ACKNOWLEDGEMENT

I would like to acknowledge my gratitude to Prof. (Miss) S. N. Kakarwal for guiding me in preparation of this report. Prof. P. J. Ahire, Head of Department of Computer Science and Engineering, has constantly encouraged me to complete my report as per the schedule. My thanks are due to him.

I would also like to thank Prof. M. B. Nagori for providing access to IEEE related resources.

Himanshu Phirke was helpful at all stages of the work involved in preparing this report. I must record my thanks to him. My brother Anand enlightened me on the conventions of writing a report of technical nature. I thank him for bringing this report to its present form.

Last but not least I would like to thank Ankit and Ajit for providing me necessary software.

(Onkar Shinde)

SYNOPSIS

FireWire has been implemented since 1995. But very few of students, end users in India know about it. This report has been written to highlight most important aspects of FireWire.

The report covers brief history, design goals and the position of FireWire in comparison to other prevalent technologies.

Major attention is given on technical details of hardware, I/O specifications, protocols, data transfer modes, data protection features of FireWire.

It also tries to cover the latest status of FireWire including latest standards, improvements in transmission of data, improvements in architecture to support more advanced features.

Lastly the report discusses implementation details, present and upcoming applications, advantages and disadvantages of FireWire.

CONTENTS

INTRODUCTION.....4

What is FireWire?.....4
Brief history of FireWire.....4

WHY FIREWIRE?.....5

Design Goals.....5
Where FireWire stands?.....6
Zorro vs. Robin Hood (FireWire vs. USB).....7

WORKING AND TECHNICAL DETAILS.....8

Connectors and Cables.....8
I/O specifications.....8
Data transfer over FireWire.....10
Isochronous transport.....19
Data protection features.....20

LATEST STATUS.....25

The next generation bus – FireWire 800 (IEEE 1394b).....25
Data transfer speeds upto 800 Mbps.....25
Distance upto 100 m.....26
Transmission mediums.....27

IMPLEMENTATION DETAILS.....28

Hardware requirements.....28
Software requirements.....28

APPLICATIONS.....30

Mass storage.....30
Video.....30
Digital audio.....31
Digital still cameras.....32
Printers and scanners.....32
Home entertainment.....33
Networking.....33
Analog-to-digital YUV video converter.....33
Analog-to-DV converter.....34

ADVANTAGES.....35

CONCLUSION.....37

GLOSSARY.....38

BIBLIOGRAPHY.....39

INTRODUCTION

What is FireWire?

FireWire is a high-speed serial input/output (I/O) technology for connecting peripheral devices to computer or to each other. It's one of the fastest peripheral standards ever developed. Originally developed by Apple, FireWire is now an official industry standard IEEE 1394.

Brief history of FireWire

- **Introduced by Apple**

  First introduced by Apple around 1990, the standard was designed to support high bandwidth requirements of devices such as digital audio/video equipments and high performance mass storage.

- **Standardized by IEEE**

  The standard was conceived by Apple and then developed within IEEE 1394 working group. This standard defines both a backplane physical layer and a point-to-point cable connected virtual bus connections. The backplane version operates at 12.5, 25 or 50 Mbps. The cable version supports data rates of 100, 200, 400 Mbps.

  Three versions of standard 1394 have been approved so far.

  I. 1394-1995 was the original standard. Driven by Apple and based largely on Apple developed technology, 1394-1995 supported data transfer rates up to 400 Mbps and distances up to 4.5 meters, and introduced the concept of self-managing, peer-to-peer multimedia interconnect.

  II. 1394a, adopted in 2000, added and clarified specifications for performance optimization and power management on FireWire buses.

  III. 1394b, adopted in 2002, raises maximum FireWire speeds with architectural specifications up to 3.2 Gbps and distances to 100 meters. It also allows more types of

media (including optical fiber cabling) to be used for FireWire connections.

- **Marketed by Apple and Sony**

  The name FireWire, which was coined by Apple, is still used by a few vendors.

  Others have adopted the name i.Link, which is trademarked by Sony Corp., and has become a popular moniker for 1394-enabled products and technology in Japan.

WHY FIREWIRE?

Interfaces like Serial, Parallel, Parallel (PCI enhanced), IDE, SCSI, 10-Base T provide connectivity solutions for different devices. Also they talk with devices with their own protocol.

USB has been advancement over these interfaces mainly due to two points:

1) Standardization of bus

2) Connectivity solution for a wide range of devices.

But when it comes to using devices that deal with hundreds of megabytes of data USB falls short. Examples of such devices are digital still cameras, DV camcorders, DVD writers, magneto-optical drives, external hard disk drives.

In such situations FireWire stands above all.

Design Goals

Designers of FireWire had several goals in mind when they created the standard. They have been successfully achieved as per details below.

- **Fast transfer of Data**

  FireWire provides data transfer rate upto 400 Mbps.

- **Lots of devices on the bus**

  Upto 63 devices can be connected on a single bus using daisy chaining.

- **Ease of use**

  No extra accessories required to plug devices to your computer.

- **Hot pluggable**

  Devices can be connected & disconnected even when power is on.

- **Provide power through cable**

  FireWire can provide upto 45W (8 to 30 Volts, 1.5 amps max) to the device, which is enough for high performance disk drives & rapid battery charging.

- **Plug and Play**

  Operating system automatically detects new hardware and demands for diver disk.

- **Low cabling & implementation cost**

  Cost is low when implementing high data dealing devices such as setting an audio recording studio.

Where FireWire stands?

Lets take a look at comparison of FireWire speed with other interfaces.

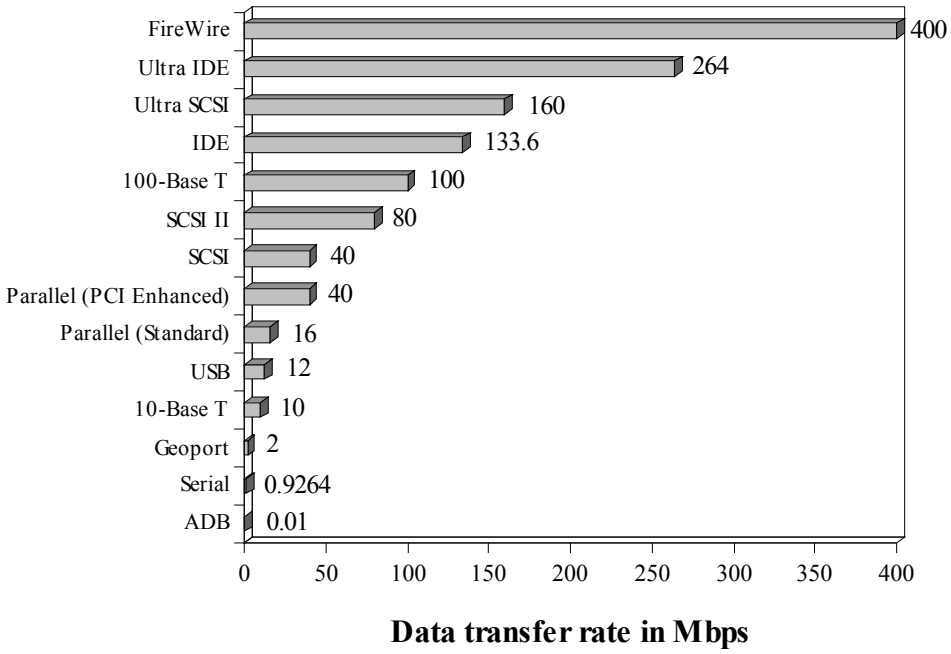


Fig. 1 - Data transfer rates over various interfaces

Zorro vs. Robin Hood (FireWire vs. USB)

Just as both Zorro and Robin Hood were great thieves both FireWire and USB are expert at stealing data.

