENDED SECURITY SENSE CODES
* NOT SUPPORTED
APCIAASSU EQU X'00' EXTENDED SECURITY SENSE CODE
* LEVEL NOT SET
APCIAFDX EQU X'03' NEGOTIATED FDX/EXPD
* CAPABILITY
APCIAFXS EQU X'02' FDX OR HDX CONVERSATIONS AND
* EXPEDITED DATA ALLOWED
APCIAFXN EQU X'01' HDX CONVERSATIONS ONLY
APCIAFXU EQU X'00' CAPABILITY IS UNKNOWN
000004 APCIAF2 DS X PARTNER APPLICATION CAPABILITY
* INDICATORS
APCIACON EQU X'C0' NEGOTIATED LEVEL OF
* SYNCHRONIZATION
APCIACNS EQU X'80' CONFIRM, SYNC POINT AND
* BACKOUT SUPPORTED
APCIACNN EQU X'40' CONFIRM SUPPORTED
APCIACNU EQU X'00' SYNCHRONIZATION LEVEL NOT
* SET
APCIASEC EQU X'20' PARTNER ACCEPTS SECURITY
* SUBFIELDS ON FMH
APCIAALV EQU X'10' PARTNER ACCEPTS REQUEST FOR
* ALREADY VERIFIED
APCIAAPR EQU X'08' PARTNER ACCEPTS REQUEST FOR
* PERSISTENT VERIFICATION
* EQU X'07' RESERVED
000005 APCIAFL3 DS X PARTNER CHARACTERISTICS
APCIALOC EQU X'E0' PARTNER LOCALITY STATUS
APCIAMUL EQU X'00' LOCALITY OF PARTNER UNKNOWN
APCIAAOMT EQU X'00' PARTNER NOT ON SAME HOST
APCIAALCL EQU X'40' PARTNER IS ON SAME HOST SYSTEM
APCIALUO EQU X'20' PARTNER LU SAME AS APPLICATION LU
* (LU=OWN)
* *
*******************************************************************************
***** VECTORS PASSED FROM APPLICATION TO VTAM AT APPCCMD ISSUANCE *****
***** **
***** Note: Highorder bit in vector key is on for all vectors sent **
***** from application to VTAM. **
***** **
*******************************************************************************
* ******************* VECTORS PASSED FROM APPLICATION TO VTAM AT APPCCMD ISSUE ******
* **
* ISTAPC82 - maps the XBUFLST RECEIVE vector. **
* *** This vector is passed to VTAM on an APPCCMD **
* *** CONTROL=RECEIVE when OPTCD specifies XBUFLST. **
*******************************************************************************
000000 ISTAPC82 DSECT MAPPING FOR XBUFLST RECEIVE VECTOR
000000 APC82LEN DS HL2 LENGTH OF VECTOR
000002 APC82KEY DS X VECTOR KEY
000003 APC82KYC EQU X'82' KEY IS X'82'
000003 APC82DTA DS 0X VECTOR DATA FIELDS
000003 APC82SFL DS X STORAGE TYPE FLAG BYTE:
* ONE OR MORE OF THE FOLLOWING
* IS REQUIRED:
* APC82ECS EQU X'80' ECSA STORAGE REQUESTED
* APC82CDS EQU X'40' DATA SPACE STORAGE REQUESTED
000004 APC82XBL DS FL4 BUFFER LENGTH (REQUIRED WHEN
* IN FILL=BUF MODE) OR ZEROS
* *
000008 APC82MXD DS FL4 MAXIMUM DATA TO BE RECEIVED
* (OPTIONAL) OR ZEROS
* *
00000C APC82TSK DS AL4 TASK TCB ADDRESS FOR CSM
* STORAGE ASSOCIATION
* (OPTIONAL) OR ZEROS
* *
*******************************************************************************
***** DATA FIELDS PASSED FROM THE APPLICATION TO VTAM. *****
***** **
***** *** Addressability: ACBAPID, ACBPASSW. ***
***** **
*******************************************************************************
000000 ISTVACAP DSECT APPLID MAPPING
* 000000 VACAPLEN DS X MAP LENGTH
000001 VACAPDTA DS 0X MAP DATA
* 000000 ISTVACPW DSECT PASSWORD MAPPING
* 000000 VACPWLEN DS X MAP LENGTH
000001 VACPWDTA DS 0X MAP DATA
* *
*******************************************************************************
***** VECTORS PASSED FROM THE APPLICATION TO VTAM. *****
***** **
***** *** Addressability: ACBAVPTR. ***
***** **
*******************************************************************************
*** Note: Highorder bit in vector key is on for all vectors sent from application to VTAM. **
*** **
***********************************************************************
*
*** MAPPING FOR VECTORLIST HEADER (LENGTH FIELD) **
000000 ISTVACAV DSECT APPLICATION VECTORLIST
* POINTEO TO BY ACBAPVTR
* WHEN PARMS=(APPLVCTR=address)
000000 VACAVLEN DS HL2 TOTAL LENGTH OF APPL VECTORS
000002 VACAVDTA DS 0X VECTOR DATA
*
***********************************************************************
*** GENERALIZED MAPPING FOR EXAMINING OR BUILDING COMMON FIELDS IN **
*** ALL ACB VECTORS IN THE VECTOR LIST POINTED TO BY ACBAVPTR **
***********************************************************************
000000 ISTVACVT DSECT VECTOR TEMPLATE
000000 VACVTLEN DS HL2 VECTOR LENGTH
000002 VACVTKEY DS X VECTOR KEY
000003 VACVTDAT DS 0X VECTOR DATA
*
***********************************************************************
*** ISTVAC81 - Application Capabilities vector **
*** - Passed to VTAM by the application at OPEN invocation **
*** for the ACB. **
*** - Bit indicators which enable/disable application use **
*** of certain VTAM functions. **
***********************************************************************
000000 ISTVAC81 DSECT APPLICATION CAPABILITIES VECTOR
000000 VAC81LEN DS HL2 VECTOR LENGTH
000002 VAC81KEY DS X VECTOR KEY
VAC81KYC EQU X'81' KEY IS X'81'
000003 VAC81CAP DS 0XL4 APPLICATION CAPABILITIES DATA
VAC81MLE EQU X'80' APPLICATION SUPPORTS HAVING ITS LOGON EXIT DRIVEN MULTIPLE TIMES
* PER SESSION REQUEST. APPLICATIONS WITH LOGON EXITS MUST SET THIS INDICATOR TO BENEFIT FROM VERIFICATION REDUCTION
VAC81FPR EQU X'40' APPLICATION INDICATES THAT IT WILL USE HPDT INTERFACE PROVIDED VIA THE OPTCD=XBUFLST FIELD ON THE APPCCMD RECEIVE MACROINSTRUCTION
* VAC81PWS EQU X'20' APPLICATION INDICATES THAT IT IS PASSWORD SUBSTITUTION CAPABLE
* VAC81ESS EQU X'10' APPLICATION INDICATES THAT IT IS CAPABLE OF EXTENDED SECURITY SENSE CODES
* VAC81FPS EQU X'08' APPLICATION INDICATES THAT IT WILL USE HPDT INTERFACE PROVIDED BY THE OPTCD=XBUFLST FIELD ON AN APPCCMD 
* MACROINSTRUCTION THAT SENDS DATA
*
***********************************************************************
*** ISTVAC82 - Local Application's DCE Capability Vector **
*** - Passed to VTAM by the application at OPEN invocation **
*** for the ACB. **
*** - Contains the Security Mechanisms data for the Local LU. **
***********************************************************************
000000 ISTVAC82 DSECT LOCAL APPLICATION'S DCE
**CAPABILITY VECTOR MAPPING**

000000 VAC82LEN DS HL2 LENGTH OF VECTOR (INCLUDING LENGTH OF THIS FIELD).

000002 VAC82KEY DS X VECTOR KEY

VAC82KYE EQU X'82' VECTOR KEY X'82'

000003 VAC82DTA DS 0X ISTVAC82 DATA

**Access-method-support vector list (ISTAMSVL)**

```plaintext
LOC  SOURCE STATEMENT
000000 ISTAMSVL DSECT MAPPING FOR RESOURCE INFORMATION
     Vector list pointed to by ACVAMSVL
000000 AMSLLEN DS HL2 Total length of vectors
000002 AMSLDATA DS 0X Vector data

***********************************************************************
*** GENERALIZED MAPPING FOR EXAMINING COMMON FIELDS IN ALL ACB ***
*** VECTORS IN THE VECTOR LIST POINTED TO BY ACBAMSVL ***
***********************************************************************

000000 ISTAMSVT DSECT VECTOR FIELDS
000000 AMSVTLEN DS X Vector length
000001 AMSVTKEY DS X Vector key

***********************************************************************
*** ISTAMS01 - maps the RELEASE LEVEL vector. ***
*** - Contains identification codes for the access method ***
*** product and its version, release, and modification ***
*** level. ***
***********************************************************************

000000 ISTAMS01 DSECT RELEASE LEVEL VECTOR
000000 AMS01LEN DS X Vector length
000001 AMS01KEY DS X Vector key

AMS01KYE EQU X'01' KEY IS X'01'
000002 AMS01DTA DS OCL4 Vector data
000002 AMS01PRD DS CL1 Product code
AMS01VTM EQU C'0' VTAM product code
AMS01VER DS CL1 Version code
AMS01REL DS CL1 Release code
AMS01MDF DS CL1 Modification code

***********************************************************************
*** ISTAMS04 - maps the COMPONENT IDENTIFICATION vector. ***
*** - This vector may be repeated. ***
*** - Each component identification vector contains product ***
*** feature information about a major component or ***
*** feature of the VTAM licensed program. When multiple ***
*** component identification vectors are present, the ***
*** first one designates the base VTAM product and later ***
*** vectors are features or other major VTAM components. ***
*** - The vector data is in the form: C'xxxx-xxxxx-xxx'. ***
***********************************************************************

000000 ISTAMS04 DSECT COMPONENT IDENTIFICATION VECTOR
000000 AMS04LEN DS X Vector length
000001 AMS04KEY DS X Vector key

AMS04KYE EQU X'04' KEY IS X'04'
000002 AMS04DTA DS CL14 Vector data

***********************************************************************
*** ISTAMS05 - maps the FUNCTION LIST vector. ***
*** - The vector data is a variable-length bit string, in ***
*** which each bit corresponds to a particular VTAM ***
*** function. If a bit is on, the corresponding function ***
*** is present in the executing release of VTAM. If a ***
*** bit is off, the function is not available. If the ***
*** vector is not present or if the bit string is shorter ***
```
than expected, you may assume that the missing bits are zero and their corresponding functions are not available.

- These indicator bits correspond to the compile-time global indicator bits in the ISTGLBAL macro.

000000 ISTAMS05 DSECT FUNCTION LIST VECTOR
000000 AMS05LEN DS X VECTOR LENGTH
000001 AMS05KEY DS X VECTOR KEY
AMS05KCYC EQU X'05' KEY IS X'05'
000002 AMS05DATA DS 0X VECTOR DATA
000002 AMS05DTS0 DS X BYTE 0 OF INDICATORS
AMS05B00 EQU X'80' NIB ENCR AND RPL CRYPT
AMS05B01 EQU X'40' ACB PARMS=NIB (COMMUNICATION NETWORK MANAGEMENT INTERFACE)
AMS05B02 EQU X'20' MULTIPLE-ADDRESS-SPACE APPLICATIONS PROGRAMS
AMS05B03 EQU X'10' AUTHORIZED PATH FOR COMMUNICATIONS MACROS
AMS05B04 EQU X'08' AUTHORIZED PATH FOR ALL RPL-BASED MACROS
AMS05B05 EQU X'04' SRBEXIT (ON APPL DEFINITION STATEMENT)
AMS05B06 EQU X'02' SSONSIP (ON APPL DEFINITION STATEMENT)
AMS05B07 EQU X'01' VTAMFRR (ON APPL DEFINITION STATEMENT)

000003 AMS05DT1 DS X BYTE 1 OF INDICATORS
AMS05B10 EQU X'80' ACB PARMS=NIB (COMMUNICATION NETWORK MANAGEMENT INTERFACE)
AMS05B11 EQU X'40' RPL OPTCD=LMPEO
AMS05B12 EQU X'20' RPL OPTCD=BUFFLST
AMS05B13 EQU X'10' RPL OPTCD=USERFH
AMS05B14 EQU X'08' ACB PARMS=USERFLO
AMS05B15 EQU X'04' RPL BRACKET-CEB
AMS05B16 EQU X'02' APPLICATION PROGRAM ASSIGNMENT OF SEQUENCE NUMBERS FOR EXPEDITED DFC
AMS05B17 EQU X'01' RESOURCE-IDENTIFICATION VECTOR LIST

000004 AMS05DT2 DS X BYTE 2 OF INDICATORS
AMS05B20 EQU X'80' ACCESS-METHOD-SUPPORT VECTOR LIST
AMS05B21 EQU X'40' RETURN OF SYSTEM RESPONSE BYTE AND EXTENDED RESPONSE BYTE FOR BSC 3270 TERMINALS ATTACHED TO ACF/NCP
AMS05B22 EQU X'20' INTREPRT
AMS05B23 EQU X'10' VTAM API IS XRF CAPABLE
AMS05B24 EQU X'08' SENSE ON -RSP(CINIT). CLSDST OPTCD=(RELEASE,SENSE)
AMS05B25 EQU X'04' UNBIND SON CODE AND SENSE. CLSDST OPTCD=(UNBIND,SONCODE).
AMS05B26 EQU X'04' UNBIND SON CODE AND SENSE. CLSDST OPTCD=(UNBIND,SONCODE).
AMS05B27 EQU X'02' HOLD/RELEASE LOGON/SCIP EXIT FOR SESSION SETUP.
AMS05B28 EQU X'01' CINIT - NETWORK ADDRESSES IN VECTOR KEY X'15'

000005 AMS05DT3 DS X BYTE 3 OF INDICATORS
AMS05B30 EQU X'80' 31-BIT API
AMS05B31 EQU X'40' NOTIFICATION OF QUEUED RESPONSES SUPPORTED. SEND OPTCD=(RSPQUED)
AMS05B32 EQU X'20' APPC IS SUPPORTED
AMS05B33 EQU X'10' ADD SUPPORT FOR USERVAR
AMS05B34 EQU X'08' VCNS API SUPPORT FOR X.25
AMS05B35 EQU X'04' VCNS API SUPPORT FOR TOKEN BUS,
AMS05B36 EQU X'02' CROSS-MEMORY API IS SUPPORTED
AMS05B37 EQU X'01' KEEPFRR SUPPORT (ON ACB STATEMENT)

AMS05DT4 DS X BYTE 4 OF INDICATORS
AMS05B40 EQU X'80' SRBEXIT SUPPORT (ON ACB STATEMENT)
AMS05B41 EQU X'40' PERSISTENT LU-LU SESSIONS
AMS05B42 EQU X'20' V.25BIS SUPPORT
AMS05B43 EQU X'10' VTAM/NPM INTERFACE SUPPORT
AMS05B44 EQU X'08' LU6 PLUS TRACKING SUPPORTED
AMS05B45 EQU X'04' BYTE 4, BIT 5: RESERVED
AMS05B46 EQU X'02' BYTE 4, BIT 6: RESERVED
AMS05B47 EQU X'01' NETWORK QUALIFIED NAMES SUPPORTED

