Proceedings of the 6th International Conference on Parallel Interconnects (PI 99).

From HiPPI-800 to HiPPI-6400: A Changing of the Guard and Gateway to the Future*

D. Tolmie⋆, T.M. Boorman⋆, A. DuBois⋆, D. DuBois⋆, W. Feng⋆†, and I. Philp⋆
{det,tmb,ajd,dhd,feng,philp}@lanl.gov

⋆ Advanced Network Research & Development
P.O. Box 1663, M.S. B255
Los Alamos National Laboratory
Los Alamos, NM 87545

† School of Electrical & Computer Engineering
1285 Electrical Engineering Building
Purdue University
W. Lafayette, IN 47907

Abstract

HiPPI-6400, a High-Performance Parallel Interface running at 6400 Mb/s (800 MB/s), is a networking technology targeted for deployment in a local-area network (LAN) or system-area network (SAN). It is a low-latency, high-bandwidth switch that has the added features of providing flow control and error detection and retransmission in hardware, thus freeing network software from having to implement these functions. And due to the very low overhead of the HiPPI-6400 physical layer and the use of separate control lines, user data rates of HiPPI-6400 are literally 6400 Mb/s or 6.4 Gb/s (eight times the usable bandwidth of Gigabit Ethernet) with an achievable bit-error rate of less than 10^{25}.

An initial 32-port HiPPI-6400 prototype, running an OS-bypass network protocol called ST between two 32-node SGI Origin 2000s, produced one-way latencies of 7 μs and sustained data rates of 3.6 Gb/s (unidirectional) and 6.4 Gb/s (bidirectional). The HiPPI-6400 bandwidth in both cases was limited by the memory architecture of the SGI Origin 2000, not by the network. When running an OS-based protocol such as TCP, the unidirectional bandwidth was 2.2 Gb/s.

Keywords: *HiPPI, gigabit networking, network hardware, OS-bypass protocol.*

*This paper is LA-UR 99-4450.

1 Introduction

HiPPI-6400 represents the next-generation high-speed interconnect beyond the current gigabit interconnect standards. Operating at 6400 Mb/s, or equivalently 800 MB/s, HiPPI-6400 ensures maximum compatibility with Ethernet, Gigabit Ethernet, ATM, and the original HiPPI installed base.

The original HiPPI standard, now denoted HiPPI-800, served as a starting point for the HiPPI-6400 standard. Standardized in 1987, HiPPI-800 pioneered the field of high-speed parallel communication. Operating at 800 Mb/s (100 MB/s), its bandwidth far surpassed what was available back in the late 80s and early 90s, e.g., 10-Mb/s Ethernet. HiPPI-800 is a circuit-switched networking technology designed to provide high bandwidth between communicating hosts. In addition to being the primary parallel interconnect used in LANL's ASCI Blue Mountain supercomputer, a 3.1-TFLOP machine consisting of 48 SGI Origin 2000 symmetric multiprocessors (SMPs), HiPPI-800 is widely used in machines at supercomputing centers throughout Europe and the United States. It has also been successfully deployed in the commercial world as well, e.g., DisneyWorld uses HiPPI-800 as its fundamental networking technology to drive its PowerWall.

Unfortunately, due to the sharply increasing demands for more bandwidth for applications such as wide-area scientific computing and visualization; the speeds of HiPPI-800, Gigabit Ethernet [4], and

Myrinet [1] will not be able to keep pace with tomorrow's bandwidth-hungry applications. For example, the anticipated bandwidth requirements between LANL's 30-TFLOP supercomputer and a visualization server are a minimum of 20 GB/s (160 Gb/s) to a maximum of 80 GB/s (640 Gb/s). To attain this kind of bandwidth with HiPPI-800 would require at least 200 HiPPI-800 ports and 15,000 cables whereas HiPPI-6400 would only require 25 HiPPI-6400 ports and 1,800 cables. Thus, the goal of HiPPI-6400, like HiPPI-800 ten years ago, is to address the need for higher-speed networking technology in both local-area networks (LANs) and system-area networks (SANs).

