Heterogeneous Computing Meets the Data Center

By Ron Wilson, Editor-in-Chief, Altera Corporation

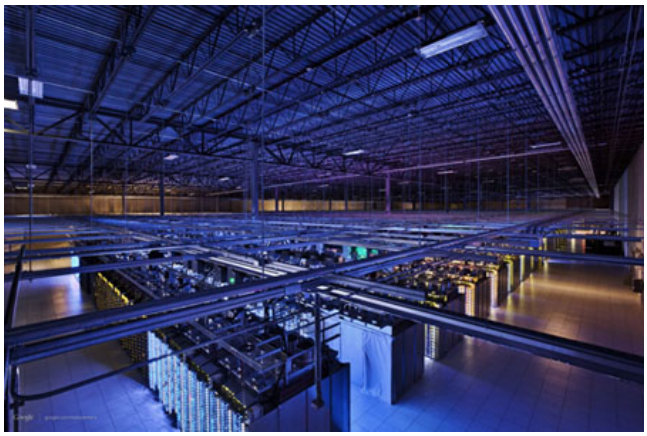
As we push single-CPU performance deeper into diminishing returns and begin extracting the last big gains from multicore processors, computing experts are turning to another old idea—heterogeneous multiprocessing—as the next way to advance. In heterogeneous systems, different kinds of processors—x86 CPUs and FPGAs, for example—cooperate on a computing task.

Heterogeneous architectures are already widely used in the mobile world, where ARM® CPUs, graphics processing units (GPUs), encryption engines, and digital signal processing (DSP) cores routinely work together, often on a single die. But what happens when these techniques arrive at that nexus of digital existence, the data center? A workshop at the 2014 International Symposium on FPGAs sought to find out.

The Orientation

To enter the data center is to step into a parallel universe—a dark, cavernous, mostly-uninhabited space in which normal notions of scale do not apply (Figure 1). The workshop's guide into this strange land was Microsoft Research extreme computing senior research engineer Andrew Putnam.

Figure 1. The data center is a huge, unfamiliar world for hardware accelerators. Photo by Connie Zhou (from Google)



“We don’t buy servers,” Putnam explained. “We buy tens of thousands of servers, by the rack.” Putnam said that the typical server is “a 1U, half-width card carrying four disk drives and two multicore CPU chips.” There is no left-over real estate on the card, or vertical clearance in the rack, for additional large chips. Nor is there much incentive to touch the cards once the racks take their places in the ranks and files of the data center floor.

“Servers fail often, but we rarely repair them,” Putnam said. Instead, failed cards just stay there, bypassed, in their racks. “So for the data center to continue operating, it is vital that the racks be fault-tolerant.”

One reason for this don’t-repair approach is the amazingly short life of a data center. “There is a new generation about every six months,” Putnam explained. “We will set one up and use it about 18 months. Then we will sell it to a different group running other applications. The total useful life is about three years.”

The requirement that the data center continue to operate without significant component repair for three years has significant implications for not only how reliable the hardware must be, but also how hardware accelerators can fit in. For example, it is tempting to cluster accelerators into an appliance. But if the appliance takes the space of eight server blades, or if the entire rack depends on the appliance being available, then the reliability requirements on the accelerator chips can skyrocket. So in a strange way—as everything in the data center can seem strange—maintenance strategy becomes a factor in accelerator hardware architecture.

The Incast Bottleneck

One point at which the need for acceleration has already outweighed the reliability question is where the racks connect to the data-center network. “We have a network incast bottleneck,” Putnam declared. Incast problems occur when, for example, a server requests data that is distributed across many other servers. If all those other servers respond at about the same time, the avalanche of returning packets can crush the original, requesting server’s network interface.

The need to streamline packet processing in such situations is one reason you find FPGAs in the network interfaces atop the racks, Putnam explained. And Microsoft, for one, is looking to go a lot further than just offloading network protocol stacks. “We are looking at a network of North-South-East-West 20 Gigabit (Gb) links between racks,” Putnam said.

Locating the Accelerator

The world Putnam describes can sound unwelcoming to hardware accelerators. Beyond their current station as network offload engines in the top of the server rack, it is difficult to see where accelerators fit into the picture. There is little room, cooling, or power budget to introduce yet another big, hot chip onto the blade. But there are, as we have seen, bandwidth and reliability issues with replacing a cluster of blades with a larger-form-factor accelerator drawer.

