THE RAPIDIO TRADE ASSOCIATION AND HYPERTTRANSPORT CONSORTIUM SAY THAT THEIR TWO STANDARDS SERVE DIFFERENT PURPOSES AND ARE NOT IN COMPETITION WITH EACH OTHER. THEN WHY ARE SO MANY CHIP VENDORS ASKING THEMSELVES WHICH ONE THEY SHOULD SUPPORT?

The popular way to increase bus bandwidth has for some time been to add lanes. However, this strategy seems to have reached the point of diminishing returns, as footprints, pin counts, and power consumption have increased to a point at which it makes sense to swing the pendulum from parallel back to higher speed serial lines. Enter HyperTransport and RapidIO.

RapidIO versus HyperTransport: A battle between equals or unintentional marketing confusion?
RapidIO and HyperTransport were from the start aware of each other. Early conversations among spec architects did not result in a merger, because both groups were focused on different applications. However, as is the case with most standards, as the specs matured, so did their perceived scopes. The question of ideal use is quickly lost as the revenue of potential secondary markets become visible. Such “market creep” only adds to the confusion over where the interfaces play and how they will compete with each other.

Both of these high-profile interconnects have a large following, as well as backing from major vendors. AMD champions HyperTransport and has public support from giants such as Broadcom and Dell. Motorola is pushing RapidIO, and IBM is also offering to support the standard. Then there’s Intel with PCI-Express (see sidebar “PCI-Express: a spec by any other name...”). No one knows how committed these vendors are, how much pressure they will exert on their partners, or how much money they will throw into the battle to hold and gain ground. However, it is not difficult to predict that each of these standards will play well into the markets that these vendors participate in.

HyperTransport pitches itself as the “real” interconnect of the three: The HyperTransport consortium claims that manufacturers have shipped more than 3 million HyperTransport nodes. Compare this fact with the industry buzz that PCI-Express won’t even be around for another two years. Are 3 million shipments big deals, especially with RapidIO parts soon becoming available? The concern in supporting an interface in its infancy is that availability of compatible devices is market-centric. For example, HyperTransport proponents claim to be making significant gains in the network-processing market, and RapidIO supporters claim to have a lock on DSPs. However, although vendors are making many claims, little silicon exists.

There are other issues to consider, as well. HyperTransport has its own unique problems. API Networks, the company that held many of the relevant patents, went belly-up, leaving questions about licensing and support somewhat unresolved. RapidIO had its own rocky start, having to delay initial misconceptions that it was competing against, of all things, InfiniBand. (InfiniBand’s claim that it, too, is a PCI replacement fueled the issue.)

ONE WINNER?

Given the perceived similarities of the interfaces and the advantages of designing to one interface, some vendors feel
strongly that one winner will emerge. Others see the various market forces—AMD, Motorola, and Intel—as too strong to take each other out. In reality, one technology or architecture does not have to win. Although you can use each interface across multiple applications and markets, each has niches that it will probably dominate.

One hope is that the variety of standards will result in a single hybrid standard mixing the best of all technologies. However, such an interface would necessarily be somewhat general and less suited for the markets each interface initially targeted. In any case, history reminds us that technological superiority is rarely the basis for whether an innovation survives. More important factors include the politics surrounding the technology (that is, some people refuse to support any technology from Microsoft or Intel), which companies are supporting it, and how deep those supporting pockets are.

Note that nothing says that an interface has to be standardized. Some vendors in the network-processing market use AGP (Advanced Graphics Protocol) from the desktop world, albeit somewhat stripped of unnecessary features, as a “proprietary” interface. Deployed interfaces, however, seem to be safer to use because they are “proven.” Also, you need not reinvent technology, as you might when creating your own interface. But some markets, such as the high end of network processing, are the bleeding edge: They have nowhere to steal from.

However, proprietary interfaces, which

---

**PCI-EXPRESS: A SPEC BY ANY OTHER NAME...**

PCI-Express, formerly known, among other things, as 3GIO, adds an interesting wrinkle to the HyperTransport/RapidIO debate. For a spec about which is little known and which most in the industry consider at least two years away, it looms high on the list of competitors with HyperTransport and RapidIO. Intel has played so close to the vest about the actual details of the spec that several IP companies that hope to design cores for the spec have yet to see any details.