Rather than simply upgrading the physical-layer speed of HiPPI from 800 Mb/s to 6400 Mb/s, we leverage our lessons learned from HiPPI-800 to design and build a better parallel interconnect for HiPPI-6400. First, HiPPI-800 requires explicit connection set-up and tear-down in order to enable communication between hosts. This connection-oriented overhead at the physical layer prevents connectionless network protocols such as UDP to make efficient use of the 800 Mb/s bandwidth. In particular, due to the additional overhead, the response-time latency is increased, and the effective bandwidth is decreased. Consequently, we use wormhole routing [2] in HiPPI-6400; in wormhole routing, messages can be sent into the network without prior knowledge of whether or not a free path is currently available to the destination host.

Second, because of the bimodal distribution of packet sizes seen in most networks, a single, large message can adversely block the transmission of other messages, particularly in HiPPI-800 as it is a circuit-switched technology. HiPPI-6400 addresses this problem by introducing virtual channels (VCs) to provide a multiplexing mechanism that can be used to minimize blocking. Multiplexing over VCs also provides the additional benefit of higher network resource utilization. In HiPPI-800, because links cannot be time-shared, physical links experience lower utilization.

Lastly, sustained network reliability over a machine as large as ASCI Blue Mountain is a severe problem (due simply to the sheer volume of components). As a result, reliable transmission over HiPPI-800 is ensured by an overlying network software protocol such as TCP. Unfortunately, network tests have shown that the network protocol stack is already the bottleneck in delivering application-to-application communication. Incorporating reliability into the network protocol stack further constricts the delivered bandwidth to an application. Therefore, for HiPPI-6400, we push the function of reliability down from the protocol stack into the network interface card (NIC). This radical design change benefits both OS-based protocols such as TCP as well as OS-bypass protocols such as AM [3], FM [7], PM [9], and ST [6, 8]. Rather than having the software deal with network retransmission, i.e., CPU executing network protocol software, the NIC would provide this support, freeing the CPU to do other useful work.

2 Related Work

At the present time, the closest competitors to HiPPI-6400 (6.4 Gb/s) are Gigabit Ethernet (1 Gb/s) and Myrinet (1.2 Gb/s). Generally, Gigabit Ethernet [4] is used in a LAN environment whereas Myrinet is used in a SAN environment, e.g., PC cluster. The biggest obstacle for Gigabit Ethernet is application-realizable bandwidth. Although the underlying physical technology is 1 Gb/s, the small maximum transmission unit (MTU) size of 1500 bytes generally causes the operating system to be interrupted more frequently (assuming that an OS-based network protocol such as TCP is being run) than with other technologies, thus reducing the realizable application bandwidth to approximately 100 Mb/s. As a result, there has been a push for supporting "jumbo datagrams" where the MTU size is on the order of 9000 bytes; this would effectively increase the realizable application bandwidth to 600 Mb/s. However, the disadvantage with "jumbo datagrams" is that they can promote blocking in the network.

Myrinet [1] is a self-initializing, low-latency switch that uses cut-through routing rather than the conventional store-and-forward routing found on the Internet. In cut-through routing, a packet is advanced into the required outgoing channel as soon as the header is received and decoded. The disadvantage of this routing is that it can promote blocking, particularly with large messages, and thus traffic can be backed up through the Myrinet switch back to the source (much like HiPPI-800). However, one of the main advantages of Myrinet is the existence of a control processor on the network interface card (NIC) which allows the functionality of the NIC to be programmed and can allow some network software functionality to be pushed down into the NIC. Thus, the issue of blocking can be addressed by ensuring that the MTU size is not "too large."

3 HiPPI-6400 Technology

The HiPPI-6400 standards effort, spearheaded by Silicon Graphics, Inc. and Los Alamos National Laboratory, originally consisted of four major parts: HiPPI-6400 Physical Layer (HiPPI-6400-

PH), HiPPI-6400 Physical Switch Control (HiPPI-6400-SC), HiPPI-6400 Optical (HiPPI-6400-OPT), and HiPPI-6400-Scheduled Transfer (HiPPI-6400-ST). Since then, the last part has been split off as a separate standard. For more detailed information, see http://www.hippi.org.

A HiPPI-6400 switch allows up to three different types of entities to be connected to it, as shown in Figure 1. An entity can be a computing node, a translator, (i.e., a bridge between HiPPI-6400 and another networking technology), or another HiPPI-6400 switch. (While the figure shows the switch with four bidirectional ports, industry will be delivering switches with 32 ports.) The physical links between the switch and a given entity are bidirectional and capable of delivering 6400 Mb/s user data bandwidth in each direction simultaneously. A link's control information is carried on separate wires in parallel with the user's data (not shown in the figure) and is thus not counted in the 6400-Mb/s bandwidth number.

