WHEN IS ZIIP OFFLOAD NOT WHAT IT SEEMS?

Mini
mizing the operating expense of your mainframe has never had a higher profile, nor been more important. Through the introduction of specialty processors (such as zIIPs), IBM has provided

significantly lower cost hardware and the promise of dramatic savings in operational costs. The primary reason this is so attractive and cost-effective is not just the cheaper hardware, but also much cheaper software, since typically, processing capacity associated with zIIPs is not included in overall z/OS MIPS/MSUs capacity by IBM or ISVs, and is therefore exempt from many related charges such as license costs, upgrades and maintenance fees.

IBM authorizes customers to use specialty engines to process certain specific types of workloads as designated by IBM. For example, certain workloads that were written to run in non-task enclave SRB mode may be made eligible for redirection to a zIIP.

The choice for IT would seem obvious: maximize the use of specialty engines to reduce cost. However, the obstacle to IT achieving these savings is the pre-requisite that only certain types of processing are eligible to execute on zIIPs. To actually realize such savings, companies must identify and migrate workloads currently running on the mainframe's traditional general purpose (GP) processors to one of the less expensive zIIPs. One way to do this might be to rewrite historical COBOL code into JAVA – JAVA workloads will always be zIIP enabled (now that we can run zAAP work on a zIIP!).

BMC Software has a way for IT to take advantage of these cost benefits with no migration, no effort, and no risk. Most of BMC's tools running in the z/OS environment are now coded to take advantage of zIIPs wherever they are encountered.

Because of the way IBM licenses access to these zIIP engines, it is not possible to specifically direct activity to be performed on a zIIP – instead, workloads are made *zIIP-Eligible*. At execution time, IBM Workload Manager (WLM) makes real-time decisions about what can be executed on a zIIP and what must remain executing on a General Processor. Some of the reasons why zIIP eligible work may not run on a zIIP include:

- No zIIPs are installed and online
- The zIIPs that are installed and online are busy doing other work
- The application which has scheduled the work has specified that only a portion of the work should be made eligible for redirection to zIIP

Because of the importance of offload eligibility, it has become a habit to look at zIIP offload percentages (as measured by various benchmarks) to determine who is running the most efficient software in any particular situation. So a tool that boasts "80 percent zIIP offload" is deemed to be better than one that only offers "50 percent offload eligibility."

Unfortunately, this crude comparison overlooks the reasons for bothering with zIIP offloading in the first place. The point is not to maximize the work that is running on the zIIPs but to minimize the work that remains executing on the GPs. I know this sounds like two ways of saying the same thing, but let me explain:

- A tool that consumes 100 MSUs and has 80 percent zIIP offload may, at best, leave 20 (chargeable) MSUs running on the GP.
- An alternative tool that consumes 25 MSUs but only boasts 30 percent zIIP offload only leaves 17.5 chargeable MSUs on the GP.

You can easily see now, that the tool with the lower zIIP offload actually costs LESS to execute than the one with the higher offload. The question always needs to be asked *"What is the zIIP eligible offload a percentage of?"* – all zIIP offload figures are not created equal.

BMC takes very seriously the need to continually review execution costs, and zIIP offload is only one of the tools we bring to bear on the problem. We usually go through these steps in determining what we can do to minimize CPU consumption:

1. Is there a way of avoiding CPU usage completely?
   An example of this is the way Next Generation Technology Reorg removes the need to decompress and recompress data during a reorg and the fact that it avoids having to call a SORT utility to reorganize Db2 data (among a long list of things that NGT Reorg no longer has to do compared with a traditional reorg utility).
2. Are there more efficient ways of performing tasks?
   In our recovery tools, for example, we have looked at the instructions we use to move data around to make sure we are doing things in the most efficient way possible. In this context, not all instructions are created equal.
   We also continually look for ways to improve our algorithms - removing the need to do repetitive tasks multiple times for example.
3. Can we make any of the remaining CPU consumption zIIP eligible?
   Using a zIIP should be the last resort after all other options have been tried. After all, the cheapest CPU second is the one you don't consume.

Other vendors hope you will be impressed with their zIIP offload benchmarks and that you won't notice the sleight of hand where they are trying to hide the real execution costs behind a mythical "best offload percentage" number. There is far more to reducing chargeable CPU than just enabling some of it to run on a zIIP.

What I haven't mentioned so far is that zIIP capacity is not infinite – there is a defined ratio (by IBM) of how many zIIPs you can have based on the number of GPs that you purchased. This started as one zIIP per General Processor, but from July 2013 onwards, it is now possible to buy two zIIPs for each GP for zEC12 and/or zBC12 (or later).

For more recent hardware, it is also possible to run zAAP (z Systems Application Assist Processor) eligible workloads on a zIIP engine – this supports running Java and XML parsing workloads on a specialty engine.

Remember above where I mentioned that one reason not to run a zIIP eligible workload is unavailability of zIIP capacity? This is a very good reason to ensure that your tools are using zIIP offload in the most efficient way possible and are not just dumping unwanted GP cycles onto a zIIP. If you are running your zIIPs at, or close to, capacity, performance can suffer. Making code eligible to run on a zIIP and then not running it there, but on a GP instead, is actually more expensive (and slower) than if you hadn't bothered with zIIP eligibility in the first place!

At the end of the day, zIIP offloading is only one way of minimizing chargeable CPU usage and people need to look at the whole picture to determine who is delivering the best value in terms of chargeable CPU usage.

Oh, and one last point. If you are aiming to reduce CPU consumption to reduce your IBM MLC licensing charges, remember that only MSUs consumed during your rolling four-hour peak contribute to that calculation. Savings elsewhere may make you feel better, but they won't reduce your software bills. In the context of zIIP redirection, only those GP cycles offloaded to a zIIP during your four-hour peak will be saving you money.