PCI-Express pitches its main strength as being the obvious successor to the backplane in the high-volume PC and server markets. Certainly, PCI-Express won’t be in every PC, because PCI and PCI-X are sufficient for cases in which they aren’t merely steppingstones to PCI-Express. However, the migration from any PCI variant to PCI-Express is fairly straightforward, you can move up the bandwidth curve without having to change or adjust the software, as you would have to with RapidIO. This factor, PCI-Express proponents state, will give PCI-Express the edge over RapidIO in the fight for the backplane.

Proponents of RapidIO don’t disagree with this argument. Rather, they say it is irrelevant, because RapidIO isn’t going for the desktop market, and its developers didn’t intend it to be compatible with PCI. Supporters of RapidIO say HyperTransport and PCI-Express claim to have leverage in the embedded world by moving technology over from the desktop world. These supporters say that this leverage simply doesn’t exist, however. Desktop technology, they say, hardly ever crosses into the market for embedded systems. Thus, they feel that PCI-Express’ strategy of showing how it will replace PCI isn’t a threat; many embedded designers simply stop the marketing pitch and say, “But I don’t use PCI.”

HyperTransport supporters tout it as a compatible technology that may be enough for chip-to-chip communication, connecting chips to I/O, or perhaps boxes that require no option slots. (But what application doesn’t require option slots for adding tomorrow what we didn’t think of putting in today?) That the technology may be enough suggests that PCI-Express will also someday try to claim these interfaces opportunities. PCI-Express does offer a bandwidth advantage over HyperTransport and RapidIO. But how long will this advantage hold? Already, HyperTransport and RapidIO are ramping up their rates. By the time PCI-Express becomes available, both HyperTransport and RapidIO will probably have specified equivalent speeds. They will also have lower rate versions entrenched in the very applications that will need this additional bandwidth, making it even more difficult for PCI-Express to claim these sockets.

Certainly, PCI-Express makes a lot of sense in the desktop world. Companies such as Dell clearly state that they will not go out of their way to “fit” either HyperTransport or RapidIO into their systems while waiting for PCI-Express. (Some suggest that PCI-X will serve this role.) Additionally, PCI-Express has adopted many of the low-level specs of InfiniBand, another interface attractive to the PC world. As a result, the argument goes, designers could use two similar specs instead of very different specs. PCI-X is said to be enough to launch 4× InfiniBand until ×8 PCI-Express is ready.

The initial PCI-Express spec is said to look a lot like PCI, except that PCI-Express is faster. Future developments will focus on mechanicals that are more modular in design. Instead of have users open a box and plug in cards, the PCI Special Interest Group, now responsible for managing the spec, hopes to create front-end loading, hot-pluggable cartridges. These new form factors will not be part of the initial spec; the group says it doesn’t know what future boxes will look like, and these form factors should not be part of the spec as it first comes out of the gate. PCI-Express is currently under member review, with Version 1.0 of the spec to be published in midyear, when anyone will be able to purchase it. That said, the industry says the spec is still two years from becoming a product.

Will PCI-Express and HyperTransport provide a one-two punch to take down RapidIO? Much of PCI-Express is still a mystery, except to the privileged few with whom Intel has chosen to work. RapidIO is effectively available now. For some, existence today is a powerful lure.
coverstory RapidIO versus HyperTransport

often provide more efficient use of bandwidth for an application, are difficult to support. This situation is especially true in the network-processing industry, with its plethora of interfaces. There is little value in and few resources for maintaining all of these interfaces. Using a standard interface could considerably decrease costs.

Yet, perhaps interfaces can become too optimized; the number of standards in the network-processing space is increasing, with CSIX (common switch interface); the upcoming Network Processor Forum interfaces; HyperTransport; RapidIO; and the quiet giant, SPI-4. SPI-4 is efficient for packets and point-to-point communications (with parity generation, training patterns, and error correction) but not for switches or subsystems.

SPI-4 gets little press, but it has quietly built tremendous momentum in the latency-critical datapath. SPI-4’s strength is as a streaming interface; HyperTransport is a memory-oriented bus suited to transactions such as read and write. It is well-thought-out and proven through widespread deployment.

TO GLUE OR NOT TO GLUE

In the PC market, the choice of processor is often the starting point for design. From this decision, you select other compatible devices; systems that avoid bridges or gateways between interfaces tend to be less complex, less expensive, and more reliable. Thus, choosing a processor that supports RapidIO, for example, will drive selection of other RapidIO devices. However, the processor is not the foundation for every application. In network processing, for example, you often first select the fabric, because reliable throughput in the fabric is more important than how a particular line card passes packets. You still have to choose among interfaces such as RapidIO, HyperTransport, and SPI-4 for both the data plane and the control plane. Given wire-speed constraints, the data plane is likely to be your next consideration. Matching these two interfaces, especially if you select a proprietary interface, limits your overall choices for network processor and coprocessors.