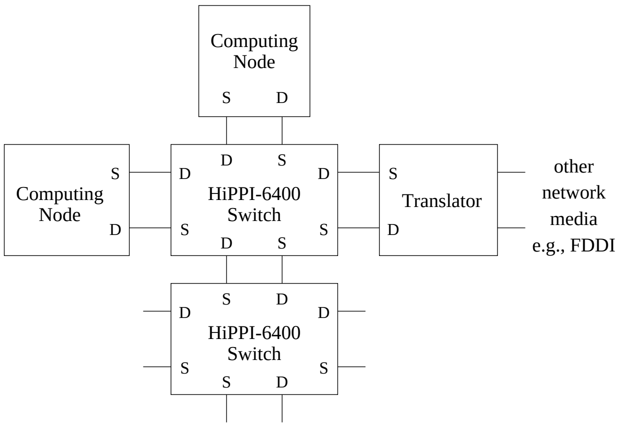

Figure 1: Using HiPPI-6400 Technology

3.1 Micropackets

A micropacket is the basic transfer unit on a physical link, consisting of thirty-two data bytes and eight control bytes (32 bytes was chosen since it is a power of 2 and fits page boundaries much better. Its function is analogous to a cell in ATM — to deliver low latency for short messages and to serve as a building block for large messages. The format of the control bytes in the micropacket is described in Table 1. Table 2 shows that the contents of the corresponding thirty-two data bytes depend in part on what the micropacket type is, i.e., the first four bits (bits 0-3) of the control bytes.

Figure 2 shows that the 24-byte HiPPI-6400 header consists of two headers — the media access control (MAC) header and the IEEE 802.2 LLC/SNAP header

Control Information (8 bytes)	
Bit #	Function
0-3	Micropacket type
4-5	Virtual channel selector
6-13	Transmit sequence number
14-21	Receive sequence number (ACK)
22	Tail bit (end of message)
23	Error (unrecoverable upstream error)
24-29	Credit update value
30-31	Virtual channel # for credit update
32-47	End-to-end CRC
48-63	Link-level CRC

Table 1: Contents of the Control Bytes in a Micropacket.

— and ten fields. The MAC header is identical to the IEEE 802.3 header except that the length field (M_len) is 32 bits rather than 16 bits as in IEEE 802.3. The reason for this difference is due to our need for HiPPI-6400 to handle larger messages. The D_ULA and S_ULA fields represent the 48-bit IEEE Universal LAN addresses for the source and the destination. The IEEE 802.2 LLC/SNAP header carries the EthernetType, which selects the upper-layer protocol.

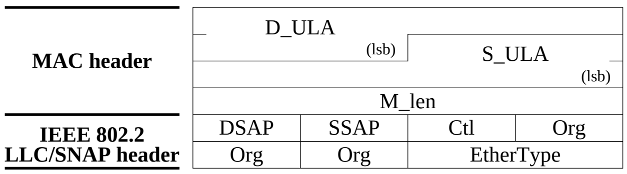

Figure 2: HiPPI-6400 Header

3.2 Virtual Channels (VCs) and Routing

HiPPI-6400 provides for four virtual channels (VCs) in each direction on each physical link. VCs are assigned based on specific message sizes (VC0 $\leq$ 2 KB, VC1 $\leq$ 128 KB, VC2 $\leq$ 128 KB, and VC3 $\leq$ 4 GB) and transfer methods. (The number of VCs is deliberately limited to four as opposed to the almost limitless number in ATM because the buffering for the HiPPI-6400 switch needs to be on-chip for performance reasons.) Each message is made up of a number of micropackets which are transmitted and delivered in-order over a single VC. The VC number does not change as the message travels from the originating source to the final destination over one or more links.

