IDF Reference Summary

Release 6
High Level Assembler for z/OS & z/VM & z/VSE

IDF Reference Summary

Release 6
Sixth Edition (July 2008)

This edition applies to IBM High Level Assembler for z/OS & z/VM & z/VSE Toolkit Feature, Release 6, Program Number 5696-234 and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

A form for reader’s comments is provided at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation
Reader Comments
DTX/E269
555 Bailey Avenue
SAN JOSE, CA 95141-1003
U.S.A.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1995, 2008. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
# Contents

**About this book** ................................................. xi
Syntax notation .................................................. xii

## Chapter 1. IDF basics ........................................... 1
Windows ............................................................ 1
IDF address expressions ........................................ 3
Addresses displayed by IDF ...................................... 3
Cursor addressing ................................................ 3
PF keys ............................................................. 4
Typeover storage modification .................................. 4

## Chapter 2. IDF commands ....................................... 7
ABEND (CMS and z/OS) ........................................... 7
ADSTOP (CMS only) ................................................ 7
ADSTOPS (CMS only) ............................................. 7
AFPR ................................................................. 7
ALARM ............................................................... 8
ALET ................................................................. 8
APROGS (CMS only) ............................................. 8
AREGS ............................................................... 8
ARRAY ............................................................... 9
AUDIT ............................................................... 9
BACK ................................................................. 9
BASE ............................................................... 9
BINARY ............................................................. 9
BIT ................................................................. 10
BOTTOM ............................................................ 10
BREAK ............................................................. 10
BRIEF ............................................................. 10
CALLERS ........................................................... 11
CHARACTER ....................................................... 11
CHECK ............................................................ 12
CLOSE .............................................................. 12
COLORS ............................................................ 12
COMMAND ........................................................ 13
COMPACT .......................................................... 13
CREGS (CMS only) .............................................. 13
CURSOR ............................................................ 13
DBREAK ............................................................ 14
DETAIL ............................................................. 14
DISASM ............................................................. 14
DOWN .............................................................. 14
DROP GLOBAL ..................................................... 15
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>REGS64 (z/OS only)</td>
<td>42</td>
</tr>
<tr>
<td>REGSTOPS (CMS only)</td>
<td>42</td>
</tr>
<tr>
<td>RESTORE</td>
<td>42</td>
</tr>
<tr>
<td>RETRIEVE</td>
<td>42</td>
</tr>
<tr>
<td>RIGHT</td>
<td>43</td>
</tr>
<tr>
<td>RLOG</td>
<td>43</td>
</tr>
<tr>
<td>RUN</td>
<td>43</td>
</tr>
<tr>
<td>RUNEXIT</td>
<td>43</td>
</tr>
<tr>
<td>R0-R15</td>
<td>43</td>
</tr>
<tr>
<td>SALIMIT</td>
<td>44</td>
</tr>
<tr>
<td>SAREGS</td>
<td>44</td>
</tr>
<tr>
<td>SAVE</td>
<td>44</td>
</tr>
<tr>
<td>SEARCH</td>
<td>44</td>
</tr>
<tr>
<td>SELFNUCX (CMS only)</td>
<td>44</td>
</tr>
<tr>
<td>SET ADSTOP (CMS only)</td>
<td>45</td>
</tr>
<tr>
<td>SET AREG</td>
<td>45</td>
</tr>
<tr>
<td>SET BREAK</td>
<td>45</td>
</tr>
<tr>
<td>SET COMMAND</td>
<td>45</td>
</tr>
<tr>
<td>SET EXITEXEC</td>
<td>46</td>
</tr>
<tr>
<td>SET GLOBAL STEM</td>
<td>46</td>
</tr>
<tr>
<td>SET GLOBAL TEXT</td>
<td>46</td>
</tr>
<tr>
<td>SET ICOUNT</td>
<td>46</td>
</tr>
<tr>
<td>SET OFFSET</td>
<td>47</td>
</tr>
<tr>
<td>SET OPTION</td>
<td>47</td>
</tr>
<tr>
<td>SET PSW</td>
<td>47</td>
</tr>
<tr>
<td>SET REGSTOP (CMS only)</td>
<td>47</td>
</tr>
<tr>
<td>SET SIZE</td>
<td>48</td>
</tr>
<tr>
<td>SHOW</td>
<td>48</td>
</tr>
<tr>
<td>SIZE</td>
<td>49</td>
</tr>
<tr>
<td>SKIPSTEP</td>
<td>50</td>
</tr>
<tr>
<td>SPACE</td>
<td>50</td>
</tr>
<tr>
<td>STATUS</td>
<td>50</td>
</tr>
<tr>
<td>STEP</td>
<td>50</td>
</tr>
<tr>
<td>STMTSTEP</td>
<td>50</td>
</tr>
<tr>
<td>STOKEY</td>
<td>51</td>
</tr>
<tr>
<td>STOREMAP</td>
<td>51</td>
</tr>
<tr>
<td>STRUCTURE</td>
<td>51</td>
</tr>
<tr>
<td>SUBSET (CMS only)</td>
<td>51</td>
</tr>
<tr>
<td>SVC (CMS only)</td>
<td>52</td>
</tr>
<tr>
<td>SWAP</td>
<td>52</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>52</td>
</tr>
<tr>
<td>TASKS (TSO only)</td>
<td>52</td>
</tr>
<tr>
<td>TITLE</td>
<td>53</td>
</tr>
<tr>
<td>TOP</td>
<td>53</td>
</tr>
<tr>
<td>TRIGGER LOAD</td>
<td>53</td>
</tr>
<tr>
<td>TYPE</td>
<td>53</td>
</tr>
<tr>
<td>UNION</td>
<td>53</td>
</tr>
</tbody>
</table>
Chapter 3. ASMDF EXTRACT Command. ........................................... 57
ADSTOPs (CMS only) ............................................. 57
ALET ....................................................... 57
AREGS .................................................... 57
ARGUMENT .................................................. 57
ARRAY ...................................................... 58
BREAK ...................................................... 58
CALLERS ................................................... 58
CMDMSG .................................................. 59
COLORS ..................................................... 59
CURSOR ..................................................... 59
DISASM ..................................................... 59
EVENT ......................................................... 60
EXITEXEC .................................................. 60
GLOBAL ..................................................... 60
GLOBAL STEM ............................................. 60
GLOBAL STEMS ............................................ 61
GSTATUS ................................................... 61
I_COUNT ..................................................... 61
LANGUAGE ARGUMENTS ............................................. 61
LANGUAGE COMMANDS ........................................ 62
LANGUAGE OPTIONS .......................................... 62
LANGUAGE STATUS .......................................... 62
LANGUAGE STEM .............................................. 62
LANGUAGE VERSION .......................................... 63
LASTMSG ...................................................... 63
LOAD ......................................................... 63
LOCATION ................................................... 63
LOCATION ALET ............................................ 64
MAP .......................................................... 64
MODE (CMS only) ............................................... 64
MODULES ..................................................... 64
MSTATUS ..................................................... 65
NAMEs ........................................................ 65
About this book

This book is intended to be used as a quick reference for the High Level Assembler Toolkit Feature Interactive Debug Facility (ASMIDF) User’s Guide.

The Interactive Debug Facility, a feature of the IBM High Level Assembler Toolkit Feature, is referred to as “ASMIDF” throughout this publication.

This book is divided into the following sections:

• ASMIDF basics
• ASMIDF commands
• ASMIDF SET command
• ASMIDF EXTRACT command
• ASMIDF options
• ASMIDF language support
• Using ASMLANGX

This book uses format conventions and syntax diagram conventions in describing language and statement elements.

