Day 1 - Working Group 3: Processor Network Interface

DK Panda, Discussion Moderator

- Let us assume that MPI will be a programming model, along with others (PGAS models, for example)

Question – Are you assuming with MPI there could be other things on the node (e.g. OpenMP)?
Answer- yes.

- Let us assume: 32K sockets
- Let us assume: 8, 16-32 multiple cores/socket, 32, 64-128 cores/socket in 2015.
- Turning to the slide with system interconnects that Jeff Vetter presented this morning.

Tomkins – 1200M messages per second is per socket. The "want" column means you push hard to get things ahead (and it is what is needed to get each machine balanced and stable). The "expect" column is without extra work.
Comment – Perhaps "want" is more like "desired".
Camp – Take the NIC bandwidth row and the message throughput row – the latter I think you get to with scaling. If we have a TF processor 100 GB/s is an aggressive NIC (and no one is thinking about working in that category). I think that is aggressive in 2011.
Comment – You will not need it.

- The broad issues of performance, power, reliability, scalability and cost – I tried to summarize what I heard discussed earlier today in terms of designed architectural support in the NIC (point-to-point, fine grain point-to-point communication with low overhead, collectives, data type support, memory registration, flow control, reliability, resilience and fault tolerance, QoS, instrumentation for performance monitoring, virtualization.
- There are two broad dimensions: on-chip NI and off-chip network interface. How many network interfaces per node? What is the balance between off-chip versus on-chip? What about on-loading versus off-loading versus hybrid?
- Now we need to come up with challenges so that tomorrow we can seek to prioritize them.

Comment - GP/GPU is PCI Express. Larrabee (in its first iteration) is PCI Express. There is a vector for system expansion.

- I believe we are focusing on network interface issues (and not taking on the GP/GPU).

Shalf – I was asking about Roadrunner and was told that they could not wait to get rid of PCI Express.
Dally – The biggest issue with NICs is getting the processor vendors to integrate things well. What we want is to extend the interface from the NOC to the system-wide network.

- You are saying the on-chip interface will be in place?

Dally – How to get to your NIC if it is not?
Comment – Concurrency is a big challenge.
Shalf – Not having the interface NIC coherent and TLB coherent means you violate two principles essential for PGAS languages. We need closer integration.
Comment – It will probably be Ethernet on the chip (that is the standard and is what will sell).
Camp – It is not clear that PCI Express Gen3 is not better than some of the alternatives – it is certainly not an order of magnitude worse. The problem with Roadrunner is not PCI Gen3 per se but how it is used. Do we have a host of little NICs? Or, do we integrate them up to a large NIC that might be several internal, larger NICs? This is an argument we are having internally.
Comment – Some of us are using the NIC for other things. If we think of the interconnect for message passing then we need to keep in mind that other people use it for other things. I do not want to see us create another network for storage.
Comment – But it would be OK to have a storage network that is optimized for storage and a network that is optimized for message passing.
Comment – That adds many dollars to the machine.
Camp – The way we think about computers today it is the memory system that counts. The processors just move from memory. You are gated by how fast you can get into and out of

memory. Memory speed is the most important thing in the system; when the memory is the slowest thing in the system - that is a big issue.
Comment – From a system perspective there are two issues: the interface the processor has to talk with the NIC and then there is the NIC. Things like PCI get in the way of writing a good NIC. You want to be able to order things and have concurrency across the interface. Better processor-NIC interface to allow efficient order and concurrency is important.
Question - Why do we not have some of the network interface features? For example, collective offload support, why do we not have it (and will we soon)?
Answer – I do not know. I do not build the NICS.
Comment – You would have to scale the number of processors doing the work for the collective on the NIC to the number of CPUs you have.
Comment – You have to put floating point on the NIC to do collectives.
Carlson – I should not care where the functionality is – it should be on the processor (it is more efficient). That should be our goal. If we were computer architects that were independent of business considerations I would want it in the logic of the CPU. If I am going off-chip to another chip, and then further, I am paying for more picoJoule than I need. I do not see how you can discuss having something off chip in the timeframe we are discussing. During the device technology group discussion this morning – 1 pJ/bit try to achieve 1 byte/second – we are talking about megawatts.
Camp – You are making a good point but there are intermediates (e.g. high speed fabric). If you try to get Intel, AMD and IBM to make these chips more complex and bigger then yield goes through the floor.
Carlson – These things have to be brought closer together. One could think of many relatively low overhead and well defined interfaces but it should be part of what I buy.
Comment – Perhaps we do not have to do anything (just wait).
Comment – Are we talking about Ethernet here? That is the only thing we can reasonably expect to be put on the processor (I do not see it happening with anything else).
Carlson – I would like to see something with the characteristics listed on the slide (e.g. fine-grain point-to-point communication with low overhead). If Ethernet becomes that, then fine (but I do not think it is about to happen). Ethernet does not send a small number of bytes at a time efficiently.

- Are we talking about Ethernet?

Comment – Ethernet has changed significantly over the past years,
Comment – There are people who want to turn Ethernet into a modern, useful data center interface. There are others who do not want to do that. Ethernet hegemony is only increasing day by day. If you want to come off processor chips go to 8.02 and fix it. The big guys (in terms of integration) are focusing on Ethernet.
Carlson – One thing large systems have been good at is adding semantics content onto messages. There are bits that have to be transported (and Ethernet can do that so long as it is relatively efficient). Say we can do the fine-grain point-to-point. I would like to have some semantics, as well. Will there be a mechanism (if we say Ethernet is the thing) to put semantic language on top of it?
Comment – Do not confuse Ethernet with IP. There is talk about pushing fiber channel on top of Ethernet and fixing Ethernet so it will connect thousands to millions of nodes without an intervening protocol. How to extend Ethernet so it can handle a larger span? If one is going to turn Ethernet into something bigger than it is now one could argue that 32K is workable. If you have a raw Ethernet and can route stuff you can put any protocol you want on top of the Ethernet.
Carlson – I understand - but I would like processor vendors to do something so the semantic content will allow for things such as atomic operations on top of that.
Comment – The Ethernet versus “Ethernot” debate is very old.