Before the rise of standard interfaces, you had more options, in one sense, because you most likely had to have glue logic to connect devices. Now, you may have a best-of-class dilemma: The parts

ROLLING YOUR OWN DIE

The most important factor driving IP (intellectual-property) cores for interface technologies is demand. The IP vendors themselves could not care less which interface wins in which markets; they simply want to sell cores. Although most vendors have chosen a single interface to concentrate on, the other interfaces are on the road map if the right customers ask for them. Be careful how much credence you give to the opinion of IP vendors on the viability of the interface they support; in this respect, they are simply hedging their bets.

Understandably, companies offering interface IP don’t care which I/O you select as long as it’s one of the cores they offer. From a low-level-signal perspective (not including the PHY), there are no real differences between HyperTransport and RapidIO. That fact is significant; if you would consider one, you would also consider the other. You would base your choice, then, on what you and your team thinks is complicated and which interface you think your customers want.

Of course, you have the option of supporting multiple interfaces to increase your market acceptability. Supporting multiple interfaces comes at a die cost but offers security in being able to fit into more applications and connect to more devices offered by other vendors. Such a strategy allows you to avoid having to choose which interface will win, because you can support several of them—again, at a cost. Chips that support multiple interfaces will need a bridging mechanism to connect the various interfaces. Altera, for example, offers its Atlantic interface, an internal interface with a generic backside that Altera claims allows you to bridge interfaces. You can also use Atlantic in an FPGA as glue logic between interfaces. For $36,000, you get a perpetual license for both HyperTransport and RapidIO.

There is talk about creating a combined HyperTransport/RapidIO core for applications that might need to support one or the other. The IP is simpler if the blocks are separate, but you can combine them, even at the PHY level. However, although the two interfaces could share a lot, inefficiencies would still occur when you merge the two. For example, the common blocks would not be optimized for either interface, and two separate paths would at some point be necessary. Thus, the block would become larger. A more realistic combination, projected to be available by the fourth quarter of 2002, is an SPI-4/HyperTransport core. Hifn, an encryption-coprocessor vendor with a vested interest in being able to communicate with as many network processors and, thus, interfaces as possible, believes in the feasibility of an SPI-4/HyperTransport/RapidIO interface using all the same pins.

It is often tempting to leave I/O until the last part of chip design. After all, you’re just adding a standard interface. However, as you progress through the design cycle, the number of constraints on the system increases. Some of these constraints can affect your ability to use an interface. High-speed interfaces can drive your choice of fab, testing, and even packaging options. Additionally, your initial assumptions may prevent the interface from working effectively or cause your design to “break” in some way.

Three common problematic assumptions include the power and ground ratio, the die-attach mechanism, and the package type. For example, you might underestimate the number of signals you can have per power and ground pair on the periphery of the die. Also, you probably won’t be able to get away with a low-quality bond-wire attachment method. And spending another $1 for higher quality packaging for a $300 part can actually result in significant cost savings when you consider that you’ll have a higher yield, be able to run a chip at a higher frequency, and increase reliability of the chip. Such issues could delay tape-out.

Choosing your interface early in the design process might be impossible, especially if marketing hasn’t decided which interface you should design in. In such cases, you should at least be aware of the design constraints for the major candidates.
you want to use have different interfaces, so you still have to use glue logic, but selecting a second-tier device with a compatible interface eliminates the glue logic and all the associated costs of an FPGA. Tough choice. A conceivable best-of-class system might have HyperTransport on the data plane, RapidIO managing the control plane, and PCI-Express over the backplane. Wouldn’t such a system be a monstrosity?

The answer depends on the application. What matters is how cleanly you segment the overall system. Often, the data plane is a pure bus that never touches the backplane and connects to the control plane through a processor that strips the protocol off the data. Specialized or exception processing of data, however, could be clean (directly in the datapath) or mixed, such as an encryption blade connected over a backplane. Avoiding conversions and bridging avoids expenses in chips, area, dollars, and power.

You may also want to connect to an interface, such as InfiniBand, that hasn’t made sufficient inroads into your application space to justify a device that specifically supports that interface. If you must have an FPGA on the board for other reasons (say, for preprocessing or post-processing), bumping up the size of the FPGA to support an interface bridge may open your options for an incremental cost. (But remember that costs rise faster when you want additional gates on the same chip instead of a second device.)