Throughout this book, we use these indicators to identify platform-specific information:

• Prefix the text with platform-specific text (for example, “Under CMS...”)
• Add parenthetical qualifications (for example, “(CMS)”)
• Marking lines above and below the text. Here are some of the marking lines we use:

------------------ z/OS ------------------

Informs you of information specific to z/OS®.

------------------ End of z/OS ------------------

------------------ z/VM ------------------

Informs you of information specific to z/VM®.

------------------ End of z/VM ------------------
About this book

Informs you of information specific to z/VSE™.

End of z/VSE

CMS is used in this manual to refer to Conversational Monitor System on z/VM.

Syntax notation

Throughout this book, syntax descriptions use the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.
  - The ➤ symbol indicates the beginning of a statement.
  - The ➔ symbol indicates that the statement syntax is continued on the next line.
  - The ➕ symbol indicates that a statement is continued from the previous line.
  - The ➖ symbol indicates the end of a statement.

Diagrams of syntactical units other than complete statements start with the ➤ symbol and end with the ➖ symbol.

- Keywords appear in uppercase letters (for example, ASPACE) or upper and lower case (for example, PATHFile). They must be spelled exactly as shown. Lower case letters are optional (for example, you could enter the PATHFile keyword as PATHF, PATHFI, PATHFIL or PATHFILE).
  - Variables appear in all lowercase letters in a special typeface (for example, integer). They represent user-supplied names or values.
  - If punctuation marks, parentheses, or such symbols are shown, they must be entered as part of the syntax.
  - Required items appear on the horizontal line (the main path).

  ➤-INSTRUCTION—required item

- Optional items appear below the main path. If the item is optional and is the default, the item appears above the main path.
Syntax notation

- When you can choose from two or more items, they appear vertically in a stack.
  If you **must** choose one of the items, one item of the stack appears on the main path.

- An arrow returning to the left above the main line indicates an item that can be repeated. When the repeat arrow contains a separator character, such as a comma, you must separate items with the separator character.

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

The following example shows how the syntax is used.
The item is optional, and can be coded or not.

The INSTRUCTION key word must be specified and coded as shown.

The item referred to by “fragment” is a required operand. Allowable choices for this operand are given in the fragment of the syntax diagram shown below “fragment” at the bottom of the diagram. The operand can also be repeated. That is, more than one choice can be specified, with each choice separated by a comma.
Chapter 1. IDF basics

On CMS and TSO you can activate IDF with the following command:

```
ASMIDF module_name (idf_options/Module_parameters)
```

On z/OS, you can activate IDF:

- In TSO batch, with the following command supplied on the DD card SYSTSIN:
  ```
  ASMIDF module_name (LU vtam_luid idf_options/module_parameters)
  ```
- In batch, with the following JCL:
  ```
  //stepname EXEC PGM=ASMIDF,
  //      PARM='module_name ( NOSVC97 LU vtam_luid idf_options /
  //          module_parameters')
  ```

On z/VSE you can activate IDF with the following JCL:

```
// EXEC ASMIDF,PARM='module_name (idf_options/module_parameters'
```

Where:

- **module_name** — The name of the module to be debugged.
- **idf_options** — Options directed to ASMIDF.
- **module_parameters** — The parameters directed to the module to be debugged.

Some command examples are:

```
ASMIDF module_name (COLORS RWGY / in out (abcd
ASMIDF module_name (PATH / in out (abcd
```

Windows

The types of windows available within IDF are:

- **AdStops window** (CMS only)
  
  - Opened by the ADSTOPS and REGSTOPS commands.
  - Closed by the ADSTOPS, REGSTOPS, or CLOSE command.
  - Displays the current PER AdStops and Register Stops.

- **Additional Floating-Point Registers window**
  
  - Opened by the AFPR command.
  - Closed by the AFPR or CLOSE command.
  - Displays the current Additional Floating-Point Registers and the Floating-Point Control Register.

- **Break window**
  
  - Opened by the BREAK command.
Windows

Closed by the BREAK or CLOSE command.
Displays the active breakpoints and watchpoints.

• Current Registers window
  Opened by the REGS, REGS64, AREGS, or CREGS command.
  Closed by the REGS or CLOSE command.
  Displays the current PSW, GPRs, and FPRs or ARs or CRs.

• Disassembly window
  Opened by the DISASM or OPEN command.
  Closed by the DISASM or CLOSE command.
  Displays disassembled instructions.

• Dump window
  Opened by the DUMP or OPEN command.
  Closed by the DUMP or CLOSE command.
  Displays storage in "dump format".

• Entry Point Names window
  Opened by the EPNAMES or OPEN command.
  Closed by the EPNAMES or CLOSE command.
  Displays information about the entry points in the module.

• Language Support Module information window
  Opened by the VARIABLE, LANGUAGE, STRUCT, or OPEN command.
  Closed by the VARIABLE or CLOSE command.
  Displays information from IDF Language Support commands.

• Minimized Windows Viewer
  Opened by the MINIMIZE command.
  Closed by the MAXIMIZE command.
  Lists the minimized windows.

• Old Registers window
  Opened by the OREGS command.
  Closed by the OREGS or CLOSE command.
  Displays the old ARs, CRs or PSW, GPRs, FPRs, and instruction at PSW.

• Options window
  Opened by the OPTIONS command.
  Closed by the OPTIONS or CLOSE command.
  Displays settings of various options.

• Skipped Subroutines window
  Opened by the SKIPSTEP command.
  Closed by the SKIPSTEP or CLOSE command.
  Displays subroutines “skipped” during single stepping, statement stepping, or the PATH or FASTPATH processing.

• Target Status window
  Opened by the STATUS command.
  Closed by the STATUS or CLOSE command.
  Displays information about the target program.
IDF address expressions

IDF address expressions are made up of terms separated by plus or minus signs. A term can consist of a program symbol, a hex constant (X’5’), a decimal constant (’4’), a character constant that is one character in length (’A’), or an implicit numeric constant (247).

Program symbols are of the form "(module.csect) symbol". If supported by the active LSM, they may also be of the form "(module.csect) STMT#nnnn". The csect in "(module.csect)" is only needed if the symbol occurs in multiple CSECTs. The module in "(module.csect)" is only needed if the symbol is not in the currently qualified module. To select the module to be the currently qualified module, use the SET QUALIFY command.

Terms may be followed with a register designator. A register designator consists of the string R0 through R15 or AR0 through AR15, enclosed in parentheses. Using AR0 through AR15 directs the DUMP, SET ALET, and EXTRACT LOCATION commands to use the ALET in the specified AR.

Terms and register designators may be followed by indirection operators (%, :>, ?, =>, ->, &, ++). If an indirection operator follows a term, IDF uses the contents of the word pointed to by the expression evaluated thus far. Similarly, if an indirection operator follows a register designator, IDF is being told how to interpret the contents of the register. The word or register is treated as:

- A 24-bit address if the % or :> operators are used.
- A 31-bit address if the ? or => operators are used.
- A 64-bit address if the & or ++ operators are used.
- The appropriate size (24-bit, 31-bit or 64-bit) depending on the AMODE of the PSW if the -> operator is used.

Addresses displayed by IDF

Whenever appropriate, IDF displays addresses in symbolic form. It is normally of the form "(module.csect) symbol+offset". If the address corresponds to IDF Language extract data, it is of the form "(module.csect) STMT#nnnn+offset". The module name is omitted if it is the currently qualified module, unless the FULLQUAL option is used.