- Are we saying that fine-grain point-to-point overhead is one of the most important challenges?

Comment – Putting reliability on that slide means there has to be a trade-off between some of the other things already listed on the slide.

- This is just a list of items. What are the most important things?

Shalf – What is virtualization in this context?

- We mean basically a virtualized NIC.

Comment – You framed collectives as on-loads versus off-loads. When you think about the large environments that we have been talking about - then I think this is the critical issue. Are certain types of collective ops going to be important? We can talk about the solution later (although it may be neither on-loading nor off-loading). I would like us to focus on the challenges (not the potential solutions).

- I agree. If we can agree to the features then we can discuss which are the most important. (How we implement the features is secondary).

Comment – You are going to need some way to identify a NIC with a processor to help MPI.

Comment – That is more an OS and batch scheduler issue. The data are there and we can get to it today - but every batch scheduler and interface does it differently. It is all there. The user should never see that (I think they do need to see topologies). I do not think that has to do with NIC and processor talking together (it is our problem further up the software stack). The hardware tells us what it looks like. It is a software problem at the library level.

Comment – I think you need to add cache injection.

- Add cache injection techniques

Comment – Do you mean bump something else out of cache? No - if the data is already in cache then you want it to update the data in cache (and not flush it).

Carlson – I would add support for synchronization. Programming models need to have a set of well defined semantics about what the programming model is. There are a number of features required to do the necessary synchronization (that is, make sure this work is done before the next work I give you).

Comment – If you have to define your minimal subset and it is going to be expensive to cross the interface - what would you include?

Carlson – Assuming there will be some programming models that put a shared memory model on top of the system and MPI is going to want to use that – you need to be able to correlate progress in the processors' view of the memory model of the system with things that are happening on the network interface.

Comment – The fact that the NI is external does not make it more expensive to have a diversity of primitives. The interface from the processor to the NIC needs to allow fine grain things at high bandwidth and ordering to the NIC without requiring round trips.

Comment – The question is from a NI towards the processor - is there a better perspective?

Comment – The PCI group has the right idea and is adding atomic processor ops into the spec. If you write something in from the network it should be coherent and update the cache. You should support a suite of atomic ops natively in the processor.

ORNL – You should evaluate the cost of having the processor doing it for you versus doing it yourself.

Comment – We are talking about adaptive routing in the fabric. At some point we have to deal with the fact that we lose order.

- Add good support from processor NI to node-level NIC and efficient handling of out of order messages (multipath, …).

ORNL – If we have a fence operation we can enforce order.

Pakin – If you have to have everyone connect to everyone else at exascale that is a bad idea.

Shalf – Right now in MPI you refer to a host name for a connection-oriented protocol – that needs to get down to a single address for your remote host. You want hardware management of how you enumerate your peers in a parallel job with the ability to remap for fault tolerance.

- Add no connection-oriented protocol.

Comment – I can give you zero latency and infinite bandwidth (so long as you do not add reliability).

Comment – If your machine is so big you know it is going to break every time - then why not assume it is going to break?
Comment – We need detection, at least.

- Add reliability (to challenges).

Comment – If the network itself is not going to give you reliability then you need to provide reliability in the NIC.
Dally – The problem with putting it in the NIC is your state requirements go way up. You make the network do it.
Comment – Doing it in the software makes things ugly, fast.
Question – How often do you lose a router, cable or something you cannot support in the network?
Dally – You lose bit errors easily. You can recover if someone cuts the cable in your router.
Carlson – How much state is there with a reasonable latency network? The trick is to have low latency and send small packets on the wire.
Comment – If you adaptively route then reliability gets trickier. At the target how do you determine a message is lost (if they come in out of order)?
Comment – The key is how a destination detects a failure.
Dally – The destination does not detect failure (it does not know a message is coming). The source detects failure. It is easier to have the sender keep a log. When you do adaptive routing and want to keep everything in order you are creating problems for yourself.

- Add end to end reliability.

Dally – I think you need to think about where you want to get to (and that does not necessarily mean MPI).
Comment – The DOE apps people I talk with all like message passing. There are a lot of people who really want a two-sided connection interface.
Comment – You want to be able to do both.
Shalf – We are running into these issues with BG/L because it has to keep extra metadata about everyone it needs to talk to. If you start with efficient one-sided you can build a two sided MPI on that.
Comment – You cannot build a two sided MPI on one sided without matching.
Dally – You do a put which spawns a thread at the far end.
Comment – I have seen few large scale users who did not want tags and/or communicators.
Comment – Whether they are needed or not they are written with MPI. We are stuck with MPI for a substantial number of apps in this time frame.
Carlson – Can we get to the layer people and say stop it?
Comment – We are talking about an entire application (and all supporting libraries).
Comment – No one is arguing that we have to purge MPI in principle.
Comment – We are saying make an efficient message passing layer and people can add MPI on top of it.

- Can we achieve the load/store latency listed in the table? (*Answer* – No, not even close).

Dally – These numbers do not focus on NIC issues.

END OF GROUP DISCUSSION SESSION