AMS05DT5 DS X BYTE 5 OF INDICATORS
AMS05B50 EQU X'80' MS TRANSPORT SUPPORTED
AMS05B51 EQU X'40' PERFORMANCE MONITOR INTERFACE SUPPORTED
AMS05B52 EQU X'20' QUEUED SESSION TERMINATION SUPPORTED
AMS05B53 EQU X'10' VTAM AGENT SUPPORTED
AMS05B54 EQU X'08' GENERIC RESOURCES SUPPORTED
AMS05B55 EQU X'04' OPTCD=KEEPSRB FOR SYNC SRB SUPPORTED
AMS05B56 EQU X'02' APPLICATION VECTORS SUPPORTED ON ACB MACRO
AMS05B57 EQU X'01' SETLOGON GNAMESUB SUPPORTED

AMS05DT6 DS X BYTE 6 OF INDICATORS
AMS05B60 EQU X'80' BYTE 6, BIT 0: RESERVED
AMS05B61 EQU X'40' BYTE 6, BIT 1: RESERVED
AMS05B62 EQU X'20' BYTE 6, BIT 2: RESERVED
AMS05B63 EQU X'10' BYTE 6, BIT 3: RESERVED
AMS05B64 EQU X'08' BYTE 6, BIT 4: RESERVED
AMS05B65 EQU X'04' BYTE 6, BIT 5: RESERVED
AMS05B66 EQU X'02' BYTE 6, BIT 6: RESERVED
AMS05B67 EQU X'01' BYTE 6, BIT 7: RESERVED

AMS05DT7 DS X BYTE 7 OF INDICATORS
AMS05B70 EQU X'80' BYTE 7, BIT 0: RESERVED
AMS05B71 EQU X'40' BYTE 7, BIT 1: RESERVED
AMS05B72 EQU X'20' BYTE 7, BIT 2: RESERVED
AMS05B73 EQU X'10' BYTE 7, BIT 3: RESERVED
AMS05B74 EQU X'08' BYTE 7, BIT 4: RESERVED
AMS05B75 EQU X'04' BYTE 7, BIT 5: RESERVED
AMS05B76 EQU X'02' BYTE 7, BIT 6: RESERVED
AMS05B77 EQU X'01' BYTE 7, BIT 7: RESERVED

******************************************************************************
*** ISTAMS06 - maps the LU6.2 SUPPORT FUNCTION LIST vector. **
*** - The vector data is a variable-length string of byte **
*** indicators, each byte corresponding to a particular **
*** LU6.2 function. Each byte will have a value showing **
*** that its corresponding function is either supported, **
*** not supported, or supported on a "pass-through" basis.**
*** (Pass-through functions are those which VTAM does not **
*** directly provide but provides the ability for the **
*** application to create the support itself.) **
*** If the vector is not present or if the byte string **
*** is shorter than expected, you may assume that the **
*** missing bytes are zero and their corresponding **
*** functions are not supported. **
*** - These indicator bytes correspond to the compile-time **
*** global indicators in the ISTGAPPC macro. **
******************************************************************************
AMS06LEN DS X VECTOR LENGTH
AMS06KEY DS X VECTOR KEY
AMS06KCYC EQU X'06'
KEY IS X'06'
AMS06DTA DS 0X VECTOR DATA
AMS06D01 DS X 01. CONVERSATIONS BETWEEN TPS
AMS06D02 DS X 02. DELAYED SESSION
AMS06D03 DS X 03. IMMEDIATE SESSION
AMS06D04 DS X 04. SYNC POINT SERVICES
AMS06D05 DS X 05. PROGRAM RECONNECT
AMS06D06 DS X 06. RESERVED
AMS06D07 DS X 07. SESSION-LEVEL LU-LU
AMS06D08 DS X 08. USERID VERIFICATION
AMS06D09 DS X 09. PROGRAM SUPPLIED USERID
AMS06D10 DS X 10. USERID AUTHORIZATION
AMS06D11 DS X 11. PROFILE VERIFICATION AND
AMS06D12 DS X 12. RESERVED
AMS06D13 DS X 13. PROFILE PASSTHROUGH
AMS06D14 DS X 14. PROGRAM-SUPPLIED PROFILE
AMS06D15 DS X 15. SEND PERSISTENT
AMS06D16 DS X 16. RECEIVE PERSISTENT
AMS06D17 DS X 17. PIP DATA
AMS06D18 DS X 18. LOGGING OF DATA IN SYSTEM
AMS06D19 DS X 19. FLUSH LU SEND BUFFER
AMS06D20 DS X 20. LUW IDENTIFIER
AMS06D21 DS X 21. PREPARE TO RECEIVE
AMS06D22 DS X 22. LONG LOCKS
AMS06D23 DS X 23. POST ON RECEIPT WITH WAIT
AMS06D24 DS X 24. POST ON RECEIPT WITH TEST
AMS06D25 DS X 25. RECEIVE-IMMEDIATE
AMS06D26 DS X 26. TEST FOR REQUEST-TO-SEND
AMS06D27 DS X 27. DATA MAPPING
AMS06D28 DS X 28. FMH APPLICATION-DATA
AMS06D29 DS X 29. GET ATTRIBUTES
AMS06D30 DS X 30. GET CONVERSATION-TYPE
AMS06D31 DS X 31. MAPPED CONVERSATION LU
AMS06D32 DS X 32. CHANGE_SESSION_LIMIT VERB
AMS06D33 DS X 33. MIN_CONWINNERS_TARGET
AMS06D34 DS X 34. RESPONSIBLE(TARGET)
AMS06D35 DS X 35. DRAIN_TARGET(NO) PARAMETER
AMS06D36 DS X 36. FORCE PARAMETER
AMS06D37 DS X 37. ACTIVATE_SESSION VERB
AMS06D38 DS X 38. DEACTIVATE_SESSION VERB
AMS06D39 DS X 39. LU-PARAMETER VERBS
AMS06D40 DS X 40. LU-LU SESSION LIMIT
AMS06D41 DS X 41. LOCALLY-KNOWN LU NAMES
AMS06D42 DS X 42. UNINTERPRETED LU NAMES
AMS06D43 DS X 43. SINGLE-SESSION
AMS06D44 DS X 44. ALTERNATE CODE PROCESSING
AMS06D45 DS X 45. MAXIMUM RU SIZE BOUNDS
AMS06D46 DS X 46. SESSION-LEVEL MANDATORY
AMS06D47 DS X 47. CONTENTION WINNER

AMS06KEY C EQU X'06'
KEY IS X'06'
AMS06D01 D 01. CONVERSATIONS BETWEEN TPS
AMS06D02 D 02. DELAYED SESSION
AMS06D03 D 03. IMMEDIATE SESSION
AMS06D04 D 04. SYNC POINT SERVICES
AMS06D05 D 05. PROGRAM RECONNECT
AMS06D06 D 06. RESERVED
AMS06D07 D 07. SESSION-LEVEL LU-LU
AMS06D08 D 08. USERID VERIFICATION
AMS06D09 D 09. PROGRAM SUPPLIED USERID
AMS06D10 D 10. USERID AUTHORIZATION
AMS06D11 D 11. PROFILE VERIFICATION AND
AMS06D12 D 12. RESERVED
AMS06D13 D 13. PROFILE PASSTHROUGH
AMS06D14 D 14. PROGRAM-SUPPLIED PROFILE
AMS06D15 D 15. SEND PERSISTENT
AMS06D16 D 16. RECEIVE PERSISTENT
AMS06D17 D 17. PIP DATA
AMS06D18 D 18. LOGGING OF DATA IN SYSTEM
AMS06D19 D 19. FLUSH LU SEND BUFFER
AMS06D20 D 20. LUW IDENTIFIER
AMS06D21 D 21. PREPARE TO RECEIVE
AMS06D22 D 22. LONG LOCKS
AMS06D23 D 23. POST ON RECEIPT WITH WAIT
AMS06D24 D 24. POST ON RECEIPT WITH TEST
AMS06D25 D 25. RECEIVE-IMMEDIATE
AMS06D26 D 26. TEST FOR REQUEST-TO-SEND
AMS06D27 D 27. DATA MAPPING
AMS06D28 D 28. FMH APPLICATION-DATA
AMS06D29 D 29. GET ATTRIBUTES
AMS06D30 D 30. GET CONVERSATION-TYPE
AMS06D31 D 31. MAPPED CONVERSATION LU
AMS06D32 D 32. CHANGE_SESSION_LIMIT VERB
AMS06D33 D 33. MIN_CONWINNERS_TARGET
AMS06D34 D 34. RESPONSIBLE(TARGET)
AMS06D35 D 35. DRAIN_TARGET(NO) PARAMETER
AMS06D36 D 36. FORCE PARAMETER
AMS06D37 D 37. ACTIVATE_SESSION VERB
AMS06D38 D 38. DEACTIVATE_SESSION VERB
AMS06D39 D 39. LU-PARAMETER VERBS
AMS06D40 D 40. LU-LU SESSION LIMIT
AMS06D41 D 41. LOCALLY-KNOWN LU NAMES
AMS06D42 D 42. UNINTERPRETED LU NAMES
AMS06D43 D 43. SINGLE-SESSION
AMS06D44 D 44. ALTERNATE CODE PROCESSING
AMS06D45 D 45. MAXIMUM RU SIZE BOUNDS
AMS06D46 D 46. SESSION-LEVEL MANDATORY
AMS06D47 D 47. CONTENTION WINNER
Resource-information vector list (ISTRIVL)

Loc  Source Statement
000000 ISTRIVL DSECT MAPPING FOR RESOURCE INFORMATION
   *  VECTOR LIST POINTED TO BY ACBRIVL
000000 RIVLLEN DS HL2 TOTAL LENGTH OF VECTORS
000002 RIVLDATA DS 0X VECTOR DATA
   *
   ***********************************************************************
   *** GENERALIZED MAPPING FOR EXAMINING COMMON FIELDS IN ALL ACB
   *** VECTORS IN THE VECTOR LIST POINTED TO BY ACBRIVL
   ***********************************************************************
000000 ISTRIVVT DSECT VECTOR TEMPLATE 0Y3A
000000 RIVVTLEN DS X VECTOR LENGTH 0Y3A
000001 RIVVTKEY DS X VECTOR KEY 0Y3A
000002 RIVVTDAT DS CL8 VECTOR DATA
   *
   ***********************************************************************
   *** ISTRIV02 - maps the application's network name vector. **
   *** - The name is specified by the name field of the **
   ***  application definition statement. **
   *** - This is obtained from the NAME ON APPL STATEMENT. **
   ***********************************************************************
000000 ISTRIV02 DSECT APPLICATION NETWORK NAME VECTOR
   *  (FROM NAME ON APPL STATEMENT)
000000 RIV02LEN DS X VECTOR LENGTH
000001 RIV02KEY DS X VECTOR KEY
RIV02KYE EQU X'02' KEY IS X'02'
000002 RIV02DATA DS CL8 VECTOR DATA
   *
   ***********************************************************************
   *** ISTRIV03 - maps the application's ACB name vector. **
   *** - This is supplied by the APPLID operand on the ACB **
   ***  statement or can be supplied by the operating **
   ***  system. During OPEN ACB, VTAM will search for the **
   ***  application's characteristics by matching the ACB **
   ***  APPLID value to an RDTE with the application's **
   ***  ACBNAME. If ACBNAME was not coded for the **
   ***  application, VTAM will search for a match with an **
   ***  RDTE containing the application's network name. **
   *** - This is obtained from the APPLID on ACB MACRO. **
   ***********************************************************************
000000 ISTRIV03 DSECT APPLICATION ACB NAME VECTOR
   *  (FROM APPLID ON ACB MACRO)
000000 RIV03LEN DS X VECTOR LENGTH
000001 RIV03KEY DS X VECTOR KEY
RIV03KYC EQU X'03' KEY IS X'03'
000002 RIV03DTA DS CL8 VECTOR DATA
*
***********************************************************************
*** ISTRIV06 - maps the network name in which the host resides. **
*** - This is obtained from the NETID START OPTION. **
*** If NETID start option is not specified, this value **
*** will be blanks. **
***********************************************************************
000000 ISTRIV06 DSECT NETWORK NAME VECTOR
* (FROM NETID START OPTION)
000000 RIV06LEN DS X VECTOR LENGTH
000001 RIV06KEY DS X VECTOR KEY
RIV06KYC EQU X'06' KEY IS X'06'
000002 RIV06DTA DS CL8 VECTOR DATA
*
***********************************************************************
*** ISTRIV07 - maps the SSCP Name vector. **
*** - This is obtained from the SSCPNAME START OPTION **
***********************************************************************
000000 ISTRIV07 DSECT SSCP NAME VECTOR
* (FROM SSCPNAME START OPTION)
000000 RIV07LEN DS X VECTOR LENGTH
000001 RIV07KEY DS X VECTOR KEY
RIV07KYC EQU X'07' KEY IS X'07'
000002 RIV07DTA DS CL8 VECTOR DATA
* (DEFAULT IS 'VTAM')
*
***********************************************************************
*** ISTRIV08 - maps the Host Subarea PU Network Name vector. **
*** - This is obtained from the HOSTPU START OPTION **
*** If HOSTPU start option is not specified, the name **
*** will default to 'ISTPUS '. **
***********************************************************************
000000 ISTRIV08 DSECT HOST SUBAREA PU NETWORK NAME VECTOR
* (FROM HOSTPU START OPTION)
000000 RIV08LEN DS X VECTOR LENGTH
000001 RIV08KEY DS X VECTOR KEY
RIV08KYC EQU X'08' KEY IS X'08'
000002 RIV08DTA DS CL8 VECTOR DATA
* (DEFAULT IS 'ISTPUS')
*
***********************************************************************
*** ISTRIV09 - maps the Host Subarea PU network address vector. **
*** - It contains the network address of the host **
*** subarea PU. **
***********************************************************************
000000 ISTRIV09 DSECT HOST SUBAREA PU NETWORK ADDRESS
*
000000 RIV09LEN DS X VECTOR LENGTH
000001 RIV09KEY DS X VECTOR KEY
RIV09KYC EQU X'09' KEY IS X'09'
000002 RIV09DTA DS XL6 VECTOR DATA
*
***********************************************************************
*** ISTRIV0A - maps the maximum subarea vector. **
*** - Contains the maximum subarea number that is valid **
*** for the host's domain. **
*** - This is obtained from the MAXSUBA START OPTION **
***********************************************************************
000000 ISTRIV0A DSECT MAXIMUM SUBAREA NUMBER VECTOR
* (FROM MAXSUBA START OPTION)
000000 RIV0ALEN DS X VECTOR LENGTH
000001 RIV0AKEY DS X VECTOR KEY
RIV0AKYC EQU X'0A' KEY IS X'0A'
000002 RIV0ADTA DS X VECTOR DATA
*
**** ISTRIV0B - maps the LU 6.2 application definition vector. **
*** After the LU 6.2 application program has issued an **
*** open ACB, the LU 6.2 application program may use **
*** this vector to determine the values coded on the **
*** APPL definition statement. **
*** - This is obtained from the APPL STATEMENT PARAMETERS **
***********************************************************************