Cursor addressing

IDF allows you to specify addresses by placing the cursor in a field on the screen. IDF determines an address in the following ways:

- If the cursor is in a GPR, the contents of that displayed register are used. If the cursor is in an AR, the DUMP and SET ALET commands use the ALET in that AR and the DUMP command uses the address in the associated GPR.
Cursor addressing

• If the cursor is in the PSW, the address part of the PSW is used.
• If the cursor is in the hex part of a disassembled instruction, then:
  – All commands except DISASM and OPEN DISASM use the address of the halfword containing the cursor.
  – If the field containing the cursor is both the first field disassembled and a branch instruction, the DISASM and OPEN DISASM commands use the effective address of the branch instruction. Otherwise they behave like other commands.
• If the cursor is in a dump field, then:
  – All commands except DUMP and OPEN DUMP use the address of the beginning of the hexadecimal field containing the cursor, or the exact address of the character on which the cursor is positioned if it is in the character portion of the display.
  – If the field is both a fullword field and the first field in the dump display, the DUMP and OPEN DUMP commands use the contents of the field. Otherwise they behave like other commands.
• If the cursor is in the protected portion of a disassemble or dump line the starting address of the line is used.

PF keys

The ENTER key and PF keys 1 through 24 can be set to any IDF command or to any IDF macro by the SET PFK command. When this is done, instead of typing the command, you can press the PF key.

The PF key settings are displayed at the bottom of the screen, unless turned off by the PFKDISP command.

Typeover storage modification

• In a Dump window, or a Disassembly window containing storage being dumped, storage may be changed by overtyping the hex or character display of that storage.
• In the current registers window, the PSW, general purpose or access registers, and floating point registers may be changed by overtyping the displayed value.
• In a Disassembly window, the hex values of the instructions may be changed by overtyping them.
• In the Additional Floating-Point Registers window, the floating point registers may be changed by overtyping the displayed values.
• In the Entry Point Names window, the short entry point name may be changed by overtyping the displayed values.
Typeover storage modification

The changes are immediately reflected on the screen as different instruction mnemonics, addresses, and so on.
Chapter 2. IDF commands

**ABEND (CMS and z/OS)**
Performs IDF cleanup, then issues OS ABEND.

```
 ABEND obend-code
```

**ADSTOP (CMS only)**
Sets one end of a PER ADSTOP range.

```
 ADStop expression
```

**ADSTOPS (CMS only)**
Displays the current Address Stops and Register Alteration Stops.

```
 ADSTops REGSTops
```

**AFPR**
Displays the Additional Floating-Point Registers and the Floating-Point Control Register.

```
 AFPR
```
ALARM

Enables or disables the terminal alarm.

ALET

Sets the ALET for a dump window.

APROGMSG (CMS only)

Enables or disables the trapping of asynchronous program-checks which occur while IDF displays the user interface.

AREGS

Rotates the register display between GPRs and ARs.
**ARRAY**

Enables variable display in the array format.

```markdown
>>--ARRy [window] [element]
```

**AUDIT**

Enables or disables the VAR basing "audit trail".

```markdown
>>--AUDit [ON|OFF]
```

**BACK**

Displays previously dumped storage (the last 10 dumps can be displayed).

```markdown
>>--BACK [window]
```

**BASE**

Sets the base of a target.

```markdown
>>--BASE [expression]
```

**BINARY**

A synonym of the FIXED command.
BIT

Sets or queries the VAR display format for BIT variables.

BOTTOM

 Displays source code at the highest available address within the current code section.

BREAK

Sets an instruction breakpoint.

BRIEF

Disables or enables the display of VAR declaration information.
CALLERS

Displays information for each generation in the program caller hierarchy.

- The information includes:
  - Location as:
    - (mod.sect)stmt#nnnn+offset
    - program_block_name+offset (if known)
  - Save Area header
  - Save Area register values
- Caller generations are numbered:
  0  current program
  1  parent (caller)
  2  grand parent (caller of caller)
  ...  and so on
- If particular caller generations are specified, only the corresponding information is shown.
- The default is "*", to show all caller generations.

Also see the SAREGS and SALIMIT commands.

CHARACTER

Sets or queries display format for CHARACTER variables.
CHECK

Enables or disables the checking of types of input values.

BOUNDs
Array index bounds
NEGATIVE
Unsigned variable values
SUBSTRING
Character or bit string substring limits
ALL | *
All of the above

CLOSE
Closes a window.

COLORS
Sets display colors.

Each value is the first letter of one of the following colors:

Blue, Green, Pink, Red, Turquoise, Yellow, White

For example:
COLORS BGRY
COMMAND

gives blue messages, green headings, red text, and yellow input.

COMMAND
Performs an IDF command.

COMMAND

COMPACT
Enables or disables the compact variable display mode.

COMPACT

CREGS (CMS only)
Rotates register display between GPRs and CRs.

CREGS

CURSOR
Positions the cursor within a window.

CURSOR

Notes:
1 2-digit hexadecimal values, separated by blanks.
DBREAK

Sets a deferred instruction breakpoint in a module.

DETAIL

Controls the display of data for Structure or Union components of intermediate depth.

DISASM

Displays a disassembly listing.

DOWN

A synonym of the NEXT command.
DROP GLOBAL

Discards information for stems from storage.

\[\text{DROP GLOBAL} \text{stem-name.}\]

DROP MODULE

Discards information about module module-name.

\[\text{DROP MODULE} \text{module-name}\]

DROP SYMBOLS

Discards IDF symbols.

\[\text{DROP SYMBOLS} \text{module-name}\]

DUMP

Provides Storage Dump in the format HEX ... HEX *char*

\[\text{DUMP} \text{window address}\]
DUMPMODE

DUMPMODE
Toggles the Dump Format between symbolic and unformatted.

EPNAMES

EPNAMES
Toggles the Entry Point Names display.

EPOFFSET

EPOFFSET
Specifies the entry-point-offset.

EXITEXEC

EXITEXEC
Toggles the Exit Routine.

EXLIMIT

EXLIMIT
Sets the maximum LSM stemmed array index during EXTRACT LANGUAGE commands execution.
EXLIMIT

- Prevents "run-away" if data being extracted is unbounded
  (for example: LANGUAGE EXTRACT VVAlu for Char(*) or Bit(*) variable).
- The default EXLimit is 20000.

FIND

An ISPF-style source text search facility, which locates the string and displays the section of code where it occurs.

* Use current search string

start-col  An integer; the column at which searching starts.
finish-col  An integer; the column at which searching finishes.
FIRST      Begin the search at the lowest address, and look for the search string in a forward direction.
LAST       Begin the search at the highest address, and look for the search string in a reverse direction.
NEXT       Begin the search at the current address, and look for the search string in a forward direction.
PREVIOUS    Begin the search at the current address, and look for the search string in a reverse direction.

A search string that is numeric or contains imbedded blanks must be enclosed in quotes. Both "...." and '...' forms are accepted.

Unless otherwise qualified, the search is performed from the current address, in the direction last specified.
FIRST

Displays the source code which corresponds to the lowest address.

FIXED

Sets or queries the VAR display format for FIXED variables.

FLOAT

Sets or queries the VAR display format for FLOAT variables.

FMT

A synonym of the FORMAT command.
FOLLOW

Directs the DUMP window to “follow” the contents of a register.

```
> FOL low
  window
  address
  OFF
```

FORMAT

Controls the display format for individual variables.

```
> FORM at
  variable-name
  ;
  variable-name
  format-type
```

`format-type` must be appropriate for variable data type:

- `*`  Reset display format to default for variable class.
- `Bit` Refer to BIT
- `Char` Refer to CHAR
- `Fixed` Refer to FIXED
- `Float` Refer to FLOAT
- `Packed` Refer to PACKED
- `Zoned` Refer to ZONED

If the `format-type` is absent, displays the current display format for the variable.

FPC

Sets the Floating Point Control register.

```
> FPC
  FPC-value
```
FPR

Sets a floating point register.

```
FPR—FPR-number—FPR-value
```

The FPR number is 0, 2, 4, or 6, except for OS/390 systems with binary floating point support, where registers 0 to 15 may be specified.