Zorro was a horse-riding swordsman and Robin Hood was an expert foot-archer. This is analogous to higher speed and advanced technology of FireWire rather than USB.

Though Zorro and Robin Hood worked with identical principle they worked for different classes of people in different conditions. That is analogous to different device class of FireWire and USB under different implementation requirements.

Let us compare FireWire with USB in different aspects.

	USB	**FireWire**
Developers	USB was developed by a joint venture between Intel, Microsoft, IBM, Compaq, DEC, NEC and Northern Telecom.	Firewire was first developed by Apple, but modified by other companies over licensing conflicts. It is also known as IEEE 1394.
Transfer rates	USB supports two different transfer rates: 1.5 Mbps for low speed devices and 12 Mbps maximum rate.	The Firewire chipset allows three different rates: 100 Mbps, 200 Mbps and 400Mbps
Maximum of devices	127	63
On bus power	Up to 2.5W	Up to 45W
Data flow	USB has three types of data communication formats depending whether the device needs an occasional signal, a continuous stream of information (real time) or a mass transfer.	Firewire packets are transmitted to all devices like USB. However, each packet is given a certain amount of time to send data until it another device gets control. This makes Firewire a time-sliced protocol.
Number of wires	Four. Two of the wires are used to supply power to each device. The other two are used to transfer data.	Six. Two of the wires are used to supply power to each device. The other four are used to transfer data.
Network Topology	Hub	Daisy Chaining
Connection	Host based	Supports Peer-to-peer

Table 1. Comparison of USB and FireWire

WORKING AND TECHNICAL DETAILS

Connectors and Cables

FireWire connectors come into two configurations:

1) 4 pin – 4 wires, 2 twisted pairs data pins.

2) 6 pin – 6 wires, 2 twisted pairs data pins, 2 power pins.

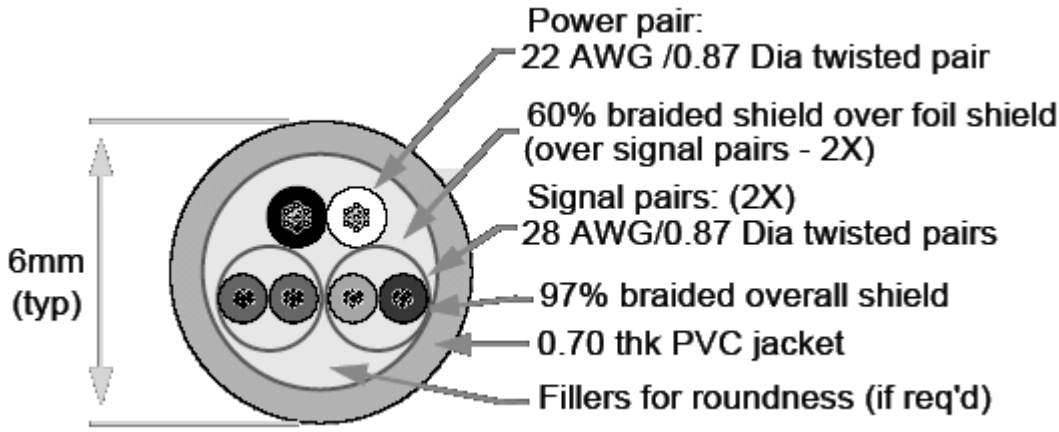


Fig. 2 - Cable cross-section

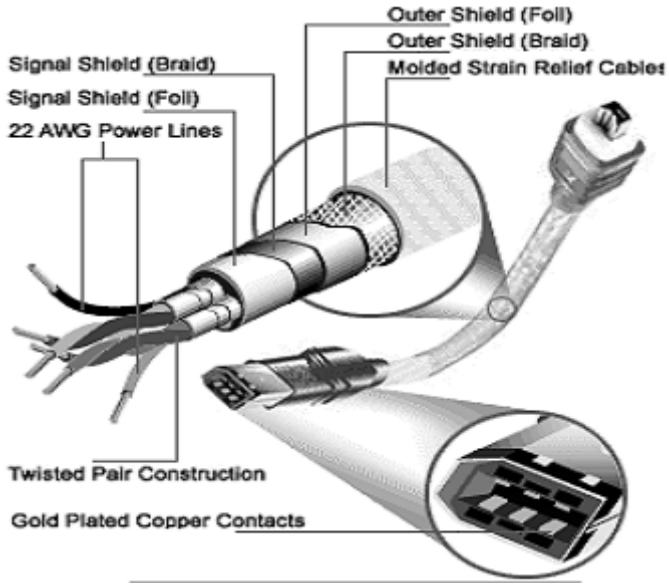


Fig. 3 - Description of cable

I/O specifications

- **Input**

  Voltage range: 8 to 33 volts (V)

  Input current range: approximately 1 watt (W)

- **Output**

Voltage range: approximately 15 to 20 V

Output current range: up to approximately 6 W, shared between the ports on the computer

- **Safety**

  The FireWire interface on the computer has three self-resetting fuses.

- **Repeater Function**

  Built-in repeater function allows FireWire data to pass between the computer and external devices even when the computer is turned off. Power from devices with power supplies runs to the repeater via the FireWire cable. The repeater does not work with devices that do not supply power.

  Repeater operates between 8 and 33 V and uses approximately 1 W.

Data transfer over FireWire

PROTOCOL STACK

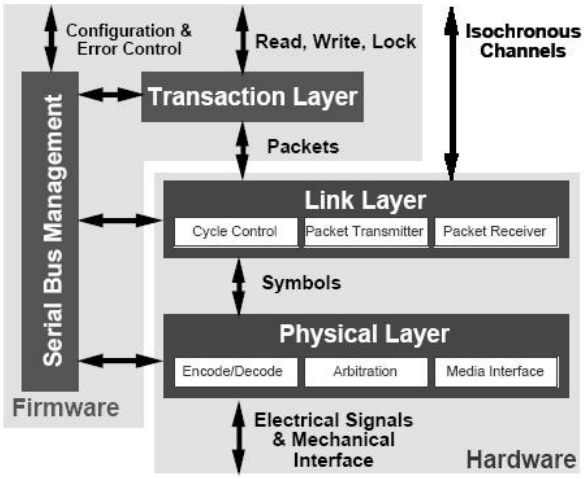


Fig. 4 - Protocol Stack

IEEE 1394 involves the low three IS0 protocol layers: the Physical Layer, the Link Layer, and the Transaction Layer, plus a Serial Bus Management process that connects to all three layers. The Physical Layer connects to the 1394 connector and the other layers connect to the application.

The Physical Layer provides the electrical and mechanical connection between the 1394 device and the 1394 cable. Besides the actual data transmission and reception tasks, the Physical Layer provides arbitration to insure all devices have fair access to the bus.

The Link Layer provides data packet delivery service for the two types of packet delivery: asynchronous and isochronous. As mentioned before, asynchronous is the conventional transmit-acknowledgment protocol and isochronous is a real-time guaranteed-bandwidth protocol for just-in-time delivery of information.

The Transaction Layer supports the asynchronous protocol write, read, and lock commands. A write sends data from the originator to the receiver and a read returns the data to the originator. Lock combines the function of the write and read commands by producing a round trip routing of data between sender and receiver including processing by the receiver.

