Tachyon: A Gigabit Fibre Channel Protocol Chip

The Tachyon chip implements the FC-1 and FC-2 layers of the five-layer Fibre Channel standard. The chip enables a seamless interface to the physical FC-0 layer and low-cost Fibre Channel attachments for hosts, systems, and peripherals on both industry-standard and proprietary buses through the Tachyon system interface. It allows sustained gigabit data throughput at distance options from ten meters on copper to ten kilometers over single-mode optical fiber.

by Judith A. Smith and Meryem Primmer

Relentlessly increasing demands of computer systems continue to stress existing communication architectures to their limits. Even as processor speeds continue to improve dramatically, they are barely keeping up with increasing numbers of concurrently running applications and CPU-intensive applications, such as multimedia, with higher data throughput requirements. Additionally, as the number of interconnects between systems and I/O devices continues to increase, I/O channels become bottlenecks to system performance. A channel such as SCSI (Small Computer Systems Interface), which operates at a maximum throughput of 20 megabytes per second in fast and wide mode, simply cannot keep pace with ever-increasing processor speeds and data rate requirements.

Another challenge of contemporary computer systems is the trend to more widely distributed systems, which require greater interface distances. Current parallel bus interconnects between systems and their I/O devices cannot operate over the distances needed for true distributed systems, such as LANs spanning campus areas and high-availability applications requiring remote mirrored disks for disaster recovery. SCSI, for example, is limited to a distance of six meters single-ended (single wire per signal) and 25 meters differential (two wires per signal).

Current peripheral interconnect protocols are limited in the number of devices they can interconnect. For example, parallel SCSI can connect eight devices and 16-bit wide SCSI can connect 16 devices. In addition, peripheral connectors are becoming too large to fit into the shrinking footprints of systems and peripherals. Other SCSI limitations include half-duplex operation only, lack of a switching capability, inability to interconnect individual buses, and the need for customized drivers and adapters for various types of attached devices.

Computer room real estate also is becoming scarce and expensive, fueled by increasing numbers of racked computers, insufficient room to connect desired numbers of peripheral devices, and more complex cabling. At the same time, data storage requirements are skyrocketing as backups of terabytes of data are becoming commonplace. An additional problem is that ever-increasing amounts of data must be backed up over too-slow LANs, making timely, low-cost backups ever more difficult to accomplish.

For all these reasons, today's parallel bus architectures are reaching their limits. Fibre Channel provides solutions to many of these limitations. Fibre Channel is a forward-thinking solution to future mass storage and networking requirements. ***Article 11*** presents a technical description of Fibre Channel.

HP and Fibre Channel

Searching for a higher-performance serial interface, HP investigated a number of technologies. HP chose Fibre Channel over other serial technologies because it supports sustained gigabit data transfer rates (the fastest throughput of any existing interface), it allows networking and mass storage on the same link, and it is an open industry standard.

Although Fibre Channel faces the challenges of lack of market awareness and industry coordination and a perception that it can be expensive, it is a stronger contender than alternative serial technologies for a number of important reasons. It is an open industry standard and an approved ANSI standard, it has vendor support from switch, hub, and disk drive suppliers, it is extensible, offering three topologies and four data transfer rates, and it supports both networking and mass storage.

Fibre Channel's increased bandwidth provides distance flexibility, increased addressability, and simplified cabling. Fibre Channel has versatility, room for growth, and qualified vendor support. Mass storage suppliers are using Fibre Channel to interconnect subsystems and systems and to control embedded disk drives. Some midrange system (server) suppliers are using Fibre Channel as a clustering interconnect and for specialized networking. Fibre Channel supporters and developers include HP, Sun Microsystems, SGI, and IBM for workstations, HP, Sun, Unisys, and Compaq in the server market, HP,

Seagate, Quantum, and Western Digital for disk drives, and Data General's Clariion Business Unit, DEC, Symbios, Fujitsu, and EMC for disk arrays, in addition to over 100 other vendors (source: The Fibre Channel Association).

Fibre Channel's main virtue is that it works as a networking interface as well as a channel interface. Fibre Channel is one of three complementary networking technologies that HP sees as the next step upwards in network performance (see Fig. 1). The other two technologies are ATM (Asynchronous Transfer Mode), and IEEE 802.12, which is also known as 100VG-AnyLAN or 100BT.[1] Each technology has a set of strengths and is best suited to a particular networking niche. Combined, these technologies support all aspects of networking.

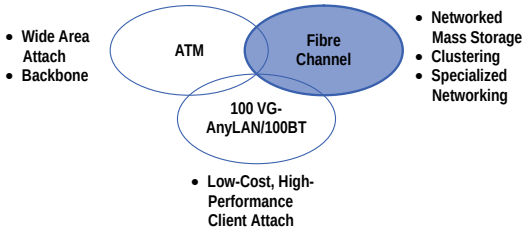


Fig. 1. *HP networking technologies.*

Both Fibre Channel and ATM are switched systems. They can share the same cable plant and encoding scheme and can work together in a network (Fig. 2). However, Fibre Channel and ATM standards are evolving independently to resolve different customer needs and objectives. ATM, which is telecommunications-based, is intended for applications that are characterized by "bursty" types of communications, thus lending itself to WAN applications. 100VG-AnyLAN or 100BT provides low-cost, high-performance client attachments. Fibre Channel is data communications-based and particularly well-adapted for networked and embedded mass storage, clustering, and specialized networking applications requiring sustained data flow rates.

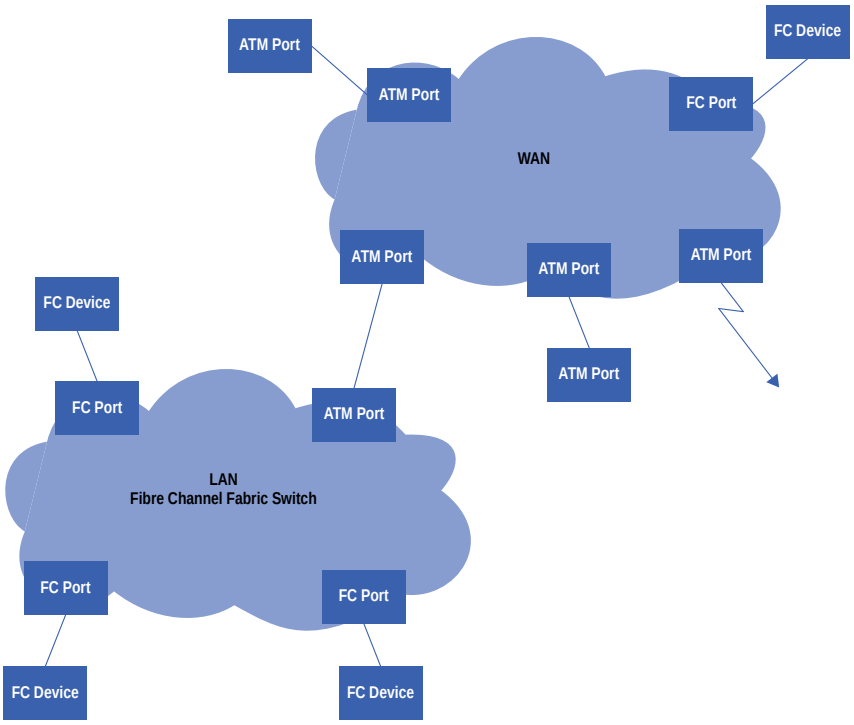


Fig. 2. *A network containing both Fibre Channel and ATM elements.*

In addition, Fibre Channel resolves the "slots and watts" problem that current symmetric multiprocessing systems have. For example, in 1995, three FDDI ports and six fast and wide SCSI ports were required to use fully the I/O capabilities of a symmetric multiprocessing HP server. Fibre Channel could support these I/O services with just three slots.

HP's vision of Fibre Channel is that it is at the core of the virtual data center containing diverse elements including:

- Fibre Channel switches connecting mainframes and supercomputers

- Network-attached disk arrays and storage archives
- ATM, FDDI, and Ethernet routers
- Imaging workstations
- Fibre Channel arbitrated loop disks and disk arrays
- High-performance mass storage peripherals
- Low-cost clients
- Clustered systems
- Video, technical, and commercial servers.