GLOBAL

Displays information about the Global Storage stems.

```
SET—GLOBAL—global-text
```

GOTO

Places an evaluated expression in the address portion of the PSW.

```
GOTO—expression
```

GPACK

Returns the Global Storage data storage areas which no longer contain stem data.

```
GPAck
```
GPR

Sets a general register.

```
>>> GPR register-number expression
```

GPRG (z/OS only)

Sets a 64-bit general register.

```
>>> GPRG register-number expression
```

GPRH (z/OS only)

Sets the upper 32-bits of a 64-bit general register.

```
>>> GPRH register-number expression
```

GSTATUS

Displays information about the storage used to contain the Global Storage stem data loaded with SET GLOBAL STEM commands.

```
>>> GSTatus
```
HIDE

HIDE

Controls the display of source code and disassembly, by hiding information. The SHOW command controls the display by showing information.

Notes:
1 An option can be chosen no more than once.

DISASM Show source code only
ALL | * Show source code only, excluding comments, declarations, macro expansions, and source lines with no corresponding object code
SOURCE Show disassembly only
COMMENTS Exclude block comment source code
DECLARES | DCL Exclude declaration source code
MACROS Exclude macro expansion source code
NOCODE Exclude source lines with no corresponding object code

HISTORY

Reviews instruction history (PATH).
ICOUNT

Displays the number of instructions executed since the last ICOUNT command.

```
$ICOUNT
```

KWDSYN

Defines a synonym of an IDF keyword.

```
$KWDSYN oldkwd newkwd
```

LANGUAGE +

Scrolls the LSM window.

```
$LANGUAGE [window] [ + - ] [scroll-number-of-lines]
```

LANGUAGE COLOR

Selects the color used to display source code.

```
$LANGUAGE COLOR BLUE
```

The default is the color used by IDF for text display.
**LANGUAGE COMMENTS**

Enables or disables the block comment display.

```
/en/lan/Comments ON OFF
```

**LANGUAGE DEBUG**

Enables or disables the display of IDF LSM interface debug information.

```
/en/lan/Debug qualifiers
```

Should only be used as directed by IBM support.

**LANGUAGE DECLARES**

Enables or disables the declare display.

```
/en/lan/Declares ON OFF
```

**LANGUAGE DROP**

Removes one or more language extract files from memory.

```
/en/lan/Drop *extract-file-name*
```

* All currently loaded extract files are removed.

extract-file-name

This extract file is removed.
LANGUAGE LOAD

Loads an extract file, optionally associating it with a specific MODULE.

```
 LANGUAGE LOAD extract-file-name [ASMLANGX] [file-type] [file-mode]
  [MODULE module-name]
```

- `extract-file-name` - z/OS PDS member name
- `file-type` - (CMS only) z/OS DD name.

Specifying this option eliminates the search using the XPATH file types. The default XPATH is "ASMLANGX".

- `MODULE module-name` - Associates an extract file with a module. See section "Options" on page 78.

LANGUAGE MACROS

Enables or disables the display of assembler source generated by macros.

```
 LANGUAGE MACros [ON] [OFF]
```

LANGUAGE OPTIONS

Displays the current value of ASMLANG settings and the Options save stack nesting level.

```
 LANGUAGE OPTIONS
```
LANGUAGE SCROLL

Sets the default scroll amount.

0  Disable scrolling
1-254  Scroll by this number of lines
MAX 1 *  Scroll by maximum amount (current size of LSM window)

LANGUAGE STATUS

Displays information about extract files currently loaded.

*  Show all extract files.
extract-file-name  The name of the extract file to display information about.

LANGUAGE STEM

Alters the name of the REXX stemmed array variable for the EXTRACT
LANGUAGE commands, and other EXTRACT commands.
LANGUAGE VERSION

Displays the ASMLANG version identifier.

```bash
$ASMLANG$VER$ION$
```

LANGUAGE XPATH (CMS and z/OS)

Defines the extract file search path file type (z/OS DD name) information.

```bash
$ASMLANGX$PATH
```

Notes:

1. Up to 10 entries can be specified.

- Used to locate extract files for which the extract file type (z/OS DD name) has not been explicitly specified.
- XPATH entries are searched in the order specified.
- If parameters are specified, then sets XPATH as specified, with up to 10 entries.
- If parameters are not specified, then resets XPATH to the default of "ASMLANGX".

LANGUAGE STATUS displays the current XPATH.

LAST

Displays source code at the highest address.

```bash
$LAST$window
```
LASTMSG

LASTMSG
Displays last two messages.

```
LASTmsg
```

LEFT

Scrolls a window left.

```
LEFT window number-of-columns
```

LIBE (CMS and z/OS)

Nominates the source of the target program which IDF is to load.

```
LIBE file-name
```
LOAD

Loads a target module and associated symbols.

- LOAD loads a target module and symbols.
- LOAD MODULE loads a module.
- LOAD SYMBOLS loads symbols for a module.

LOCATE

XEDIT-style source text search facility which locates the string and displays the section of code where it occurs. The search begins at first source line on screen.

If "-" is specified, the search is performed towards the beginning of the source information.

The trailing delimiter is only required if the string contains trailing blanks.
LOCATION

LOCATION
Sets the main storage to MEMAREA (MEMAREA is a REXX variable).

MACRO
Issues an IDF macro.

MAJOR
Disables or enables display of data for Structure or Union major component.
MAP

Displays information about modules.

* Information shown for all modules.
  module-name
  Information shown for this module.

The information includes:
  • Module location
  • CSECT location
  • Extract file associated with each CSECT

MAXIMIZE

Maximizes a window.

MINIMIZE

Minimizes a window.
MODE (CMS only)

Sets the file mode for command and macro logging and play back.

```
>>> MODE file-mode
```

MODULE

Prevents IDF from loading a target module. (Use only within a macro.)

```
>>> MODULE
```

MODULE

Sets the base and size of a module from system control blocks.

```
>>> MODULE module-name
   CDE
   NUCext
   TRANs
```

MODULE BASE

Sets the base of module `modname`.

```
>>> MODULE module-name BASE module-start-address
```
MODULE SIZE

Sets the size of module modname.

```
>>> MODULE modname SIZE module-length
```

MOVE

Moves a window around on the screen.

```
>>> MOVE window Location

ADStops
AFPR
BReak
DISasm
DUMP
OPTIONS
OREGs
REGs
SKIPstep
STATUS
LSMinfo
```

Location:

```
+------------------+
|                   |
|                   |
|                   |
|                   |
|                   |
+------------------+
```

```
MPACK

MPACK
Returns unused areas in the extract data storage pool to allow use by other programs.

MRUN

MRUN
Execute program until next event. (Use only within a macro.)

MSG

MSG
Sets the next message.

MSGID (CMS and z/OS)

MSGID
Toggles the display of the message identifier.
MSGMODE

Displays status and informational messages when various IDF commands have been issued via PF keys.

MSTATUS

Displays extract data memory status:
- number of compile areas
- extract data storage consumption (total, direct, pooled)
- extract data storage pool utilization, including number of AREAs in the pool which are unused

MSTEP

Executes the next program instruction. (Use only within a macro.)

NAMES

- If name patterns are specified, displays the symbol names associated with those patterns.
- Otherwise, displays all symbol names.
NAMES

- All eligible symbols are shown:
  - within the current extract file with valid scoping
  - within the External Symbols List for other extract files which are loaded

- special pattern match meta-characters:
  - ?  matches a single arbitrary character
  - %  matches zero or more arbitrary characters
  - \  A backslash (\) followed by any character matches that character.
    Most useful when you need to match a real "?", "%", or "\".

NEXT
Scrolls a window forward.

OFFSET
Sets or queries the current offset.

OPEN
Opens a window.
OPTIONS

OPTIONS
Toggles the options window.

ORDER
Makes a window the first displayed.

OREGS
Toggles the old registers window.

PACKED
Selects the default VAR display format for Packed Decimal variables.
PARMS

Displays the Parameter List for module names.

PAUSE

Delays the execution of IDF for a number of seconds.

PER (CMS only)

Enables or disables PER.

PFK

Assigns a command to a PF key.
PFKDISP

PFKDISP
Toggles the display of the PF keys settings.

PLOCATES
Displays Pointer Locates information for variables.

PLOCATES is a command alias which is active after ASMLANG has been activated.

PRESERVE
Saves LSM Options and Settings in a 32 element stack.

PREVIOUS
Scrolls a window backward.
PROGCK (CMS only)

PROGCK (CMS only)
Simulates program check "nn".