In contrast to the virtual connections of ATM and the end-to-end connections used in HiPPI-800, HiPPI-

Micropacket Type (Bits 0-3)	Data Bytes	Transmit Sequence #	Receive Sequence #	Credit Update
Header	24-byte header, 8-byte user data	Yes	Yes	Yes
Data	32-bytes of user data	Yes	Yes	Yes
Administration	Administrative message	Yes	Yes	Yes
Credit-only	32 bytes of 0s	Yes	Yes	Yes
Null	32 bytes of 0s	Invalid	Yes	Invalid
Reset or Initialize	32 bytes of 0s	Invalid	Invalid	Invalid

Table 2: Capabilities for each Micropacket Type

6400 switches use wormhole routing. In wormhole routing, a message can be sent into the network without prior knowledge of whether or not a free path is available to the destination host. Thus, time-consuming circuit set-up and tear-down are not required, and the switches do not need to maintain large amounts of state information. On the other hand, if there is contention for the output port of a switch on the same VC, the latter arriving message is blocked until the earlier arriving message has completed transmission. To prevent a large message from blocking a small message, the messages are transmitted over different VCs. This is in contrast to HiPPI-800 where a large message may block any number of messages queued for the same output port.

3.3 Flow Control

HiPPI-6400 provides for link-level, hop-by-hop, credit-based flow control (a la X.25 but without being connection-oriented) to prevent overrunning buffers at intermediate switches and at the destination host. Figure 3 shows that credits are assigned on a VC basis. Thus, congestion on one VC will not stall traffic on another VC. As in TCP, the destination grants credits to match the number of free receive buffers for a particular VC. The source end of the link then consumes credits as it moves micropackets from the VC buffers to the output buffer. We again stress that the flow control is on a per-link (or per-hop) basis. Thus, the source may be the originating source host or an intermediate switch; likewise, the destination may be an intermediate switch or the destination host.

In order to achieve 6400 Mb/s, buffers had to be on-chip and limited to 10 KB for each of the four VCs. This buffer-size constraint limits the physical-link length to roughly one kilometer; otherwise, speed degradation occurs. At 6400 Mb/s, 5 ns/m propagation delay, and 10 KB in flight, the maximum round-trip distance is 2.5 km. However, because this number does not include any processing overhead, the link distance was specified as a maximum of 1 km.

To ensure reliable transmission between a source and destination, as shown in Figure 3, HiPPI-6400 uses a go-back-N retransmission scheme. If an error is detected in a micropacket or a micropacket fails to arrive at the destination host, then that micropacket and all the micropackets transmitted after it must be retransmitted. An error in a micropacket is detected by either the link-level cyclic redundancy check (LCRC) or the end-to-end cyclic redundancy check (ECRC); these CRCs are checked at the destination side of a link at the input buffer (see Figure 3).

The destination/receiver acknowledges micropackets by returning the highest sequence number of contiguously acceptable micropackets. Hence, if a micropacket is received in error, then the receive sequence number stays on the value of the last correctly received micropacket. However, rather than rely on duplicate ACKs to signal retransmission as in TCP, a timeout mechanism at the sender detects that a transmitted micropacket was not acknowledged and retransmits all micropackets starting with the unacknowledged one. While a timeout mechanism may not be appropriate for a link with a long delay, it is preferred when the link delay is low, such as with HiPPI-6400 at 10 ms and when adequate buffering is available. The 8-bit sequence numbers allow up to 256 unacknowledged micropackets or nearly 10 KB, the size of the receive buffer.

Finally, we note that retransmission is independent of the both the VC used and the credit information. That is, retransmission occurs between the output buffer of the source and the input buffer of the destination (on a link basis) while VC and credit information pertain only to the VC buffers, as illustrated in Figure 3.

3.4 Error Detection

HiPPI-6400 implements error detection through the use of two distinct 16-bit cyclic redundancy checks