Serial Bus Management provides overall configuration control of the serial bus in the form of optimizing arbitration timing, guarantee of adequate electrical power

for all devices on the bus, assignment of which 1394 device is the cycle master, assignment of isochronous channel ID, and basic notification of errors. Bus management is built upon IEEE 1212 standard register architecture.

CSR ADDRESSING

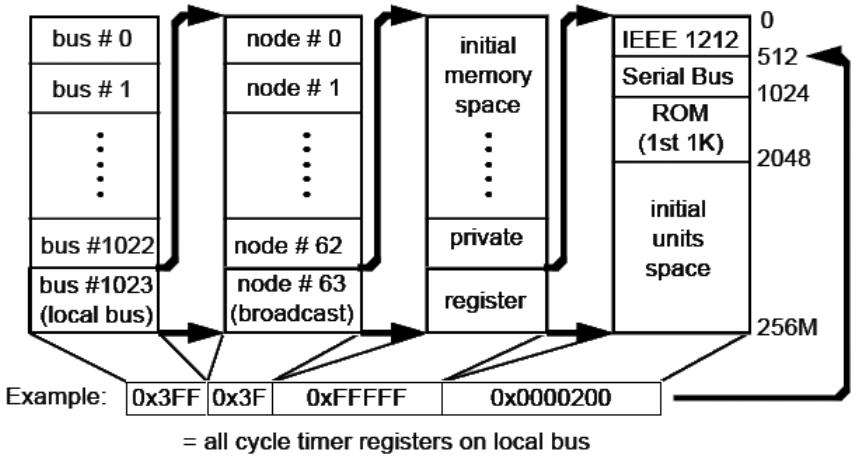


Fig. 5 - CSR Addressing details

FireWire is an interconnect (bus) that conforms to the CSR architecture, ISO/IEC 13213:1994. FireWire permits communications between nodes over shared physical media at speeds that range, at present, from 100 to 400 Mbps.

The CSR architecture describes a memory-mapped address space that Serial Bus implements as a 64-bit fixed addressing scheme. Within the address space,

1. Ten bits are allocated for bus ID (up to a maximum of 1,023 buses)
2. Six are allocated for node physical ID (up to 63 per bus)
3. Remaining 48 bits (offset) describe a per node address space of 256 terabytes.

The CSR architecture, by convention, splits a node's address space into two regions with different behavioral characteristics.

1. The lower portion, up to but not including 0xFFFF F000 0000, is EXPECTED to behave as memory in response to read and write transactions.
2. The upper portion is more like a traditional IO space: read and write transactions in this area usually have side effects.

Control and status registers (CSRs) that have FIFO behavior customarily are implemented in this region.

Within the 64-bit address, the 16-bit node ID (bus ID and physical ID) is analogous to a network hardware address---but 1394 node IDs are variable and subject to reassignment each time one or more nodes are added to or removed from the bus.

NOTE: Although the 16-bit node ID contains a bus ID, at present there is no standard method to connect separately enumerated Serial Buses. Active development of a standard for Serial Bus to Serial Bus bridges is underway in the IEEE P1394.1 working group. Unless extended by some future standard, the IPv4 over 1394 protocols specified by this document may not operate correctly across bridges.

The 1394 link layer provides a packet delivery service with both confirmed (acknowledged) and unconfirmed packets. Two levels of service are available: "asynchronous" packets are sent on a best-effort basis while "isochronous" packets are guaranteed to be delivered with bounded latency. Confirmed packets are always asynchronous but unconfirmed packets may be either asynchronous or isochronous. Data payloads vary with implementations and may range from one octet up to a maximum determined by the transmission speed (at 100 Mbps, named S100, the maximum asynchronous data payload is 512 octets while at S400 it is 2048 octets).

NOTE: Extensions in IEEE P1394b contemplate additional speeds of 800, 1600 and 3200 Mbps.

LINK ENCAPSULATION AND FRAGMENTATION

All IP datagrams (broadcast, unicast or multicast), 1394 ARP requests/responses and MCAP advertisements/solicitations that are transferred via 1394 block write requests or stream packets SHALL be encapsulated within the packet's data payload. The maximum size of data payload, in octets, is constrained by the speed at which the packet is transmitted.

Speed	Asynchronous	Isochronous
S100	512	1024
S200	1024	2048

S400	2048	4096
S800	4096	8192
S1600	8192	16384
S3200	16384	32768

Table 1 - Maximum data payloads (octets)

The maximum data payload for asynchronous requests and responses may also be restricted by the capabilities of the sending or receiving node(s); this is specified by max_rec in either the bus information block or 1394 ARP response.

For either of these reasons, the maximum data payload transmissible between IP-capable nodes may be less than the default 1500 octet maximum transmission unit (MTU) specified by this document. This requires that the encapsulation format also permit 1394 link-level fragmentation and reassembly of IP datagrams.

Global asynchronous stream packet (GASP) format

Some IP datagrams, as well as 1394 ARP requests and responses, may be transported via asynchronous stream packets. When asynchronous stream packets are used, their format SHALL conform to the global asynchronous stream packet (GASP) format specified by IEEE P1394a. The GASP format illustrated below is INFORMATIVE and reproduced for ease of reference, only.

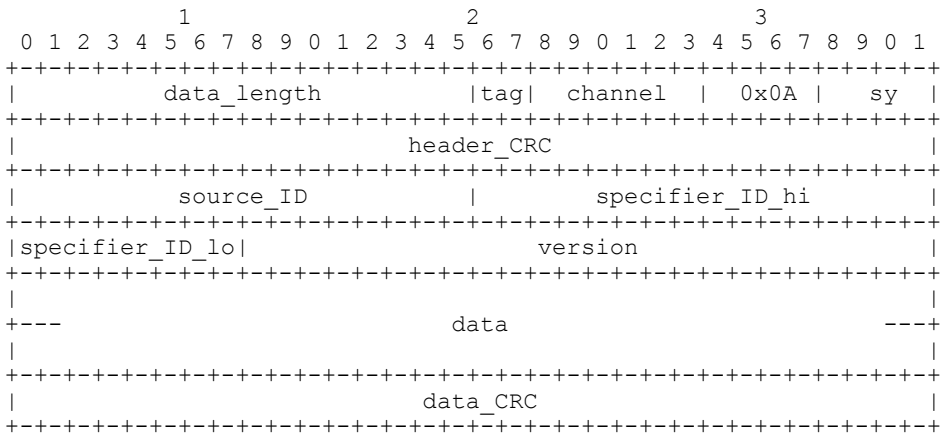


Fig. 6 - GASP format

The source_ID field SHALL specify the node ID of the sending node and SHALL be equal to the most significant 16 bits of the sender's NODE_IDS register.

The specifier_ID_hi and specifier_ID_lo fields together SHALL contain the value 0x00 005E, the 24-bit organizationally unique identifier (OUI) assigned by the IEEE Registration Authority (RA) to IANA.

The version field SHALL be one.

NOTE: Because the GASP format utilizes the first two quadlets of data payload in an asynchronous stream packet format, the maximum payloads cited in Table 1 are effectively reduced by eight octets. In the clauses that follow, references to the first quadlet of data payload mean the first quadlet usable for an IP datagram or 1394 ARP request or response. When the GASP format is used, this is the third quadlet of the data payload for the packet.

Encapsulation header

All IP datagrams transported over 1394 are prefixed by an encapsulation header with one of the formats illustrated below.

