FireWire® for Audio Applications

by
Ross Alexander Hendler

Submitted in partial fulfillment of the requirements for the

Masters of Music in Music Technology

in the Department of Music and Performing Arts Professions
in the Graduate School of Education
New York University

Advisors: Kenneth J. Peacock, Robert J. Rowe

Spring 2003

FireWire® for Audio Applications

Ross Alexander Hendler

Submitted in partial fulfillment of the requirements for the
Masters of Music in Music Technology
in the Department of Music and Performing Arts Professions

Abstract

The purpose of this thesis is to examine the peripheral standard FireWire as an audio interface and to project where this technology is headed. Many technologies have been replaced very quickly due to the rapid advances in computer technology. However, this does not seem to be the case with the FireWire technology. This thesis will examine why FireWire became so popular and why it works as an audio interface.

Current high-end audio devices make use of low-latency, multi-channel inputs and outputs, sampling rates and bit rates. These are all important factors which dictate where technologies will progress based on their priority, cost and technological developments. FireWire appears to be an ideal technology for the audio world for numerous reasons, reasons which will be explored in this thesis.

Contents:

List of Figures

Introduction

The preservation of sound has undergone vast changes over the last century. The invention of the tinfoil-coated cylinder by Edison and then the wax cylinder by Bell and Tainter, were the earliest steps in the process of modern recording. Flat disc technology, which allowed the mass production of recorded music, quickly followed. The flat disc is an example of an analog device. It uses a continuously variable physical phenomenon (sound) to precisely reproduce another dynamic phenomenon (the sound in the real world which it records). The sounds are encoded in a groove on the phonograph record that varies continuously in width and shape. When a stylus passes along the groove, the analog information -- the signal that is capable of precisely representing the sound as it occurs in the real world - is picked up and then electronically amplified in order to reproduce the original sounds. Significant, is that any number of minor imperfections will be translated by the playback device into additional sounds, or noise.

In the early 1980s, a profound departure occurred from the analog age with the introduction of the compact disc by Philips and Sony. This ushered in the digital age. The digital technology of the compact disc allows for sounds to be translated into binary code and recorded on the disc as discrete pits. Digital devices can better deal with the problem of unwanted information, or noise. Noise is less of a problem because most noise will not be encoded initially, and noise that does get encoded is easily recognized and eliminated during the retranslation process. The one drawback of a digital process is that it cannot be expected to reproduce every single aspect of a continuous phenomenon. An analog device will produce a more complete, or truer, rendering of a continuous phenomenon. However, digital technology has democratized the process of recording and composing and allows for a larger group of people to be involved with the recording process.

Organization

Chapter 1, **Background**, focuses on the practice of digital communication. This chapter discusses how various interfaces communicate and finishes with a comparison of the current most popular digital interfaces, USB and FireWire.

Chapter 2, **FireWire Technology**, gives a presentation and discussion of FireWire technology, what it is and why it works as an audio interface.

Chapter 3, **Handling Audio With FireWire**, gives a side-by-side look at the digital audio demands for effective transfer and how FireWire provides for these demands.

Chapter 4, **Applications**, shows how, when and where FireWire audio devices can be used.

Chapter 5, **Conclusions and Future Developments**, ties up by summarizing and discussing where the technology is headed.

1 Background

The transfer of information can be broken down as the sending of information from Point A to Point B. As such, there are many interactions that can occur during this transaction. Point A could be sending a package to Point B which is received with no further interaction. Point A could be sending an instruction to Point B about which Point B must make a decision. This way of communicating information, much like a conversation between two people, could occur very quickly or carry on for a lengthy period of time.

Following is a passage from “Data Communication Explained” [Data01] which helps to explain this process.

> *Consider Morse Code, a two-state data communication system that functions very similarly to today's computerized data communication systems. Developed by Samuel E.B. Morse in the 19th century, Morse code uses electrical current to transmit a series of dashes and dots that represent letters of the alphabet, numbers, a comma and a period. A basic Morse Code transaction works as follows: A "message" is given to an operator who translates that message into dots and dashes (Point A), then the transmitting operator uses the telegraph key to send an electrical signal to the receiving operator at the desired location to indicate that a message is about to come through. The receiving operator (point B) sends back an acknowledgment that he is ready, and the transmitting operator then sends the message which the receiving operator takes down. When the message is completely transmitted, the transmitting operator signals to the receiving operator that he is done, and the transmission line is closed. The receiving operator then translates the code back into the original message, and delivers it to the designated recipient.*
> *Clearly, in a system of this type, accuracy is extremely important. As only two characters--dot or dash--are used to create a code for an entire language system, the transmitting and receiving operators must be extremely accurate. (Indeed, it makes a big difference whether the message says "Give one million dollars to Ted" or "Give one million dollars to Ned"--an easy mistake to make using Morse code, as the letter T is "-" and the letter N is "-.") This system can only work if both sides of the data communication system know the code and can encrypt and decode messages. It is also essential that the transmitter not send faster than the receiver can take-down the information. Even using expert operators, static on the line could obscure the signals making a dash sound like a dot and thereby corrupting the message. Thus, it becomes obvious that the most important aspect of designing a data communication system is ensuring not only that Point B can receive and understand the data transmitted by Point A, but also that the data remain uncorrupted during transmission. [Data01]*