Interoperability and the establishment of a critical mass of Fibre Channel products are the keys to the success of Fibre Channel. HP is committed to Fibre Channel and is working with partners and standards bodies to ensure interoperability. HP is an active participant in the ANSI Fibre Channel Working Group, the Fibre Channel Association (FCA), and the Fibre Channel Systems Initiative, which has been integrated into the FCA. In 1994 HP purchased Canstar, a Fibre Channel switch company, which is now HP's Canadian Networks Operation. HP has developed Fibre Channel disk drives, gigabit link modules and transceivers,[2] system interfaces, and the Tachyon protocol controller chip, which is the subject of this article. HP is using Fibre Channel's versatility and speed for high-availability mass storage solutions and clustered system topologies.

Tachyon Chip

The system interconnect laboratory of the HP Networked Computing Division became interested in Fibre Channel in 1993 as a method of entering the high-speed serial interconnect market because Fibre Channel was the first technology that could be used for both networking and mass storage. HP decided to develop the Tachyon chip in mid-1993 after investigating which Fibre Channel controller chip to use in a Fibre Channel host adapter card under development for the HP 9000 Series 800 K-class server.[3] The investigation determined that no available chipset would meet the functional or performance requirements, so the decision was made to develop a controller internally.

The Tachyon chip (Fig. 3) implements the FC-1 and FC-2 layers of the five-layer Fibre Channel standard (see ***Article 11***). Tachyon's host attach enables low-cost gigabit host adapters on industry-standard buses including PCI, PMC, S-Bus, VME, EISA, Turbo Channel, and MCA. It is easily adaptable both to industry-standard and proprietary buses through the Tachyon system interface (a generic interface) and provides a seamless interface to GLM-compliant modules and components. GLM (gigabaud link module) is a profile defined by the FCSI (Fibre Channel Systems Initiative) and adopted by the FCA (Fibre Channel Association). It is a subset of the Fibre Channel FC-0 layer.[4]

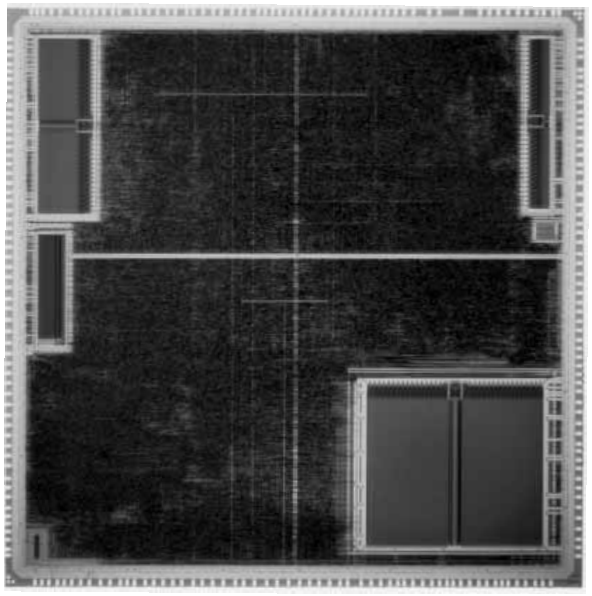


Fig. 3. *HP Tachyon Fibre Channel controller chip.*

Tachyon provides gigabit data throughput at distance options from 10 meters on copper to 10 kilometers over single-mode optical fiber. Tachyon host adapters save system slots, minimizing cost and cabling infrastructure.

Tachyon achieves high performance and efficiency because many of its lower-level functions are implemented in hardware, eliminating the need for a separate microprocessor chip. Functions such as disassembly of outbound user data from

sequences into frames, reassembly of inbound data, flow control, data encoding and decoding, and simple low-level error detection at the transmission character level are all built into hardware. One set of hardware supports all upper-level protocols. Errors and exceptions are offloaded to host-based upper-level software to manage.

Tachyon High-Level Design Goals

The Tachyon designers made several high-level design decisions early in the project. The primary design goal was to deliver sustained, full-speed gigabit performance while imposing the minimum impact on host software overhead. To accomplish this, Tachyon supports all Fibre Channel classes of service (see ***Article 11***), automatically acknowledges inbound frames for class 1 and class 2, handles NL_Port and N_Port initialization entirely in hardware, manages concurrent inbound and outbound sequences, and uses a messaging queue to notify the host of all completion information. To offload networking tasks from hosts, Tachyon is designed to assist networking protocols by supporting IP checksums and two different modes for splitting network headers and data.

The second major design goal was that Tachyon should support SCSI encapsulation over Fibre Channel (known as *FCP*). From the beginning of the project, Tachyon designers created SCSI hardware assists to support SCSI initiator transactions. These hardware assists included special queuing and caching. Early in the design, Tachyon only supported SCSI initiator functionality with its SCSI hardware assists. It became evident from customer feedback, however, that Tachyon must support SCSI target functionality as well, so SCSI target functionality was added to Tachyon SCSI hardware assists.

Tachyon Feature Set

To take advantage of Fibre Channel's high performance, Tachyon:[5]

- Provides a single-chip Fibre Channel solution.
- Manages sequence segmentation and reassembly in hardware.
- Automatically generates acknowledgement (ACK) frames for inbound data frames.
- Automatically intercepts and processes ACK frames of outbound data frames.
- Processes inbound and outbound data simultaneously with a full-duplex architecture.
- Allows chip transaction accesses to be kept at a minimum by means of host-shared data structures.

To provide the most flexibility for customer applications, Tachyon:

- Supports link speeds of 1063, 531, and 266 Mbaud.
- Supports Fibre Channel class 1, 2, and 3 services.
- Supports Fibre Channel arbitrated loop (FC-AL), point-to-point, and fabric (crosspoint switched) topologies.
- Provides a glueless connection to industry-standard physical link modules such as gigabaud link modules.
- Supports up to 2K-byte frame payload size for all Fibre Channel classes of service.
- Supports broadcast transmission and reception of FC-AL frames.
- Allows time-critical messages to bypass the normal traffic waiting for various resources via a low-latency, high-priority outbound path through the chip.
- Provides a generic 32-bit midplane interface—the Tachyon system interface.

To provide support for customer networking applications, Tachyon:

- Manages the protocol for sending and receiving network sequences over Fibre Channel.
- Provides complete support of networking connections.
- Computes exact checksums for outbound IP packets and inserts them in the data stream, thereby offloading the host of a very compute-intensive task.
- Computes an approximate checksum for inbound IP packets that partially offloads the checksum task from the host.
- Contains hardware header/data splitting for inbound SNAP/IP sequences.

To provide support for customer mass storage applications, Tachyon:

- Supports up to 16,384 concurrent SCSI I/O transactions.
- Can be programmed to function as either an initiator or a target.
- Assists the protocol for peripheral I/O transactions via SCSI encapsulation over Fibre Channel (FCP).

To reduce host software support overhead, Tachyon:

- Allows chip transaction accesses to be kept at a minimum by means of host-shared memory data structures.
- Manages interrupts to one or zero per sequence.
- Performs FC-AL initialization with minimal host intervention.

To provide standards compliance, Tachyon:

- Complies with Fibre Channel System Initiative (FCSI) profiles.

- Complies with industry-standard MIB-II network management.

To ensure reliability, Tachyon:

- Supports parity protection on its internal data path.
- Has an estimated MTBF of 1.3 million hours.

Fabrication

Tachyon is fabricated by LSI Logic Corporation using a 0.5-µm 3.3V CMOS process, LCB500K. The chip dissipates just under 4 watts and is contained in a 208-pin MQUAD package with no heat sink.

Tachyon Functional Overview

The host interface of the Tachyon chip is a set of registers used for initialization, configuration, and control and a set of data structures used for sending and receiving data and for event notification. This interface is very flexible and allows the customer to design an interface to Tachyon that best meets the capability, performance, and other requirements of a specific application.

Transmitting a Fibre Channel Sequence

To transmit an outbound sequence (see Fig. 4), the host builds several data structures and sets up the data to be transmitted. A data structure called the *outbound descriptor block* is built first. The outbound descriptor block provides much of the information Tachyon needs to send a sequence. The outbound descriptor block points to a data structure called the *extended descriptor block*, which points to data buffers containing the data for the sequence. The host then creates the Tachyon header structure, which contains important Fibre Channel-specific information such as which Fibre Channel class of service to use during sequence transmission. The host sets up the outbound descriptor block to point to the Tachyon header structure. The host then adds the outbound descriptor block to the outbound command queue.