If an entire IP datagram may be transmitted within a single 1394 packet, it is unfragmented and the first quadlet of the data payload SHALL conform to the format illustrated below.

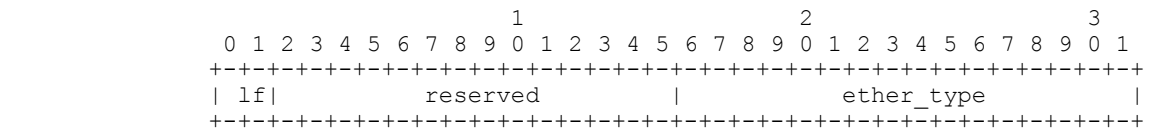


Fig. 7 - Unfragmented encapsulation header format

The lf field SHALL be zero.

The ether_type field SHALL indicate the nature of the datagram that follows, as specified by the following table.

ether_type	Datagram
0x0800	IPv4
0x0806	1394 ARP
0x8861	MCAP

NOTE: Other network protocols, identified by different values of ether_type, may use the encapsulation formats defined herein but such use is outside of the scope of this document.

In cases where the length of the datagram exceeds the maximum data payload supported by the sender and all recipients, the datagram SHALL be broken into link fragments; the first two quadlets of the data payload for the first link fragment SHALL conform to the format shown below.

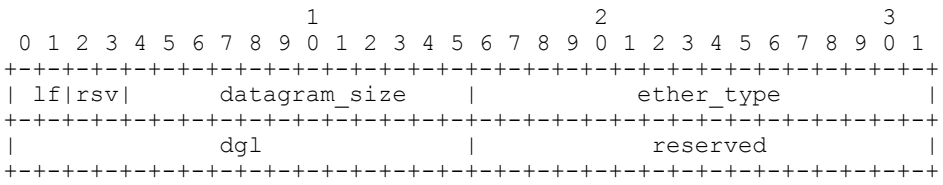


Fig. 8a - First fragment encapsulation header format

The second and subsequent link fragments (up to and including the last) SHALL conform to the format shown below.

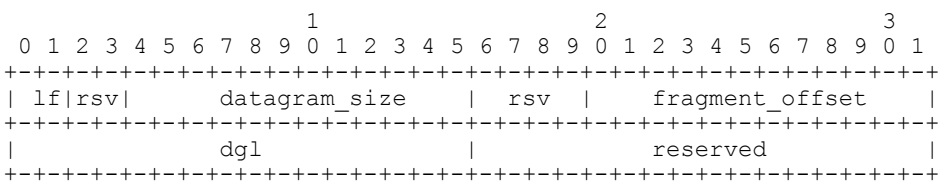


Fig. 8b - Subsequent fragment(s) encapsulation header format

The definition and usage of the fields is as follows:

The lf field SHALL specify the relative position of the link fragment within the IP datagram, as encoded by the following table.

lf	Position
0	Unfragmented
1	First
2	Last
3	Interior

datagram_size: The encoded size of the entire IP datagram. The value of datagram_size SHALL be the same for all link fragments of an IP datagram and

SHALL be one less than the value of Total Length in the datagram's IP header (see STD 5, RFC 791).

ether_type: This field is present only in the first link fragment and SHALL have a value of 0x0800, which indicates an IPv4 datagram.

fragment_offset: This field is present only in the second and subsequent link fragments and SHALL specify the offset, in octets, of the fragment from the beginning of the IP datagram. The first octet of the datagram (the start of the IP header) has an offset of zero; the implicit value of fragment_offset in the first link fragment is zero.

dgl: The value of dgl (datagram label) SHALL be the same for all link fragments of an IP datagram. The sender SHALL increment dgl for successive, fragmented datagrams; the incremented value of dgl SHALL wrap from 65,535 back to zero.

All IP datagrams, regardless of the mode of transmission (block write requests or stream packets) SHALL be preceded by one of the above described encapsulation headers. This permits uniform software treatment of datagrams without regard to the mode of their transmission.

Link fragment reassembly

The recipient of an IP datagram transmitted via more than one 1394 packet SHALL use both the sender's source_ID (obtained from either the synchronous packet header or the GASP header) and dgl to identify all the link fragments from a single datagram.

Upon receipt of a link fragment, the recipient may place the data payload (absent the encapsulation header) within an IP datagram reassembly buffer at the location specified by fragment_offset. The size of the reassembly buffer may be determined from datagram_size.

If a link fragment is received that overlaps another fragment identified by the same source_ID and dgl, the fragment(s) already accumulated in the reassembly buffer

SHALL be discarded. A fresh reassembly may be commenced with the most recently received link fragment. Fragment overlap is determined by the combination of fragment_offset from the encapsulation header and data_length from the 1394 packet header.

Upon detection of a Serial Bus reset, recipient(s) SHALL discard all link fragments of all partially reassembled IP datagrams and sender(s) SHALL discard all not yet transmitted link fragments of all partially transmitted IP datagrams.

SERIAL BUS ADDRESS RESOLUTION PROTOCOL (1394 ARP)

Methods to determine the hardware address of a device from its corresponding IP address are inextricably tied to the transport medium utilized by the device. In the description below and throughout this document, the acronym 1394 ARP pertains solely to an address resolution protocol whose methods and data structures are specific to 1394.

1394 ARP requests SHALL be transmitted by the same means as broadcast IP datagrams; 1394 ARP responses MAY be transmitted in the same way or they MAY be transmitted as block write requests addressed to the sender_unicast_FIFO address identified by the 1394 ARP request. A 1394 ARP request/response is 32 octets and SHALL conform to the format illustrated by following figure.

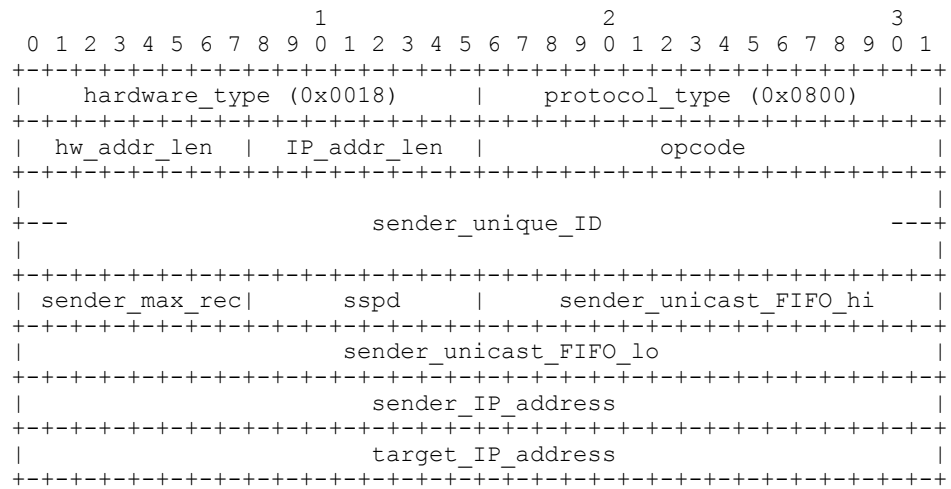


Fig. 9 - 1394 ARP request/response format

1394 ARP requests and responses transported by asynchronous stream packets SHALL be encapsulated within the GASP format specified by IEEE P1394a (see also 4.1). The recipient of a 1394 ARP request or response SHALL ignore it unless the most significant ten bits of the source_ID field (whether obtained from the GASP header of an asynchronous stream packet or the packet header of a block write request) are equal to either 0x3FF or the most significant ten bits of the recipient's NODE_IDS register.