000000 ISTRIV0B DSECT LU 6.2 APPL DEFINITION VECTOR
* (FROM APPL STATEMENT PARAMETERS)
000000 RIVOBLEN DS X VECTOR LENGTH
000001 RIVOBKEY DS X VECTOR KEY
RIVOBKYE EQU X'0B' KEY IS X'0B'
000002 RIVOBDTA DS 0X VECTOR DATA
000002 DS X RESERVED
RIVOBLSL EQU X'C0' SESSION-LEVEL LU-LU VERIFICATION
* BIT MASK
RIVOBLSR EQU X'80' REQUIRED
RIVOBSDL EQU X'40' OPTIONAL
RIVOBSLN EQU X'00' NONE
000003 RIVOBCLS DS X CONVERSATION SECURITY ACCEPTANCE
RIVOBCLN EQU X'01' NONE
RIVOBCLC EQU X'02' CONV
RIVOBCLA EQU X'03' ALREADY
RIVOBCLP EQU X'04' PERSISTV
RIVOBCLV EQU X'05' AVPV

***********************************************************************

*** ISTRIV0C - maps the common application definition vector. **
*** After the application program has issued an open for **
*** its ACB, the application may examine this vector to **
*** determine the values coded on the APPL definition **
*** statement for common application definition keywords. **
*** - This is obtained from the APPL STATEMENT PARAMETERS **
***********************************************************************

000000 ISTRIV0C DSECT APPLICATION DEFINITION VECTOR
* FOR ALL APPLICATION PROGRAMS @N1A
000000 RIV0CLEN DS X VECTOR LENGTH @N1A
000001 RIV0CKEY DS X VECTOR KEY @N1A
RIV0CKYC EQU X'0C' KEY IS X'0C' @N1A
000002 RIV0CDTA DS 0X VECTOR DATA @N1A
000002 RIV0CAUT DS X AUTHORIZATION SETTINGS @N1A
RIV0CACQ EQU X'80' AUTH=ACQ @N1A
RIV0CASD EQU X'40' AUTH=ASDP @N1A
RIV0CCNM EQU X'20' AUTH=CNM @N1A
RIV0CPAS EQU X'10' AUTH=PASS @N1A
RIV0CPPO EQU X'08' AUTH=PPO @N1A
RIV0CSPO EQU X'04' AUTH=SPO @N1A
RIV0CTSO EQU X'02' AUTH=TSO @N1A
RIV0CVPA EQU X'01' AUTH=VPACE @N1A
000003 RIV0CFL1 DS X MISCELLANEOUS FLAGS 1 @N1A
RIV0CFL1 EQU X'80' DRAINL=ALLOW @N1A
RIV0CFL2 EQU X'40' DRESPL=ALLOW @N1A
RIV0CTSA EQU X'20' ATNLOSS=ALL @N1A
RIV0CFL3 EQU X'10' SYNCLVL=SYNPT @N1A
RIV0CFL4 EQU X'08' OPERCNOS=ALLOW @N1A
000005 DS X RESERVED @N1A
000006 RIV0BDSL DS HL2 DSESLSM VALUE @N1A
000006 RIV0BDMKL DS HL2 DMINNWL VALUE @N1A
000006 RIV0BDMKR DS HL2 DMINWR VALUE @N1A
000006 RIV0BDAUT DS HL2 AUTOSES VALUE @N1A
*

***********************************************************************
### ISTRIV11 - maps the APPCCMD vector area length vector
- It contains the absolute minimum length and the recommended minimum length for full use of the APPCCMD vector area.

```assembly
000000 ISTRIV11 DSECT APPCCMD VECTOR AREA LENGTH VECTOR
   * _0L3C
000000 RIV11LEN DS X VECTOR LENGTH
000001 RIV11KEY DS X VECTOR KEY
RIV11KYC EQU X'11' KEY IS X'11'
000002 RIV11AML DS XL4 ABSOLUTE MINIMUM APPCCMD VECTOR
   * AREA LENGTH _0L3A
000006 RIV11RML DS XL4 RECOMMENDED MINIMUM APPCCMD
   * VECTOR AREA LENGTH _0L3C
```

### ISTRIV12 - maps the application to VTAM vector keys vector
- It contains a list of all ACB vector keys that VTAM will process. Constants for the ACB vectors are located in ISTVACBV.

```assembly
000000 ISTRIV12 DSECT APPLICATION TO VTAM VECTOR KEYS
   * FOR ACB MACRO
000000 RIV12LEN DS X VECTOR LENGTH
000001 RIV12KEY DS X VECTOR KEY
RIV12KYC EQU X'12' KEY IS X'12'
000002 RIV12DATA DS OCL1 VECTOR DATA
   *
```

### ISTRIV13 - maps the Performance Monitor vector
- Identifies a table of Performance Data vector fields (within ISTXPL) that have been retired by the Performance Monitor Interface since its inception.

```assembly
000000 ISTRIV13 DSECT PERFORMANCE MONITOR VECTOR _0L1A
   *
000000 RIV13LEN DS X VECTOR LENGTH _0L1A
000001 RIV13KEY DS X VECTOR KEY _0L1A
RIV13KYC EQU X'13' KEY IS X'13' _0L1A
000002 RIV13ENT DS HL2 NUMBER OF ENTRIES IN TABLE _0L2A
   (ZERO IF NONE RETIRED)
```
Extended buffer list entry (ISTBLXEN)

```assembly
LOC       SOURCE STATEMENT
000000 ISTBLXEN DSECT
000000 BLXEN_CSM    DS  OCL2B    THIS AREA MAPS THE CSM
                     BUFFER DESCRIPTOR
*     000000      DS  X        RESERVED. THIS FIELD MUST BE
                     SET TO ZERO.
     000001 BLXEN_SOURCE DS  X        BUFFER SOURCE
     000001       EQU X'80'     INDICATES THAT THE STORAGE
                     REFERENCED IN THE LIST IS
                     CSM ECSA
     000001       EQU X'40'     INDICATES THAT THE STORAGE
                     REFERENCED IN THE LIST IS
                     CSM DATA SPACE
                     *     000002      DS  X        BUFFER TYPE
                     *     000002       EQU X'80'     INDICATES THAT THE STORAGE IS
                     *                     IN A GUARANTEED TO BE FIXED
                     *                     STATE
                     *     000002       EQU X'40'     INDICATES THAT THE STORAGE IS
                     *                     IN A GUARANTEED TO BE PAGEABLE
                     *                     STATE
                     *     000002       EQU X'20'     INDICATES THAT THE STORAGE
                     *                     IS ELIGIBLE TO BE PAGEFREED BY
                     *                     CSM
                     *     000003      DS  XL1      RESERVED
     000004 BLXEN_CTKN DS  XL12     CSM TOKEN
     000010 BLXEN_ALET DS  F        CSM DATA SPACE ALET
     000014 BLXEN_AREA DS  A        POINTER TO DATA
     000018 BLXEN_RLEN DS  F        LENGTH OF DATA
     00001C BLXEN_RLENA DS  F        LENGTH OF DATA ACCEPTED BY
                     ... VTAM ON A REQUEST TO SEND
                     ... DATA.
                     ... THIS FIELD SHOULD BE SET
                     ... TO ZERO BY THE APPLICATION.
                     ... VTAM SETS THIS FIELD
                     ... TO REFLECT THE AMOUNT OF
                     ... DATA REFERENCED BY XBULST
                     ... THAT HAS BEEN ACCEPTED BY
                     ... VTAM.
     000020 BLXEN_VAFLAGS DS  X        VTAM and APPL FLAGS
     000020       EQU X'80'     VTAM HAS ACCEPTED OWNERSHIP
                     OF THE CSM BUFFER
     000021      DS  XL15      RESERVED
```
Chapter 4. Summary of register usage

Table 3 shows what VTAM does with the general-purpose registers before it returns control to the application program at the next sequential instruction. It indicates which registers are left unchanged by the VTAM macroinstructions and which ones can be modified between the time the macroinstruction is executed and control is returned to the application program. The table also shows the disposition of the registers when any of the exit routines receive control. Refer to z/OS Communications Server: SNA Programmer’s LU 6.2 Guide for further details on how to handle macroinstruction errors.

Table 3. Register contents upon return of control

<table>
<thead>
<tr>
<th>Upon return from OPEN and CLOSE macroinstructions</th>
<th>Register 0</th>
<th>Register 1</th>
<th>Register 2-12</th>
<th>Register 13</th>
<th>Register 14</th>
<th>Register 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN and CLOSE</td>
<td>Unpredictable</td>
<td>Unpredictable</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>1</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon return from RPL-based macroinstructions, including CHECK</td>
<td>See footnote 2</td>
<td>Address of RPL</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>1</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon return from GENCB</td>
<td>Error return code or control block address</td>
<td>Control block address (^3) (^4)</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>1</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon return from SHOWCB, MODCB, or TESTCB</td>
<td>Error return code (^4)</td>
<td>Unpredictable</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>1</td>
<td>Unpredictable</td>
</tr>
<tr>
<td>Upon invocation of LERAD or SYNAD exit routines</td>
<td>Recovery action return code</td>
<td>Address of RPL</td>
<td>Unmodified</td>
<td>Unmodified</td>
<td>1</td>
<td>Return address</td>
</tr>
<tr>
<td>Upon invocation of other EXLST exit routines</td>
<td>Unpredictable</td>
<td>Address of VTAM-supplied parameter list</td>
<td>Unpredictable</td>
<td>Unpredictable</td>
<td>Return address</td>
<td>Address of exit routine</td>
</tr>
<tr>
<td>Upon invocation of RPL-based exit routines</td>
<td>Unpredictable</td>
<td>Address of RPL</td>
<td>Unpredictable</td>
<td>Unpredictable</td>
<td>Return address</td>
<td>Address of exit routine</td>
</tr>
</tbody>
</table>

1. Register 13 must indicate the address of an 18-word save area when the macroinstruction is executed.
2. If the operation completed normally, register 15 is set to 0. For some macroinstructions completing normally but with a special condition, register 0 is also set. If an error occurred and the LERAD or SYNAD exit routine has been invoked, registers 0 and 15 contain the values set in them by the exit routine. If an error occurred and no LERAD or SYNAD exit routine exists, VTAM sets register 15 to 4 and places a recovery action return code in register 0 (if the error is that the ACB is not open, register 15 is set to decimal 32 and the RPL request code is set in register 0).
3. When GENCB completes successfully (register 15 is set to 0), register 1 contains the address of the generated control blocks and register 0 contains the length of the control blocks, in bytes.
4. If GENCB, SHOWCB, MODCB, or TESTCB completes unsuccessfully (with register 15 not set to 0), register 1 is unpredictable and register 0 contains an error code (if register 15 is set to 4 or 12) or else is unpredictable.
Appendix A. Architectural specifications

This appendix lists documents that provide architectural specifications for the SNA Protocol.

The APPN Implementers’ Workshop (AIW) architecture documentation includes the following architectural specifications for SNA APPN and HPR:

- APPN Architecture Reference (SG30-3422-04)
- APPN Branch Extender Architecture Reference Version 1.1
- APPN Dependent LU Requester Architecture Reference Version 1.5
- APPN Extended Border Node Architecture Reference Version 1.0
- APPN High Performance Routing Architecture Reference Version 4.0
- SNA Formats (GA27-3136-20)
- SNA Technical Overview (GC30-3073-04)

For more information, see the AIW documentation page at http://www.ibm.com/support/docview.wss?rs=852&uid=swg27017843.

The following RFC also contains SNA architectural specifications:

- RFC 2353 APPN/HPR in IP Networks APPN Implementers’ Workshop Closed Pages Document

RFCs can be obtained from:

Government Systems, Inc.
Attn: Network Information Center
14200 Park Meadow Drive
Suite 200
Chantilly, VA 22021

Many RFCs are available online. Hardcopies of all RFCs are available from the NIC, either individually or by subscription. Online copies are available using FTP from the NIC at http://www.rfc-editor.org/rfc.html.

Use FTP to download the files, using the following format:

```
RFC:RFC-INDEX.TXT
RFC:RFCnnnn.TXT
RFC:RFCnnnn.PS
```

where:

- `nnnn` is the RFC number.
- `TXT` is the text format.
- `PS` is the postscript format.

You can also request RFCs through electronic mail, from the automated NIC mail server, by sending a message to service@nic.ddn.mil with a subject line of RFC `nnnn` for text versions or a subject line of RFC `nnnn.PS` for PostScript versions. To request a copy of the RFC index, send a message with a subject line of RFC INDEX.

For more information, contact nic@nic.ddn.mil.
Appendix B. Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you can view the information through the z/OS Internet Library website or the z/OS Information Center. If you continue to experience problems, send an email to mhvrcfs@us.ibm.com or write to:

IBM Corporation  
Attention: MHVRCFS Reader Comments  
Department H6MA, Building 707  
2455 South Road  
Poughkeepsie, NY 12601-5400  
USA

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. See z/OS TSO/E Primer, z/OS TSO/E User's Guide, and z/OS ISPF User's Guide Vol I for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at www.ibm.com/systems/z/os/zos/bkserv/

One exception is command syntax that is published in railroad track format, which is accessible using screen readers with the Information Center, as described in "Dotted decimal syntax diagrams."

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always
present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should see separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- A question mark (?) means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

- An exclamation mark (!) means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted
decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1!(KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1!(KEEP), and 2.1.1 (DELETE), the default option KEEP applies only to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- An asterisk (*) means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:
1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.
Notices

This information was developed for products and services offered in the USA.

IBM may not offer all of the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
P.O. Box 12195
3039 Cornwallis Road
Research Triangle Park, North Carolina 27709-2195
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations might not appear.

COPYRIGHT LICENSE:

This information might contain sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing
application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_.

IBM is required to include the following statements in order to distribute portions of this document and the software described herein to which contributions have been made by The University of California. Portions herein © Copyright 1979, 1980, 1983, 1986, Regents of the University of California. Reproduced by permission. Portions herein were developed at the Electrical Engineering and Computer Sciences Department at the Berkeley campus of the University of California under the auspices of the Regents of the University of California.

Portions of this publication relating to RPC are Copyright © Sun Microsystems, Inc., 1988, 1989.

Some portions of this publication relating to X Window System** are Copyright © 1987, 1988 by Digital Equipment Corporation, Maynard, Massachusetts, and the Massachusetts Institute Of Technology, Cambridge, Massachusetts.

Some portions of this publication relating to X Window System are Copyright © 1986, 1987, 1988 by Hewlett-Packard Corporation.

Permission to use, copy, modify, and distribute the M.I.T., Digital Equipment Corporation, and Hewlett-Packard Corporation portions of this software and its documentation for any purpose without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of M.I.T., Digital, and Hewlett-Packard not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T., Digital, and Hewlett-Packard make no representation about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.


Copyright © 1988, 1993 The Regents of the University of California.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment:
This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software program contains code, and/or derivatives or modifications of code originating from the software program "Popper." Popper is Copyright ©1989-1991 The Regents of the University of California. Popper was created by Austin Shelton, Information Systems and Technology, University of California, Berkeley.

Permission from the Regents of the University of California to use, copy, modify, and distribute the "Popper" software contained herein for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies. HOWEVER, ADDITIONAL PERMISSIONS MAY BE NECESSARY FROM OTHER PERSONS OR ENTITIES, TO USE DERIVATIVES OR MODIFICATIONS OF POPPER.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THE POPPER SOFTWARE, OR ITS DERIVATIVES OR MODIFICATIONS, AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE POPPER SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

Copyright © 1983 The Regents of the University of California.

Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by the University of California, Berkeley. The name of the University may not be used to endorse or promote products derived from this software without specific prior
written permission. THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Copyright © 1991, 1993 The Regents of the University of California.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment:
   This product includes software developed by the University of California, Berkeley and its contributors.
4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 1990 by the Massachusetts Institute of Technology

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Furthermore if you modify this software you must label your software as modified software and not distribute it in such a fashion that it might be confused with the original M.I.T. software. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Copyright © 1998 by the FundsXpress, INC.
Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of FundsXpress not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. FundsXpress makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.


Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright © 1995-1998 Eric Young (eay@cryptsoft.com)

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscape's SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)." The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related.

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include acknowledgment:

   "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

The license and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License.]

This product includes cryptographic software written by Eric Young.


Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND INTERNET SOFTWARE CONSORTIUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INTERNET SOFTWARE CONSORTIUM BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright © 2004 IBM Corporation and its licensors, including Sendmail, Inc., and the Regents of the University of California.

Copyright © 1999,2000,2001 Compaq Computer Corporation

Copyright © 1999,2000,2001 Hewlett-Packard Company
Copyright © 1999,2000,2001 IBM Corporation

Copyright © 1999,2000,2001 Hummingbird Communications Ltd.


Copyright © 1999,2000,2001 Sun Microsystems, Inc.

Copyright © 1999,2000,2001 The Open Group

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

X Window System is a trademark of The Open Group.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

You can obtain softcopy from the z/OS Collection (SK3T-4269), which contains BookManager and PDF formats.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [http://www-01.ibm.com/software/support/systemsz/lifecycle/](http://www-01.ibm.com/software/support/systemsz/lifecycle/)
- For information about currently-supported IBM hardware, contact your IBM representative.
Programming interface information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/OS Communications Server.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies.

A current list of IBM trademarks is available on the web at [Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml).

Intel is a registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Java™ and all Java-based trademarks are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Adobe and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.
Bibliography

This bibliography contains descriptions of the documents in the z/OS Communications Server library.

z/OS Communications Server documentation is available in the following forms:

- Online at the z/OS Internet Library web page at www.ibm.com/systems/z/os/zos/bkserv/
- In softcopy on CD-ROM collections. See "Softcopy information" on page xiv.

z/OS Communications Server library updates

An index to z/OS Communications Server book updates is at http://www.ibm.com/support/docview.wss?uid=swg21178966. Updates to documents are also available on RETAIN® and in information APARs (info APARs). Go to http://www.ibm.com/software/network/commserver/zos/support to view information APARs. In addition, Info APARs for z/OS documents are in z/OS and z/OS.e DOC APAR and PTF ++HOLD Documentation, which can be found at http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ZDOCAPAR

z/OS Communications Server information

z/OS Communications Server product information is grouped by task in the following tables.

Planning

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: New Function Summary</td>
<td>GC27-3664</td>
<td>This document is intended to help you plan for new IP or SNA function, whether you are migrating from a previous version or installing z/OS for the first time. It summarizes what is new in the release and identifies the suggested and required modifications needed to use the enhanced functions.</td>
</tr>
<tr>
<td>z/OS Communications Server: IPv6 Network and Application Design Guide</td>
<td>SC27-3663</td>
<td>This document is a high-level introduction to IPv6. It describes concepts of z/OS Communications Server’s support of IPv6, coexistence with IPv4, and migration issues.</td>
</tr>
</tbody>
</table>

Resource definition, configuration, and tuning

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Configuration Guide</td>
<td>SC27-3650</td>
<td>This document describes the major concepts involved in understanding and configuring an IP network. Familiarity with the z/OS operating system, IP protocols, z/OS UNIX System Services, and IBM Time Sharing Option (TSO) is recommended. Use this document with the z/OS Communications Server: IP Configuration Reference.</td>
</tr>
</tbody>
</table>
## z/OS Communications Server: IP Configuration Reference

**Title**: z/OS Communications Server: IP Configuration Reference  
**Number**: SC27-3651  
**Description**: This document presents information for people who want to administer and maintain IP. Use this document with the z/OS Communications Server: IP Configuration Guide. The information in this document includes:  
- TCP/IP configuration data sets  
- Configuration statements  
- Translation tables  
- Protocol number and port assignments

## z/OS Communications Server: SNA Network Implementation Guide

**Title**: z/OS Communications Server: SNA Network Implementation Guide  
**Number**: SC27-3672  
**Description**: This document presents the major concepts involved in implementing an SNA network. Use this document with the z/OS Communications Server: SNA Resource Definition Reference.

## z/OS Communications Server: SNA Resource Definition Reference

**Title**: z/OS Communications Server: SNA Resource Definition Reference  
**Number**: SC27-3675  
**Description**: This document describes each SNA definition statement, start option, and macroinstruction for user tables. It also describes NCP definition statements that affect SNA. Use this document with the z/OS Communications Server: SNA Network Implementation Guide.

## z/OS Communications Server: SNA Resource Definition Samples

**Title**: z/OS Communications Server: SNA Resource Definition Samples  
**Number**: SC27-3676  
**Description**: This document contains sample definitions to help you implement SNA functions in your networks, and includes sample major node definitions.

## z/OS Communications Server: IP Network Print Facility

**Title**: z/OS Communications Server: IP Network Print Facility  
**Number**: SC27-3658  
**Description**: This document is for systems programmers and network administrators who need to prepare their network to route SNA, JES2, or JES3 printer output to remote printers using TCP/IP Services.

## Operation

### Title: z/OS Communications Server: IP User’s Guide and Commands

**Number**: SC27-3662  
**Description**: This document describes how to use TCP/IP applications. It contains requests with which a user can log on to a remote host using Telnet, transfer data sets using FTP, send and receive electronic mail, print on remote printers, and authenticate network users.

### Title: z/OS Communications Server: IP System Administrator’s Commands

**Number**: SC27-3661  
**Description**: This document describes the functions and commands helpful in configuring or monitoring your system. It contains system administrator’s commands, such as TSO NETSTAT, PING, TRACERTE and their UNIX counterparts. It also includes TSO and MVS commands commonly used during the IP configuration process.

### Title: z/OS Communications Server: SNA Operation

**Number**: SC27-3673  
**Description**: This document serves as a reference for programmers and operators requiring detailed information about specific operator commands.

### Title: z/OS Communications Server: Quick Reference

**Number**: SC27-3665  
**Description**: This document contains essential information about SNA and IP commands.
### Customization

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
</table>
| z/OS Communications Server: SNA Customization | SC27-3666 | This document enables you to customize SNA, and includes the following information:  
  - Communication network management (CNM) routing table  
  - Logon-interpret routine requirements  
  - Logon manager installation-wide exit routine for the CLU search exit  
  - TSO/SNA installation-wide exit routines  
  - SNA installation-wide exit routines |

### Writing application programs

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Sockets Application Programming Interface Guide and Reference</td>
<td>SC27-3660</td>
<td>This document describes the syntax and semantics of program source code necessary to write your own application programming interface (API) into TCP/IP. You can use this interface as the communication base for writing your own client or server application. You can also use this document to adapt your existing applications to communicate with each other using sockets over TCP/IP.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP CICS Sockets Guide</td>
<td>SC27-3649</td>
<td>This document is for programmers who want to set up, write application programs for, and diagnose problems with the socket interface for CICS® using z/OS TCP/IP.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP IMS Sockets Guide</td>
<td>SC27-3653</td>
<td>This document is for programmers who want application programs that use the IMS™ TCP/IP application development services provided by the TCP/IP Services of IBM.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Programmer’s Guide and Reference</td>
<td>SC27-3659</td>
<td>This document describes the syntax and semantics of a set of high-level application functions that you can use to program your own applications in a TCP/IP environment. These functions provide support for application facilities, such as user authentication, distributed databases, distributed processing, network management, and device sharing. Familiarity with the z/OS operating system, TCP/IP protocols, and IBM Time Sharing Option (TSO) is recommended.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programming</td>
<td>SC27-3674</td>
<td>This document describes how to use SNA macroinstructions to send data to and receive data from (1) a terminal in either the same or a different domain, or (2) another application program in either the same or a different domain.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programmer’s LU 6.2 Guide</td>
<td>SC27-3669</td>
<td>This document describes how to use the SNA LU 6.2 application programming interface for host application programs. This document applies to programs that use only LU 6.2 sessions or that use LU 6.2 sessions along with other session types. (Only LU 6.2 sessions are covered in this document.)</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Programmer’s LU 6.2 Reference</td>
<td>SC27-3670</td>
<td>This document provides reference material for the SNA LU 6.2 programming interface for host application programs.</td>
</tr>
<tr>
<td>z/OS Communications Server: CSM Guide</td>
<td>SC27-3647</td>
<td>This document describes how applications use the communications storage manager.</td>
</tr>
<tr>
<td>Title</td>
<td>Number</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>

## Diagnosis

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: IP Diagnosis Guide</td>
<td>GC27-3652</td>
<td>This document explains how to diagnose TCP/IP problems and how to determine whether a specific problem is in the TCP/IP product code. It explains how to gather information for and describe problems to the IBM Software Support Center.</td>
</tr>
<tr>
<td>z/OS Communications Server: ACF/TAP Trace Analysis Handbook</td>
<td>GC27-3645</td>
<td>This document explains how to gather the trace data that is collected and stored in the host processor. It also explains how to use the Advanced Communications Function/Trace Analysis Program (ACF/TAP) service aid to produce reports for analyzing the trace data information.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Diagnosis Vol 1, Techniques and Procedures and SNA Diagnosis Vol 2, FFST Dumps and the VIT</td>
<td>GC27-3667 GC27-3668</td>
<td>These documents help you identify an SNA problem, classify it, and collect information about it before you call the IBM Support Center. The information collected includes traces, dumps, and other problem documentation.</td>
</tr>
<tr>
<td>z/OS Communications Server: SNA Data Areas Volume 1 and z/OS Communications Server: SNA Data Areas Volume 2</td>
<td>GC31-6852 GC31-6853</td>
<td>These documents describe SNA data areas and can be used to read an SNA dump. They are intended for IBM programming service representatives and customer personnel who are diagnosing problems with SNA.</td>
</tr>
</tbody>
</table>

## Messages and codes

<table>
<thead>
<tr>
<th>Title</th>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Communications Server: SNA Messages</td>
<td>SC27-3671</td>
<td>This document describes the ELM, IKT, IST, IUT, IVT, and USS messages. Other information in this document includes: • Command and RU types in SNA messages • Node and ID types in SNA messages • Supplemental message-related information</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 1 (EZA)</td>
<td>SC27-3654</td>
<td>This volume contains TCP/IP messages beginning with EZA.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 2 (EZB, EZD)</td>
<td>SC27-3655</td>
<td>This volume contains TCP/IP messages beginning with EZB or EZD.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 3 (EZY)</td>
<td>SC27-3656</td>
<td>This volume contains TCP/IP messages beginning with EZY.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP Messages Volume 4 (EZZ, SNM)</td>
<td>SC27-3657</td>
<td>This volume contains TCP/IP messages beginning with EZZ and SNM.</td>
</tr>
<tr>
<td>z/OS Communications Server: IP and SNA Codes</td>
<td>SC27-3648</td>
<td>This document describes codes and other information that appear in z/OS Communications Server messages.</td>
</tr>
</tbody>
</table>
Index

A

AAREA keyword
  accepting a session 150
  allocation request for specific session 24
  allocation request, conditional, without wait 60
  allocation requests for contention-winner session 36
  allocation requests that can be queued 12
  any-mode RECEIVES 351
  as pointer to RPL extension 399
  CNOS requests 156
  conditional deallocation 103
  conditional deallocation that includes data 111
  deallocation due to program errors 73
  deallocation due to service errors 80
  deallocation due to timer errors 88
  deallocation due to user-specified criteria 96
  defining LU-mode table values 170
  displaying LU-mode table values 176
  entering receive state conditionally 246
  entering receive state conditionally and including data 255
  entering receive state unconditionally 278
  entering receive state unconditionally and including data 267
  flushing the buffer 491
  immediate allocation request 49
  indicating the beginning of a synchronization exchange 545
  indicating the end of a synchronization exchange 550
  obtain status on information available on a specified conversation 573
  obtain status on information from any active conversation 555
  obtain status on information immediately available from any active conversation 561
  obtain status on information immediately available on a specified conversation 566
  queued deallocation due to program, service, timer, or user errors 143
  receive normal information immediately available from any conversation 363
  receive normal information immediately available on a specific conversation 375
  receiving an FMH-5 332
  receiving expedited data from the specified conversation 310
  receiving expedited data immediately available in any-mode 293
  receiving expedited data immediately from the specified conversation 302
  receiving expedited data in any-mode 285
  rejecting a session 165
  releasing a suspended session 535
  replying to a confirmation request 434
  requesting permission to enter send state 499
  resetting the continuation mode 417
  restoring a mode 183
  sending an error notification 478
  sending data 441
  sending data and flushing the buffer 465
  sending data conditionally 425, 453

AAREA keyword (continued)
  sending expedited data on a full-duplex session 507
  specific-mode RECEIVES 387
  suspending a subsequent session 540
  terminating a session 399
  terminating a session due to errors 411
  terminating session due to protocol error or cleanup deactivation 405
  unconditional deallocation 134
  unconditional deallocation that includes data 123

abnormal conversation deallocation
  LU Services errors (ABNDSERV) 78, 139
  reference material (macroinstruction syntax and operands) 70, 93
  timing errors (ABNDTIME) 85, 139
  transaction program errors (ABNDPROG) 70, 139
  user-defined errors (ABNDUSER) 93, 139

ACB keyword
  accepting a session 150
  allocation request for contention-winner session 37
  allocation request for specific session 24
  allocation request, conditional, without wait 60
  allocation requests that can be queued 12
  any-mode RECEIVES 351
  CNOS requests 156
  conditional deallocation 103
  conditional deallocation that includes data 112
  deallocation due to program errors 73
  deallocation due to service errors 80
  deallocation due to timer errors 88
  deallocation due to user-specified criteria 96
  defining LU-mode table values 170
  displaying LU-mode table values 176
  entering receive state conditionally 246
  entering receive state conditionally and including data 255
  entering receive state unconditionally 279
  entering receive state unconditionally and including data 267
  flushing the buffer 491
  immediate allocation request 49
  indicating the beginning of a synchronization exchange 546
  indicating the end of a synchronization exchange 550
  obtain status on information available on a specified conversation 573
  obtain status on information from any active conversation 555
  obtain status on information immediately available from any active conversation 561
  obtain status on information immediately available on a specified conversation 566
  queued deallocation due to program, service, timer, or user errors 143
  receive normal information immediately available from any conversation 363
  receive normal information immediately available on a specific conversation 375
  receiving an FMH-5 332
  receiving expedited data from the specified conversation 310