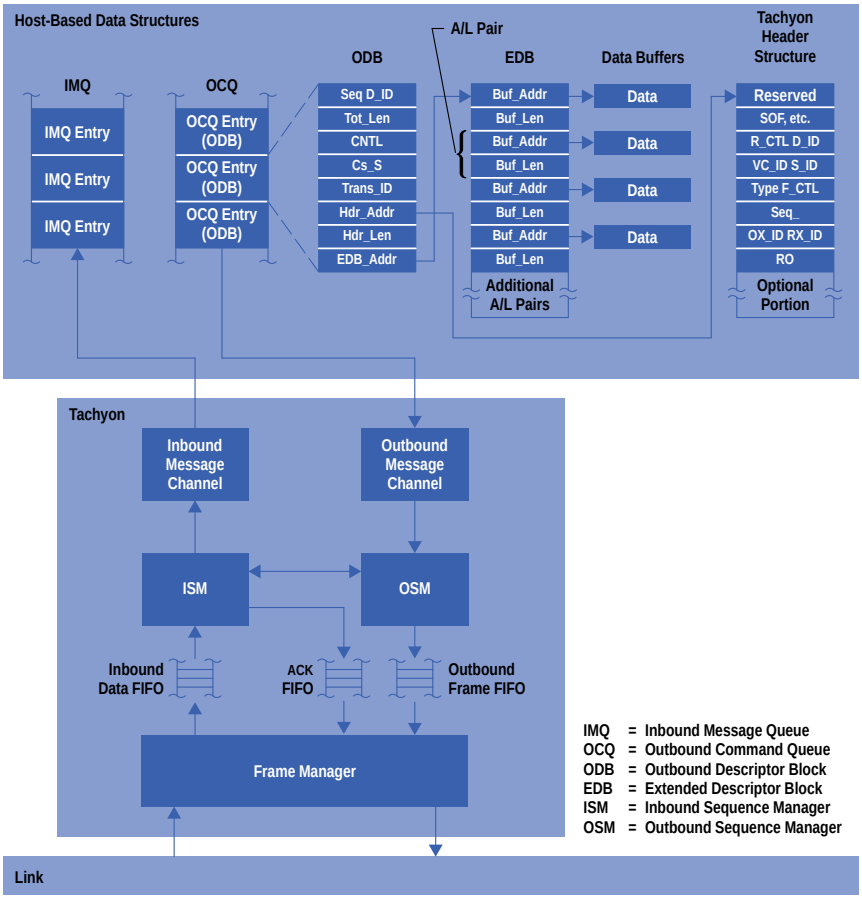


Fig. 4. *Transmit process overview.*

When Tachyon sees the new entry in the outbound command queue, it gets the outbound descriptor block from host memory via DMA. As Tachyon reads the Tachyon header structure to send the first frame of the sequence, it copies the header structure to internal registers for use in generating Fibre Channel headers for subsequent frames.

If this is class 1 service, after sending the first frame, Tachyon waits until it receives the ACK for the first frame of the sequence before continuing. Tachyon then inserts an identifier value, called the *responder exchange identifier* (RX_ID), which is returned in the ACK, into the Fibre Channel header on all subsequent frames of this sequence. Tachyon continues to transfer data from the host via DMA in frame-sized blocks and sends the frames with headers automatically generated from the previously stored header.

Tachyon keeps track of the frame count for the sequence. The Fibre Channel header for each frame contains an incremental count of the number of frames transmitted for the sequence along with the relative position of that frame within the sequence. As Tachyon sends the frames for the sequence, it also tracks flow control for the sequence using a Fibre Channel flow control method called end-to-end credit (EE_Credit). EE_Credit determines the number of frames that Tachyon can send to the remote destination without receiving an ACK. Each time Tachyon sends a frame, EE_Credit decrements. Each time Tachyon receives an ACK from the destination, EE_Credit increments. If EE_Credit goes to zero, Tachyon stops transmitting frames and starts a watchdog timer called the ED_TOV timer (error detect timeout value). The ED_TOV timer counts down until ACKs arrive. If ACKs arrive, Tachyon resumes transmission. If the ED_TOV timer expires, Tachyon sends a completion message to the host indicating that the sequence has timed out.

Once Tachyon has transmitted the last frame of a sequence and received all of the ACKs for the sequence, it sends a completion message to the host via the inbound message queue. This tells the host that it can deallocate all memory associated with this outbound descriptor block and inform any processes waiting on its completion. If a sequence terminates abnormally, Tachyon will notify the host of the error in the completion message.

Receiving a Fibre Channel Sequence

Fig. 5 shows an overview of the receive process. To enable Tachyon to receive data, the host first supplies Tachyon with two queues of inbound buffers. Two inbound queues are required because single-frame sequence reception and multiframe sequence reception are handled independently. Tachyon needs minimal resources to receive a single-frame sequence, but for multiframe sequences the chip needs to use additional resources to manage the reassembly of frames. Because of this resource demand, Tachyon can reassemble only one incoming multiframe networking sequence at a time. Tachyon supports the reception of single-frame sequences while reassembling a multiframe sequence. This process allows a host to receive short sequences while Tachyon is reassembling a longer incoming sequence.

Single-Frame Sequence Reception. The hosts uses the single-frame sequence buffer queue to inform Tachyon of the location of host-based buffers that Tachyon should use to receive sequences contained within a single frame. As Tachyon receives a single-frame sequence, it places the entire sequence, which includes the Tachyon header structure followed by the sequence data, in the buffer defined by the address from the single-frame sequence buffer queue. If the sequence is larger than one buffer size, the remaining data of the sequence is packed into the next buffers, as required, until all of the sequence is stored. Next, Tachyon sends an inbound_sfs_completion message to the host via the inbound message queue and generates an interrupt to the host.

Multiframe Sequence Reception. The host uses the multiframe sequence buffer queue to inform Tachyon of the location of host-based buffers that Tachyon should use to receive and reassemble incoming data that has been split into an arbitrarily large number of frames. When the first frame of a new sequence arrives, Tachyon copies the Tachyon header structure into the beginning of the next available multiframe sequence buffer. Tachyon packs the data payload of the frame into the next buffer following the buffer with the Tachyon header structure. As each new frame arrives, Tachyon discards the Fibre Channel header information and sends the data to the host. Tachyon packs this data into each of the buffers on the multiframe sequence buffer queue, obtaining a new buffer when the current buffer is full, until the entire sequence is stored. Once all the frames arrive and the sequence is reassembled in memory, Tachyon notifies the host that the entire sequence has been received by generating a completion message and placing it into the inbound message queue. Tachyon then generates a host interrupt to process the entire sequence.

Tachyon can also handle multiframe sequences that are received out of order. When Tachyon detects an out-of-order frame, Tachyon generates a completion message that indicates the in-order portion of the sequence and the last sequence buffer used. Tachyon passes the completion message to the inbound message queue, but does not generate an interrupt until all frames of the sequence are received. Next, Tachyon obtains the next available sequence buffer and copies the Tachyon header structure of this out-of-order frame into it. Then, into the next sequence buffer, it copies the data payload of this out-of-order frame. At this point, if the frames that follow the out-of-order frame are in order, Tachyon discards the Tachyon header structures and packs the data into the host buffers. Tachyon packs this data into each of the buffers on the multiframe sequence buffer queue, obtaining a new buffer when the current buffer is full, until the entire sequence is stored. If another frame arrives out of order from the previous out-of-order portion, Tachyon generates a new completion message and the process is repeated. When it receives the final frame of the sequence, Tachyon passes it to the host and generates a completion message. At this time, Tachyon generates a host interrupt to process the entire sequence. With the information in each of the Tachyon header structures that Tachyon passed to the host for each in-order portion and the information in the completion messages, the host has enough information to reorder the out-of-order multiframe sequence.