One advantage of keeping the FPGA is that it opens your options to not just a single interface but to whatever interface you want to load into the part. Additionally, you need only enough gates to support a single interface and do not have to support the overhead of maintaining every interface you might support across product lines and applications.

**HEAD TO HEAD**

It’s difficult to discuss the differences between HyperTransport and RapidIO because whether a particular aspect is important or complex depends on your background and application. Several experts designing to one or both of the standards see a variety of perceived advantages and disadvantages between the specs. Again, whether these differences are significant depends on your application and experience. In any case, a consensus exists that both RapidIO and HyperTransport are well-thought-out, comprehensive, stable specifications ready for you to develop to them.

At the PHY level, RapidIO has standardized LVDS (low-voltage differential signaling); HyperTransport, on LDT (lightning data transfer). LVDS is an older technology, which implies reliability and market acceptance, but maturity can mean the technology is based on assumptions that are perhaps out of date. LVDS’ developers designed it for bipolar processes and migrated it to CMOS. LDT, on the other hand, was designed for CMOS, so some say it is easier to work with than LVDS in CMOS, especially as processes continue to shrink. Others think both technologies can be pushed to 2 Gbps, which is the point at which clock-and-data-recovery mechanisms become necessary to recover data, significantly raising the complexity of both standards.

Because RapidIO lets you skew individual data channels in relation to each other, a deskew mechanism must be built into the PHY layer. Thus, RapidIO must periodically insert a training packet into the data stream (overhead estimated at probably less than 1%). HyperTransport, on the other hand, has budgeted the amount of skew allowed between channels. As frequencies rise, this budget increases the difficulty of board design, makes plug-in boards a bit trickier, and may someday limit the total bandwidth that can be achieved without redefining the PHY (making migration difficult). On the flip side, although HyperTransport’s tight specifications could be problematic, RapidIO, using in-phase clocking, potentially loses performance, because noise events are not coherent between clock and

---

**FOR MORE INFORMATION...**

For more information on products such as those discussed in this article, go to www.ednmag.com/info and enter the reader service number. When you contact any of the following manufacturers directly, please let them know you read about their products in EDN.

- **Altera**
  1-408-544-7000
  www.altera.com
  Enter No. 301

- **AMD**
  www.amd.com
  Enter No. 302

- **Broadcom**
  1-949-450-8700
  www.broadcom.com
  Enter No. 303

- **Dell**
  www.dell.com
  Enter No. 304

- **Fairchild**
  1-207-775-8100
  www.fairchildsemi.com
  Enter No. 305

- **GDA Technologies**
  1-408-432-3090
  www.gdatech.com
  Enter No. 306

- **Hifn**
  1-408-399-3500
  www.hifn.com
  Enter No. 307

- **IBM**
  www.ibm.com
  Enter No. 308

- **Intel**
  www.intel.com
  Enter No. 309

- **Microsoft**
  www.microsoft.com
  Enter No. 310

- **Motorola**
  www.motorola.com
  Enter No. 311

- **NurLogic**
  1-877-687-5644
  www.nurlogic.com
  Enter No. 312

- **PLX Technology**
  1-408-744-0960
  www.plxtech.com
  Enter No. 313

- **PMC-Sierra**
  www.pmc-sierra.com
  Enter No. 314

- **TriCN**
  1-415-625-3606
  www.tricn.com
  Enter No. 315

- **Tundra Semiconductor**
  1-613-592-0714
  www.tundra.com
  Enter No. 316

---

**OTHER RESOURCES**

- **HyperTransport Consortium**
  www.hypertransport.org

- **Network Processor Forum**
  www.npforum.com

- **PCI Special Interest Group**
  www.pcisig.com

- **RapidIO Trade Association**
  www.rapidio.org

---

**SUPER INFO NUMBER**

For more information on the products available from all of the vendors listed in this box, enter no. 317 at www.ednmag.com/info.
data, reducing the effective eye pattern you can capture.

HyperTransport (LDT) uses a lower voltage supply and, hence, less power than RapidIO (LVDS). Therefore, it may require an additional power supply. The lower voltage may give you more room to get down to smaller process technologies, but board attenuation over long distances or high frequencies becomes more noticeable at lower voltages. However, some proponents claim that RapidIO minimizes attenuation because of the lower reference voltage. The HyperTransport spec limits attenuation to 25% between driver and receiver; RapidIO limits it to more than 50%, allowing the spec to better support the classic attenuation trade-offs of longer transmissions, passing over connectors, or using a cheaper pc board. HyperTransport also has a tighter ground shift of 50 mV, compared with RapidIO’s 1V tolerance.