Field usage in a 1394 ARP request/response is as follows:
hardware_type: This field indicates 1394 and SHALL have a value of 0x0018.

protocol_type: This field SHALL have a value of 0x0800; this indicates that the protocol addresses in the 1394 ARP request/response conform to the format for IP addresses.

hw_addr_len: This field indicates the size, in octets, of the 1394-dependent hardware address associated with an IP address and SHALL have a value of 16.

IP_addr_len: This field indicates the size, in octets, of an IP version 4 (IPv4) address and SHALL have a value of 4.

opcode: This field SHALL be one to indicate a 1394 ARP request and two to indicate a 1394 ARP response.

sender_unique_ID: This field SHALL contain the node unique ID of the sender and SHALL be equal to that specified in the sender's bus information block.

sender_max_rec: This field SHALL be equal to the value of max_rec in the sender's configuration ROM bus information block.

sspd: This field SHALL be set to the lesser of the sender's link speed and PHY speed. The link speed is the maximum speed at which the link may send or receive packets; the PHY speed is the maximum speed at which the PHY may send, receive or repeat

packets. The table below specifies the encoding used for sspd; all values not specified are RESERVED for future standardization.

Value	Speed
0	S100
1	S200
2	S400
3	S800
4	S1600
5	S3200

Table 2 - Speed codes

sender_unicast_FIFO_hi and sender_unicast_FIFO_lo: These fields together SHALL specify the 48-bit offset of the sender's FIFO available for the receipt of IP datagrams in the format specified by section 6. The offset of a sender's unicast FIFO SHALL NOT change, except as the result of a power reset.

sender_IP_address: This field SHALL specify the IP address of the sender.

target_IP_address: In a 1394 ARP request, this field SHALL specify the IP address from which the sender desires a response. In a 1394 ARP response, it SHALL be IGNORED.

Isochronous transport

Isochronous, just in time delivery, allows low-cost implementations of time-critical multimedia integaces.

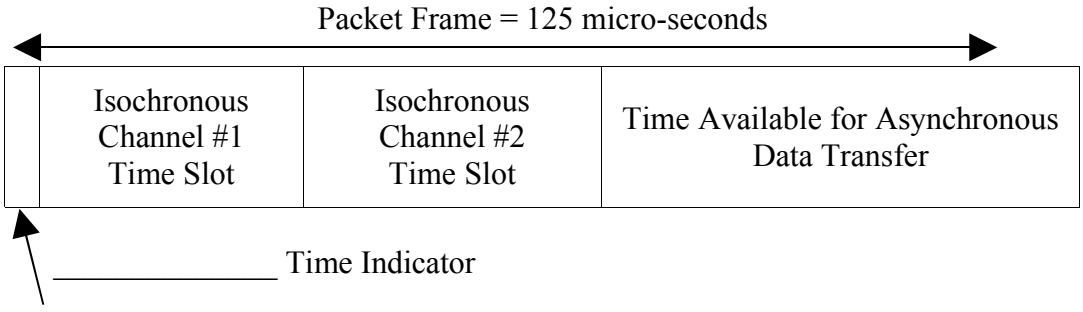
Both asynchronous and isochronous data transfers are supported by Firewire. The asynchronous format transfers data and transaction layer information to an explicit address. The isochronous format broadcasts data based on channel numbers rather than specific addressing. Isochronous packets are issued on the average of each 125 μs in support of time-sensitive applications. Providing both asynchronous and isochronous formats on the same interface allows both non-real-time critical applications such as printers, and scanners and real-time critical applications such as video and audio to operate on the same bus transfers. Isochronous transfers provide guaranteed transmission opportunities at defined intervals; if a packet is not received successfully, it is not resent. In asynchronous transfers, the intervals between

transmissions can vary, and data can be resent if it is missed. Isochronous, just in time delivery, allows low-cost implementations of time-critical multimedia devices.

For example, sending a live TV broadcast requires isochronous transmission to ensure that each frame arrives on time and in the correct order. By contrast, storing data on a hard disk drive can be done asynchronously. It's okay for dropped data to be resent, and it doesn't matter in what order the data arrives, because each packet is tagged with an address or sequence number to reliably identify it.

FireWire is one of very few interfaces that combine both isochronous and asynchronous capabilities. FireWire can reserve up to 80 percent of its bandwidth for one or more isochronous channels, making it an excellent interface for applications that require real-time data transmission.

With isochronous transport, the sender requests an isochronous channel with a specific bandwidth. Isochronous channel IDS are transmitted followed by the packet data. The receiver monitors the incoming data's channel ID and accepts only data with the specified ID. User applications are responsible for determining how many isochronous channels are needed and their required bandwidth. Although up to 64 isochronous channels may be defined, the diagram below shows 2 channels.



Data protection features

Digital content over 1394 can be robustly protected using Digital Transmission Copy Protection (DTCP). DTCP support device authentication, content encryption, and renewability, should a DTCP device ever be compromised. Encoding rules can be specified for content, e.g., "copy freely," "copy once," or "copy never." The motion

picture industry recognizes DTCP as satisfactory to them in permitting the transmission of their content over 1394.

The Digital Transmission Content Protection Specification defines a cryptographic protocol for protecting audio/video entertainment content from unauthorized copying, intercepting, and tampering as it traverses digital transmission mechanisms such as a high-performance serial bus that conforms to the IEEE 1394-1995 standard. Only legitimate entertainment content delivered to a source device via another approved copy protection system (such as the DVD Content Scrambling System) will be protected by this copy protection system. The use of this specification and access to the intellectual property and cryptographic materials required to implement it will be the subject of a license. The Digital Transmission Licensing Administrator (DTLA) is responsible for establishing and administering the content protection system described in this specification. While DTCP has been designed for use by devices attached to serial buses as defined by the IEEE 1394-1995 standard, the developers anticipate that it will be appropriate for use with future extensions to this standard, other transmission systems, and other types of content as authorized by the DTLA.

Following are the four layers of copy protection:

- **Copy control information (CCI)**

  Content owners need a way to specify how their content can be used ("copy-one-generation," "copy-never," etc.). This content protection system is capable of securely communicating copy control information (CCI) between devices in two ways:

  1. The Encryption Mode Indicator (EMI) provides easily accessible yet secure transmission of CCI via the most significant two bits of the ***sy*** field of the isochronous packet header.
  2. CCI is embedded in the content stream (e.g. MPEG). This form of CCI is processed only by devices which recognize the specific content format.

- **Device authentication and key exchange (AKE)**

Before sharing valuable information, a connected device must first verify that another connected device is authentic. To balance the protection requirements of the content industries with the real-world requirements of PC and consumer electronics (CE) device users, this specification includes two authentication levels, Full and Restricted.

1. Full Authentication can be used with all content protected by the system.
2. Restricted Authentication enables the protection of "copy-one-generation" and "no-more-copies" content only. Copying devices such as digital VCRs employ this kind of authentication.

- **Content encryption**

Devices include a channel cipher subsystem that encrypts and decrypts copyrighted content. To ensure interoperability, all devices must support the specific cipher specified as the baseline cipher. The subsystem can also support additional ciphers, whose use is negotiated during authentication.

- **System renewability**

Devices that support Full Authentication can receive and process system renewability messages (SRMs) created by the DTLA and distributed with content and new devices. System renewability ensures long-term integrity of the system through the revocation of compromised devices.