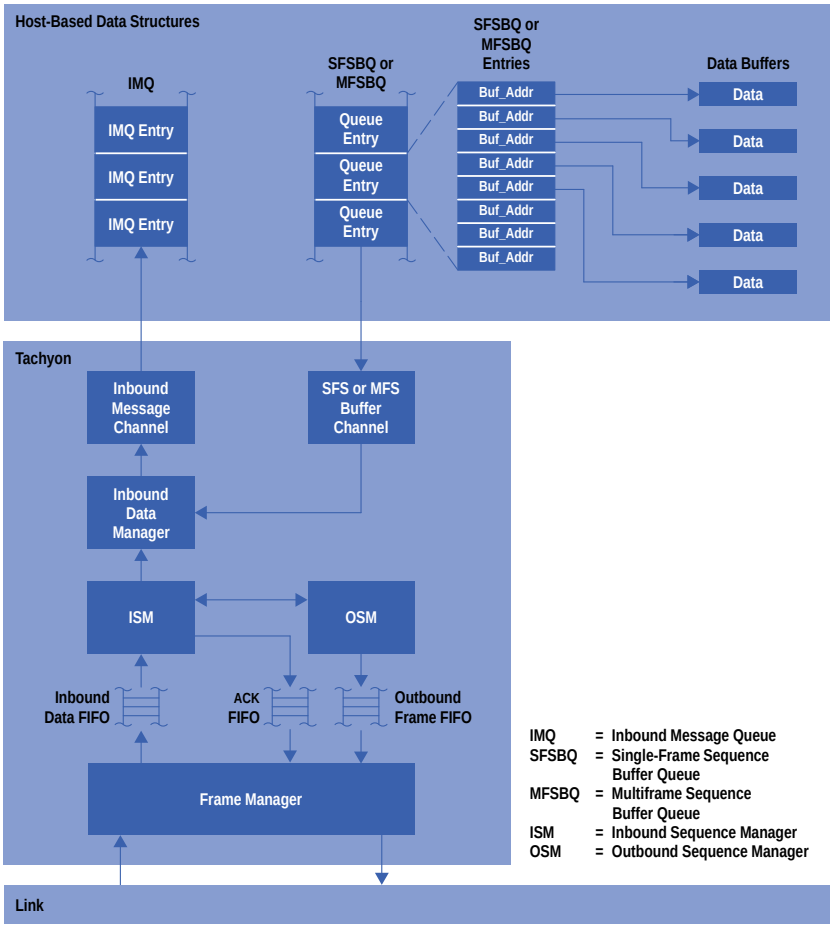


Fig. 5. *Receive process overview.*

Tachyon Internal Architecture

Tachyon's internal architecture is illustrated in Fig. 6. Each functional block in the architecture is described below.

Outbound Message Channel. The outbound message channel block manages the outbound command queue. It maintains the outbound command queue as a standard circular queue. The outbound message channel informs the outbound sequence manager block when an outbound command is waiting in host memory to be processed. When requested by the outbound sequence manager, the outbound message channel then reads one 32-byte entry from the outbound command queue and delivers it to the outbound sequence manager block for processing.

High-Priority Message Channel. The high-priority message channel block manages the high-priority command queue. The host can use the high-priority channel to send urgent single-frame sequences that need to bypass the dedicated outbound command queue. For example, the host could use the high-priority command queue to send special Fibre Channel error recovery frames that might not otherwise be transmitted because of a blocked outbound command queue. The high-priority message channel maintains the high-priority command queue as a standard circular queue. The high-priority message channel informs the outbound sequence manager block when a high-priority outbound command is waiting in host memory to be processed. When requested by the outbound sequence manager, the high-priority message channel reads one entry from the high-priority command queue and delivers it to the outbound sequence manager block for processing.

Outbound Block Mover. The outbound block mover block's function is to transfer outbound data from host memory to the outbound sequence manager via DMA. It takes as input an address/length pair from the outbound sequence manager block, initiates the Tachyon system interface bus ownerships, and performs the most efficient number and size of transactions on the Tachyon system interface bus to pull in the data requested.

Outbound Sequence Manager. The outbound sequence manager block is responsible for managing all outbound sequences. The outbound message channel, the high-priority message channel, and the SCSI exchange manager notify the outbound sequence manager block when they have data to send. The outbound sequence manager must determine which channel has priority. High-priority sequences have first priority, but the outbound sequence manager determines priority between

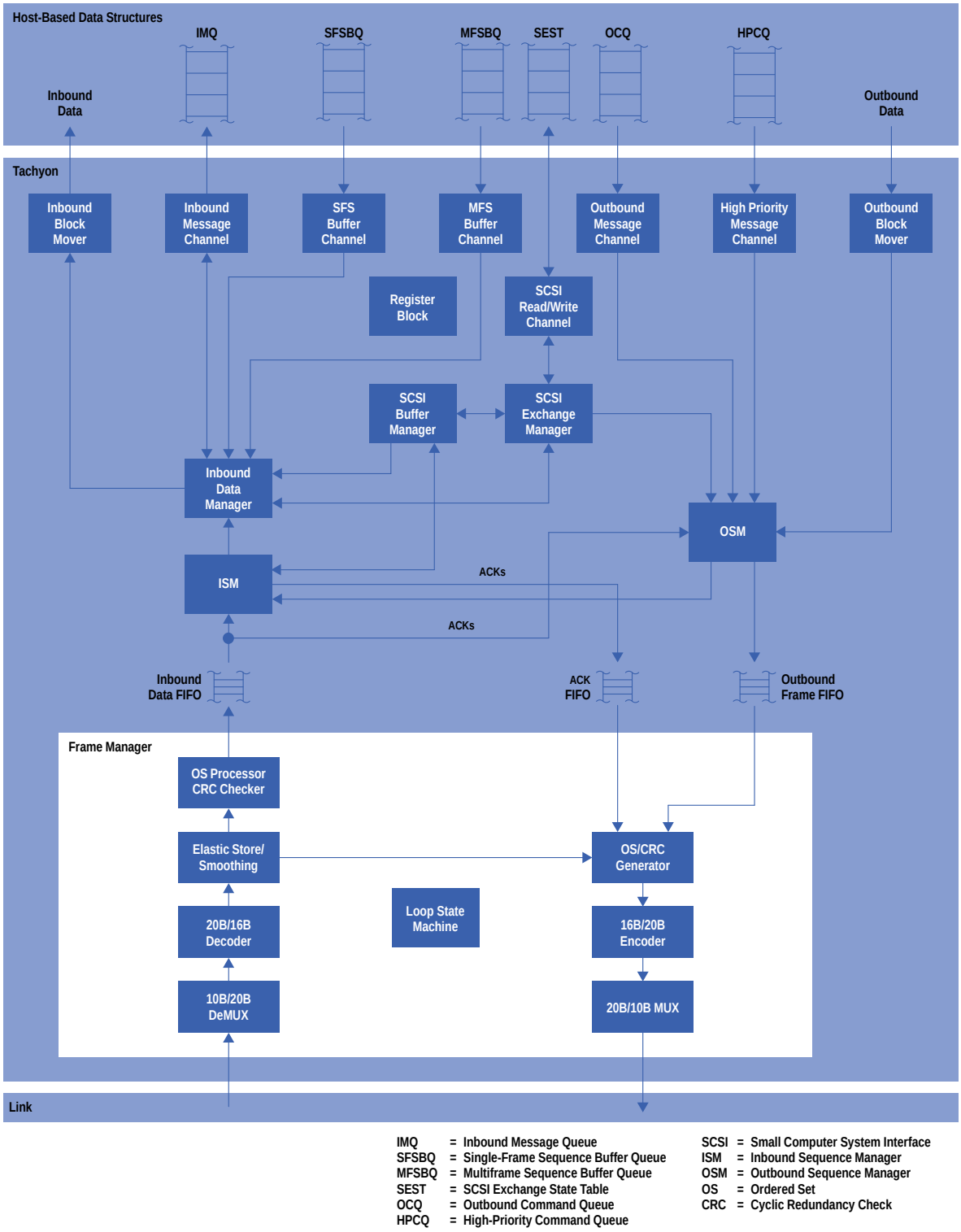


Fig. 6. *Tachyon internal architecture.*

networking and SCSI transactions using a fairness algorithm. Once priority is determined, the outbound sequence manager programs the outbound message channel to retrieve a data sequence from host memory and segment it into individual frames for transmission. The outbound sequence manager transmits the sequence, performs a TCP or UDP-type checksum on the sequence, verifies that each frame is acknowledged by the receiving node, handles errors if required, and sends a completion message to the host through the inbound message channel.

Outbound Frame FIFO. The outbound frame FIFO buffers data before transmission to prevent underrun. This FIFO is sized to hold one maximum-size frame. As Tachyon sends the current frame onto the link, the outbound frame FIFO is simultaneously filled with the next frame, maximizing outbound performance and reducing latency.