© Copyright IBM Corp. 2000, 2013
ACB keyword (continued)

receiving expedited data immediately available in any-mode 294
receiving expedited data immediately from the specified conversation 302
receiving expedited data in any-mode 286
rejecting a session 165
releasing a suspended session 535
replying to a confirmation request 434
requesting permission to enter send state 499
resetting the continuation mode 418
restoring a mode 183
sending an error notification 478
sending data 441
sending data and flushing the buffer 465
sending data conditionally 425, 453
sending expedited data on a full-duplex session 507
specific-mode RECEIVES 387
suspending a subsequent conversation 540
terminating a session 400
terminating a session due to errors 412
terminating a session due to protocol errors or cleanup deactivation 406
unconditional deallocation 134
unconditional deallocation that includes data 123
accepting a session macroinstruction 148
accepting a session, macroinstruction 148
access method control block (ACB) keyword
accepting a session 150
allocation request for contention-winner session 37
allocation request for specific session 24
allocation request, conditional, without wait 60
allocation requests that can be queued 12
any-mode RECEIVES 351
CNOS requests 156
conditional deallocation 103
conditional deallocation that includes data 112
deallocation due to program errors 73
deallocation due to service errors 80
deallocation due to timer errors 88
deallocation due to user-specified criteria 96
defining LU-mode table values 170
displaying LU-mode table values 176
entering receive state conditionally 246
entering receive state conditionally and including data 255
entering receive state unconditionally 279
entering receive state unconditionally and including data 267
flushing the buffer 491
immediate allocation request 49
indicating the beginning of a synchronization exchange 546
indicating the end of a synchronization exchange 550
obtain status on information available on a specified conversation 573
obtain status on information from any active conversation 555
obtain status on information immediately available from any active conversation 561
obtain status on information immediately available on a specified conversation 566
queued deallocation due to program, service, timer, or user errors 143
receive normal information immediately available from any conversation 363

access method control block (ACB) keyword (continued)
receive normal information immediately available on a specific conversation 375
receiving an FMH-5 332
receiving expedited data from the specified conversation 310
receiving expedited data immediately available in any-mode 294
receiving expedited data immediately from the specified conversation 302
receiving expedited data in any-mode 286
rejecting a session 165
releasing a suspended session 535
replying to a confirmation request 434
requesting permission to enter send state 499
resetting the continuation mode 418
restoring a mode 183
sending an error notification 478
sending data 441
sending data and flushing the buffer 465
sending data conditionally 425, 453
sending expedited data on a full-duplex session 507
specific-mode RECEIVES 387
suspending a subsequent conversation 540
terminating a session 400
terminating a session due to errors 412
terminating a session due to protocol errors or cleanup deactivation 406
unconditional deallocation 134
unconditional deallocation that includes data 123
accessibility 687
activating a session, macroinstruction for 148
ACTSESS qualify value 148
allocating a conversation
APPCCMD CONTROL=ALLOC macroinstruction
macroinstruction syntax and operands 9, 46
session assignment algorithm conditional request, without wait 57
immediate requests 46
request for specific session 22
requests for contention-winner sessions 33
requests that can be queued 9
ALLOCQ qualify value
session assignment algorithm 9
syntax and operands 9
any-mode RECEIVES, macroinstruction 348
APPCCMD macroinstruction
coding comments 3
continuation lines 3
AREA keyword,
as pointer to
DEFINE/DISPLAY control block 176
session limits control block 157
returned value, used with
obtain status on information available on a specified conversation 573
obtain status on information from any active conversation 555
obtain status on information immediately available from any active conversation 561
obtain status on information immediately available on a specified conversation 566
obtain status on information immediately available on any-mode 286
used with
accepting a session 151
allocation request for contention-winner session 37
allocation request for specific session 25
allocation request that can be queued 12
AREA keyword, (continued)  
used with (continued)  
allocation request, conditional, without wait 60  
any-mode RECEIVES 352  
CNOS requests 157  
conditional deallocation that includes data 112  
deallocation due to program errors 73  
deallocation due to service errors 81  
deallocation due to timer errors 88  
deallocation due to user-specified criteria 96  
defining LU-mode table values 170  
displaying LU-mode table values 176  
entering receive state conditionally and including data 256  
entering receive state unconditionally and including data 268  
immediate allocation request 49  
queued deallocation due to program, service, timer, or user errors 143  
receive normal information immediately available from any conversation 363  
receive normal information immediately available on a specific conversation 375  
receiving an FMH-5 332  
receiving expedited data from the specified conversation 311  
receiving expedited data immediately available in any-mode 294  
receiving expedited data immediately from the specified conversation 302  
receiving expedited data in any-mode 286  
restoring a mode 184  
sending an error notification 478  
sending data 442  
sending data and flushing the buffer 466  
sending data conditionally 454  
sending expedited data on a full-duplex session 508  
specific-mode RECEIVES 388  
unconditional deallocation that includes data 124  

AREALEN keyword, used with  
any-mode RECEIVES 352  
displaying LU-mode table values 176  
obtain status on information available on a specified conversation 574  
obtain status on information from any active conversation 556  
obtain status on information immediately available from any active conversation 561  
obtain status on information immediately available on a specified conversation 567  
receive normal information immediately available from any conversation 363  
receive normal information immediately available on a specific conversation 375  
receiving an FMH-5 332  
receiving expedited data from the specified conversation 311  
receiving expedited data immediately available in any-mode 294  
receiving expedited data immediately from the specified conversation 302  
receiving expedited data in any-mode 286  
restoring a mode 184  
specific-mode RECEIVES 388  

ARG keyword  
relationship to CID 165  
used with accepting a session 151  
used with rejecting a session 165  

AVFA RPL field  
returned value, used with  
allocation request for contention-winner session 41  
allocation request for specific session 29  
allocation request that can be queued 17  
allocation request, conditional, without wait 64  
CNOS request 160  
immediate allocation request 53  

B  
becoming receiving LU 244, 276  
becoming sender  
requesting permission to enter send state macroinstruction 496  

BIND image  
DSECT 633  
relationship to RESTORE 181  

BINDAREA DSECT  
profile 1 636  
profile 2 639  
profile 3 640  
profile 4 640  
profile 6 642  

BRANCH keyword, used with  
accepting a session 151  
allocation request for contention-winner session 37  
allocation request for specific session 25  
allocation request that can be queued 12  
allocation request, conditional, without wait 60  
any-mode RECEIVES 352  
CNOS requests 157  
conditional deallocation 103  
conditional deallocation that includes data 112  
deallocation due to program errors 74  
deallocation due to service errors 81  
deallocation due to timer errors 88  
deallocation due to user-specified criteria 96  
defining LU-mode table values 170  
displaying LU-mode table values 177  
indicating the beginning of a synchronization exchange 546  
indicating the end of a synchronization exchange 550  
obtain status on information available on a specified conversation 574  
obtain status on information from any active conversation 556  
obtain status on information immediately available from any active conversation 561  
obtain status on information immediately available on a specified conversation 567  
queued deallocation due to program, service, timer, or user errors 144  
flushing the buffer 491  
immediate allocation request 49  

Index 707
BRANCH keyword, used with (continued) receive normal information immediately available on a specific conversation 375 receiving an FMH-5 332 receiving expedited data from the specified conversation 311 receiving expedited data immediately available in any-mode 294 receiving expedited data immediately from the specified conversation 302 receiving expedited data in any-mode 286 rejecting a session 165 releasing a suspended session 535 replying to a confirmation request 434 requesting permission to enter send state 499 resetting the continuation mode 418 restoring a mode 184 sending an error notification 478 sending data 442 sending data and flushing the buffer 466 sending data conditionally 425, 454 sending expedited data on a full-duplex session 508 specific-mode RECEIVES 388 terminating a session 400 terminating a session due to errors 412 terminating a session due to protocol errors or cleanup deactivation 406 unconditional deallocation 135 unconditional deallocation that includes data 124 used with suspending a subsequent conversation 541

CONFIRM qualify value (continued) PREPRCV CONTROL value 244 SEND CONTROL value 422 confirmation request replying to a confirmation request macroinstruction, used with 432 sending an error notification macroinstruction, used with 475 CONFRMD qualify value 432 CONFTXT keyword, used with accepting a session 151 CONMODE keyword, used with allocation request for contention-winner session 37 allocation request for specific session 25 allocation request that can be queued 13 allocation request, conditional, without wait 60 any-mode RECEIVES 353 conditional deallocation 103 conditional deallocation that includes data 113, 124, 135 entering receive state conditionally 247 entering receive state conditionally and including data 256 entering receive state unconditionally 279 entering receive state unconditionally and including data 268 flushing the buffer 492 immediate allocation request 49 ISTRPL6 584 receive normal information immediately available from any conversation 364 receive normal information immediately available on a specific conversation 376 receiving expedited data from the specified conversation 311 receiving expedited data immediately available in any-mode 294 receiving expedited data immediately from the specified conversation 302 receiving expedited data in any-mode 286 replying to a confirmation request 435 requesting permission to enter send state 500 resetting the continuation mode 418 sending an error notification 479 sending data 442 sending data and flushing the buffer 466 sending data conditionally 426, 455 sending expedited data on a full-duplex session 508 specific-mode RECEIVES 389 CONSTATE RPL field returned value, used with allocation request for contention-winner session 42 allocation request for specific session 29 allocation request that can be queued 13 allocation request, conditional, without wait 65 any-mode RECEIVES 355 conditional deallocation 105 conditional deallocation that includes data 116 deallocation due to program errors 75 deallocation due to service errors 82 deallocation due to timer errors 90 deallocation due to user-specified criteria 98 entering receive state conditionally 250 entering receive state conditionally and including data 260 entering receive state unconditionally 281 entering receive state unconditionally and including data 272

canceling APPCCMD macroinstructions abnormal deallocation (syntax and operands) 70, 93 terminating a session (syntax and operands) 409 CD keyword, used with any-mode RECEIVES 352 ISTRPL6 583 receive normal information immediately available from any conversation 364 receive normal information immediately available on a specific conversation 376 specific-mode RECEIVES 388 CGID keyword, used with allocation request for specific session 25 terminating a session due to protocol errors or cleanup deactivation 406 CGID RPL field returned value, used with allocation request for contention-winner session 41 allocation request that can be queued 17 allocation request, conditional, without wait 65 immediate allocation request 53 receiving an FMH-5 335 change number of sessions (CNOS) macroinstruction 154 changing continuation modes use of RESETRCV macroinstruction variation 415 CHECK control value 69 CHECK macroinstruction 69 CNOS (change number of sessions) macroinstruction 154 CNOS session limits data structure—ISTSLCNS DSECT 653 coding comments 3, 4 Communications Server for z/OS, online information xv CONFIRM qualify value DEALLOC CONTROL value 101

708 z/OS V2R1.0 Communications Server: SNA Programmer’s LU 6.2 Reference
CONSTATE RPL field (continued)
returned value, used with (continued)
flushing the buffer 494
immediate allocation request 53
obtain status on information available on a specified
conversation 575
obtain status on information immediately available on a
specified conversation 568
queued deallocation due to program, service, timer, or
user errors 145
receive normal information immediately available from
any conversation 366
receive normal information immediately available on a
specific conversation 379
receiving an FMH-5 335
receiving expedited data from the specified
conversation 313
receiving expedited data immediately available in
any-mode 296
receiving expedited data immediately from the specified
conversation 304
receiving expedited data in any-mode 288
relying to confirmation request 437
requesting permission to enter send state 502
resetting the continuation mode 420
sending an error notification 482
sending data 446
sending data and flushing the buffer 470
sending data conditionally 428, 458
sending expedited data on full-duplex session 511
specific-mode RECEIVES 391
terminating a session 401
unconditional deallocation 137
unconditional deallocation that includes data 128
contention winner
requests for allocation 33
continuation lines 3, 4
CONTROL keyword
ALLOC value
ALLOCD QUALIFY value 9
CONVGRP QUALIFY value 21
CONWIN QUALIFY value 33
IMMED QUALIFY value 46
WHENFREE QUALIFY value 57, 232
CHECK value 69
DEALLOC value
ABNDPROG QUALIFY value 70
ABNDSEV QUALIFY value 78
ABNDTIME QUALIFY value 85
ABNDUSER QUALIFY value 93
CONFIRM QUALIFY value 101
DATACON QUALIFY value 109
DATAFLU QUALIFY value 121
FLUSH QUALIFY value 132
DEALLOCQ value
ABNDPROG QUALIFY value 139
ABNDSEV QUALIFY value 139
ABNDTIME QUALIFY value 139
ABNDUSER QUALIFY value 139
OPRCNTL value
ACTSESS QUALIFY value 148
CNOS QUALIFY value 154
DACTSESS QUALIFY value 163
DEFINE QUALIFY value 168
DISPLAY QUALIFY value 174
RESTORE QUALIFY value 181
CONTROL keyword (continued)
PREPRCV value
CONFIRM QUALIFY value 244
DATACON QUALIFY value 253
DATAFLU QUALIFY value 265
FLUSH QUALIFY value 276
RCVEXPD value
ANY QUALIFY value 283
IANY QUALIFY value 291
ISPEC QUALIFY value 299
SPEC QUALIFY value 307
RCVFHMS value 328
RECEIVE value
ANY QUALIFY value 348
IANY QUALIFY value 360
ISPEC QUALIFY value 372
SPEC QUALIFY value 384
REJECT value
CONV QUALIFY value 397
CONVGRP QUALIFY value 403
SESSION QUALIFY value 409
RESETRCV value 415
SEND value
CONFIRM QUALIFY value 422
CONFMD QUALIFY value 432
DATA QUALIFY value 439
DATACON QUALIFY value 451
DATAFLU QUALIFY value 463
ERROR QUALIFY value 475
FLUSH QUALIFY value 489
RQSEND QUALIFY value 496
SENDXPD value 504
SETSESS value
RESUME QUALIFY value 533
SUSPEND QUALIFY value 538
SYNCBEG QUALIFY value 543
SYNCEND QUALIFY value 548
TESTSTAT value
ALL QUALIFY value 553
IALL QUALIFY value 558
ISPEC QUALIFY value 564
SPEC QUALIFY value 571
conversation allocation
APPCCMD CONTROL=ALLOC macroinstruction
macroinstruction syntax and operands 9, 46
session assignment algorithm
conditional request, without wait 57, 232
immediate requests 46
request for specific session 21
requests for contention-winner sessions 33
requests that can be queued 9
conversation data, receiving
APPCCMD macroinstructions used 348, 384
conversation data, sending
APPCCMD macroinstructions used
syntax and operands 422, 496
buffer list (OPTCD=BUFFLST) considerations, used with
conditional deallocation that includes data 114
entering receive state conditionally and including
data 259
sending data 115, 445
sending data and flushing the buffer 259, 271, 469
sending data conditionally 457
sending expedited data on a full-duplex session 510
unconditional deallocation that includes data 127
flushing the buffer, used with
conditional deallocation 101
Index 709
conversation data, sending (continued)
flushing the buffer, used with (continued)
conditional deallocation that includes data 109
entering receive state conditionally 244
entering receive state conditionally and including data 253
entering receive state unconditionally 276
entering receive state unconditionally and including data 265
flushing the buffer 489
sending data and flushing the buffer 463
sending data conditionally 422, 451
unconditional deallocation 132
unconditional deallocation that includes data 121
macroinstructions for sending new data, used with conditional deallocation that includes data 109
entering receive state conditionally and including data 265
sending data 439
sending data and flushing the buffer 463
sending data conditionally 451
unconditional deallocation that includes data 121
sending expedited data on a full-duplex session 504
conversation deallocation
APPCCMD CONTROL=DEALLOC
macroinstruction syntax and operands 70, 132
APPCCMD CONTROL=DEALLOCQ
macroinstruction syntax and operands 139
use of macroinstruction 139
conversation identifier (continued)
keyword, used with (continued)
unconditional deallocation 135
unconditional deallocation that includes data 125
returned value, used with
allocation request for contention-winner session 42
allocation request for specific session 29
allocation request that can be queued 17
allocation request, conditional, without wait 65
any-mode RECEIVEs 356
immediate allocation request 53
receive normal information immediately available from any conversation 367
receiving an FMH-5 335
receiving expedited data in any-mode 289
conversation security
conversation level 642
CONVGRP quality value
session assignment algorithm 22
CONVID key word, used with conditional deallocation 104
conditional deallocation that includes data 113
deallocation due to program errors 74
deallocation due to service errors 81
deallocation due to timer errors 89
deallocation due to user-specified criteria 96
entering receive state conditionally 247
entering receive state conditionally and including data 257
entering receive state unconditionally 280
flushing the buffer 492
indicating the beginning of a synchronization exchange 546
obtain status on information available on a specified conversation 574
obtain status on information immediately available on a specified conversation 567
queued deallocation due to program, service, timer, or user errors 144
receive normal information immediately available on a specific conversation 377
replying to a confirmation request 435
requesting permission to enter send state 500
resetting the continuation mode 419
sending an error notification 479
sending data 443
sending data and flushing the buffer 467
sending data conditionally 426, 455
sending expedited data on a full-duplex session 509
specific-mode RECEIVEs 389
suspending a subsequent conversation 541
terminating a session 400
returned value, used with
allocation request for specific session 29
allocation request that can be queued 17
allocation request, conditional, without wait 65
any-mode RECEIVEs 356
immediate allocation request 53
receive normal information immediately available from any conversation 367
receiving an FMH-5 335
CONVID returned value, used with receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 297
receiving expedited data immediately from the specified conversation 303
receiving expedited data in any-mode 289