The color-coded selections of this passage bring up three very important points regarding data communication:

1. Data is similarly understood by all parties and there are no misunderstandings. In other words, everyone needs to be speaking the same language.

2. Data must be sent with stability in order to avoid corruption. An example of this is the popular game in which a group of people form a circle and one person whispers an instruction into a neighbors' ear. That neighbor must whisper the same instruction into his/her neighbors' ear, and on, and on until the source of the instruction has been reached. Many times the instruction will become drastically altered so that it becomes something entirely different. This is the equivalent of how information can be corrupted during data communication.

3. Data must be sent at a specific speed. This is especially important for time-critical applications, such as audio and video.

1.1 Data Communication

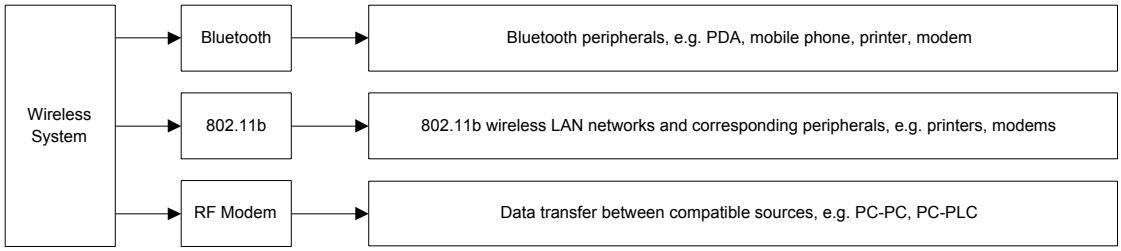
In the world of data communication, it is important to have an effective way of communicating and transferring data so that the data is received in the way it was intended to be received. To accomplish this, data must be sent with precise speed and accuracy. However, the specific requirements of different types of data vary, as do the methods of transfer. Therefore, specific types of input/output equipment have been created to deal with the various types of data. The rest of this chapter will focus on the various features of input/output equipment used for the communication of digital data.

The specifications of input/output equipment, referred to as communication peripherals, could be either ideal or unsuited for the transfer of certain types of data. For example, the use of a keyboard or mouse with a computer has a much different demand than the use of audio or video equipment with a computer. Some types of data transfer, such as the backing up of information on a hard drive, require verification and confirmation to ensure that there is no data corruption. Other types of data transfer, such as audio/video streams, need to be done very quickly without confirmation, which would otherwise slow down the transfer process. Therefore, the equipment used for the transfer of data must correspond with the type of data that will be transferred.

1.1.1 Equipment Features

In order to accurately transmit many different types of data, numerous interfaces and protocols have been created. When selecting the appropriate equipment for a data communication application, it is important to examine both the application and the communication peripherals that must be incorporated into the system. [Data01]

Digital communication systems can be broken down into two categories: Wired Systems and Wireless Systems. Wireless systems allow for the communication between devices without wired connections. These types of systems operate using methods such as radio frequencies to enable communication. Wireless systems do not currently have the high speeds that wired systems have, however advances in wireless technology have been occurring at a remarkable pace. Current high-end wireless interfaces include Bluetooth, 802.11b and RF Modem. Wired Systems include all communication systems that communicate via wired connections. Wired systems can be found in both portable and desktop interfaces. Current high end wired interfaces include PCI/Compact PCI, ISA, PCMCIA/CardBus USB (universal serial bus) and FireWire. (see fig.1.1) This thesis focuses on the wired interface, FireWire.