ACK FIFO. The ACK FIFO holds Fibre Channel class 1 and class 2 ACKs until they can be sent out by the frame manager.

Frame Manager. The frame manager is Tachyon's interface to the physical link module. The frame manager implements the N_Port state machine described in the FC-PH specification and the loop state machine described in the FC-AL specification. The frame manager can be configured to support link speeds of 1063, 531, and 266 megabits per second. It also implements initialization in hardware for both NL_Ports and N_Ports.

The frame manager implements portions of the FC-1 and FC-2 specifications. It is responsible for the FC-1 functions of transmitting and receiving Fibre Channel frames and primitives. It calculates and verifies the CRC for frame data, checks parity of the transmit data, and generates parity for the receive data. It generates primitives, encodes and receives Fibre Channel frames and primitives, decodes 10-bit or 20-bit physical link module data, and implements the running disparity algorithm in FC-1. The frame manager can be configured to generate interrupts to the host when certain link configuration changes occur to which the host must respond. The interrupt process occurs as part of N_Port initialization and loop initialization and any time the link has been disrupted.

The ordered set and CRC generator encapsulates data into FC-1 frames, generates a 32-bit cyclic redundancy check (CRC), writes it into the frame, and passes the encapsulated data to the 16B/20B encoder. The 16B/20B encoder converts outbound 16-bit-wide data into two 8-bit pieces, each of which is encoded into a 10-bit transmission character using the 8B/10B encoding algorithm.

The 20B/10B multiplexer is responsible for selecting the proper width, either 10 bits or 20 bits, of the data path for the specific type of physical link module being used. The data width and speed are specified by the parallel ID field of the physical link module interface.

The 10B/20B demultiplexer is responsible for receiving incoming encoded data, either 10 bits wide or 20 bits wide, from the physical link module and packing it into 20 bits for decoding. The data width and speed are specified by the parallel ID field of the physical link module interface.

The 20B/16B decoder is responsible for converting 20-bit-wide data received from the 10B/20B demultiplexer into two 8-bit data bytes.

The elastic store and smoothing block is responsible for retiming received data from the receive clock to the transmit clock. The elastic store provides a flexible data FIFO buffer between the receive clock and transmit clock domains. Receiver overrun and underrun can be avoided by deleting and inserting duplicate primitives from the elastic store as needed to compensate for differences in receive clock and transmit clock frequencies (within specifications).

The ordered set processor and CRC checker block is responsible for detecting incoming frame boundaries, verifying the cyclic redundancy check, passing the data to the inbound FIFO, and decoding and recognizing primitives.

Inbound Data FIFO. The inbound data FIFO buffers frame data while the frame manager verifies the CRC. It also serves as high-availability temporary storage to facilitate the Fibre Channel flow control mechanisms. This FIFO is sized to hold a maximum of four 2K-byte frames (including headers).

Inbound Sequence Manager. The inbound sequence manager is responsible for receiving and parsing all link control and link data frames. It interacts with the SCSI buffer manager block to process SCSI frames. The inbound sequence manager block is also responsible for coordinating the actions taken when a link reset is received from the frame manager block and for passing outbound completions to the inbound data manager block. The inbound sequence manager also manages class 1 Fibre Channel connections.

Inbound Data Manager. The inbound data manager routes incoming frames to their appropriate buffers in host memory, transferring SCSI FCP_XFER_RDY frames to the SCSI exchange manager, sending completion messages to the inbound message queue, and sending interrupts to the interrupt controller inside the inbound message channel.

Inbound Block Mover. The inbound block mover is responsible for DMA transfers of inbound data into buffers specified by the multiframe sequence buffer queue, the single-frame sequence buffer queue, the inbound message queue, or the SCSI buffer manager. The inbound block mover accepts an address from the inbound data manager, then accepts the subsequent data stream and places the data into the location specified by the address. The inbound block mover also accepts producer index updates and interrupt requests from the inbound message channel and works with the arbiter to put the interrupt onto the Tachyon system interface bus.

Inbound Message Channel. The inbound message channel maintains the inbound message queue. This includes supplying the inbound data manager with the address of the next available entry in the inbound message queue and generating a completion message when the number of available entries in the inbound message queue is down to two.

The inbound message channel also generates interrupts for completion messages, if necessary, and handles message and interrupt ordering. The completion message must be in the host memory before the interrupt. The inbound message channel also handles interrupt avoidance, which minimizes the number of interrupts (strobes on the INT pin on the Tachyon system interface bus) asserted to the host for each group of completions sent to the host.

Single-Frame Sequence Buffer Channel. The inbound single-frame sequence buffer channel manages the single-frame sequence buffer queue. It supplies addresses of empty single-frame sequence buffers to the inbound data manager and generates a completion message when the supply of single-frame sequence buffers runs low.

Multiframe Sequence Buffer Channel. The inbound multiframe sequence buffer channel manages the multiframe sequence buffer queue. It supplies addresses of empty multiframe sequence buffers to the inbound data manager and generates a completion message when the supply of multiframe sequence buffers runs low.

SCSI Buffer Manager. The SCSI buffer manager is responsible for supplying the inbound data manager with addresses of buffers to be used for inbound SCSI data frames. The SCSI buffer manager maintains a cache of 16 unique SCSI descriptor blocks. Each block contains eight SCSI buffer addresses. Depending upon the direction of the exchange, the originator exchange identifier (OX_ID) or the responder exchange identifier (RX_ID) of the current frame is provided by the inbound data manager block and is used to point to the correct entry in the SCSI exchange state table.

SCSI Exchange Manager. In conjunction with the SCSI exchange state table data structure, the SCSI exchange manager provides Tachyon with the hardware assists for SCSI I/O transactions. It converts SCSI exchange state table entries to outbound descriptor block format for the outbound sequence manager block's use. The SCSI exchange manager accepts SCSI outbound FCP_XFER_RDY frames from the inbound data manager block. It then uses the OX_ID or RX_ID given in the frame as an offset into the state table, reads the entry, and builds the outbound descriptor block.

Register Block. The register block consists of control, configuration, and status registers. These registers are used for initialization, configuration, control, error reporting, and maintenance of the queues used to transfer data between Tachyon and the host. Each register is 32 bits wide and may be readable or both readable and writable by the host depending upon its function.

SCSI Read/Write Channel. The SCSI read/write channel manages requests from the SCSI exchange manager and interfaces to the Tachyon system interface arbiter block.

Tachyon SCSI Hardware Assists

Tachyon supports SCSI I/O transactions (exchanges) by two methods. The first method uses host-based transaction management. In this method, the host transmits and receives the various sequences using the general transmit and receive processes. By using the host-based transaction management method, Tachyon reassembles only one SCSI unassisted multiframe sequence at a time. The second method uses Tachyon's SCSI hardware assists. With this method, Tachyon assists the host transaction management through the use of a shared host data structure called the SCSI exchange state table.

By using Tachyon's SCSI hardware assists, the host can concurrently reassemble up to 16,384 SCSI assisted sequences. Tachyon maintains an on-chip cache for up to 16 concurrent inbound transactions. Tachyon uses a *least recently used* caching algorithm to allow the most active exchanges to complete their transfers with the minimum latency.

The protocol for SCSI encapsulation by Fibre Channel is known as FCP. Fig. 7 shows an overview of the FCP read exchange and Fig. 8 shows an overview of the FCP write exchange. The exchanges proceed in three phases: command, data, and status.

SCSI Exchange State Table. The SCSI exchange state table is a memory-based data structure. Access is shared between Tachyon and the host. The SCSI exchange state table is an array of 32-byte entries. The starting address of the table is programmable and is defined by the SCSI exchange state table base register. The length of the table is also programmable and is defined by the SCSI exchange state table length register. Each used table entry corresponds to a current SCSI exchange, or I/O operation. Each entry contains two kinds of information: information supplied by the host driver for Tachyon to manage the exchange, and information stored by Tachyon to track the current state of the exchange. For initiators in SCSI write transactions, the outbound SCSI exchange state table entries contain information indicating where outbound data resides in memory and what parameters to use in the transmission of that data on the Fibre Channel link. For initiators in SCSI read transactions, the inbound SCSI exchange state table entries contain information indicating where inbound data is to be placed in memory. SCSI exchange state table entries are indexed by an exchange identifier (X_ID) —either the originator exchange identifier (OX_ID) or the responder exchange identifier (RX_ID). In an initiator application, the OX_ID provides the index into the SCSI exchange state table. In a target application, the RX_ID provides the index into the SCSI exchange state table.