CONVSECP RPL field returned value, used with allocation request for specific session 29
allocation request, conditional, without wait 65
allocation requests for contention-winner session 42
allocation requests that can be queued 17
immediate allocation requests 54

CONWIN qualify value session assignment algorithm 33

CONXMOD keyword, used with allocation request for contention-winner session 38
allocation request for specific session 26
allocation request that can be queued 13
allocation request, conditional, without wait 61
any-mode RECEIVES 353
conditional deallocation 104
conditional deallocation that includes data 113, 125, 136
entering receive state conditionally 248
entering receive state conditionally and including data 257
entering receive state unconditionally 280
entering receive state unconditionally and including data 269
flushing the buffer 492
immediate allocation request 50
ISTRPL6 584
receives normal information immediately available from any conversation 365
receives normal information immediately available on a specified conversation 377
receiving an FMH-5 333
receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 295
receiving expedited data immediately from the specified conversation 303
receiving expedited data in any-mode 287
replying to a confirmation request 435
requesting permission to enter send state 500
resetting the continuation mode 419
sending data 443
sending data and flushing the buffer 467
sending data conditionally 426, 455
sending expedited data on full-duplex session 509
specific-mode RECEIVES 389
cross-memory mode, restrictions 69

CRYPT keyword, used with conditional deallocation that includes data 114
entering receive state 257
entering receive state unconditionally 269
sending data 443
sending data and flushing the buffer 467
sending data conditionally 456
unconditional deallocation that includes data 125

CRYPTLVL RPL field returned value, used with allocation request for contention-winner session 42
allocation request for specific session 29
allocation request that can be queued 18
allocation request, conditional, without wait 65
immediate allocation request 54
receiving an FMH-5 335

D DACTSESS qualify value used with terminating a session 163
DATA qualify value 439, 504
data truncation
sending an error notification macroinstruction 475
data, receiving
APPCCMD macroinstructions used 348, 384
data, sending
APPCCMD macroinstructions used syntax and operands 422, 496
buffer list (OPTCD=BUFFLST) considerations
conditional deallocation that includes data 114
entering receive state conditionally and including data 259
sending data 115, 445
sending data and flushing the buffer 259, 271, 469
sending data conditionally 457
sending expedited data on a full-duplex session 510
unconditional deallocation that includes data 127
flushing the buffer, used with conditional deallocation 101
conditional deallocation that includes data 109
entering receive state conditionally and including data 253
entering receive state unconditionally 276
entering receive state unconditionally and including data 265
flushing the buffer 489
sending data and flushing the buffer 463
sending data conditionally 422, 451
unconditional deallocation 132
unconditional deallocation that includes data 121
macroinstructions for sending new data, used with conditional deallocation that includes data 109
entering receive state conditionally and including data 253
entering receive state unconditionally and including data 265
sending data 439
sending data and flushing the buffer 463
sending data conditionally 451
unconditional deallocation that includes data 121
sending expedited data on a full-duplex session 504

DATACON qualify value
DEALLOC CONTROL value 109
PREPRCV CONTROL value 233
SEND CONTROL value 451

DATAFLU qualify value
DEALLOC CONTROL value 121
PREPRCV CONTROL value 265
SEND CONTROL value 463
dequitting sessions unused with terminating a session macroinstruction 163
deactivation type code (DEACTYP)
keyword, used with
terminating session due to error 412
terminating session due to protocol error or cleanup
deactivation 406
DEACTYP (deactivation type code)
keyword, used with
terminating session due to error 412
terminating session due to protocol error or cleanup
deactivation 406
deallocating a conversation
APPCCMD CONTROL=DEALLOC
macroinstruction syntax and operands 70, 93
APPCCMD CONTROL=DEALLOCQ
macroinstruction syntax and operands 139
use of macroinstruction 139
deallocating conversations
macroinstruction forms of CONTROL=DEALLOC 70, 132
deallocation, abnormal
LU Services errors (ABNDSERV) 78, 139
reference material (macroinstruction syntax and operands) 70, 93
timing errors (ABNDTIME) 85, 139
transaction program errors (ABNDBROG) 70, 139
user-defined errors (ABNDUSER) 93, 139
DEFINE qualify value
syntax and operands 168
defining LU-mode table values 168
design considerations for LU 6.2
restoring a mode 181
disability 687
DISPLAY qualify value
syntax and operands 174
displaying LU-mode data 174
DNS, online information xvii
DSECT
ISTDBIND 633
ISTFMS 643
ISTRPL6X 645
ISTSLCNS 653
ISTSILD 654
ISTSREST 656
ISTSTATD 656
ISTUSFBC 657
E
ECB (event control block) (continued)
keyword, used with (continued)
entering receive state unconditionally 280
entering receive state unconditionally and including data 270
flushing the buffer 493
immediate allocation request 50
indicating the beginning of a synchronization exchange 546
indicating the end of a synchronization exchange 551
obtain status on information available on a specified conversation 574
obtain status on information from any active conversation 556
obtain status on information immediately available from any active conversation 562
obtain status on information immediately available on a specified conversation 567
queued deallocation due to program, service, timer, or user errors 144
receive normal information immediately available from any conversation 365
receive normal information immediately available on a specific conversation 377
receiving an FMH-5 333
receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 295
receiving expedited data immediately from the specified conversation 303
receiving expedited data in any-mode 287
rejecting a session 165
resetting the continuation mode 419
sending an error notification 480
sending data 444
sending data and flushing the buffer 468
sending expedited data conditionally 427, 456
sending expedited data on a full-duplex session 509
specific-mode RECEIVES 390
suspension of a subsequent conversation 541
terminating a session 400
terminating a session due to errors 412
terminating a session due to protocol errors or cleanup deactivation 406
unconditional deallocation 136
unconditional deallocation that includes data 126
entering RECEIVE state
PREPRCV macroinstructions (syntax and operands) 244, 276
entering SEND state
requesting permission to enter send state
macroinstruction 496
ERROR qualify value 475
event control block (ECB) keyword, used with
accepting a session 151
allocation request for contention-winner session 38
allocation request for specific session 26
allocation request that can be queued 13
allocation request, conditional, without wait 61
any-mode RECEIVES 354
CNOS requests 157
conditional deallocation 104
conditional deallocation that includes data 114
deallocation due to program errors 74
deallocation due to service errors 81
deallocation due to timer errors 89
deallocating due to user-defined criteria 96
defining LU-mode table values 171
displaying LU-mode table values 177
entering receive state conditionally 248
entering receive state conditionally and including data 258
event control block (ECB) (continued)
keyword, used with (continued)
conditional deallocation 104
conditional deallocation that includes data 114
deallocation due to program errors 74
deallocation due to service errors 81
deallocation due to timer errors 89
deallocation due to user-specified criteria 96
defining LU-mode table values 171
displaying LU-mode table values 177
entering receive state conditionally 248
entering receive state conditionally and including data 258
entering receive state unconditionally 280
entering receive state unconditionally and including data 270
flushing the buffer 493
immediate allocation request 50
indicating the beginning of a synchronization exchange 546
indicating the end of a synchronization exchange 551
obtain status on information available on a specified conversation 574
obtain status on information from any active conversation 556
obtain status on information immediately available from any active conversation 562
obtain status on information immediately available on a specified conversation 567
queued deallocation due to program, service, timer, or user errors 144
receive normal information immediately available from any conversation 365
receive normal information immediately available from any conversation 365
receive normal information immediately available on a specific conversation 377
receiving an FMH-5 333
receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 295
receiving expedited data immediately available in any-mode 295
receiving expedited data immediately available in any-mode 295
receiving expedited data in any-mode 287
rejecting a session 165
releasing a suspended session 536
replying to a confirmation request 436
requesting permission to enter send state 501
resetting the continuation mode 419
restoring a mode 184
sending an error notification 480
sending data 444
sending data and flushing the buffer 468
sending data conditionally 427, 456
sending expedited data on a full-duplex session 509
specific-mode RECEIVES 390
suspending a subsequent conversation 541
terminating a session 400
terminating a session due to errors 412
terminating a session due to protocol errors or cleanup deactivation 406
unconditional deallocation 136
unconditional deallocation that includes data 126

EXIT keyword, used with (continued)
allocation request, conditional, without wait 61
any-mode RECEIVES 354
CNOS requests 157
conditional deallocation 105
conditional deallocation that includes data 114
deallocation due to program errors 74
deallocation due to service errors 82
deallocation due to timer errors 89
deallocation due to user-specified criteria 97
defining LU-mode table values 171
displaying LU-mode table values 177
entering receive state conditionally 248
entering receive state conditionally and including data 258
entering receive state unconditionally 280
entering receive state unconditionally and including data 270
flushing the buffer 493
immediate allocation request 51
indicating the beginning of a synchronization exchange 546
indicating the end of a synchronization exchange 551
queued deallocation due to program, service, timer, or user errors 287
receive normal information immediately available from any conversation 365
receive normal information immediately available from any conversation 365
receive normal information immediately available on a specific conversation 377
receiving an FMH-5 333
receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 295
rejecting a session 166
releasing a suspended session 536
replying to a confirmation request 436
requesting permission to enter send state 501
resetting the continuation mode 419
restoring a mode 184
sending an error notification 480
sending data 444
sending data and flushing the buffer 468
sending data conditionally 427, 456
specific-mode RECEIVES 390
suspending a subsequent conversation 541
terminating a session 400
terminating a session due to errors 412
terminating a session due to protocol errors or cleanup deactivation 407
unconditional deallocation 136
unconditional deallocation that includes data 126

EXPDLEN RPL field
returned value, used with
any-mode RECEIVES 356
conditional deallocation 106
conditional deallocation that includes data 116
deallocation due to program errors 76
deallocation due to service errors 83
deallocation due to timer errors 90
deallocation due to user-specified criteria 98
entering receive state conditionally and including data 260
entering receive state unconditionally and including data 272
receive normal information immediately available from any conversation 367
EXPDLEN RPL field (continued)
returned value, used with (continued)
   receive normal information immediately available on a
specific conversation 379
   receiving expedited data from the specified
conversation 314
   receiving expedited data immediately available in
any-mode 297
   receiving expedited data immediately from the specified
conversation 305
   receiving expedited data in any-mode 289
   sending an error notification 482
   sending data 447
   sending data and flushing the buffer 470
   sending data conditionally 428, 458
   sending expedited data on full-duplex session 511
specific-mode RECEIVES 392
unconditional deallocation 137
unconditional deallocation that includes data 128

EXPDRCV RPL field
returned value, used with
   any-mode RECEIVES 356
   conditional deallocation 106
   conditional deallocation that includes data 116
   deallocation due to program errors 76
   deallocation due to service errors 83
   deallocation due to timer errors 91
   deallocation due to user-specified criteria 98
   entering receive state conditionally and including
data 261
   entering receive state unconditionally and including
data 272
   receive normal information immediately available from
any conversation 367
   receive normal information immediately available on a
specific conversation 379
   receiving expedited data from the specified
conversation 314
   receiving expedited data immediately available in
any-mode 297
   receiving expedited data immediately from the specified
conversation 305
   receiving expedited data in any-mode 289
   sending an error notification 482
   sending data 447
   sending data and flushing the buffer 470
   sending data conditionally 428, 458
   sending expedited data on full-duplex session 511
specific-mode RECEIVES 392
unconditional deallocation 137
unconditional deallocation that includes data 128

F

FDB2 RPL field (continued)
returned value, used with (continued)
deallocation due to timer errors 91
deallocation due to user-specified criteria 98
defining LU-mode table values 173
displaying LU-mode table values 179
entering receive state conditionally 250
entering receive state conditionally and including
data 261
entering receive state unconditionally 281
entering receive state unconditionally and including
data 272
flushing the buffer 495
immediate allocation request 54
indicating the beginning of a synchronization
exchange 547
indicating the end of a synchronization exchange 552
obtain status on information available on a specified
conversation 576
obtain status on information from any active
conversation 557
obtain status on information immediately available on a
specified conversation 569
queued deallocation due to program, service, timer, or
user errors 146
receive normal information immediately available from
any conversation 367
receive normal information immediately available on a
specific conversation 379
receiving an FMH-5 336
receiving expedited data from the specified
conversation 314
receiving expedited data immediately available in
any-mode 297
receiving expedited data immediately from the specified
conversation 305
receiving expedited data in any-mode 289
rejecting a session 167
releasing a suspended session 537
replying to a confirmation request 437
requesting permission to enter send state 502
resetting the continuation mode 421
restoring a mode 186
sending an error notification 482
sending data 447
sending data and flushing the buffer 471
sending data conditionally 428, 458
sending expedited data on full-duplex session 511
specific-mode RECEIVES 392
suspending a subsequent conversation 542
terminating a session 401
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup
deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 128
feedback code data structure—ISTUSFBC DSECT 657
feedback information
request-to-send indicator 496
FILL keyword
receive normal information immediately available on a
specific conversation 377
used with ISTRPL6 585
used with specific-mode RECEIVES 390
FLUSH qualify value
DEALLOC CONTROL value 132
PREPRCV CONTROL value 276