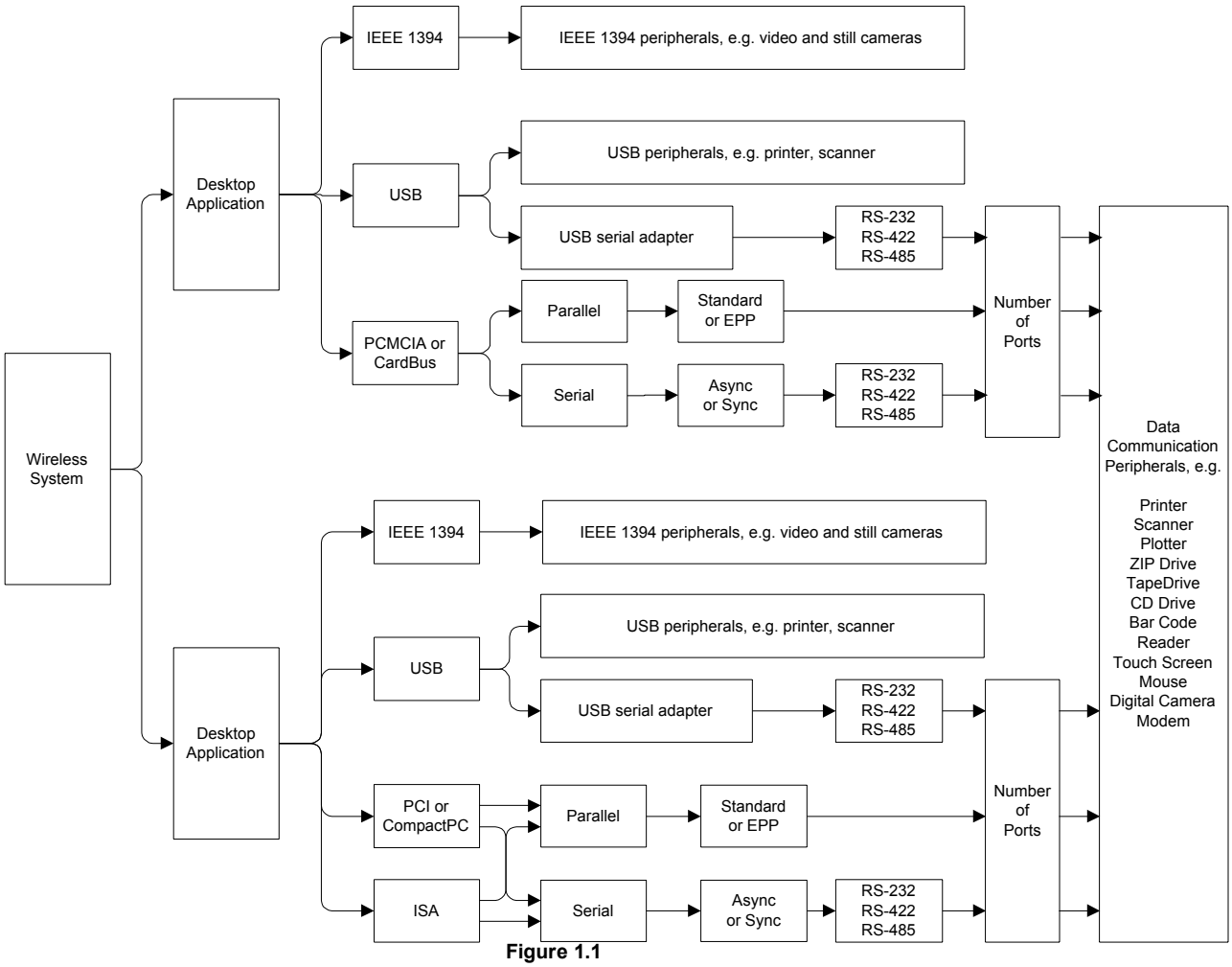


Figure 1.1

Desktop Interfaces	**Portable Interfaces**
FireWire/IEEE 1394	FireWire/IEEE 1394
USB	USB
PCI or CompactPCI	PCMCIA or CardBus
ISA	

Figure 1.1 Assembling A Data Communication System
(from "Communication Overview" http://www.quatech.com/figures/com_ov2001.pdf)

At one time there were many differences between wired desktop and wired portable interfaces. However, as indicated by the previous list, current interfaces serve for both desktop and portable systems. There are many reasons for this including the aspect of having complete compatibility between multiple systems.

Wired communication system peripherals make use of serial buses/ports and parallel cables/ports. A parallel cable allows for parallel data transfer, the transmission of data of more than one bit/several bits at a time, while a serial bus allows for serial data transfer, the transmission of data one bit at a time. (see fig.1.2)