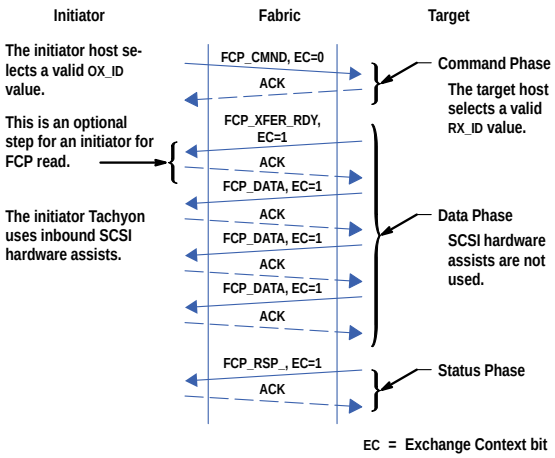


Fig. 7. *FCP read exchange overview.*

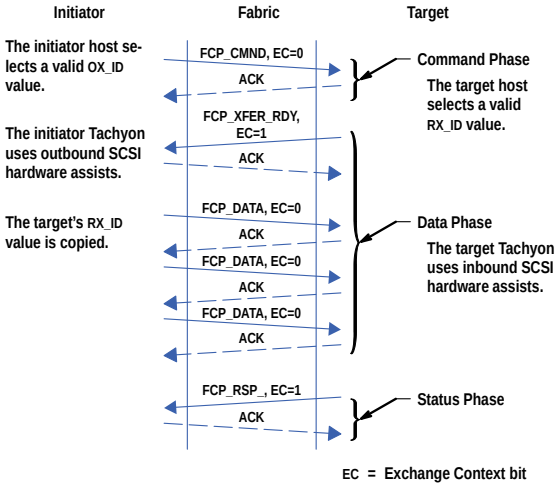


Fig. 8. *FCP write exchange overview.*

FCP Read Exchange—Tachyon as an Initiator. Fig. 9 shows the FCP read exchange host data structures for an initiator Tachyon. For the initiator host to receive inbound SCSI data, it selects a valid OX_ID value that points to an unused location in the SCSI exchange state table. The OX_ID value identifies this particular exchange. Using the OX_ID value, the initiator host builds a data structure called an inbound SCSI exchange state table entry. The inbound SCSI exchange state table entry includes the address of the SCSI descriptor block. The SCSI descriptor block defines the host buffers that Tachyon will use to store the received read data.

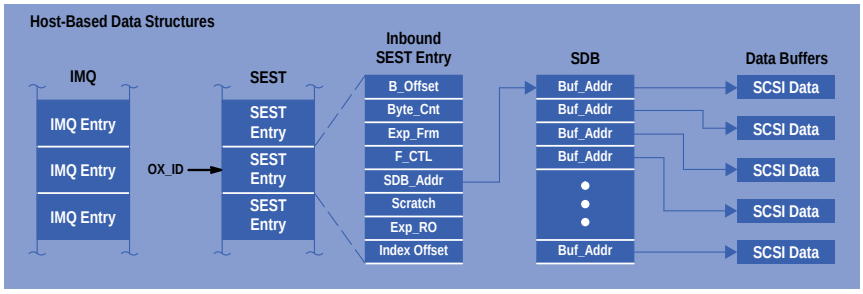


Fig. 9. *FCP read exchange initiator host data structures.*

In the command phase, once the host creates the inbound SCSI exchange state table entry, it creates an FCP_CMND for an FCP read exchange. The initiator Tachyon sends the FCP_CMND to the target via the outbound command queue.

In the data phase, the initiator Tachyon may receive an FCP_XFER_RDY from the target. This is an optional step for an initiator in an FCP read because the data frames contain all the information needed to process them. When Tachyon receives the optional FCP_XFER_RDY from the target, it acknowledges the frame if appropriate and discards the FCP_XFER_RDY. As each data frame is received, the SCSI exchange manager uses the OX_ID to access the appropriate inbound SCSI exchange state table entry for the address of the SCSI data buffer. The SCSI descriptor block and the relative offset of the data frame determine where data is to be placed in host memory. Tachyon maintains an internal cache of 16 inbound SCSI exchange state table entries. If the SCSI exchange state table information associated with the received frame is not in cache, Tachyon writes the least recently used cache entry back to the host SCSI exchange state table. Tachyon then fetches into cache the inbound SCSI exchange state table entry associated with the received frame and transfers the read data to host memory via DMA. The initiator Tachyon automatically handles both single and multiple data phases for inbound data transfers.

In the status phase, when the data phase is complete, the initiator Tachyon receives an FCP_RSP from the target. The FCP_RSP is a Fibre Channel information unit that contains status information that indicates that the SCSI exchange has completed. The initiator Tachyon passes the FCP_RSP to the host via the single-frame sequence channel and sends a completion message to the initiator host. This informs the initiator host that the exchange is completed.

FCP Read Exchange—Tachyon as a Target. For FCP read exchanges for the target Tachyon, SCSI hardware assists are not used. Fig. 10 shows the read exchange target host data structures.

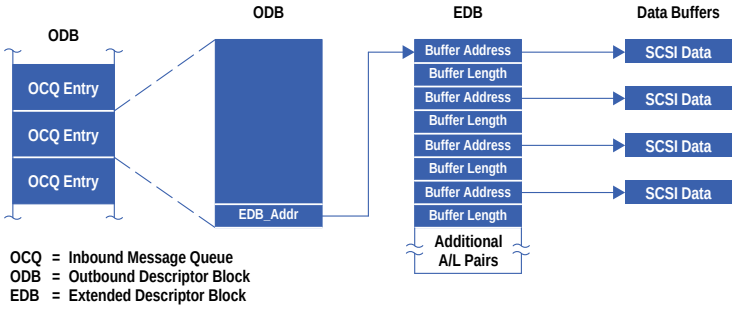


Fig. 10. *FCP read exchange target host data structures.*

In the command phase, the target Tachyon receives an FCP_CMND for an FCP read from the initiator and sends the FCP_CMND to the target host via the single-frame sequence channel. If configured to automatically acknowledge, the target Tachyon immediately returns an ACK (for class 1 and class 2) to the initiator. The target host selects a valid, unused RX_ID value. The RX_ID is placed into the header of the ACK and sent via the high-priority command queue.

In the data phase, the target host builds an outbound descriptor block that contains the extended descriptor block address. The target host builds an extended descriptor block that defines where the read data is located in the target host memory. The target host may send an FCP_XFER_RDY to the initiator host to indicate that it is ready to send the requested data. The target Tachyon sends the FCP_XFER_RDY(s) with the appropriate RX_ID value to the initiator Tachyon. Using the outbound command queue, the target Tachyon then sends the appropriate SCSI read data to the initiator via the outbound command queue.

In the status phase, the target Tachyon sends an FCP_RSP to the initiator to indicate that the exchange has completed.

FCP Write Exchange—Tachyon as an Initiator. Fig. 11 shows the FCP write exchange initiator host data structures. For the initiator host to perform an outbound data transfer, it selects an unused SCSI exchange state table entry whose index will be used as the OX_ID value. Using the OX_ID value, the initiator host builds an outbound SCSI exchange state table entry. The outbound SCSI exchange state table entry includes information about the frame size, the Fibre Channel class, and writable fields that the initiator Tachyon uses to manage the SCSI transfer. The outbound SCSI exchange state table entry also contains a pointer to the extended descriptor block that contains pointers to the data to be sent.

In the command phase, once the host creates the outbound SCSI exchange state table entry, the host creates the FCP_CMND for an FCP write exchange. The initiator Tachyon sends the FCP_CMND to the target via the outbound command queue.

In the data phase, when the initiator Tachyon receives the FCP_XFER_RDY from the target, it uses its SCSI hardware assists and manages the data phase for this FCP_XFER_RDY independent of the host. Tachyon uses the information in the SCSI exchange state table and the FCP_XFER_RDY to build an outbound descriptor block to be sent through the outbound state machine. If the outbound sequence manager is busy, the SCSI exchange state machine adds the request to a linked list of outbound transactions waiting for transmission. As the outbound sequence manager becomes available to process a SCSI transfer, the SCSI exchange state manager dequeues a waiting transaction and passes it to the outbound sequence manager. The outbound sequence manager transmits the write data to the target Tachyon.