Some vendors, including Oracle, have attacked the problem at the rack level. “We tend to sell multiple-rack solutions, often for vertical applications—a cloud-in-a-box,” said Oracle Labs

technical director Eric Sedlar. “This allows us to optimize a system at rack level for a particular application. For example, we might sell a rack configured for the oil and gas exploration business.” And by isolating an application onto a few racks at most, it relieves the no-repair constraints that the racks would face in the middle of a giant data center.

Even so, Sedlar reported a fair amount of skepticism about putting accelerators next to CPUs on the blade, except in the case of GPUs. “We tend to use FPGA chips in clusters, not close to the compute engine,” he said.

One reason for this might be that Oracle Labs sees the function—and the location—of accelerators to be an application-specific question. What functions you need to accelerate—and, therefore, where you should place the accelerator—can depend on the structure of the problem. For example, Sedlar cited database acceleration. In fact some database operations lend themselves to hardware acceleration. But just how to accelerate can be a complex question.

“SQL domain-by aggregation is a useful example,” Sedlar offered. “The task can be DRAM-limited. Or it can be cache-, instruction-, or TLB-prefetch-bound. It depends on the cardinality, the particular data set, and the operations you are performing.”

The only constant, he says, is that if you execute the entire operation on CPUs, you will have to struggle against excessive data moves. Beyond that, just where you place the accelerator—in the rack, on the network, or in the storage subsystem—and what you ask it to do will be heavily influenced by the situation. As the data set or the task changes, you may want to rethink both location and function.

For example, in an aggregation operation that spans many disk drives, an accelerator at the drive or array level that can perform decompression, formatting, and selection of relevant fields as a flow-through process can make a huge improvement in the traffic load on the network, DRAM, and caches. But another type of database operation might require an entire data set to stream through the caches and be processed cooperatively by CPUs and FPGAs.

Acceleration at the CPU

Other speakers in the workshop provided a quite different point of view. If you are a system architect, acceleration is a system issue. If you are a CPU architect, it is a chip-level issue.

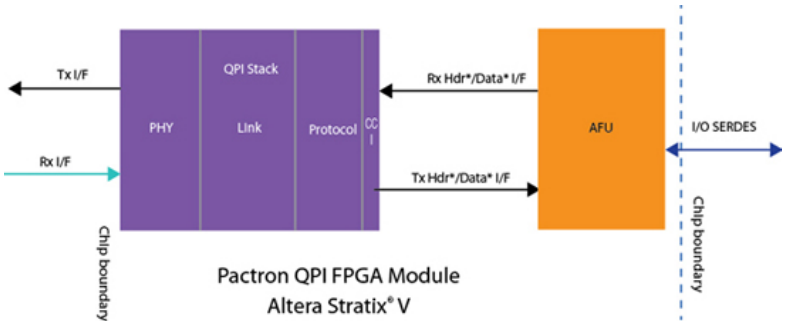
IBM Research chief scientist Peter Hofstee listed a number of IBM applications using accelerations in the form of FPGAs working directly with CPUs. They all share a common model, he explained.

“We start out with CPUs and accelerators sharing the same memory model,” Hofstee said. “And we make sure that any task we can do on the accelerator we can also do on the CPU. We always want the hardware to be optional. From those base assumptions we go on to work on locality and predictability.”

By increasing the task’s locality of reference, IBM tries to get everything the task needs as close as possible to the CPUs—ideally in cache—with as few moves as possible. This tends to decouple task latency from storage latencies. Then the developers can identify the operations with the least predictable execution times and move them onto faster, more deterministic FPGAs.

IBM believes so deeply in this approach that they have provided for it in the core of the Power 8 architecture, Hofstee said. The multicore, cache-rich CPU chip generates, in effect, a memory-coherent external extension of the internal cache bus, using PCI Express® (PCIe®) Gen3 electricals. This bus allows an FPGA with an appropriate bus controller to share the view of memory the CPUs have (Figure 2). Not only does this make cooperative processing much easier, but it allows moving tasks back and forth between CPUs and FPGAs. “That is important,” Hofstee observed, “because over time a lot of things will end up back on the CPU.”

Figure 2. A coherent cache-bus interface such as IBM CAPI or (here) Intel QPI can be implemented in an FPGA, in effect placing an FPGA-based accelerator on the CPU chip’s internal cache bus.