```
PROGck check-code
```

PSW
A synonym of the GOTO command.

PSWSTEAL (CMS only)
Declares a PSW "stealing" location.

```
PSWSTEAL address
```

QUALIFY
Sets the currently qualified module.

```
QUALify default-module-name
```

QUIET
Disables or enables display of informational messages.

```
QUIET ON OFF
```
QUIETLY

QUIETLY
Temporarily suppresses the display of I, W and E messages during execution of a command.

QUIT
Returns to z/OS, TSO, CMS, or z/VSE.

RCQUIT
Returns to z/OS, TSO, CMS, or z/VSE with a return code.

REFRESH
Refreshes windows.
REGS

REGS
Toggles the Register display.

REGS64 (z/OS only)
Toggles the Current Registers and Old Registers windows between displaying 31-bit and 64-bit registers.

REGSTOPS (CMS only)
A synonym of the ADSTOPS command.

RESTORE
Restores LSM Options and Settings from a 32 element stack.

RETRIEVE
Puts the previous command in the command area.
**RIGHT**

Scrolls a window to the right.

```
RIGHT [window] [number-of-columns]
```

**RLOG**

Executes commands stored in the command log.

```
RLog [search-string]
```

**RUN**

Runs the program until the next event.

```
RUN
```

**RUNEXIT**

Executes the current exit routine (PFKey).

```
RUNExit
```

**R0-R15**

Sets a General Purpose register.

```
Rn-expression
```
SALIMIT

SALIMIT
Sets the maximum Program Caller hierarchy depth for the CALLERS command.

```
+++SALimit--max-caller-display-depth---
```

The default value is 100, the range is 1 to 999999.

SAREGS

Enables or disables the display of Save Area header and registers for the CALLERS command.

```
+++SAREgs--ON--OFF---
```

SAVE

A synonym of the PRESERVE command.

SEARCH

Searches for a string in storage.

```
+++SEARCH--window--string---
```

SELFINUCX (CMS only)

Sets the offset in module of nucleus extension.

```
+++SELFINucx--SYMbol--symbol-name--VALUE--start-offset-value---
```
SET ADSTOP (CMS only)

Sets or clears one end of a PER ADSTOP range.

```
>>> SET ADSTOP ON expression OFF
```

SET AREG

Sets ARn.

```
>>> SET AREG access-register-number expression
```

SET BREAK

Sets or clears a breakpoint at an address.

```
>>> SET BREAK ON OFF address command
```

SET COMMAND

Places text on the command line.

```
>>> SET COMMAND text
```
SET EXITEXEC

IDF-exit-exec is the current exit.

SET GLOBAL STEM

Writes data in a REXX stemmed array.

SET GLOBAL TEXT

Sets the IDF global area to text.

SET ICOUNT

Sets the instructions counted.
## SET OFFSET

Toggles the display of addresses using offsets.

```
>> SET OFFSET [ON | OFF]
```

## SET OPTION

Enables or disables an IDF option.

```
>> SET OPTION [ON | OFF] [option] (1)
```

### Notes:

1. This alternative form may be used only when it does not conflict with other command names.

## SET PSW

Sets the current PSW to a value.

```
>> SET PSW [hex]
```

## SET REGSTOP (CMS only)

Toggles PER monitoring of General Purpose Register (GPR) contents.

```
>> SET REGSTOP [ON | OFF] [general-purpose-register-number]
```
SET SIZE

Sets the size of the program.

SHOW

Controls source code and disassembly display, by showing information. The HIDE command controls the display by hiding information.

Notes:
1 An option can be chosen no more than once.

BOTH Show both source code and disassembly
SOURCE Show source code only
separator A comma, blank, or semicolon
COMMENTS Show block comment source code
DECLARES | DCL Show declaration source code
LINE Show source line number with source text
MACROS Show macro expansion source code
SHOW

NOCODE
   Show source lines with no corresponding object code

STATEMENT | STMT
   Show source statement number with source text

ALL | *
   Show all source code and disassembly

DISASM
   Show disassembly only

Initial settings: BOTH, COMMENTS, DCL, MACROS, NOCODE, STMT

SIZE

Resizes a window.

Location:

Location: row + column

+ +

STANdard

STD
**SKIPSTEP**

Sets a subroutine to be skipped.

```plaintext
>>> SKIPstep address
```

**SPACE**

Toggles the insertion of a blank line between variables or sets of components.

```plaintext
>>> SPACE ON OFF
```

**STATUS**

Toggles the program status window.

```plaintext
>>> STATUS
```

**STEP**

Steps to the next instruction in the program.

```plaintext
>>> STEP
```

**STMTSTEP**

Steps to the next statement in the target program.

```plaintext
>>> STMTstep
```
STOKEY

Displays the Storage Key.

\[ \text{STOKEY} \quad \text{address} \]

STOREMAP

Displays information about storage allocation.

\[ \text{STOREmap} \quad \text{address} \]

STRUCTURE

Enables variable display in a structure format.

\[ \text{STRUCTure} \quad \text{window} \quad \text{variable-name} \]

SUBSET (CMS only)

Enters a CMS Subset.

\[ \text{SUBset} \]
SVC (CMS only)

Monitors SVCs.

SWAP

Displays the application screen.

SYMBOL

Adds a symbol to the symbol table.

TASKS (TSO only)

Display information about currently executing tasks
TITLE

Sets the value of the title text.

\[ \text{TITLE} \quad \text{window} \quad \text{title-text} \]

TOP

Displays source code at the lowest address within the current code section.

\[ \text{TOP} \quad \text{window} \]

TRIGGER LOAD

Installs deferred breakpoints in a loaded module.

\[ \text{TRIGGER-LOAD} \quad \text{module-name} \]

TYPE

Displays type attributes for variables.

\[ \text{TYPE} \quad \text{window} \quad \text{variable-name} \]

UNION

A synonym of the STRUCTURE command.
UNTIL

UNTIL
Executes a program until an address (not including the address).

/SM590000/SM590000 UNTil/SM590000/SM630000

UP
This is a synonym of the PREVIOUS command.

VALUE
Evaluates an expression and displays it.

/SM590000/SM590000 VALue/SM590000/SM630000

VARIABLE
Enables variable display.

/SM590000/SM590000 VARIABLEwindow/SM590000/SM630000

VCHANGE
Logs commands (special purpose).

/SM590000/SM590000 VChange/SM590000/SM630000
VERSION

Displays the IDF Version.

```plaintext
>>> VERSION
```

VS

Special command logging.

```plaintext
>>> VS
```

VSEP

Enables or disables the blank line separating multiple variables.

```plaintext
>>> VSEP ON
OFF
```

WATCH

Sets a "watchpoint" condition that must be true before a particular breakpoint takes effect.