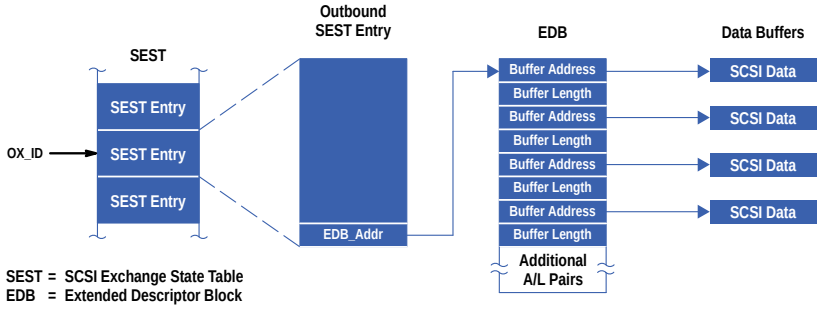


Fig. 11. *FCP write exchange initiator host data structures.*

If this FCP_XFER_RDY is part of a multiple-data-phase transfer, the initiator Tachyon passes this FCP_XFER_RDY as a single-frame sequence to the initiator host along with a completion message. The initiator host is responsible for managing the data phase for this multiple-data-phase transfer by using the general sequence moving services. The OX_ID field in the FCP_XFER_RDY is an index into the SCSI exchange state table and identifies the appropriate outbound SCSI exchange state table entry that points to the extended descriptor block in which the write data is located. Using the information in the outbound SCSI exchange state table entry, the initiator host builds an outbound descriptor block. The initiator Tachyon uses the outbound descriptor block to transmit the write data to the target.

In the status phase, after the data phase completes, the initiator Tachyon receives an FCP_RSP from the target. The initiator Tachyon passes the FCP_RSP to the host and sends a completion message to the initiator host. This informs the initiator host that the exchange has completed.

FCP Write Exchange—Tachyon as a Target. Fig. 12 shows the FCP write exchange target host data structures.

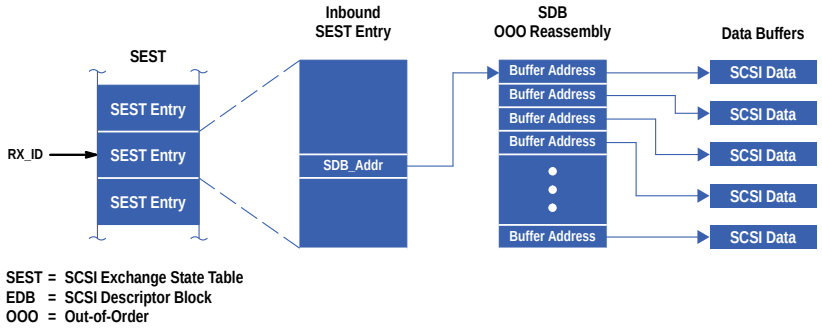


Fig. 12. *FCP write exchange target host data structures.*

In the command phase, the target Tachyon receives an FCP_CMND for an FCP write from the initiator. If configured to do so, the target Tachyon immediately returns an ACK to the initiator for class 1 and class 2. If Tachyon is not configured to return an ACK, the target host is responsible for sending the ACK. The target host selects a valid RX_ID value, places the RX_ID into the ACK header and sends the ACK via the high-priority command queue.

In the data phase, the target host selects an unused SCSI exchange state table entry whose index will be the RX_ID value. Using the RX_ID value, the target host builds the inbound SCSI exchange state table entry that points to the SCSI descriptor block. The SCSI descriptor block contains as many buffer addresses as necessary to receive the data. If the target host has enough buffers and is ready to receive all of the data from the initiator, the host programs the inbound SCSI exchange state table entry and the target Tachyon sends the FCP_XFER_RDY to the initiator via the outbound command queue. The initiator will send the data when it is ready.

The target Tachyon checks the RX_ID of the data frame to locate the appropriate inbound SCSI exchange state table entry. The inbound SCSI exchange state table entry helps Tachyon determine exactly where within the buffers the data is to be placed. When the last data frame is received, the target Tachyon sends a completion message to the target host to inform the target host that all frames for the SCSI sequence have been received.

In the status phase, the target host sends an FCP_RSP to the initiator via the outbound command queue.

Tachyon System Interface

The Tachyon system interface describes the backplane protocol for the Tachyon chip. It is a flexible backplane interface that allows customers to interface to Tachyon using many existing backplane protocols, including PCI, MCA, S-bus, EISA, VME, and Hewlett-Packard's High-Speed Connect (HSC) bus. The Tachyon system interface is capable of 100-Mbyte/s data transfers. Fig. 13 shows the Tachyon system interface signals.

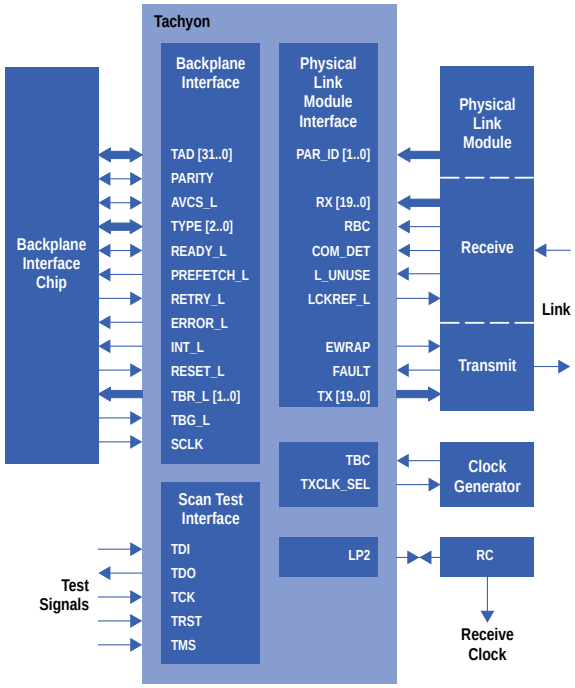


Fig. 13. *Tachyon system interface.*

The Tachyon system interface provides a basic transaction protocol that uses two major operations: write transactions and read transactions. Every transaction has a master and a responder. If the host is the master of a transaction, Tachyon is the responder in that transaction, and vice versa.

The master of a transaction drives an address and transaction type onto the TAD[] and TYPE[] buses, respectively, while asserting AVCS_L to indicate the start of the transaction. If Tachyon masters a transaction, the host, as the responder, uses READY_L as its acknowledgment signal. Similarly, if the host masters the transaction, Tachyon, as the responder, uses READY_L as its acknowledgment signal. A Tachyon system interface bus master has the choice of using one-word, two-word, four-word, or eight-word read and write transactions on the Tachyon system interface bus.

Tachyon System Interface Streaming

To maximize performance, the Tachyon system interface allows the host to configure the length of Tachyon's bus tenancy. When Tachyon obtains mastership of the Tachyon system interface and has more than one transaction to perform, Tachyon may extend its bus tenancy and perform several Tachyon system interface transactions (up to the maximum programmed limit) before releasing mastership. Table I shows how streaming increases performance over nonstreaming.

Table I
Tachyon Performance Savings with Streaming

	Tachyon Stream Size			
	1 Transaction (no streaming)	4 Transactions	16 Transactions	64 Transactions
Savings on Writes	0%	14.2%	18.5%	19.6%
Savings on Reads	0%	5.7%	7.2%	7.5%

Tachyon Host Adapter Requirements

A generic Fibre Channel host bus adapter board using the Tachyon chip contains the following:

- Backplane Connector. Connects the backplane interface chip to the system bus.
- Backplane Interface Chip. Enables the connection of the Tachyon system interface bus to PCI, EISA, HP-HSC or other bus.
- Tachyon Chip. HP's Fibre Channel interface controller.
- Physical Link Module. Tachyon interfaces to many GLM-compliant physical link modules currently in the marketplace. Types of modules include:
  - 1063-Mbit/s DB9 copper connectors for distances up to 30 meters.
  - 1063-Mbit/s shortwave laser physical link modules for distances up to 500 meters.
  - 1063-Mbit/s longwave laser physical link modules for distances up to 10 kilometers.
  - 266-Mbit/s shortwave laser physical link modules for distances up to 2 kilometers.