Intel’s view is similar to IBM’s. Intel director of accelerator technology PK Gupta said “Our motivation is to apply heterogeneous computing to get single-thread performance acceleration. To that end, we are using two sockets: a Xeon CPU next to a high-end FPGA, connected by our QuickPath Interconnect (QPI) coherent cache interface.”

Gupta explained that the coherent interface allowed the CPU software and the FPGA-based accelerators to share a virtual memory space. This results in a simpler programming model, and allows either in-line or offload acceleration.

Programming Models

One issue came up repeatedly during the workshop: how will data center accelerators appear to programmers? It is a complex and vital question.

It is complex because the programming environment in the data center is incredibly rich. “ Inside Google we have literally thousands of services running on the servers,” Ross said. “And we use about 30 languages in production. There will never be a time when we are down to just one. But software designers, if they have to use accelerators, want to at least start out in a familiar language.”

The question is vital as well. “Five years from now, no acceleration technology will be in the data center unless it seems approachable to software developers,” Microsoft’s Putnam declared. “Even then mostly it will be used within service calls, not explicitly programmed by software engineers.

“There will be critical pieces that we know we want to accelerate, like map-reduce and memcached,” Putnam added. Sedlar had also mentioned some database and Java functions in his world. “The most important thing,” Ross contributed, “is to present the problem as a software issue.”

Ross illustrated his point by fishing for an analogy. “There is a really dreadful-looking deep-sea creature called the Patagonian Toothfish,” he said. “It tastes wonderful, but you can imagine how far it would get with that name at the grocery store. But renamed as Chilean Sea Bass, it is a premium item. It’s the same idea here: offer software developers the Bass, not the Toothfish.”

The speakers explained, from their various contexts, the scope of Ross’s mandate. From a CPU developer’s point of view, keeping focus on software means providing a coherent, unified memory model so development tools can retarget tasks between CPUs and accelerators and allow the two to work cooperatively.

For rack-level system developers, software focus may mean providing a rich set of domain-specific libraries for frequently-used source languages—and yes, FORTRAN is still out there--so software developers can explicitly call functions that will execute on the accelerators. Equally important, it means providing source-level profiling and debug tools so application

developers and—just as critical—library developers can do their work in a familiar environment, protected from the complexities of the accelerator hardware or the details of how threads get mapped onto hardware.

For masters of the giant data centers, there is another layer of meaning that may seem quite foreign to others. For most system developers, the point of acceleration is to speed completion of a task. But as in so many other instances, once we enter the parallel universe of data centers, the obvious can be misleading.

“Tail latency is the most important issue,” Ross said. He explained that a data center, unlike a supercomputer or a domain-specific cloud-in-a-box, is always running a rich mix of tasks. There are always more batch jobs waiting the queue.

“One resource will always get used up first,” Ross explained. “Then batch jobs come in to soak up any resources that are left idle.” So from the data center manager’s point of view, the challenge is not to execute some specific job in minimum time. It is to come as close as possible to full utilization without missing quality-of-service goals on critical tasks.

From this point of view, the most desirable attribute is not raw speed: it is consistency. And that is where hardware accelerators can shine. “The data center components are unpredictable—because CPUs are unpredictable,” Ross said. “But FPGAs can be deterministic.”

The ability to identify tasks with large execution-time uncertainty, and then reduce the uncertainty by accelerating the key part of the task, meets the manager’s needs. Increasing the predictability of latencies makes load balancing easier and more effective. And it makes it possible to guarantee quality of service on critical tasks. Thus hardware acceleration can have major benefits for the data center even if the accelerated task is not significantly faster than the fastest possible execution time on a CPU. As IBM’s Hofstee had said earlier, sometimes it is predictability that counts.

From unusual notions of reliability to architectural questions, to new kinds of development tools, to emphasis on determinism rather than raw speed, hardware accelerators face a strange new world in the data center. As the need for heterogeneous multiprocessing in data centers grows, it is likely that all sorts of chips, from custom designs to GPUs to FPGAs, will find their way into data centers at various points.

At the same time, rapid evolution has already begun in the way these devices are being programmed, for example in the adoption of Open Computing Language (OpenCL™) for generating kernel accelerators on GPUs and FPGAs. These last changes perhaps have the largest significance for designers in the embedded world, who may find that they now have access to new levels of hardware acceleration through conventional programming languages—thanks to those strange, huge, dark buildings.

Read More System Design Articles | Subscribe to System Design Articles