```plaintext
>>> WATCH
```

- **address**: Address of the breakpoint.
- **comparator**: The condition being checked, for example = or LT.
WATCH

_instruction_
An S/370 comparison instruction.

_command_
A command that is issued when the breakpoint is taken.

WHERE
Displays the symbolic name for an address.

/XEDEXIT (CMS only)
Xedit exit macro (EXIT ASM).

/ZONED
Selects the default VAR display format for Zoned Decimal variables.
Chapter 3. ASMDIF EXTRACT Command

ADSTOPs (CMS only)
All storage modification stops.

```
>---EXTract-ADSTOPs
```

Sets ADSTOP.n

ALET
The ALET used to qualify the dataspace to be displayed in a Dump window.

```
>---EXTract-LOCATION-ALET-alet-number-of-bytes-start-address
```

Sets ALET

AREGS
The Access Registers

```
>---EXTract-AREGs
```

Sets AR.n, OAR.n

ARGUMENT
An address argument from the command line or cursor position.

```
>---EXTract-ARGUMENT-ARGs-argument
```

© Copyright IBM Corp. 1995, 2008

57
ARRAY

Sets SOURCE, FIELD, EXACT, INDIRECT

ARRAY

Returns information about array elements.

```
EXTract—ARRAY—array-element-name
```

Sets stemname.0, stemname.n

BREAK

One or all breakpoints.

```
EXTract—BREAK—breakpoint-address
```

Sets BREAK.n, PBREAK.n

CALLERS

Returns information for each generation in the program caller hierarchy.

```
EXTract—CALLers—program-caller-generation
```

Sets stemname.0, stemname.n
CMDMSG

Contents of the command line and message lines.

```
$ EXTract CMDMsg
```

Sets COMMAND, MSG1, MSG2

COLORS

Color settings.

```
$ EXTract COLORS
```

Sets COLORS

CURSOR

Current position of the cursor.

```
$ EXTract CURsor
```

Sets DISPLAY, SOURCE, FIELD, EXACT, INDIRECT, HEXCURSR, CPDISASM, CPDUMP, NPDISASM, NPDUMP

DISASM

Data about an instruction.

```
$ EXTract DISasm instruction-address
```

Sets INSTR, NINSTR, CSECT
EVENT

Data about the last event

```
EVENT
```

Sets EVENT, COMMAND

EXITEXEC

The name of the currently assigned exit routine.

```
EXITEXEC
```

Sets EXITEXEC

GLOBAL

Return the current setting of the ASMIDF global variable.

```
GLOBAL
```

Sets GLOBAL

GLOBAL STEM

Return the data of the Global Storage stems.

```
GLOBAL stem-name
```
GLOBAL STEMS

GLOBAL STEMS
Return the names of all currently defined Global Storage stems.

```plaintext
>>EXTract—GLOBAL—STEMs
```

Sets GLOBALS.0, GLOBALS.n

GSTATUS
Returns information about the storage used to contain the Global Storage data.

```plaintext
>>EXTract—GSTatus
```

ICOUNT
Number of instructions executed since the last ICOUNT command.

```plaintext
>>EXTract—ICOunt
```

Sets ICOUNT

LANGUAGE ARGUMENTS
Returns the current command arguments for each LSM information window

```plaintext
>>EXTract—LANGuage ARGs
```

Arguments
### LANGUAGE COMMANDS

**LANGUAGE COMMANDS**

Returns the current command for each LSM information window

```
>>>EXTract-LANGuage-Commands-CMDs
```

### LANGUAGE OPTIONS

**LANGUAGE OPTIONS**

Returns information about the current value of the various ASMIDF Language Support settings.

```
>>>EXTract-LANGuage-OPTions
```

### LANGUAGE STATUS

**LANGUAGE STATUS**

Returns information about the extract files that have been loaded

```
>>>EXTract-LANGuage-STAtus
```

### LANGUAGE STEM

**LANGUAGE STEM**

Returns the name of the REXX stemmed variable array

```
>>>EXTract-LANGuage-STEM
```
LANGUAGE VERSION

Returns the ASMIDF Language Support version

`=>-EXTract—LANguage—VERsion—`-

LASTMSG

Returns the last ten messages issued by SET MSG

`=>-EXTract—LASTMsg—`-

Returns LASTMSG.n and LASTMSGM.n

LOAD

Obtain information about the target program

`=>-EXTract—LOAD—`-

Sets NAME, AREA, SYMBOL, ORIGIN, EPOFFSET, OFFSET, SIZE, LSM, LOADLIB

LOCATION

Extracts bytes of main memory

`=>-EXTract—LOCATION—number-of-bytes—start-address—`-

Sets MEMAREA
LOCATION ALET

LOCATION ALET
Storage from a dataspace

```
EXtract LOCATION ALET alet number-of-bytes start-address
```

Sets MEMAREA

MAP
Returns information about the location of all modules and code sections known to ASMI DF.

```
EXtract MAP
```

MODE (CMS only)
Current file mode

```
EXtract MODE
```

Sets MODE

MODULES
Information about defined modules

```
EXtract MODULES
```

Sets MODULES.n
MSTATUS

MSTATUS
Returns information about the storage used to contain extract data information

NAMES

NAMES
Returns information about symbol names.

OPTIONS

OPTIONS
A list of all ASMDIF options and their current settings.

PER (CMS only)

PER (CMS only)
Value of the PER setting.

Sets PER
PFK

PFK
Current PFK definitions.

PLIST
Arguments at the time of ASMIIDF invocation.

PLOCATES
Returns information about the variables that may be located with Locator (pointer) variables.

QUALIFY
Name of the currently qualified module

Sets PFK, PLIST, PLOCATES, QUALIFY
QUERY SETTING

QUERY SETTING
Returns the current value of an indicator or option item

\[\text{Query argument} \]

Sets QUERY.n

REGS
The GPRs, FPRs, and PSW

\[\text{Reg} \]

Sets GPR.n, OGPR.n, FPR.n, OFPR.n, PSW, OPSW, FPC

REGSTOPS (CMS only)
List of registers that are being monitoring.

\[\text{RegStops} \]

Sets GPR.0 - GPR.15

SCOPE
Returns information about the statement scope block that corresponds to a memory address.

\[\text{Scope address} \]
SCRVAR

Returns the contents of an LSM information window

```
>>> Extract-SCRvar
```

SELFNUCX

Current value of the self-load offset

```
>>> Extract-SELFNuCx
```

Sets SELFNUCX

SKIPSTEP

One or all currently skipped subroutines.

```
>>> Extract-SKIPstep
```

Sets SKIP.n

SOURCE

Returns the source records that correspond to a memory address

```
>>> Extract-SOUrce
```
STOREMAP

Return Storage Allocation Map information.

\[ \text{EXTract--STOREmap} \]
\[ \downarrow \text{addr-expr} \]

STRUCTURE

Returns information about structure and union components

\[ \text{EXTract--STRUCTure} \]
\[ \downarrow \text{component-name} \]

SVC (CMS only)

Current SVC tracing state.

\[ \text{EXTract--SVC} \]

Sets SVC

SYMBOLS

Information about symbols known to ASMDIF

\[ \text{EXTract--SYMBOLS} \]
\[ \downarrow \text{module-name} \]

Sets SYMBOL_n
TASKS

Returns information about the currently executing tasks

```
>>EXTTRACT-TASKS
```

TYPE

Returns information about the type attributes for variables.

```
>>EXTTRACT-TYPE variable-name
```

VALUE

Value of an expression.