Given below is the overview of content protection. In this overview, the source device has been instructed to transmit a copy protection stream of content. In this and subsequent diagrams, a source device is one that can send a stream of content. A sink device is one that can receive a stream of content. Multifunction devices such as PCs and

record/playback devices such as digital VCRs can be both source and sink devices.

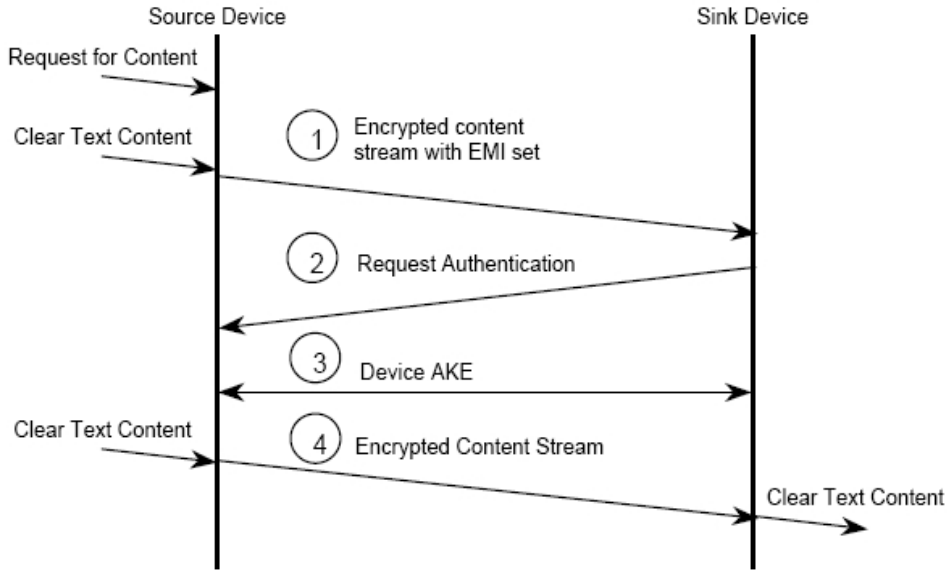


Fig. 10 – Data transfer using DTCP

1. The source device initiates the transmission of a stream of encrypted content marked with the appropriate copy protection status (e.g., “copy-one-generation,” “copy-never,” or “no-more-copies”) via the EMI bits.
2. Upon receiving the content stream, the sink device inspects the EMI bits to determine the copy protection status of the content. If the content is marked “copy-never,” the sink device requests that the source device initiate Full AKE. If the content is marked “copy-one-generation” or “no-more-copies” the sink device will request Full AKE, if supported, or Restricted AKE. If the sink device has already performed the appropriate authentication, it can immediately proceed to Step 4.
3. When the source device receives the authentication request, it proceeds with the type of authentication requested by the sink device, unless Full AKE is requested but the source device can only support Restricted AKE, in which case Restricted AKE is performed.
4. Once the devices have completed the required AKE procedure, a content channel encryption key can be

exchanged between them. This key is used to encrypt the content at the source device and decrypt the content at the sink.

LATEST STATUS

The next generation bus – FireWire 800 (IEEE 1394b)

Now FireWire 800, the next generation of FireWire technology, promises to spur the development of more innovative high-performance devices and applications. FireWire800 (an implementation of the IEEE 1394b standard approved in 2002) doubles the throughput of the original technology, dramatically increases the maximum distance of FireWire connections, and supports many new types of cabling.

Now transferring data at up to 800 Mbps, FireWire 800 delivers more than double the effective bandwidth of the USB 2.0 peripheral standard. That means you can send more than a CD's worth of data every ten seconds.

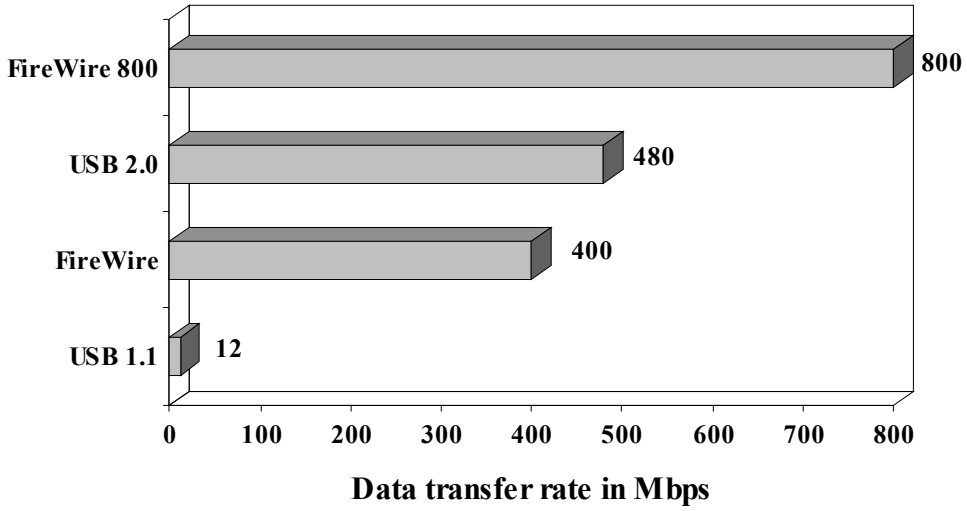


Fig. 11 – FireWire 800 more faster than USB 2.0

Data transfer speeds upto 800 Mbps

FireWire 800 is capable of transferring data at 800 Mbps. Twice the speed of the original FireWire. This performance increase has been achieved primarily by using the same highly efficient encoding scheme used by Gigabit Ethernet and Fiber Channel. In fact, the FireWire roadmap outlined in the IEEE 1394b standard will eventually take the theoretical bit rate to 1600 Mbps and then up to a staggering 3200 Mbps. That's 3.2 gigabits per second, which will make FireWire indispensable for

transferring massive data files and for even the most demanding video applications, such as working with uncompressed high-definition (HD) video or multiple standard-definition (SD) video streams.

Distance upto 100 m

Not only is FireWire 800 twice as fast as before, but it can be used over much longer distances. The 1394b specification allows the use of various types of cabling, each offering different speed/distance capabilities, as shown below:

Cable type	100 Mbps	200 Mbps	400 Mbps	800 Mbps	1600 Mbps	3200 Mbps
9-pin shielded twisted pair copper	4.5 m	4.5 m	4.5 m	4.5 m	4.5 m	4.5 m
CAT-5 unshielded twisted pair copper (standard Ethernet cable)	100 m	-	-	-	-	-
Step-index plastic optical fiber	50 m	50 m	-	-	-	-
Hard polymer-clad plastic optical fiber	100 m	100 m	-	-	-	-
Glass optical fiber	100 m	100 m	100 m	100 m	100 m	100 m

Table 3 - Data transfer rates on FireWire 800 with different cables

FireWire 800 (1394b) hubs even make it possible to connect FireWire 400 (1394a) devices up to 100 meters apart. Neither the computer nor the remote devices need to support FireWire 800, because the new hubs and their associated cables work with FireWire 400 equipment. New FireWire 800 (and other 1394b) devices can, of course, communicate over long connections directly through their own FireWire 800 ports; no hub is required to gain this distance benefit in a pure FireWire 800 connections.

Transmission mediums

Following are the various transmission mediums that can be used with FireWire 800.