RapidIO is a switched-based topology, meaning you always have another device between nodes (an extra chip whose expense you might need to measure in power consumption and area rather than price). Note that RapidIO supports single point-to-point architectures. However, RapidIO switches are not required to interpret the transaction protocol to make forwarding decisions, meaning that the switch does not need to know a read message from a write message. As a result, adding transactions does not require the replacement of existing switches; in other words, RapidIO is forward-compatible. The spec also supports proprietary transaction types.

HyperTransport, on the other hand, employs a tree, or chain, topology, in which each node passes data that isn’t intended for it on to the next node, a process that has both latency and throughput implications. Both technologies target chip-to-chip (mezzanine) and board-to-board (backplane) interfaces. RapidIO also targets chassis-to-chassis (extended-backplane) applications. HyperTransport acknowledges itself as complementary to extended-backplane technologies, such as InfiniBand and 1/10 Gigabit Ethernet.

RapidIO has defined a directory-based coherency protocol for globally shared memory. Coherency plays a role in applications in which you must divide a task among several processors that all need access to the same memory. Coherency refers to the integrity of memory; another processor may not use data that is sitting in a cache, which another processor may have changed but has not yet written back to memory.

RapidIO’s directory-based mechanism tracks which device has or owns the latest copy of any globally shared cache or memory line. When the controller begins a look-up, it searches the directory. If another device owns the data, the controller tells the other device to release it. You can implement the directory in a distributed fashion, on each memory controller, or centralized in a single location.

HyperTransport has no publicly available coherency mechanism. (Apparently, AMD has defined one but has kept it internal.) Therefore, you would probably use a snoop-based mechanism running on the HyperTransport interface if you need coherency. Snoop-based coherency requires a processor to broadcast requests to other processors to discover whether they are holding a particular datum in their cache. Requests for data that are not on another processor, which probably encompasses most requests, merely burden these processors and add latency to such accesses.

HyperTransport has already announced its first set of enhancements to address several of its shortcomings in comparison with RapidIO, beginning the one-upmanship dance between the standards. Although having two high-profile, competing standards is a good thing—they are exposing each other’s shortcomings and driving each other to be more efficient—the difficulty arises in understanding comparisons between the specs. For example, error correction, not considered a necessity in the PC world, is considered necessary in highly available applications, such as those in telecommunications and data communications. Enhancements to HyperTransport address the need for error correction that the original spec failed to meet. However, these network extensions won’t be available until the second half of 2002. Your challenge is whether to compare RapidIO with what HyperTransport is or what it is soon to become. And what about upcoming enhancements to RapidIO?

Regarding error correction, RapidIO prevents dropping of packets or transactions and recovers from small to moderate burst errors through link-by-link error detection and hardware recovery. It assumes that real-time systems don’t have the luxury of waiting for software to recover from errors and that failure to detect an error is unacceptable.

RapidIO has several mechanisms for detecting and recovering from faults. LVDS, for example, uses 8b/10b encoding to prevent data and clock loss; this protection, however, comes with 25% overhead. (Only 1.6 Gbps of a 2-Gbps link is data.) RapidIO also does packet checking and can reinitialize based on the severity of a fault, supporting various levels of fault tolerance and recovery. Synchronization occurs between link pairs using a circular acknowledge ID, which helps detect lost packets, and in-band transmission of that ID, which you can insert into the middle of another packet. Thus, one device can precisely acknowledge every packet received directly with the sending device. If the two devices are out of sync, hardware attempts to resynchronize them, recovers any lost packets, and retransmits if necessary.

HyperTransport currently generates a CRC (cyclic-redundancy check) on every 512 data units. The mechanism is effective at detecting errors but has no granularity for narrowing down which datum has the error. If an error occurs, the system must discard all the data related to that CRC—a process that becomes complicated as data propagates through the daisy chain of HyperTransport devices. Because you can’t tell if a read has been turned into a write, you should consider the system unreliable and restart it.