```
>>EXTTRACT-VALUE-address
```

Sets EXPR

VARIABLE

Returns information about variables

```
>>EXTTRACT-VARIABLE variable-name
```
VDECLARE

Returns attribute information about variables

VERSION

ASMDIF Version message

Sets VERSION

VLOC

Returns location information about variables

VVALUE

Returns data value information about variables
Information about the screen and open windows

Sets WINDOW.n
Chapter 4. ASMDIF Options

Options may be turned on by:

<option> ON
SET <option> ON
SET OPT ON <option>

Options may be turned off by:

<option> OFF
SET <option> OFF
SET OPT OFF <option>

1ADSTop (CMS)  When PER is enabled, treats the four address ranges as a single address range.

AMODE24       Forces the target program to run in AMODE-24.

AMODE31       Forces the target program to run in AMODE-31.

AMODE64 (z/OS) Forces the target program to run in AMODE-64.

ASCII         Displays a dump in ASCII.

AUTOLOAD      ASMDIF should/should not automatically try to load LSM extract files when Statement stepping.

AUTOSize      ASMDIF should/should not resize window.

BCX           Displays branches in extended mnemonics.

CKSubcm       Insures ASMDIF’s Subcom is valid before running macros.

CMDLog        Logs user entered commands.

CMPExit       Indicates Exit is written in compiled code.

COLORS mhti   msg/head/text/input
              Blue/Green/Pink/Red/Turquoise/Yellow/White

COMMAND       PLIST for target is actually a command to invoke.

DMS0 (CMS)    Loads symbols that start with "DMS0".

EXITexec execname
              Specifies the name of the EXIT EXEC that should be used to determine breakpoint applicability.

FASTPath      Uses fast version of PATH.

FULLQual      Symbolic addresses should always be fully qualified.
ASMIDF Options

HEXDisp Displays offset in hexadecimal.
HEXInput Numbers without explicit base are hexadecimal.
IMPMacro Permits implicit macros from command line.
INVPsw Accepts invalid PSWs on a SET PSW command.
ISA address (CMS) Defines the address of a 16-byte double-aligned interrupt save area.
LIBE fn/$ (CMS and z/OS) Loads from specified DDname.
LINE X'num' (CMS) Uses a terminal other than the virtual console.
LSMDebug Displays LSM debugging information.
LUname lu_unit (z/VSE and z/OS) Defines the VTAM logical unit name of the terminal used by IDF.
MACROLog Logs commands entered from macros.
MODE xx (CMS) The CMDLOG and PATHDATA files are read from, or written to, the minidisk at the specified filemode.
MODMap (CMS) Uses fn MAP before LOAD MAP for symbol information.
NOAUTOLd Do not automatically try to load LSM extract files when Statement stepping.
NOAUTOSz Do not automatically resize windows.
NOBcx Do not display branches in extended mnemonics.
NODSects Do not load symbols in DSECTs.
NOIMPMac Disallows the implied execution of macros from the command line.
NOINVPsw Does not accept invalid PSWs on a SET PSW command.
NOMODMap (CMS) Prefers the "LOAD MAP" file to the "modname MAP" file.
NOProfil Do not run a profile macro.
NOSTOPNp Do not put internal breakpoints at a NOP(R) after a BAL(R).
NOSTOPSt Do not stop statement stepping when not in a statement.
ASMIDF Options

NOSVC97 (z/OS)  Do not use SVC 97 for events.

NUCext (CMS)  Runs the program as a CMS nucleus extension.

OFFSet  Displays address in offset format.

OLDBREAK  Uses the old operation of the Break command.

PA5spgm  Passes program interrupts to the target.

PATH  Displays the number of times each instruction has executed.

PATHFile  Writes the number of times each instruction has executed to a file.

PROfile name  Runs REXX procedure ‘name’ as the profile.

QWDump  Forces unformatted Dump display to begin on a fullword.

RISk  Ignores as many “errors” as possible.

RLog  Replays all previously logged user commands.

ROWstyle  Uses row style for display of registers.

SBORDer  Uses simple border characters.

SCDactiv  Collapses ASMIDF Subcom before running target.

SELFNucx symbol (CMS)  The code is self-nucxloading.

STOPNOP  Places internal breakpoints at a NOP(R) after a BAL(R).

STOPStmt  Stops statement stepping when not in a statement.

SVC97 (TSO)  Uses SVC 97 for events.

SWAp  Enables the capture of a target program’s screen image.

SY Stem (CMS)  Runs the program in system key (key=0).

TRACeall  All instructions are traced in single stepped mode.

TRANs (CMS)  Runs the program as a transient.

UNFIdump  Displays Dump in unformatted mode.
<table>
<thead>
<tr>
<th>ASMIDF Options</th>
</tr>
</thead>
</table>
Chapter 5. ASMDIF Language Support

Some examples in this section use the < > characters as follows:

< item >
   Item (such as parameter or word) is optional

->
   Represents the based-pointer notation

Introduction

ASMLANG is a Language Support Module (LSM) subsystem which acts as an extension to ASMDIF and provides source-level debugging capabilities for assembler programs. ASMLANG uses extract files which contain the language source and variable information. The extract files are created by the ASMLANGX utility using the SYSADATA files provided by the IBM High Level Assembler (HLASM).

A word about variables

Information for all variables in the user’s program is extracted into a common format by ASMLANGX.

ASMLANG displays variables using terminology similar to PL/I. Where necessary, extensions have been made - for example, FLOAT, PACKED DECIMAL and ZONED DECIMAL are terms used by ASMLANGX.

Invocation

ASMLANG is integrated with the base debugger module, ASMDIF, and the LSM support is activated during ASMDIF initialization.

To explicitly load ASMLANG extract files from the ASMDIF command line or via a macro, you can issue the following command:

  LOAD LANGUAGE efn eft efm (options

To implicitly load ASMLANG extract files you can use an LSM command such as STMTSTEP.

Parameters:

  EFN    Extract file name
     • On z/OS, the PDS member name of the extract file created by ASMLANGX.
     • On CMS, the file name of the extract file created by ASMLANGX.
Invocation

- On z/VSE, the file name of the extract file created by ASMLANGX.

EFT Extract file type
- On z/OS, the DD name allocated to the extract file created by ASMLANGX.
- On CMS, the file type of the extract file created by ASMLANGX.
- On z/VSE, not used.
- Specifying this option eliminates the search using the XPATH file types (DD names).
- The default XPATH is "ASMLANGX".

EFM Optional
- On CMS, the FM of the extract file created by ASMLANGX.
- On z/OS, not used, ignored if specified.
- On z/VSE, not used, ignored if specified.

Options

MODULE modname
Module with which to associate the extract file.

If this option is not specified:
- If extract file contains information which requires load-time resolution it defaults to the qualified target module.
- Otherwise, the extract file is "generic" where it is freely associated with any relevant CSECTs in all MODULEs.

Displaying source

Use the ASMIDF DISASM command:

```
DISASM (module.csect)stmt#nnn
DISASM 0(PSW)
```

Displaying variables

Use the VAR command:

```
VAR var
```

Multiple variables may be displayed:

```
VAR var1<;var2<;var3;...;varn>>
```

Locating expressions may be used:

```
VAR ptr1->ptr2->based_var
```
Displaying variables

VAR array_var(1,3,4)
VAR struct_var.member[1,3]
VAR triglips.too??(1,3??)

Substring specification may be used:
VAR chrstr(1);chrstr(2:3);chrstr(1::4)
VAR cstring[0::4]

ADDR() function can be used:
VAR Addr(buff(1))->based_var

Displaying structures

Use the STRUCT command:
STR strname
STR strname.substructure

The expression syntax is the same as for the VAR command.

Displaying array elements

Use the ARRAY command:
Arr arrayvar1(2);array2[3,55]

The expression syntax is the same as for the VAR command.

When the display is scrolled, the array indexing is also scrolled.

Altering variables

1. Display variables using any of the previous commands.
2. Type over the current variable contents
3. Press the Enter key

Displaying type attributes

Use the TYPE command:
TYPE var

Type attribute information includes:
• Fundamental data type
• User defined data type
• Type hierarchy

Type attributes for multiple variables may be displayed:
Displaying type attributes

TYPE var1<;var2<;var3;...;varn>>

LANGUAGE command aliases

With ASMIDF a facility called "Command Alias" is available which allows ASMLANG to add additional commands without the LANGUAGE prefix.

Command Alias is an ASMIDF facility that enables:

<table>
<thead>
<tr>
<th>Alias</th>
<th>Equivalent ASMLANG command</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>LANGUAGE LOCATE /</td>
</tr>
<tr>
<td>-=/</td>
<td>LANGUAGE LOCATE -=/</td>
</tr>
<tr>
<td>BOTtom</td>
<td>LANGUAGE BOTTOM</td>
</tr>
<tr>
<td>CALLers</td>
<td>LANGUAGE CALLERS</td>
</tr>
<tr>
<td>F</td>
<td>LANGUAGE FIND</td>
</tr>
<tr>
<td>FIRst</td>
<td>LANGUAGE FIRST</td>
</tr>
<tr>
<td>GOTo</td>
<td>PSW</td>
</tr>
<tr>
<td>HIDe</td>
<td>LANGUAGE HIDE</td>
</tr>
<tr>
<td>LASt</td>
<td>LANGUAGE LAST</td>
</tr>
<tr>
<td>LLocate</td>
<td>LANGUAGE LOCATE</td>
</tr>
<tr>
<td>Locate</td>
<td>LANGUAGE LOCATE</td>
</tr>
<tr>
<td>MAP</td>
<td>LANGUAGE MAP</td>
</tr>
<tr>
<td>MPAck</td>
<td>LANGUAGE MPACK</td>
</tr>
<tr>
<td>MStatus</td>
<td>LANGUAGE MSTATUS</td>
</tr>
<tr>
<td>NAMes</td>
<td>LANGUAGE NAMES</td>
</tr>
<tr>
<td>PARms</td>
<td>LANGUAGE PARMS</td>
</tr>
<tr>
<td>PLocates</td>
<td>LANGUAGE PLOCATES</td>
</tr>
<tr>
<td>SHOw</td>
<td>LANGUAGE SHOW</td>
</tr>
<tr>
<td>TOP</td>
<td>LANGUAGE TOP</td>
</tr>
<tr>
<td>TYPE</td>
<td>LANGUAGE TYPE</td>
</tr>
<tr>
<td>UNIon</td>
<td>STRuct</td>
</tr>
</tbody>
</table>

Hints and tips

1. Maximum LSM window display lines
   When displaying multiple variables use the LANGUAGE VSEP OFF command. This increases the number of screen lines available to display the variable information.

2. z/OS Dataset Conventions
   On z/OS, ASMIDF and ASMLANGX commands do not change. The CMS conventions are used, with the following mapping of the CMS file conventions to z/OS:

   CMS  | z/OS Equivalent
   ----|------------------
   fn   | PDS member name (ignored if using sequential file)
   ft   | DDNAME, which in turn points to the z/OS dataset name
Hints and tips

fm not used on z/OS

You must allocate the DDs using ALLOC (CLIST or EXEC) or DD (JCL).

3. z/VSE Dataset Conventions

On z/VSE, ASMIDF and ASMLANGX are invoked from JCL, with the following mapping of the CMS file conventions to z/VSE:

<table>
<thead>
<tr>
<th>CMS</th>
<th>z/VSE Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>z/VSE librarian member name.</td>
</tr>
<tr>
<td>ft</td>
<td>DLBL name, which in turn points to the z/VSE file name.</td>
</tr>
<tr>
<td>fm</td>
<td>not used on z/VSE</td>
</tr>
</tbody>
</table>
Hints and tips
Chapter 6. Using ASMLANGX

Invocation

ASMLANGX CMS and TSO syntax

```
//ASMLANGX--file-name

//option
```

ASMLANGX z/OS EXEC syntax

```
//stepname--EXEC--PGM=ASMLANGX--,PARM='file-name

//option
```

ASMLANGX z/VSE EXEC syntax

```
//EXEC--ASMLANGX--,PARM='output-file-name

//option
```

Parameters:

**input_file_name**

Input file name
- On TSO, the PDS member name of the SYSADATA input file.
- On CMS, the file name (FN) of the SYSADATA input file.
- On z/OS in batch, this defaults to the SYSADATA file created by the High Level Assembler.
Invocation

- On z/VSE, this defaults to the SYSADATA file created by the High Level Assembler.
- Dummy token required only if the extract file name is to be modified.

output_file_name

Output file name.
- On TSO, or CMS, this is optional and defaults to the same name as the input file with a file type (CMS) of ASMLANGX, or the PDS name of the ASMLANGX DD (TSO) name.
- On z/OS, this must be specified with PARM='output_file_name'.
- On z/VSE, this must be specified with PARM='output_file_name'.

Options

General Options:

ASM Extract is for Assembler.
CONDASM Include conditional assembly statements.
DCL Suppress source for declarations. This is the default option.
DEBUG Log standard and internal diagnostic messages.
To be used as directed by IBM service personnel.
ERROR List invalid or incomplete extract records.

IFM filemode

input file mode
- On CMS, the file mode to search for the input files. Standard search order used if not found.
- On z/OS, not used.
- On z/VSE, not used.

INCL Extract source from INCLUDE files. This is the default option.
LOUD Issue progress or error messages.
MACDEF Include inline macro definitions.
NOCONDASM Exclude conditional assembly statements.
Options

NOINCL
Suppress source from INCLUDE files. Variable information is still extracted.

NOMACDEF
Exclude inline macro definitions.

NOPACK
Disables packing of source statement text

NODCL
Suppress source for declarations (including associated block comments). Variable information is still extracted.

NOSEQ
Suppress source record sequence numbers.

OFM filemode
Output file mode
  • On CMS, the file mode of the output file, default “A1”.
  • On z/OS, not used.
  • On z/VSE, not used.

OFN filename
Output file name
  • On CMS, the file name of the output file.
  • On z/OS, the PDS member name of the output file.
  • On z/VSE, the librarian member name of the output member.

OFT filetype
Output file type
  • On CMS, the file type of the output file.
  • On z/OS, the DD name of the output file.
  • On z/VSE, not used.

PACK Compress redundant characters in source statement text

PFM filemode
Primary input file mode
  • On CMS, the initial file mode to search for the primary input file.
    Standard search order used if not found. Overrides IFM
  • On z/OS, not used.
  • On z/VSE, not used.

PFT filetype
Primary input file type
  • On CMS, the file type (FT) of the input file.
  • On z/OS, the DD name of the input file.
  • On z/VSE, the DLBL name of the input file.

QUIET Suppress display of progress and error messages.
Options

SEQ Retain source record sequence numbers.

Default Options:
• ASM
• PFT SYSADATA
• OFT ASMLANGX
• PACK
• NOSEQ
• DCL
• INCL

Examples
Here is a sample CMS ASMLANGX invocation.
• using “fn SYSADATA”

```
ASMLANGX fn (ASM LOUD ERROR
```
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer 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.

Licensees of this program who wish 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:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie New York 12601-5400
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 document 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.

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
Notices

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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

IBM  S/370
The IBM Logo  VTAM
ibm.com  z/OS
OS/390  z/VM
REXX  z/VSE
Readers’ Comments — We’d Like to Hear from You

High Level Assembler for z/OS & z/VM & z/VSE
IDF Reference Summary
Release 6

Publication No. GC26-8712-05

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your 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. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.
Send your comments to the address on the reverse side of this form.
If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

E-mail address
Readers’ Comments — We’d Like to Hear from You
GC26-8712-05

Please do not staple

Fold and Tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS MAIL   PERMIT NO. 40   ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE
IBM Corporation
Reader Comments
DTX/E269
555 Bailey Avenue
San Jose, CA
U.S.A. 95141-9989