IBM Systems MEDIA

IBM z13 Configuration Best Practices

If you want to take advantage of everything the mainframe offers, and do so in the most efficient way possible, you may need to think a bit more about how you're configuring your images.

By Kevin McKenzie

08/07/2017

In 1965, a year after the System/360 was introduced, Gordon Moore coined Moore's Law, which says that every 18 months, the number of transistors that can fit in a given space doubles. And for 40 years, that meant that every 18 months, the speed of the processor would double. The computer industry was able to take advantage of that constant free capacity gain. But in the 2000s, that came to an end: Physical limits related to heat dispersion and electron behavior were being reached that meant that chips couldn't go any faster. Since then, IBM has been continuing to grow central electronics complex (CEC) capacity through the number of processors and their capabilities, and that's led to changes in the way systems behave at scale.

Scaling out, instead of just using the benefits of faster processors, has meant that we've had to make a lot of changes to the way both the hardware and software behave, and make them cooperate more closely. This has led to things like HiperDispatch, microcode and PR/SM dynamic allocation of processors.

The problem is that system programmers and capacity planners have come up with rules of thumb for how best

to allocate system resources and run systems, but following those rules can lead to performance inefficiencies when applied to modern systems. While IBM has been trying to inform clients of these differences, we haven't been able to reach everyone.

The two areas clients have the most problems are with how they allocate logical processors and with how busy they run their CECs. All of these recommendations are related to performance, not to stability. You are free to configure your CECs and LPARs however you choose; just realize that some configurations may lead to a less efficient system (i.e., one that takes more CPU time to perform a function than it would in more optimal configuration). These recommendations are not IBM z13 specific, but not following these recommendations can lead to more overhead on the z13 than prior processors.

Let's look at each of these challenges and solutions to them.

Logical Processors

After PR/SM was first introduced, people started worrying about LPAR overhead, the amount of time it takes PR/SM to manage access to system resources. A rule of thumb developed of not having more than a 2:1 logical CPU to physical CPU ratio. Then IBM introduced Workload Manager (WLM) Vary CPU Management, which saw PR/SM and WLM cooperating to take logical processors offline if the amount of overhead they were causing was greater than the performance benefit they provided. At that point, some people decided that the prior 2:1 logical to physical ratio rule of thumb no longer applied, and began recommending much higher ratios once WLM Vary CPU Management was turned on.

IBM next introduced HiperDispatch, which has some similarities to WLM Vary CPU Management, but some important differences in the way excess CPUs are managed. But some clients are still running essentially the same way they did in 2004, prior to HiperDispatch, with many systems having access to many more logical CPs than they are entitled to by their weight.

Last year, IBM's Washington Systems Center released a Technote (http://ibm.biz/TD106388) that provides new best practices for defining logical central processors (CP) to an LPAR. The recommendation is to define the number of logical processors based on their processor share (i.e., the number of processors they're entitled to by weight), and add one or two for expansion in case of a surge of work. The Technote goes into a number of reasons for the recommendation. One reason that's z13 specific: with the z13, PR/SM clusters processors and system memory as close together as possible, and changes the location of processors and memory as needed in response to LPAR and CEC configuration changes. When you define an LPAR that has a lot more logical CPs than it's entitled to, it makes it more difficult for PR/SM to figure out the most efficient CEC topology.

System Utilization

One of the design points for IBM Z and z/OS has always been balance between CPU performance, I/O capabilities and access to memory. This means that, unlike other platforms, you can run z/OS at very high CPU and I/O utilizations and see a near-linear scaling of performance. If you can do N transactions with 25 percent of the processor capacity, you can do 2N transactions with about 50 percent of the capacity. So clients have, for many years, run their CECs and LPARs at close to 100 percent busy for very long periods of time to extract the maximum value from the platform. From a high-availability point of view, running at 100 percent for extended periods of time has never been recommended because it doesn't leave room for temporary workload spikes, or for any sort of recovery processing that might need to be done in the event of a subsystem failure.

However, as the CPUs have gotten more complex, we've added things like branch prediction, speculative execution, transactional execution, pipelining, simultaneous multithreading and millicode. And, while the

processor, memory and I/O subsystem have all gotten faster, the processor improvements have outpaced the memory and I/O improvements. Relatively speaking, memory and I/O interrupts are more problematic than they used to be.

Modern processors are no longer as predictable as they were. With prior generations of processors, any given instruction took a known, consistent number of machine cycles. With modern processors, that's not the case; how many machine cycles an instruction takes to execute can vary based on a host of factors. Two of them you can control: how well the program behaves and how busy the CEC is.

In terms of how a program behaves, for most programs that have been written recently, and/or are recompiled on a regular basis, you probably won't have any problems. However, there are certain things you should be aware of. An in-depth explanation is available in "IBM z Systems Processor Optimizer Primer (http://ibm.biz/zSystemsProcessorOptimizationPrimer)," but one of the things it's critical to be aware of is not storing near the instruction stream. IBM has been warning against mixing instructions and data for at least 15 years. With separate instruction and data caches, and modern processor design assumptions, such code can be exceptionally inefficient.

Another thing you can control is how busy the CEC is. Again, for availability reasons, IBM recommends against running a CEC 100 percent utilized on a regular basis, but there are also CPU time/transaction impacts to consider as well. Again, because CPUs have outpaced memory and I/O, cache misses take relatively longer to retrieve data for each successive processor generation, and as the CPU gets busier, cache misses are more and more likely. So transactions will take longer, in terms of time spent on the CPU, on a very busy CEC.

Tools

To plan your CEC configuration, IBM made available an LPAR Design tool (http://www-03.ibm.com/systems/z/os/zos/features/wlm/WLM_Further_Info_Tools.html#Design) that you can use. Enter information about the CEC and the LPARs defined on the CEC, and it will tell you how many high, medium and low processors of each type each LPAR will have. You can also use the IBM Topology Report tool (http://www-03.ibm.com/systems/z/os/zos/features/wlm/WLM_Further_Info_Tools.html#Topology) in conjunction with WLM SMF 99.14 records to see how your CEC is laid out at a given point in time.

With APAR OA48570, IBM introduced SMF 98 records that contain frequently collected, low-level dispatching data indicating system efficiency and its influencers. This is in contrast to most other SMF data, which is mostly summary data over a long reporting intervals. With SMF98 data, you can see information about how efficiently your LPARs are running, and what is influencing that efficiency. There's a demo (http://ibm.biz/try-zoi) that shows the sort of information you can get from SMF98s. Most usefully, IBM developers have been looking at the data to identify patterns that correlate with low system efficiency, and can highlight those areas of time and what can be done to improve system efficiency (e.g., reducing the number of initiators or threads, reducing the use of vertical low CPs or identifying programs that are causing high rates of locking).

Configuration Advantage

This isn't the first time that the mainframe has gone through a transition like this; in the 1990s, IBM switched from using CPUs built using bipolar transistors to ones using CMOS technology. The reason the switch was made was because bipolar chips run much hotter than CMOS chips. However, CMOS chips were much slower than bipolar chips, so it took several years before CMOS machines had the same capacity as the bipolar machines they were replacing.

Unlike then, there isn't a new technology we can switch to; but also unlike then, the microarchitectural changes

are increasing the capacity of the mainframe dramatically. But if you want to take advantage of everything the mainframe offers, and do so in the most efficient way possible, you may need to think a bit more about how you're configuring your images.

Anne Romanowski contributed to the technical and editorial review of this article.

About the author

Kevin McKenzie has been with IBM for 18 years, all in test, with a focus on performance sensitive items.