- **UTP-5 (ISO/IEC 11801 ch7)**

  Cat-5 Unshielded Twisted Pair and RJ-45 connector
  Well-known for ethernet

  TX on pins 1/2, RX on pins 7/8

  Transformer isolation

  Similar electrical specification as other standards (1V pk-pk binary)

  Adaptive equalization in the receiver

  Allows 100m operation

- **Plastic Optic Fiber / Hard Polymer Clad Fiber**

  POF: 1000 μm step index multimode plastic optic fiber
  Suitable for up to 50m

  HPCF: 225 μm graded index multimode hard polymer clad fiber
  Suitable for up to 100m

  Same transceiver spec for both

  Low cost, easy to install

  Fiber alleviates emissions and interference problems

  PN connector

- **Glass fiber (MMF)**

  Leverage VCSEL (Vertical Cavity Surface Emitting Laser) technology

  Leverage Fibre Channel and Gigabit Ethernet specifications

  50 micron multimode fiber (MMF)

  LC connector

  Leverage Fibre Channel/ Gigabit Ethernet specifications

IMPLEMENTATION DETAILS

FireWire devices can be connected to a computer in two ways.

- **On-board FireWire Bus/Port**

  FireWire devices can be connected to a computer through an onboard port provided by motherboard manufacturer. Current very few vendors provide this facility.

  Ex.

  Apple Mac based on G4 processors provides FireWire ports.

  Asus, which manufactures motherboards for Intel and AMD processor based systems provides 2 FireWire ports on their A7N8X model.

- **FireWire PCI Adaptor**

  FireWire products can also be connected to computer using FireWire PCI adaptors that fit into standard PCI expansion slots.

Hardware requirements

- **For PC**

  233 MHz processor or better, 400 MHz plus recommended for digital video

  PCI bus 2.1 (For FireWire PCI Adaptor or PCI Card Bus only)

- **For Macintosh**

  G3 or better Macintosh

  PCI bus 2.1 (For FireWire PCI Adaptor or PCI Card Bus only)

Software requirements

Following are the operating system requirements for FireWire devices. No special software other than device drivers is required to use FireWire devices.

- **For PC**

  Windows: -

  Windows 98SE, 2000, Millenium (ME), XP

FireWire Software RAID requires software supported only on Windows 2000, but will function as multi-drive storage clusters on all other supported Windows systems.

A special patch from Microsoft is available for Windows 98SE and is recommended for users of FireWire storage devices.

Linux: -

Linux kernel version 2.4 or above

FireWire RAID will function as multi-drive storage clusters Linux systems.

- **For Macintosh**

Mac OS 9.1 and above

FireWire Software RAID requires special software supported only on Mac OS 8.6 to Mac OS 9.2, but will function as multi-drive storage clusters on Mac OS X systems.

Mac users using a system prior to OS 9.1 may require FireWire drivers from Apple in order to use any FireWire products.

APPLICATIONS

Following is the list of areas where FireWire has been implemented or in process of implementation. Some FireWire products are available while some are upcoming product.

Mass storage

Storage devices, especially portable ones are being radically transformed by the adoption of FireWire. Not only does FireWire permit an external hard disk drive to be mounted by simply connecting a single plug, it can even provide enough electricity to power the drive. FireWire mass storage devices include hard disk drives, magneto-optical drives, high-capacity removable drives, tape drives, and CD/DVD products, including both read-only and read/write drives.
Ex.

- Iomega external hard drive with 30 GB capacities.
- AITE130UL – AIT-2 USB/i.Link® Combo External Drive. A StorStation tape drive by Sony. External and implements i.Link® (Another variant if FireWire by Sony) interface. Capacity 130GB

Video

Digital video (DV) camcorders capture video and audio and can send a perfect copy to a computer for editing, adding special effects, and making other modifications to create a finished video. FireWire provides the high-speed connection required to download digital video quickly. FireWire 800 even has the necessary throughput for bandwidth-intensive applications that were not possible over the original FireWire, such as multiple-stream, uncompressed, standard-definition video. The long-distance capability of FireWire 800 also gives production studios and similar businesses more flexibility to locate each piece of equipment where it's most appropriate, rather than having to put everything adjacent to the computer.

Digital camcorders, whose internal electronics are all digital, store the incoming audio and video on tape in a digital format called DV rather than in an analog format such as High 8. DV produces full-size, full-motion video of 720 by 480 pixels at 30 frames per second. With FireWire, bringing DV into a computer is nothing more than a simple file transfer between the camera and the computer at 35 megabytes per second.

Digital camcorders vary in price from $850 to $5,000 and up. Most cameras use a single charge-coupled device (CCD, the electronic device that captures the image) to digitize the incoming light; higher-end cameras employ three CCDs, one each for red, green, and blue. Higher-end cameras generally work better with low light conditions, and often have interchangeable lenses.
Ex.

- DCRTRV19KITB - DCR-TRV19 MiniDV Handycam® Camcorder by Sony. Implements i.Link®* DV Interface (IEEE 1394)

Digital audio

FireWire delivers the bandwidth required for high-quality digital audio. Even FireWire 400 has enough bandwidth over a single connector for hundreds of channels of noise-free, high-resolution digital audio and up to 256 channels of MIDI. FireWire 800 can handle twice as many simultaneous real-time streams. Support for cabling up to 100 meters gives you more configuration options with FireWire 800 than solutions such as USB, enabling you to use a Macintosh system as a virtual patch bay that connects audio devices in situations ranging from a personal studio to a huge multi-room production facility. You can even hot-swap devices in and out of the audio processing chain as your needs change.

Digital recording, processing, and storage bring many advantages to today's musicians, producers, engineers and, of course, listeners. These advantages include clean transmission and audio integrity even with multiple generations of copying. FireWire offers a high-speed, flexible bridge between professional digital audio components. In addition, because FireWire has been widely embraced by consumer

electronics manufacturers, you can easily integrate video cameras, receivers, and other consumer gear with professional audio equipment.
Ex.

- IPod – A portable digital audio player with capacities 10GB, 20GB, 40GB. Implements FireWire interface, which is used for data transfer as well as battery charging.

Digital still cameras

Digital cameras are one of the fastest-growing peripheral segments in computing. They allow you to capture high-quality still images and transfer them digitally to your Macintosh system, eliminating the need for traditional film developing and scanning.

FireWire provides a means for transferring images from the camera to the computer that is much faster and more convenient than serial, parallel, or even USB connections, especially for high-resolution images that can be hundreds of megabytes in size. The high bandwidth of FireWire 800 will be increasingly important as consumer product manufacturers offer cameras with higher and higher mega pixel ratings.

FireWire connections appeared first on higher-end cameras, which generally have larger file sizes (for higher-quality images) and sell at higher prices. These cameras are pushing the state of the art in CCD technology, with the ability to capture up to 6 million pixels with a single shutter click. When printed out on high-quality color photo printers, the images from these cameras are often indistinguishable from traditional photos at first glance.

Printers and scanners

The benefits of FireWire printers and scanners include faster direct connections for high-quality imaging applications and the ability to free up Ethernet bandwidth by sharing printers within small FireWire-equipped workgroups.

Printers and scanners also benefit from the built-in power capabilities of FireWire. Portable printers don't need a separate battery, and consumer and film scanners don't need wall-mounted power supplies.

First FireWire printers are expected to be higher-end professional products with higher prices than their USB counterparts. USB printers are aimed at the cost-conscious home user, while FireWire printers will be professional workplace products at first.