A block diagram of a typical host bus adapter is shown in Fig. 14. An HP-HSC Fibre Channel adapter is shown in Fig. 15. This adapter was developed by HP to provide Fibre Channel connection to the HP 9000 K-Class servers[3] for fast networking and mass storage.

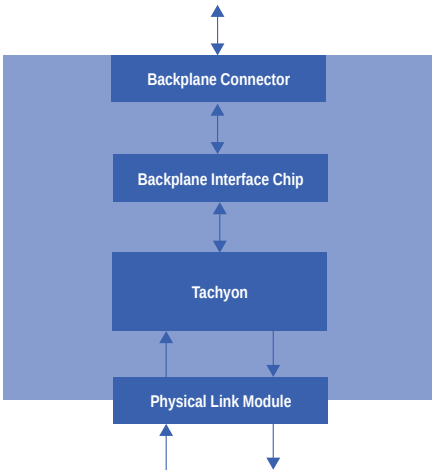


Fig. 14. *Block diagram of a typical host adapter board.*

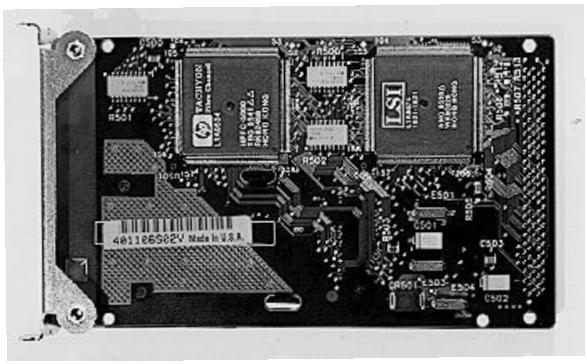
Development Tools

Effective tools were key to the success of the Tachyon chip.[6] The following tools were used in the development project:

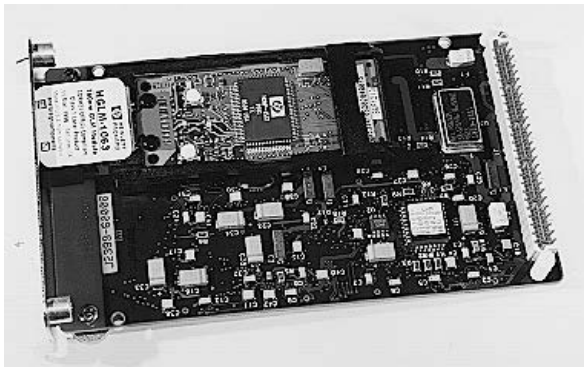
- Cadence Verilog-XL Turbo was used for interactive and batch Register Transfer Language (RTL) simulations, gate functional simulations, and dynamic simulation.
- Chronologic Verilog Compiled Simulator (VCS) was used for batch RTL simulations.
- Quad Motive was used for static timing analysis.
- Synopsys was used for chip synthesis.
- Veritools Undertow was used as a waveform viewer.
- Inter-HDL Verilint was used to check syntax and semantics for RTL code.
- SIL Synth was used to manage and launch synthesis jobs.
- History Management System (HMS),[5] written by Scott A. Kramer, was used as a revision control system.
- LSI Logic Corporation's Concurrent Modular Design Environment (CMDE) was used for floorplanning bonding, delay calculation, and layout.
- Hewlett Packard Task Broker[6] was used to distribute jobs to the various compute engines.

Verification Environment

The verification environment used for Tachyon was very sophisticated and automated. The Tachyon chip model (as either a Verilog RTL code or a gate-level netlist) was tested in simulation using Verilog's programming language interface capability. Test modules written in C or in Verilog provided stimulation to the Tachyon chip. Other test modules then verified the functionality of the chip and reported errors.



(a)



(b)

Fig. 15. *(a) HP HSC Fibre Channel adapter, side A. (b) HP HSC Fibre Channel adapter, side B.*

Conclusion

The Tachyon Fibre Channel interface controller provides a high-performance, single-chip solution for mass storage, clustering, and networking applications. Tachyon has been selected by many other companies as the cornerstone of their Fibre Channel product designs. As an understanding of the capabilities of Fibre Channel technology grows in the marketplace, Tachyon is expected to be present in a large number of new Fibre Channel products.

Acknowledgements

We wish to acknowledge the contributions of Randi Swisley, section manager for Tachyon and the HP HSC FC adapter projects, Bob Whitson, Bryan Cowger, and Mike Peterson, technical marketing, Tachyon chip architects Bill Martin, Eric Tausheck, and Mike Thompson, Fibre Channel standards committee members Kurt Chan and Steve Dean, project managers Margie Evashenk and Tom Parker, VLSI design team lead Joe Steinmetz, hardware and VLSI design team members Catherine Carbonaro, Dave Clark, Mun Johl, Ted Lopez, Bill Martin, George McDavid, Joseph Nuno, Pery Pearson, and Christi Wilde, Tachyon simulation team members Narayan Ayalasomayajula, Tony de la Serna, Murthy Kompella, Brandon Mathew, Mark Shillingberg, John Schimandle, Gordon Matheson, and Matt Wakeley, Tachyon and HSC FC adapter bringup team members Navjyot Birak, Bob Groza, and Donna Jollay, and customer support specialist Rick Barber, who is happy to answer questions from U.S. customers at 1-800-TACHYON. Special thanks to learning products engineer Mary Jo Domurat, who wrote the Tachyon user's manual, and disk support engineer Leland Wong, who provided photographs for this article.

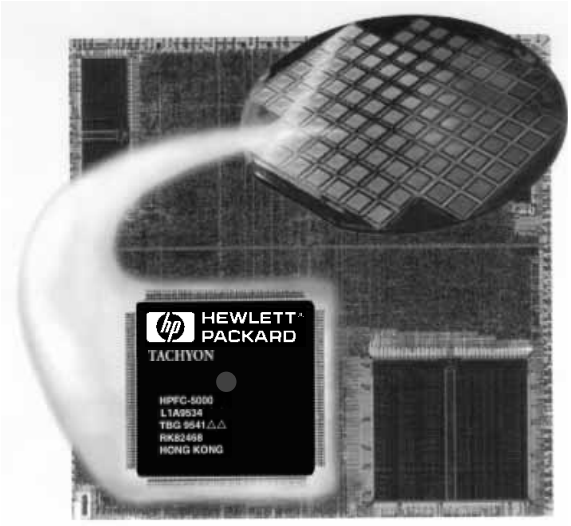
References

1. A.R. Albrecht and P.A. Thaler, "Introduction to 100VG-AnyLAN and the IEEE 802.12 Local Area Network Standard," *Hewlett-Packard Journal*, Vol. 46, no. 4, August 1995, pp. 6-12.
2. J.S. Chang, et al, "A 1.0625-Gbit/s Fibre Channel Chipset with Laser Driver," *Hewlett-Packard Journal*, Vol. 47, no. 1, February 1996, pp. 60-67.
3. M.J. Harline, et al, "Symmetric Multiprocessing Workstations and Servers System-Designed for High Performance and Low Cost," *Hewlett-Packard Journa*l, Vol. 47, no. 1, February 1996, pp. 8-17.
4. *Gigabaud Link Module*, FCSI-301, Rev. 1.0, Fiber Channel Association.

5. S.A. Kramer, "History Management System," *Proceedings of the Third International Workshop on Software Configuration Management (SCM3)*, June 1991, p. 140.
6. T.P. Graf, et al, "HP Task Broker: A Tool for Distributing Computational Tasks," *Hewlett-Packard Journal*, Vol. 44, no. 4, August 1993, pp. 15-22.

Bibliography

1. *Tachyon User's Manual*, Draft 4, Hewlett-Packard Company.
2. *Fibre Channel Arbitrated Loop Direct Disk Attach Profile (Private Loop)*, Version 2.0, ad hoc vendor group proposal.



When we started to write this article, the possibility existed that the HP Journal would be able to feature the Tachyon chip on the cover. We came up with a cover concept, and many people worked very hard on the cover design. Unfortunately, it turned out that Tachyon could not be featured on the cover. We would like to thank the many people who contributed to this graphic: Margie Evashenk for her concept drawing, LSI Logic Corporation who contributed the photomicrograph, Leland Wong who supplied photographs of Tachyon, and graphic artist Marianne deBlanc who put all the pieces together so beautifully.

▶ Go to Table of Contents
▶ Go to HP Journal Home Page