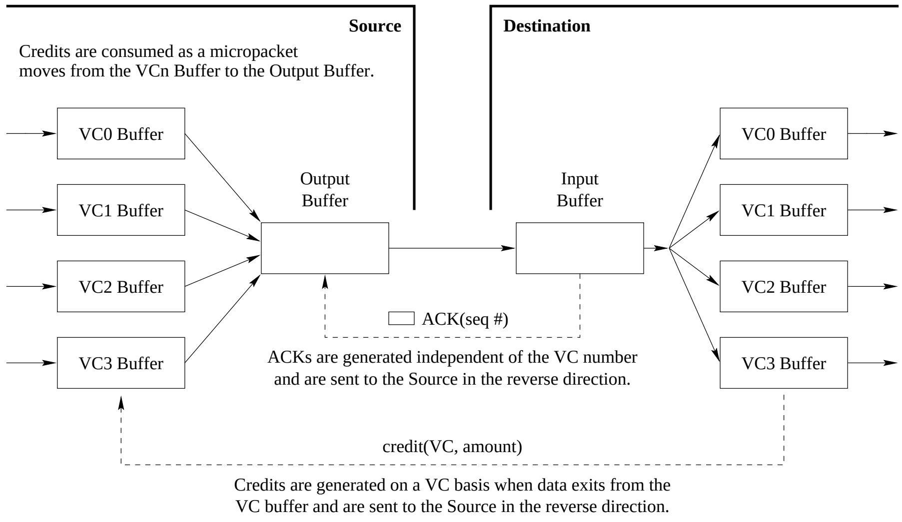

Figure 3: Flow Control and Retransmission

(CRCs): a link-level CRC (LCRC) and an end-to-end CRC (ECRC). The ECRC polynomial $(x^{16}+x^{12}+x^3+x+1)$ covers the data bytes of all the micropackets in a message; this includes the header micropacket and all of the data micropackets up to this point in a message. However, ECRC does not cover the control bits. Being an end-to-end CRC, the ECRC remains unchanged from the source host to the destination host, e.g., through switches and bridges. Of important note is the fact that the ECRC is accumulated over an entire message. For example, as shown in Table 3, the ECRC of the second micropacket covers the information from both the first and second micropacket; the ECRC of the third micropacket covers the information from the first, second, and third micropacket, and so on.

The LCRC polynomial $(x^{16}+x^{12}+x^5+1)$ covers all of the data and control bits of a micropacket with the exception of itself, i.e., bits 48-63 of the control information. It is initialized for each micropacket and must be calculated from scratch for each link since some values change from hop to hop, e.g., received sequence number and credit information.

Both CRCs are checked at the input buffer of each HiPPI-6400 node, whether it be a switch or an end device. The combination of the two 16-bit CRCs provides a stronger check than a single 16-bit CRC for link-level checking of individual micropackets. Hoffman [5] showed that no errors go undetected unless at least six bits in a micropacket are in error, i.e., 1 in 1.86 billion bits. Not only must there be at least six bits in error, but the bits must be strategically located and non-contiguous. With respect to a single 32-bit CRC, two separate 16-bit CRCs are easier to calculate than a single 32-bit one.

3.5 Hardware Design

In HiPPI-6400, data is transmitted in parallel over the cable (hence its name) and strobed with the clock signal. Figure 4 shows the signal lines that enable parallel data transfer between two end devices. This parallel architecture allows the use of CMOS circuits and available drivers and receivers, thus providing a tremendous cost and time-to-market savings. (A serial implementation of HiPPI-6400 would have required a serial link rate of roughly 10 Gb/s, which is too costly with optics and impossible to do with copper cable.)

3.5.1 Media Interfaces

HiPPI-6400 defines a parallel copper cable interface for a 16-bit data system using a total of twenty-three 500-Mbaud signals in each direction (16 for data, 4 for control, 1 for framing, and 2 for clocks), as shown in Figure 4. The cable assembly provides differential paths for 46 signals, 23 in each direction. The characteristic impedance and maximum distance of the cable are 150 W and 40 m, respectively. While the cable to support this speed and distance is expensive, it is available through commercial vendors.

HiPPI-6400 also defines a local electrical interface with the intent to drive parallel optical transceivers on the same circuit board. The optical interface is defined for an 8-bit rather than a 16-bit data system, using a total of twelve 1-Gbaud signals in each direction (8 for data, 2 for control, 1 for framing, and 1 for clock); see the numbers in parentheses in Figure 4. Due to the

Packet	Contents	LCRC Coverage	ECRC Coverage
1	Header, Bytes 0-7	Header, Bytes 0-7, c00-c47	Header, Bytes 0-7
2	Bytes 8-39	Bytes 8-39, c00-c47	Header, Bytes 0-39
3	Bytes 40-71	Bytes 40-71, c00-c47	Header, Bytes 0-71
4	Bytes 72-103	Bytes 72-103, c00-c47	Header, Bytes 0-103
5	Bytes 104-135	Bytes 104-135, c00-c47	Header, Bytes 0-135

c00-c47 represent bits 00-47 of the control information in a micropacket.