714  z/OS V2R1.0 Communications Server: SNA Programmer’s LU 6.2 Reference
FLUSH qualify value (continued)
SEND CONTROL value 489
flushing the send buffer
flushing the buffer macroinstruction 489
used with entering receive state unconditionally
macroinstruction 276
used with unconditional deallocation
FMH-5
DSECT 643
FMH5LEN RPL field
returned value, used with
allocation request for contention-winner session 43
allocation request for specific session 30
allocation request that can be queued 18
allocation request, conditional, without wait 66
any-mode RECEIVEs 356
conditional deallocation 106
conditional deallocation that includes data 116
deallocation due to program errors 76
deallocation due to service errors 83
deallocation due to timer errors 91
deallocation due to user-specified criteria 98
entering receive state conditionally 250
entering receive state conditionally and including
data 261
entering receive state unconditionally 281
entering receive state unconditionally and including
data 272
flushing the buffer 495
immediate allocation request 54
obtain status on information available on a specified
corverstation 576
obtain status on information from any active
corverstation 557
obtain status on information immediately available from
any active conversation 563
receive normal information immediately available on a
specified conversation 569
queued deallocation due to program, service, timer, or
user errors 146
receive normal information immediately available from
any conversation 367
receive normal information immediately available on a
specific conversation 379
receiving an FMH-5 336
receiving expedited data from the specified
corverstation 314
receiving expedited data immediately available in
any-mode 297
receiving expedited data immediately from the specified
conversation 305
replying to a confirmation request 437
resetting the continuation mode 421
sending an error notification 482
sending data 447
sending data and flushing the buffer 471
sending data conditionally 428, 458
sending expedited data on full-duplex session 512
specific-mode RECEIVEs 392
terminating a session 402
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup
deactivation 408
unconditional deallocation 138
FMH5LEN RPL field (continued)
returned value, used with (continued)
unconditional deallocation that includes data 128
FMH5RCV RPL field
returned value, used with
allocation request for contention-winner session 43
allocation request for specific session 30
allocation request that can be queued 18
allocation request, conditional, without wait 66
any-mode RECEIVEs 356
conditional deallocation 106
conditional deallocation that includes data 116
deallocation due to program errors 76
deallocation due to service errors 83
deallocation due to timer errors 91
deallocation due to user-specified criteria 99
entering receive state conditionally 250
entering receive state conditionally and including
data 261
entering receive state unconditionally 281
entering receive state unconditionally and including
data 272
flushing the buffer 495
immediate allocation request 54
obtain status on information available on a specified
corverstation 576
obtain status on information from any active
corverstation 557
obtain status on information immediately available from
any active conversation 563
receive normal information immediately available on a
specific conversation 380
receiving an FMH-5 336
receiving expedited data from the specified
corverstation 314
receiving expedited data immediately available in
any-mode 297
receiving expedited data immediately from the specified
conversation 305
receiving expedited data in any-mode 289
replying to a confirmation request 437
requesting permission to enter send state 503
resetting the continuation mode 421
sending an error notification 483
sending data 447
sending data and flushing the buffer 471
sending data conditionally 428, 458
sending expedited data on full-duplex session 512
specific-mode RECEIVEs 392
terminating a session 402
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup
deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 129

G

global variables
declaring and setting with ISTGAPPC
macroinstruction 577
global variables (continued)
table of 579

I
IBM Software Support Center, contacting xii
identifier, conversation keyword, used with
conditional deallocation 104
conditional deallocation that includes data 113
deallocation due to program errors 74
deallocation due to service errors 81
deallocation due to timer errors 89
deallocation due to user-specified criteria 96
entering receive state conditionally 247
entering receive state conditionally and including data 257
entering receive state unconditionally 280
entering receive state unconditionally and including data 269
flushing the buffer 492
indicating the beginning of a synchronization exchange 546
obtain status on information available on a specified conversation 574
obtain status on information immediately available on a specified conversation 567
queued deallocation due to program, service, timer, or user errors 144
receive normal information immediately available on a specific conversation 377
replying to a confirmation request 435
requesting permission to enter send state 500
resetting the continuation mode 419
sending an error notification 479
sending data 443
sending data and flushing the buffer 467
sending data conditionally 426, 455
sending expedited data on a full-duplex session 509
specific-mode RECEIVES 389
suspending a subsequent conversation 541
terminating a session 400
unconditional deallocation 135
unconditional deallocation that includes data 125
returned value, used with
allocation request for contention-winner session 42
allocation request for specific session 29
allocation request that can be queued 17
allocation request, conditional, without wait 65
any-mode RECEIVES 356
immediate allocation request 53
receive normal information immediately available from any conversation 367
receiving an FMH-5 335
receiving expedited data in any-mode 289
IMMED qualify value
session assignment algorithm 46
use of 46
Information APARs xiii
Internet, finding z/OS information online xv
ISTDBIND DSECT 633
ISTFM5 DSECT 643
ISTGAPPC macroinstruction 577
ISTRPL6 macroinstruction 580
ISTRPL6X DSECT 645
ISTSLCNS DSECT 653
ISTSLD DSECT 654

K
keyboard 687

L
license, patent, and copyright information 691
limited resource session 643
LIST keyword, used with
ISTRPL6 585
restoring a mode 184
LOCKS keyword, used with
entering receive state conditionally 248
entering receive state conditionally and including data 258
ISTRPL6 585
LOGMODE keyword, used with
 allocation request for contention-winner session 38
allocation request that can be queued 14
allocation request, conditional, without wait 62
CNOS requests 157
defining LU-mode table values 171
displaying LU-mode table values 177
immediate allocation request 51
ISTRPL6 586
restoring a mode 185
returned value, used with
allocation request for specific session 30
receiving an FMH-5 336
LOGRCV RPL field
returned value, used with
any-mode RECEIVES 356
conditional deallocation 106
conditional deallocation that includes data 117
entering receive state conditionally 250
entering receive state conditionally and including data 261
entering receive state unconditionally and including data 273
receive normal information immediately available from any conversation 368
receive normal information immediately available on a specific conversation 380
sending an error notification 483
sending data 447
sending data and flushing the buffer 471
sending data conditionally 429, 459
specific-mode RECEIVES 393
LU 6
restoring a mode 181
LU-mode table
changing information in table 168
querying information in table 174
LUAFFIN keyword, used with
allocation request for a contention-winner session 39
allocation request for a specific session 26
allocation request that can be queued 14
allocation request, conditional, without wait 62, 586
CNOS requests 158

716 z/OS V2R1.0 Communications Server: SNA Programmer’s LU 6.2 Reference
LUAFFIN keyword, used with (continued)
reserving a contention-winner session without establishing conversation 215
reserving a conversation session without establishing conversation 237
reserving a session without establishing conversation 192
with a specified conversation group 203
LUNAME
keyword, used with
allocation request for contention-winner session 39
allocation request that can be queued 14
allocation request, conditional, without wait 62
CNOS requests 158
defining LU-mode table values 172
displaying LU-mode table values 178
immediate allocation request 51
ISTRPL6 586
restoring a mode 185
returned value, used with
allocation request for specific session 31
receiving an FMH-5 336

M
macroinstruction
operands 2
syntax 1
mainframe
education xiii
mode
cross-memory, restrictions 69
restoring 181

N
NAMEUSE
keyword, used with
allocation request for contention-winner session 39
allocation request that can be queued 15
allocation request, conditional, without wait 62
CNOS requests 158
NETID
keyword, used with
allocation request for contention-winner session 40
allocation request that can be queued 15
allocation request, conditional, without wait 63
CNOS requests 159
defining LU-mode tables 172
displaying LU-mode table values 178
immediate allocation request 51
ISTRPL6 586
restoring a mode 185
returned value, used with
allocation request for specific session 31
receiving an FMH-5 336

O
operand format 3
operand notation 3
OPTCD keyword, used with
accepting a session 152
allocation request for contention-winner session 40
allocation request for specific session 27
allocation request that can be queued 15
OPTCD keyword, used with (continued)
allocation request, conditional, without wait 63
any-mode RECEIVES 354
CNOS requests 159
conditional deallocation 105
conditional deallocation that includes data 114
deallocation due to program errors 74
deallocation due to service errors 82
deallocation due to timer errors 89
deallocation due to user-specified criteria 97
defining LU-mode table values 172
displaying LU-mode table values 178
entering receive state conditionally 249
entering receive state conditionally and including data 259
entering receive state unconditionally 280
entering receive state unconditionally and including data 270
flushing the buffer 493
immediate allocation request 51
indicating the beginning of a synchronization exchange 546
indicating the end of a synchronization exchange 551
obtain status on information from any active conversation 556
obtain status on information immediately available from any active conversation 562
obtain status on information available on a specified conversation 574
obtain status on information immediately available on a specified conversation 568
queued deallocation due to program, service, timer, or user errors 144
receive normal information immediately available from any conversation 365
receive normal information immediately available on a specific conversation 378
receiving an FMH-5 334
receiving expedited data from the specified conversation 312
receiving expedited data immediately available in any-mode 295
receiving expedited data immediately from the specified conversation 304
receiving expedited data in any-mode 287
rejecting a session 166
releasing a suspended session 536
replying to a confirmation request 436
requesting permission to enter send state 501
resetting the continuation mode 419
restoring a mode 185
sending an error notification 480
sending data 444
sending data and flushing the buffer 468
sending data conditionally 427, 456
sending expedited data on a full-duplex session 509
specific-mode RECEIVES 390
suspending a subsequent conversation 541
terminating a session 400
terminating a session due to errors 413
terminating a session due to protocol errors or cleanup deactivation 407
unconditional deallocation 136
unconditional deallocation that includes data 126

Index 717
PARAMETERS, SESSION
DSECT 633
relationship to
RESTORE 181

PIP (program initialization parameters) data
DSECT 643
preparing to receive data 244, 276
PREPRCV macroinstructions (syntax and operands) 244, 276
prerequisite information xiii
program initialization parameters (PIP) data
DSECT 643
PRSISTVP RPL field
returned value, used with
allocation request for contention-winner session 43
allocation request for specific session 31
allocation request that can be queued 19
allocation request, conditional, without wait 66
CNOS request 161
immediate allocation request 54

PSERVIC
profile 1 636
profile 2 639
profile 3 640
profile 4 640
profile 6 642

QUALIFY keyword
ABNDPROG value 70, 139
ABNDSERV value 78, 139
ABNDTIME value 85, 139
ABNDUSER value 93, 139
ACTSESS value 148
ALLOCOD value 9
ANY value
RCPRI
returned value, used with
accepting a session 153
allocation request for contention-winner session 43
allocation request for specific session 31
allocation request that can be queued 19
allocation request, conditional, without wait 66
any-mode RECEIVES 357
CNOS requests 161
conditional deallocation 107
conditional deallocation that includes data 118
deallocation due to program errors 76
deallocation due to service errors 83
deallocation due to timer errors 91
deallocation due to user-specified criteria 99
defining LU-mode table values 173
displaying LU-mode table values 179
entering receive state conditionally 251
entering receive state conditionally and including
data 262
entering receive state unconditionally 282
entering receive state unconditionally and including
data 273
flushing the buffer 495
immediate allocation request 55
indicating the beginning of a synchronization exchange 547
indicating the end of a synchronization exchange 552
obtain status on information available on a specified conversation 576
obtain status on information from any active conversation 557
obtain status on information immediately available from
any active conversation 563
obtain status on information immediately available on a
specified conversation 569
queued deallocation due to program, service, timer, or user errors 146
receive normal information immediately available from
any conversation 368
receive normal information immediately available on a
specific conversation 381
receiving an FMH-5 336
receiving expedited data from the specified conversation 314
receiving expedited data immediately available in
any-mode 297
receiving expedited data immediately from the specified conversation 305
RCPRI (continued)
returned value, used with (continued)
receiving expedited data in any-mode 289
rejecting a session 167
releasing a suspended session 537
replying to confirmation request 437
requesting permission to enter send state 503
resetting the continuation mode 421
restoring a mode 186
sending an error notification 484
sending data 448
sending data and flushing the buffer 472
sending data conditionally 429, 460
sending expedited data on full-duplex session 512
specific-mode RECEIVEs 393
suspending a subsequent conversation 542
terminating a session 402
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 129
RCSEC
returned value, used with
accepting a session 153
allocation request for specific session 31
allocation request that can be queued 19
allocation requests for contention-winner session 43
any-mode RECEIVEs 357
CNOS requests 161
conditional deallocation 107
conditional deallocation that includes data 118
deallocation due to program errors 76
deallocation due to service errors 84
deallocation due to timer errors 91
deallocation due to user-specified criteria 99
defining LU-mode table values 173
displaying LU-mode table values 179
entering receive state conditionally 251
entering receive state conditionally and including data 262
entering receive state unconditionally 282
entering receive state unconditionally and including data 273
flushing the buffer 495
immediate allocation request 55
indicating the beginning of a synchronization exchange 547
indicating the end of a synchronization exchange 552
obtain status on information available on a specified conversation 576
obtain status on information from any active conversation 557
obtain status on information immediately available from any active conversation 563
obtain status on information immediately available on a specified conversation 569
queued deallocation due to program, service, timer, or user errors 146
receive normal information immediately available from any conversation 368
receive normal information immediately available on a specific conversation 381
receiving an FMH-5 336
receiving expedited data from the specified conversation 314
RCSEC (continued)
returned value, used with (continued)
receiving expedited data immediately available in any-mode 297
receiving expedited data immediately from the specified conversation 306
receiving expedited data in any-mode 289
rejecting a session 167
releasing a suspended session 537
replying to a confirmation request 437
requesting permission to enter send state 503
resetting the continuation mode 421
restoring a mode 186
sending an error notification 484
sending data 448
sending data and flushing the buffer 472
sending data conditionally 429, 460
sending expedited data on full-duplex session 512
specific-mode RECEIVEs 394
suspending a subsequent conversation 542
terminating a session 402
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 129
RCVFMH5 control value 328
RECEIVE state, entering
PREPRCV macroinstructions (syntax and operands) 244, 276
receive-any mode
used with any-mode RECEIVEs macroinstruction 348
receive-specific mode
any-mode RECEIVEs macroinstruction, used with 384
receiving data
APPCCMD macroinstructions used 348, 384
RECLEN keyword, used with
accepting a session 152
allocation request for contention-winner session 40
allocation request for specific session 28
allocation request that can be queued 16
CNOS requests 159
conditional deallocation that includes data 115
deallocation due to program errors 75
deallocation due to service errors 82
deallocation due to timer errors 90
deallocation due to user-specified criteria 97
defining LU-mode table values 172
entering receive state conditionally and including data 260
entering receive state unconditionally and including data 271
immediate allocation request 52
queued deallocation due to program, service, timer, or user errors 145
sending an error notification 481
sending data 446
sending data and flushing the buffer 470
sending data conditionally 458
sending expedited data on a full-duplex session 510
unconditional deallocation that includes data 127
returned value, used with
any-mode RECEIVEs 357
displaying LU-mode table values 179
request parameter list (RPL) (continued)