Home entertainment

Set-top boxes, personal video recorders (PVRs), game consoles, home stereo equipment, DVD players, digital TVs, interactive TVs, and computers all have a need to communicate using a common plug-and-play, high-speed interconnection that's capable of efficiently transmitting video, audio, graphics, and Internet data. More and more of these devices are already available with FireWire ports, so they can easily be integrated into the modern home entertainment system.
Ex.

- Sony Playstation 2 implements FireWire interface.

Networking

Workgroup computers that have FireWire or other 1394 ports can be linked via FireWire and communicate using standard IP networking protocols. FireWire can also be used to cost-effectively share a printer, scanner, camera, or other device.

Networking with FireWire offers several advantages over Ethernet. FireWire 400 is faster than 100BASE-T, the most common Ethernet speed, and FireWire 800 comes close to the speed of 1000BASE-T. FireWire 800, with its ability to guarantee the timing interval of data packets, can also deliver smooth real-time video.

Analog-to-digital YUV video converter

This converter lets you use existing analog camcorders, TVs, and VCRs with a FireWire-based Macintosh system. Instead of taking up a PCI slot, it handles analog-to-digital conversion through a small external box connected to a FireWire port on the computer. The resulting video files are QuickTime movies in the raw video format—they are not DV, MPEG, M-JPEG, or Sorenson format until you choose to convert them to a specific format using QuickTime.

Analog-to-DV converter

This product converts incoming analog audio and video to a DV-compressed digital stream before sending it to your Macintosh system over FireWire. The conversion happens in real time and essentially makes all analog audio/video products appear to the computer as DV devices with high-quality video streams. It also allows for the export of DV video to analog video.

ADVANTAGES

- **A digital interface**

  No need to convert digital data into analog for better signal integrity.

- **A physically small thin serial cable**

  Replaces today's bulky and expensive interfaces.

- **Easy to use**

  No need for terminators, device IDs, screws, or complicated set-ups.

- **Hot pluggable**

  Devices can be added and removed while the bus is active.

- **Scalable**

  The Standard defines 100, 200, and 400 Mbps devices and can support the multiple speeds on a single bus.

- **Fast, guaranteed bandwidth**

  The Standard supports guaranteed delivery of time critical data, which enables smaller buffers (lower cost).

- **Flexible**

  The Standard supports free form daisy chaining and branching for peer-to-peer implementations. FireWire is a peer-to-peer interface. This allows dubbing from one camcorder to another without a computer. It also allows multiple computers to share a given peripheral without any special support in the peripheral or the computers. It is a result of all of these features that FireWire has become the digital interface of choice and its acceptance is growing.

- **On Bus Power**

  While USB 2.0 allows at most 2.5W of power — enough for a simple, slow device like a mouse — FireWire devices can provide or consume up to 45W of power, plenty for high-performance disk drives and rapid battery charging. That’s why iPod only needs one cord for both data and power.

- **Real-Time Data Delivery**

Unlike many other data transfer technologies, FireWire can guarantee real-time delivery of data. This is critical for streaming media applications such as audio and video, where delayed or out-of-order frames are unacceptable. The data traffic between FireWire nodes is divided into isochronous and asynchronous transfers. Isochronous transfers provide guaranteed transmission opportunities at defined intervals; if a packet is not received successfully, it is not resent. In asynchronous transfers, the intervals between transmissions can vary, and data can be resent if it's missed. For example, sending a live TV broadcast requires isochronous transmission to ensure that each frame arrives on time and in the correct order. By contrast, storing data on a hard disk drive can be done asynchronously. Its okay for dropped data to be resent, and it doesn't matter in what order the data arrives, because each packet is tagged with an address or sequence number to reliably identify it. FireWire is one of very few interfaces that combine both isochronous and asynchronous capabilities. FireWire can reserve up to 80 percent of its bandwidth for one or more isochronous channels, making it an excellent interface for applications that require real-time data transmission.

- **Boot Your OS from 1394**

Windows 2000 and its next revision, Windows Whistler, allow for booting from 1394 hard drives on computers with BIOS support for 1394 booting. Hardware manufacturers are working toward delivering the ability to boot from these drives now.

CONCLUSION

It's clear that Apple's Firewire has satisfied the definition of a good bus design —it has expanded far beyond its original intent. The 1394-1995, 1394a-2000 serial bus has the bandwidth capacity to displace most other peripheral connection communication methods in use today, including parallel, RS232, SCSI, USB 1.0 and USB 2.0, and consolidate them into a unified high-performance serial bus. As consumer multimedia and computing applications continue to merge, the peer-to-peer flexibility and the serial bus advantages of the IEEE 1394 standard will become increasingly popular. As demonstrated by the increasing bandwidth of 1394b, there's no immediate end in sight to the amount of performance that can be extracted from this protocol. With backward compatibility in mind, contributors to the evolving standard are opening up a new way of computing that will provide increasing flexibility and portability of applications.

Design of copy protection systems and especially with their deployment schedule. Every effort is made to assist the studios in choosing a system design that is appropriate to the task and then assist in the implementation as a part of the IEEE 1394 interface. For the first time, cable, satellite, broadcast, and the CE manufacturers are all on the same side of an issue, all equally concerned with copy protection cost, complexity, customer service and satisfaction issues, and the impact on the deployment schedule for digital services.

Finally, the hot plugging, power sourcing and dynamic reconfiguration abilities make 1394 a user-friendly environment. The features of 1394 will allow plugging in a computer expansion system as easily as plugging into AC power, providing communications on demand without having to shut down and reconfigure each time an I/O device is added or removed.

GLOSSARY

ADB: Apple Desktop Bus. The first bus created by Apple.

GeoPort: This is a port created by Apple for use with Macintosh devices

10-Base T: A networking port standard transmitting data at 10 Mbps upto 100 m and common for LANs

Parallel (Standard): Standard 16-bit port for used in primitive PCs and Commodore 64 computers. Handshaking (coordination between devices) was done by the driver.

Parallel (PCI enhanced): This is the current standard in current PC computers. Handshaking is done through the port itself.

SCSI II: An improved 16-bit version of SCSI (Small Computer Serial Interface) with addition wires to transmit data in parallel.

IDE: Integrated Drive Electronics

USB: Universal Serial Bus.

Datagram: An Internet message that conforms to the format specified by STD 5, RFC 791

Octet: 8 bits of data.

Quadlet: Four octets, or 32 bits, of data.

DTCP: Digital Transmission Content Protection.

BIBLIOGRAPHY

- Everything related to basic information of IEEE standards, drafts of standards (accessible to members only) and other related information can be found on IEEE website.

  http://www.ieee.org

- IEEE 1394 trade association established to help vendors include FireWire in their products.

  http://www.1394ta.org

- Apple's page about FireWire. You can also find links to tech-brief and factsheet.

  http://www.apple.com/firewire

- A range of FireWire products including harddisks, CD drives, Combo drives (CD-R/CD-RW/DVD).

  http://www.firewiredirect.com

- System Requirements

  http://www.firewiredirect.com/site/support/supportpages/system.shtml

- Consumer electronics products of FireWire

  http://www.1394ta.org/About/products/consumer_products.html

- RFC of IPv4 over FireWire

  http://www.faqs.org/rfcs/rfc2734.html

- Asus's website to find their motherboards implementing FireWire.

  http://www.asus.com

- Consumer electronics products by Sony implementing i.Link (another implementation of IEEE 1394).

  http://www.sony.com

- A guide for students on writing technical reports.

  http://www.kevinboone.com/howto_report.html