Table 3: Coverage of Cyclic Redundancy Codes.

fact that the optical interface was not as far along in design and standardization, it was split off into a separate standards document called the High-Performance Parallel Interface — 6400 Mb/s Optical Specification (HiPPI-6400-OPT). At the present time, a number of optical variants are being explored. One variant uses 850-nm laser arrays and 62.5/125-micron multimode fiber and may use an open-fiber control system to detect an open fiber and power down the lasers to avoid potential eye damage. A second variant uses the same lasers and fiber but decreases the power to avoid eye-safety problems. And a third variant uses 1300-nm lasers and either single-mode or multimode fiber.

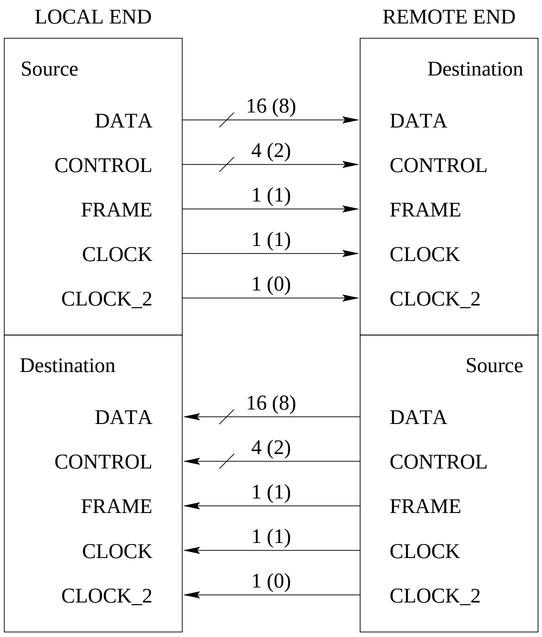

Figure 4: HiPPI-6400 Signal Lines for Copper Cable (Optical Fiber)

3.5.2 AC Coupling

To effectively drive long cables, signals should be AC-coupled and DC-balanced. AC coupling separates the ground paths between the end devices and avoids ground loops while DC balance indicates that signal is above the switching threshold as much as it is below the threshold. These characteristics greatly improve jitter and signal quality over encoding schemes such as Manchester encoding. HiPPI-6400 specifies 4B/5B encoding which inserts extra bits into the bit stream so as to break up long sequences of 0s or 1s. Specifically, every 4 bits of actual data are encoded in a 5-bit code (w, x, T, Y, and z in Figure 5; hence the name 4B/5B.

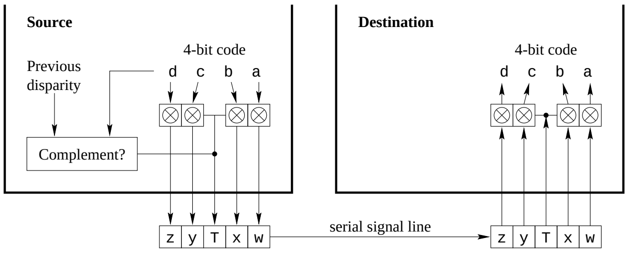

Figure 5: 4B/5B Encoder/Decoder

Figure 5 shows a simplified schematic of a 4B/5B encoder/decoder. For each signal line (only one such line is shown in the figure), a running count called the Disparity Count is kept of all the ones and zeros transmitted on that line since the link was reset. The Disparity Count is incremented for each "1" transmitted and decremented for each "0" transmitted. The actual 5-bit code which is transmitted depends on the current value of the Disparity Count and the input data 4-bit code (a, b, c, and d). For example, if the Disparity Count is positive (i.e., more "1"s than "0"s) and the incoming 4-bit data stream also has more "1"s than "0"s, then the incoming 4-bit code is complemented in order to generate more "0"s, and the "T" bit is set to 0. At the receiving end, the incoming bits pass straight through if $T = 1$ or are complemented if $T = 0$.

3.5.3 Deskewing

The CLOCK signal strobes the other HiPPI-6400 signals. Because it is carried on a separate line, it negates the need for clock recovery circuits on every data line. HiPPI-6400 allows up to 10 ns of differential skew between the signal lines at the receiver, and the deskew circuits dynamically adjust every 10 ms in a process called "training." Because the deskew adjustment takes one micropacket time (40 ns) every 10 ms, only 0.04is incurred as overhead.