keyword, used with (continued)
receive normal information immediately available from any conversation 366
receive normal information immediately available on a specific conversation 378
receiving an FMH-5 334
receiving expedited data from the specified conversation 313
receiving expedited data immediately available in any-mode 296
receiving expedited data immediately from the specified conversation 304
receiving expedited data in any-mode 288
rejecting a session 166
releasing a suspended session 537
replying to a confirmation request 437
requesting permission to enter send state 502
resetting the continuation mode 420
restoring a mode 186
resending an error notification 481
sending data 446
sending data and flushing the buffer 470
sending data conditionally 428, 458
sending expedited data on a full-duplex session 304
specific-mode RECEIVEs 391
specific-mode RECEIVEs 394
terminating a session due to errors 413
terminating a session due to protocol errors or cleanup deactivation 407
unconditional deallocation 137
unconditional deallocation that includes data 128
RESETRCV control value 415
responding negatively to session requests used with terminating a session 163
responding positively to session requests 148
RESTORE macroinstruction 181
RESTORE qualify value 181
restoring modes and sessions 181
return codes listed and described 591
RCPRI, RCSEC combinations 591
RTNCD, FDB2 information 630
returned information request-to-send indicator 496
RFC (request for comments) accessing online xv
RPL (request parameter list) keyword, used with accepting a session 152
allocation request for contention-winner session 40
allocation request for specific session 28
allocation request that can be queued 16
allocation request, conditional, without wait 64
any-mode RECEIVEs 355
CNOS requests 160
conditional deallocation 105
conditional deallocation that includes data 116
CONTROL=CHECK 70
deallocation due to program errors 75
deallocation due to service errors 82
deallocation due to timer errors 90
deallocation due to user-specified criteria 97
defining LU-mode table values 173
displaying LU-mode table values 179
entering receive state conditionally 249
entering receive state conditionally and including data 260
entering receive state unconditionally 281
entering receive state unconditionally and including data 272
flushing the buffer 494
immediate allocation request 52
indicating the beginning of a synchronization exchange 547
indicating the end of a synchronization exchange 552
obtain status on information available on a specified conversation 575
obtain status on information from any active conversation 557
obtain status on information immediately available from any active conversation 563
obtain status on information immediately available on a specified conversation 568
queued deallocation due to program, service, timer, or user errors 145
request parameter list (RPL) (continued)
keyword, used with (continued)
receive normal information immediately available from any conversation 366
receive normal information immediately available on a specific conversation 378
receiving an FMH-5 334
receiving expedited data from the specified conversation 313
receiving expedited data immediately available in any-mode 296
receiving expedited data immediately from the specified conversation 304
receiving expedited data in any-mode 288
rejecting a session 166
releasing a suspended session 537
replying to a confirmation request 437
requesting permission to enter send state 502
resetting the continuation mode 420
restoring a mode 186
resending an error notification 481
sending data 446
sending data and flushing the buffer 470
sending data conditionally 428, 458
sending expedited data on a full-duplex session 304
specific-mode RECEIVEs 391
specific-mode RECEIVEs 394
terminating a session due to errors 413
terminating a session due to protocol errors or cleanup deactivation 407
unconditional deallocation 137
unconditional deallocation that includes data 128
RESETRCV control value 415
responding negatively to session requests used with terminating a session 163
responding positively to session requests 148
RESTORE macroinstruction 181
RESTORE qualify value 181
restoring modes and sessions 181
return codes listed and described 591
RCPRI, RCSEC combinations 591
RTNCD, FDB2 information 630
returned information request-to-send indicator 496
RFC (request for comments) accessing online xv
RPL (request parameter list) keyword, used with accepting a session 152
allocation request for contention-winner session 40
allocation request for specific session 28
allocation request that can be queued 16
allocation request, conditional, without wait 64
any-mode RECEIVEs 355
CNOS requests 160
conditional deallocation 105
conditional deallocation that includes data 116
CONTROL=CHECK 70
deallocation due to program errors 75
deallocation due to service errors 82
deallocation due to timer errors 90
deallocation due to user-specified criteria 97
defining LU-mode table values 173
displaying LU-mode table values 179
entering receive state conditionally 249
RPL (request parameter list) (continued)

keyword, used with (continued)

entering receive state conditionally and including data 260
entering receive state unconditionally 281
entering receive state unconditionally and including data 272
flushing the buffer 494
immediate allocation request 52
indicating the beginning of a synchronization exchange 547
indicating the end of a synchronization exchange 552
obtain status on information available on a specified conversation 575
obtain status on information from any active conversation 557
obtain status on information immediately available from any active conversation 563
obtain status on information immediately available from a specified conversation 568
queued deallocation due to program, service, timer, or user errors 145
receive normal information immediately available from any conversation 366
receive normal information immediately available on a specific conversation 378
receiving an FMH-5 334
receiving expedited data from the specified conversation 313
receiving expedited data immediately available in any-mode 296
receiving expedited data immediately from the specified conversation 304
receiving expedited data in any-mode 288
rejecting a session 166
releasing a suspended session 537
replying to a confirmation request 437
resetting the continuation mode 420
restoring a mode 186
sending an error notification 481
sending data 448
sending data and flushing the buffer 470
sending data conditionally 428, 458
sending expedited data on a full-duplex session 511
specific-mode RECEIVES 391
suspending a subsequent conversation 542
terminating a session 401
terminating a session due to errors 413
terminating a session due to protocol errors or cleanup deactivation 407
unconditional deallocation 137
unconditional deallocation that includes data 128

RPL extension control block 645
RPL extension layout, figure 650
RPL extension—ISTRPL6X DSECT 645
RTNCD field (continued)

returned value, used with (continued)
deallocation due to program errors 76
deallocation due to service errors 84
deallocation due to timer errors 91
deallocation due to user-specified criteria 99
defining LU-mode table values 173
displaying LU-mode table values 180
entering receive state conditionally 251
entering receive state conditionally and including data 262
entering receive state unconditionally 282
entering receive state unconditionally and including data 274
flushing the buffer 495
immediate allocation request 55
indicating the beginning of a synchronization exchange 548
indicating the end of a synchronization exchange 552
obtain status on information available on a specified conversation 577
obtain status on information from any active conversation 558
obtain status on information immediately available from any active conversation 563
obtain status on information immediately available from a specified conversation 570
queued deallocation due to program, service, timer, or user errors 147
receive normal information immediately available from any conversation 369
receive normal information immediately available on a specific conversation 381
receiving an FMH-5 337
receiving expedited data from the specified conversation 315
receiving expedited data immediately available from any-mode 298
receiving expedited data immediately available in any-mode 290
receiving expedited data immediately from the specified conversation 306
receiving expedited data in any-mode 289
rejecting a session 167
releasing a suspended session 537
replying to a confirmation request 438
requesting permission to enter send state 503
resetting the continuation mode 421
restoring a mode 186
sending an error notification 484
sending data 448
sending data and flushing the buffer 470
sending data conditionally 428, 458
sending expedited data on full-duplex session 512
specific-mode RECEIVES 394
suspending a subsequent conversation 543
terminating a session 402
terminating a session due to errors 414
terminating a session due to protocol errors or cleanup deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 129
RTSRTRN keyword, used with
allocation request for contention-winner session 40
allocation request for specific session 28
allocation request that can be queued 16
allocation request, conditional, without wait 64
immediate allocation request 52
ISTRPL6 588
RTSRTRN keyword, used with (continued)
receiving an FMH-5 334

S
security
conversation level 642
sending data
APPCCMD macroinstructions used
syntax and operands 422, 496
buffer list (OPTCD=BUFSLST) considerations
conditional deallocation that includes data 114
entering receive state conditionally and including
data 259
sending data 115, 445
sending data and flushing the buffer 259, 271, 469
sending data conditionally 457
sending expedited data on a full-duplex session 510
unconditional deallocation that includes data 127
flushing the buffer, used with
conditional deallocation 101
conditional deallocation that includes data 109
entering receive state conditionally 244
entering receive state conditionally and including
data 253
entering receive state unconditionally 276
entering receive state unconditionally and including
data 265
flushing the buffer 489
sending data and flushing the buffer 463
sending data conditionally 422, 451
unconditional deallocation 132
unconditional deallocation that includes data 121
macroinstructions for sending new data, used with
conditional deallocation that includes data 109
entering receive state conditionally and including
data 253
entering receive state unconditionally and including
data 265
sending data 439
sending data and flushing the buffer 463
sending data conditionally 451
unconditional deallocation that includes data 121
sending expedited data on a full-duplex session 504
SENSE
keyword, used with
accepting a session 153
deallocation due to user-specified criteria 97
ISTRPL6 589
sending an error notification 481
terminating a session 401
terminating a session due to errors 413
terminating a session due to protocol errors or cleanup
deactivation 407
returned value, used with
allocation request for contention-winner session 44
allocation request for specific session 31
allocation request that can be queued 19
allocation request, conditional, without wait 67
any-mode RECEIVEs 357
CNOS request 161
conditional deallocation 107
conditional deallocation that includes data 118
entering receive state conditionally 251
entering receive state conditionally and including
data 262
SENSE (continued)
returned value, used with (continued)
entering receive state unconditionally and including
data 274
receive normal information immediately available from
any conversation 369
receive normal information immediately available on a
specific conversation 381
receiving an FMH-5 337
sending an error notification 484
sending data 448
sending data and flushing the buffer 472
sending data conditionally 430, 460
specific-mode RECEIVEs 394
unconditional deallocation that includes data 129
SESSID
keyword, used with
indicating the beginning of a synchronization
exchange 547
indicating the end of a synchronization exchange 552
terminating a session 413
returned value, used with
allocation request for contention-winner session 44
allocation request for specific session 32
allocation request that can be queued 19
allocation request, conditional, without wait 67
immediate allocation request 55
receiving an FMH-5 337
releasing a suspended session 537
suspending a subsequent conversation 542
SESSIDL
keyword, used with
indicating the beginning of a synchronization
exchange 547
indicating the end of a synchronization exchange 552
terminating a session 413
returned value, used with
allocation request for contention-winner session 44
allocation request for specific session 32
allocation request that can be queued 19
allocation request, conditional, without wait 67
immediate allocation request 55
receiving an FMH-5 337
suspending a subsequent conversation 542
session activation, macroinstruction for 148
session deactivation
used with terminating a session macroinstruction 163
session limit
changing with CNOS requests 154
session limits, changing
with CNOS request macroinstruction 154
session parameters.
DSECT 633
relationship to
RESTORE 181
session requests, accepting
with a session macroinstruction 148
session requests, rejecting
used with terminating a session 163
SETSESS control value 533, 538, 543, 548
shortcut keys 687
SIGDATA RPL extension field
returned value, used with
any-mode RECEIVEs 358
conditional deallocation that includes data 118
SIGDATA RPL extension field (continued)

- returned value, used with (continued)
  - entering receive state conditionally and including data 263
  - entering receive state unconditionally and including data 274
  - receive normal information immediately available on a specific conversation 381
  - receiving expedited data from the specified conversation 315
  - receiving expedited data immediately available in any-mode 298
  - receiving expedited data immediately from the specified conversation 306
  - receiving expedited data in any-mode 290
  - sending an error notification 484
  - sending data 449
  - sending data and flushing the buffer 473
  - sending data conditionally 430, 461
  - sending expedited data on full-duplex session 512
  - specific-mode RECEIVES 394
  - unconditional deallocation that includes data 129

Signal RU 496

SIGRCV RPL extension field

- returned value, used with
  - any-mode RECEIVES 358
  - conditional deallocation that includes data 118
  - entering receive state conditionally and including data 263
  - entering receive state unconditionally and including data 274
  - receive normal information immediately available from any conversation 369
  - receive normal information immediately available on a specific conversation 382
  - receiving expedited data from the specified conversation 315
  - receiving expedited data immediately available in any-mode 298
  - receiving expedited data immediately from the specified conversation 306
  - receiving expedited data in any-mode 290
  - sending an error notification 484
  - sending data 449
  - sending data and flushing the buffer 473
  - sending data conditionally 430, 461
  - sending expedited data on full-duplex session 512
  - specific-mode RECEIVES 394
  - unconditional deallocation that includes data 130

STSHBF (continued)

- returned value, used with (continued)
  - deallocation due to user-specified errors 99
  - entering receive state conditionally and including data 263
  - entering receive state unconditionally and including data 275
  - queued deallocation due to program, service, timer, or user errors 147
  - sending an error notification 485
  - sending data 449
  - sending data and flushing the buffer 473
  - sending data conditionally 461
  - unconditional deallocation that includes data 130

STSHDS

- returned value, used with
  - conditional deallocation that includes data 119
  - deallocation due to service errors 84
  - deallocation due to timer errors 91
  - deallocation due to user-specified errors 99
  - entering receive state conditionally and including data 263
  - entering receive state unconditionally and including data 275
  - queued deallocation due to program, service, timer, or user errors 147
  - sending an error notification 485
  - sending data 449
  - sending data and flushing the buffer 473
  - sending data conditionally 461
  - unconditional deallocation that includes data 130

sync level 642

T

TCP/IP

- online information xv
- Technotes xiii
- terminating session
  - terminating a session due to errors (syntax and operands) 409
- trademark information 699
- truncating
  - sending an error notification macroinstruction 475
- TYPE keyword, used with
  - ISTRPL6 589
  - sending an error notification 481

U

USERFLD

- keyword, used with
  - accepting a session 152
  - allocation request for contention-winner session 41
  - allocation request for specific session 28
  - allocation request that can be queued 16
  - allocation request, conditional, without wait 64
  - CNOS request 160
  - defining LU-mode table values 173
  - displaying LU-mode table values 179
  - immediate allocation request 52
  - ISTRPL6 589
  - receiving an FMH-5 334
  - rejecting a session 166
  - restoring a mode 186
USERFLD (continued)
returned value, used with
accepting a session 153
any-mode RECEIVEs 358
CNOS request 161
conditional deallocation 108
deallocation due to program errors 77
deallocation due to service errors 84
deallocation due to timer errors 92
deallocation due to user-specified criteria 99
defining LU-mode table values 173
displaying LU-mode table values 180
entering receive state conditionally 252
entering receive state conditionally and including data 263
entering receive state unconditionally 282
entering receive state unconditionally and including data 275
flushing the buffer 495
obtain status on information immediately available on a specified conversation 377
obtain status on information immediately available on a specified conversation 370
queued deallocation due to program, service, timer, or user errors 147
receive normal information immediately available from any conversation 369
receive normal information immediately available on a specific conversation 382
receiving expedited data from the specified conversation 315
receiving expedited data immediately available in any-mode 298
receiving expedited data immediately from the specified conversation 306
receiving expedited data in any-mode 290
rejecting a session 167
replying to a confirmation request 438
requesting permission to enter send state 503
resetting the continuation mode 421
restoring a mode 186
sending an error notification 485
sending data 449
sending data and flushing the buffer 473
sending data conditionally 430, 461
sending expedited data on full-duplex session 513
specific-mode RECEIVEs 395
terminating a session 402
terminating a session due to protocol errors or cleanup deactivation 408
unconditional deallocation 138
unconditional deallocation that includes data 130

what-received RPL extension field (continued)
returned value, used with (continued)
receive normal information immediately available on a specific conversation 382
specific-mode RECEIVEs 395
WHENFREE qualify value
session assignment algorithm 57
winner, contention requests for allocation 33

Z
z/OS Basic Skills Information Center xiii
z/OS, documentation library listing 701

V
VTAM, online information xv

W
what-received RPL extension field
returned value, used with
any-mode RECEIVEs 358, 370
receive normal information immediately available from any conversation 370
Communicating your comments to IBM

If you especially like or dislike anything about this document, use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Send your comments to us in any of the following ways:

• To send comments by FAX, use this number: 1+919-254-1258
• To send comments electronically, use this address:
  – comsvrcf@us.ibm.com
• To send comments by post, use this address:
  International Business Machines Corporation
  Attn: z/OS Communications Server Information Development
  P.O. Box 12195, 3039 Cornwallis Road
  Department AKCA, Building 501
  Research Triangle Park, North Carolina 27709-2195

Make sure to include the following information in your note:

• Title and publication number of this document
• Page number or topic to which your comment applies