The integrity of the HyperTransport link is arguably as effective as an SRAM DDR bus, which has no error-handling or -correction capability, yet is considered relatively reliable. However, pressure from vendors designing applications that need to achieve a high level of fault tol-
erance has prompted the HyperTransport Consortium to revise the error-detection scheme of Version 1.0 as part of its upcoming enhancements to the spec. These enhancements include the ability to narrow down the CRC to apply to fewer packets, such as four or eight, and the ability to roll the system back to a last known good checkpoint. Additionally, it is no longer necessary to chase an error through the daisy chain. At each stage, the data is held until all the data and the CRC for that data arrives; that is, the system does not act on the data until it determines the data to be good. Buffering the data in this manner prevents any state changes from occurring until the data is verified but adds latency to transactions. To reduce this latency, links can send packets before the checksum arrives to avoid store-and-forward penalties, so latency is limited to buffering at the destination node. Error handling is piggybacked onto the flow-control packet that handles the credit-based mechanism of the interface. Note that this feature is not backward-compatible, because if a node doesn’t support the feature, the mechanism is unusable.

RapidIO is a switched bus targeting both point-to-point and peer-to-peer applications. Initially, HyperTransport supports only point-to-point transactions, but the network extensions will support peer-to-peer transactions and formalize definitions for HyperTransport switches. Direct peer-to-peer transfers allow nodes to talk directly to each other without passing messages through a host. Applications include those in which several processors must be able to communicate with each other. In a network-processing application, for example, one classification engine might need to send a partial result to another classification engine. Thus, each classification engine needs only one bus to communicate with the others, as well as a network processor, rather than two separate buses, reducing pin count. It is questionable whether memory vendors will be willing to pay for IP licenses to support HyperTransport or RapidIO, so these interfaces are unlikely to replace current memory interfaces.

RapidIO is source-addressed, meaning that the initiator of a message includes a source ID and a destination ID with the data. Through the use of routing tables in switches, the transaction finds its way to its destination and supports redundant paths and changing routing en route to increase fault tolerance. Additionally, RapidIO can track errors by target rather than individual addresses. RapidIO also supports a class of messaging for passing transient data. For example, Processor A processes a large amount of data and then passes it to Processor B, never to touch it again. Traditional DMA (direct-memory-access) schemes require Processor A to track Processor B’s memory space. Using messaging with large data sets (in which the message overhead is small compared with the overall data-set size), Processor A no longer needs visibility into Processor B’s memory space, simplifying design.

Enhancements to message passing will allow HyperTransport to stream packets to an address—rather than merely read and write single transactions to given addresses—over 16 point-to-point channels per link. Flow control can use the channels to create end-to-end channels through nodes or support quality of service. Arbitration algorithms support a weighted round-robin mode as well as limited bandwidth for high-priority data. Antistarvation logic guarantees a minimum bandwidth, so errant applications can’t consume a link. Additionally, the 16 messaging channels ease the passing of SPI-4 traffic. In contrast, RapidIO supports streaming semantics but does not support flow-control channelization or arbitration.

Making Your Bet

Few players have committed to locking in on either interface. The IP vendors just want to get into chips; they’ll support whatever the market calls for (see sidebar “Rolling your own die”). Vendors designing devices that hang off other devices, such as coprocessors, will be forced to support all the major interfaces; not supporting an interface is a tough reason to fail to secure a design win. Thus, processor vendors will probably determine which interface will win in which market.

The odd paradox is that most of the processor vendors don’t want to make that choice. Several vendors refuse to explain why they picked one interface over another beyond revealing, “That’s what our customers asked for.” Of course, they also said they would support other interfaces if a customer with enough money asked them to, but they’d rather support one than several. And this choice is important. The interface a vendor chooses to support may well define which players the vendor can partner with. Thus, although you will be limited in your interface options by chip vendors’ choices, they claim that you are driving their decision. In this light, your first decision should be whether to care about which interface you use. If you do care, then you need to express your preference to your chip vendors. Chances are, no choice is ideal. However, if you choose to be silent, you will lose your voice. Decide which deficiencies you can live with, and cast your vote.

Keep in mind, however, that these specs are still somewhat in flux because of their competition with each other. Such competition promises to foster innovation with the strengths of one standard driving the other. And, because not much silicon has yet emerged, there is room for change. HyperTransport, the standard with the most silicon, has already moved to make its first set of enhancements. RapidIO is also looking at increasing speeds. Both standards’ developers anticipated many of these changes. However, to grab market share, each standards group has stepped up the schedule for improvements. The effect of competition should continue to be positive, forcing greater scrutiny of each standard before it becomes too entrenched to change.

Will there be one winner? Within particular niches, yes. For the overall market, no. In the end, however, each standard will be better, given the other.