Figure 6 shows a block diagram of the deskew circuit on one signal line; there are 20 of these circuits per interface chip. Each received signal drives a tapped delay line, and the output is selected from one of the taps.

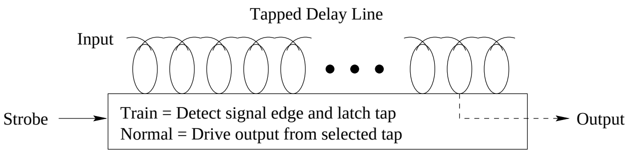

Figure 6: Deskew Circuit

3.5.4 An Initial Prototype

In late 1998, SGI demonstrated an initial prototype of the HiPPI-6400-PH standard called the Super-HiPPI Media Access Control (SuMAC) chip. SGI is producing this SuMAC chip for use in the 3.1-TFLOP ASCI Blue Mountain supercomputer and other machines which currently use HiPPI-800 interconnect technology. It demonstrates latencies of 90 ns in one direction and 120 ns in the other for a total end-to-end latency of 210 ns. Because the switch path between two hosts would most likely contain two SuMAC chips, one for input and one for output, the end-to-end latency would roughly be 540 ns. At 40 ns/micropacket and a cable delay of 1.5 ns/m, this translates to a worst-case latency of about 10 ms. Typical latencies would be on the order of 1 ms.

Our initial testing of the prototype SuMAC also demonstrated nearly line-rate bandwidth speeds. Our benchmark tests show that the SuMAC chip can achieve raw bandwidths of at least 6.08 Gb/s in each direction. Unfortunately, we were unable to test beyond this bandwidth due to limitations in our test equipment, not because of limitations in the SuMAC chip.

The initial prototype, however, was not without its share of problems. For example, initial testing of the SuMAC led us to believe that cable distances might need to be limited to less than 30 meters due to inexplicable bit errors. Subsequent examination of the cables showed that the equalization circuits, which are built into the back shell of long cables, were installed on the wrong end. Current versions of the SuMAC are now free of known bugs.

Along with the SuMAC, SGI has also developed another prototype chip, the SHAC (Super HiPPI Access Controller) which contains speed enhancements for TCP/IP and a new protocol OS-bypass protocol called Scheduled Transfer (ST). This new chip performs IP checksums, and for ST, performs a number of low-level messaging operations such as clear-to-send, enhancing the HiPPI-6400 network performance.

We initially benchmarked SGI's HiPPI-6400 implementation on two 32-node SGI Origin 2000s with alpha ST drivers. The network interface cards (NICs) contained both SuMAC and SHAC 2.0 chips and were installed on the Origin XIO bus. An Essential 32-port HiPPI-6400 switch connected the machines together, so all of our benchmark measurements included the switch-induced delays. The combination of the two chips exhibited latencies of 7 μs and sustained data rates of 3.6 Gb/s for a single direction. Running in both directions simultaneously with ST produced sustained bandwidths of 6.4 Gb/s. The HiPPI-6400 bandwidth numbers in both cases was limited by the memory architecture of the Origin 2000, not by the network. When running TCP/IP over HiPPI-6400, our tests produced a unidirectional bandwidth of 2.2 Gb/s.

4 The Transition from HiPPI-800 to HiPPI-6400

Because mass production and deployment of HiPPI-6400 is not expected for at least several months, our ASCI Blue Mountain supercomputer needs to rely on HiPPI-800 as its high-speed parallel interconnect for at least that long. Unfortunately, as mentioned earlier, HiPPI-800 does not ensure the reliable transmission of information at the hardware level like HiPPI-6400 does, therefore reliable transmission must be implemented in the software. As a result, SGI and LANL have installed a thin messaging layer in SGI's version of MPI over HiPPI-800.

While HiPPI-6400 technology is not currently being deployed widely, it is commercially available. For instance, HiPPI-6400 is also commercially known as and available as Gigabyte System Network (GSN), and HiPPI-6400 NICs containing the SuMAC and SHAC 2.0 chips are available for SGI machines. Unfortunately, production-quality software for this leading-edge hardware is not yet available.

Additional HiPPI-640 hardware developments include the SPIRIT chip from Power Micro Research (PMR). The SPIRIT chip, similar in nature to SGI's proprietary SHAC chip, will run on PCI-X based PC implementations. Other companies such as Genroco, Inc. and ODS, Inc. have developed or are developing switches, bridges, and PCI-X NICs. In the meantime, several of the major workstation manufacturers (Sun, IBM, and HP) are working to develop HiPPI-6400 NICs for their machines as part of the ASCI PathForward project.

When HiPPI-6400 arrives and the ST protocol is standardized and deployed, the software-based reliable messaging layer will be removed from MPI, alleviating the network/application bandwidth bottleneck. LANL is preparing for this transition by installing a network testbed to allow for testing of NICs, low-cost switches, bridges and software as they are manufactured. In the long term, LANL plans to build HiPPI-6400 production-quality networks, leveraging the above technologies.

5 Conclusion

Because HiPPI-800, Gigabit Ethernet, and Myrinet are unable to keep up with today's and tomorrow's bandwidth-hungry applications, e.g., remote visualization, a faster high-speed interconnect is needed. In this paper, we presented such an interconnect called HiPPI-6400. HiPPI-6400 represents the next-generation interconnect beyond the current gigabit interconnect standards. In addition to ensuring maximum compatibility with the Ethernet, Gigabit Ethernet, ATM, and HiPPI-800 installed base, HiPPI-6400 also leveraged many of the "best" features of each in its design.

Our initial benchmarks on HiPPI-6400 technology demonstrate that we are able to exceed the *delivered* bandwidth numbers of any other commercially available networking technology. At sustained data rates of 3.6 Gb/s (unidirectional) and 6.4 Gb/s (bidirectional), HiPPI-6400 far exceeds even the maximum bandwidth of Gigabit Ethernet and Myrinet.

As LANL changes its guard from HiPPI-800 to HiPPI-6400 production-quality networks, LANL will be providing a high-bandwidth gateway to applications such as remote visualization which routinely require bandwidths on the order of hundreds of gigabits per second. To attain these bandwidth numbers, HiPPI-6400 will be striped as many as 25-ways. In addition, there is ongoing work in HiPPI-6400-OPT to develop and standardize electro-optical interfaces for higher performance, and perhaps, to drive optical signals directly into silicon.

Acknowledgments

The Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy. SGI, Inc., with Dr. Greg Chesson leading their efforts, has contributed the majority of HiPPI-6400 technical innovations. The HiPPI standards committee has worked hard in standardizing and documenting HiPPI-6400 and the Scheduled Transfer as an ANSI standard.

References

[1] N. Boden, D. Cohen, R. E. Felderman, A. Kulawik, C. L. Seitz, J. N. Sizovic, and W.-K. Su, "Myrinet: A Gigabit per Second Local Area Network," *IEEE Micro*, February 1995.

[2] W. Dally and C. L. Seitz, "Deadlock Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Transactions on Computers*, vol. 36, pp. 547-553, 1987.

[3] T. von Eicken, D. Culler, S. Goldstein, and K. Schauser, "Active Messages: A Mechanism for Integrated Communication and Computation," *Proceedings of the International Symposium on Computer Architecture*, 1992.

[4] Gigabit Ethernet Alliance, *Gigabit Ethernet: Accelerating the Standard for Speed*, Whitepaper, 1997.

[5] J. Hoffman, *HiPPI-6400: Analysis of a High-Throughput Network Interface*, M.S. Thesis, University of Arizona, 1996.

[6] I. Philp and Y.-L. Liong, "The Scheduled Transfer Protocol (ST)," *IEEE Workshop on communication, Architecture, and Applications for Network-Based Parallel Computing (CANPC '99)*, Orlando, FL, January 1999.

[7] S. Pakin, M. Lauria, and A. Chien, "High-Performance Messaging on Workstations: Illinois Fast Messages (FM) for Myrinet," *Proceedings of Supercomputing '95*, San Diego, CA, 1995.

[8] High-Performance Parallel Interface (HIPPI) Standards Activities, http://www.hippi.org/cDOCS.html

[9] H. Tezuka, A. Hori, Y. Ishikawa, and M. Sato, "PM: An Operating System Coordinated High-Performance Communication Library," *High-Performance Computing and Networking '97*, April 1997.