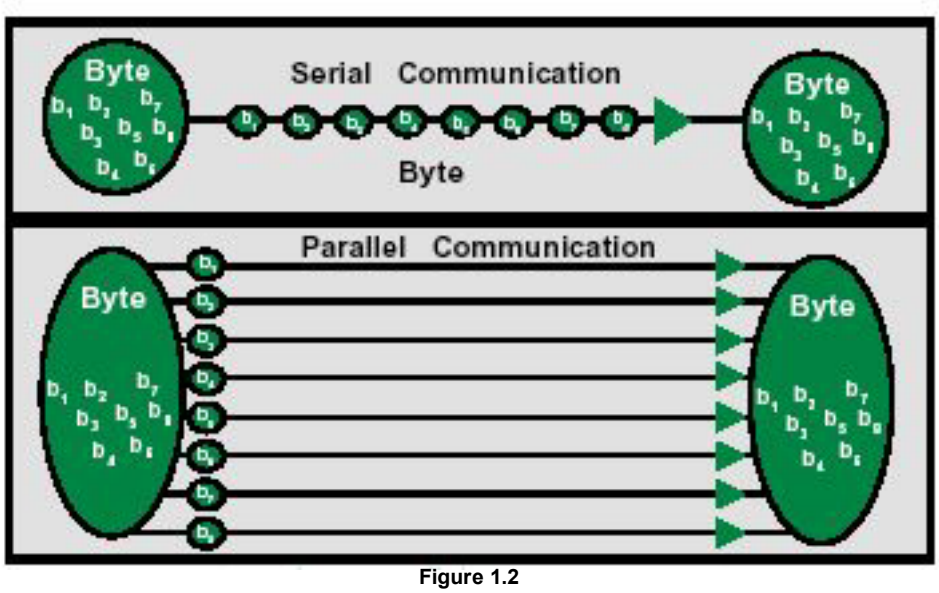


Figure 1.2

At first glance, a parallel cable seems more useful for transporting large amounts of data since there are more wires to carry signals. However, parallel cables can be problematic because they have more electrical interference between the multiple wires, have a large physical cable size, and can produce synchronization problems between wires at high speeds. The large physical cable size of parallel cables also makes them more expensive since there are more wires and shielding. [Moor01]

Figure 1.2 Serial vs. Parallel Communication
(from "Communication Overview" http://www.quatech.com/figures/com_ov2001.pdf)

A serial bus provides a simple point-to point connection allowing scalable capability with technology improvements. A serial cable and its connector are less expensive to manufacture and require less room than a parallel cable. [Moor01] A serial bus can connect a lot of peripherals easily and inexpensively and its data transmission commands are simple and straightforward, such as a *write* to the address of the peripheral or a *read* from the address. [Moor01] These factors have added to the popularity of serial communication over parallel communication.

1.1.2 Transmission Forms

Data communication can take on many different forms of transfer. In the digital world, the most often used methods of data communication are asynchronous, synchronous and isochronous. Following are descriptions of these methods of communication.

Asynchronous communication has been adopted as a simple, cost effective form of data communication. Asynchronous communication is performed between two (or more) devices which operate on independent clocks. Therefore, even if the two clocks agree for a time, there is no guarantee that when point A begins transmitting, point B will begin receiving, or that point B will continue to sample at the rate point A transmits. To combat this timing problem, asynchronous communication requires additional bits to be added around actual data in order to maintain signal integrity. [Data01] This includes the use of start bits (indicating the receiver that a word/chunk of data broken up into individual bits is about to begin), stop bits (indicating the receiver that a word has come to an end), and parity bits (making sure that the data received is composed of the same number of bits in the same order in which they were sent.) All this information makes asynchronous communication ideal for applications which require integrity of data during transfer due to the extensive amount of additional information sent through transfers. However, the additional information sent during transfers slows down the transfer process, which makes asynchronous communication inefficient for time-critical applications such as audio/video.

Synchronous communication differs in how the transmission is broken down. During synchronous communication, the communication partners have a short conversation before data exchange begins. In this conversation, they align their clocks and agree upon the parameters of the data transfer, including the time interval between bits of data. Any data that falls outside these parameters is assumed to be either an error or a placeholder used to maintain synchronization. (Synchronous lines must remain constantly active in order to maintain synchronization, thus the need for placeholders between valid data.) [Data01]

The following passage (from “Data Communication Explained” [Data01]) and illustration (fig.1.3) make convenient analogies of the differences between asynchronous and synchronous communication.

> *Think of the difference in terms of a friendly chat. With asynchronous communication you would need to stop after every word to make sure the listener understood your meaning, and knew that you were about to speak the next word. With synchronous communication, you would establish with your listener that you were speaking English, that you will be speaking words at measured intervals, and that you would utter a complete sentence, or paragraph, or extended soliloquy, before pausing to confirm understanding. Further, you would establish with your listener beforehand that any extraneous noises you make during the speech or between speeches (coughing, burping, hiccupping) should be ignored. Clearly the second approach is much faster, even though initializing communication may take slightly longer. In fact, by replacing the start, stop and parity bits around individual words with start, stop and control (processing instructions and error checking) sequences around large continuous data blocks, synchronous communication is about 30% faster than asynchronous communication, before any other factors are considered. [Data01]*

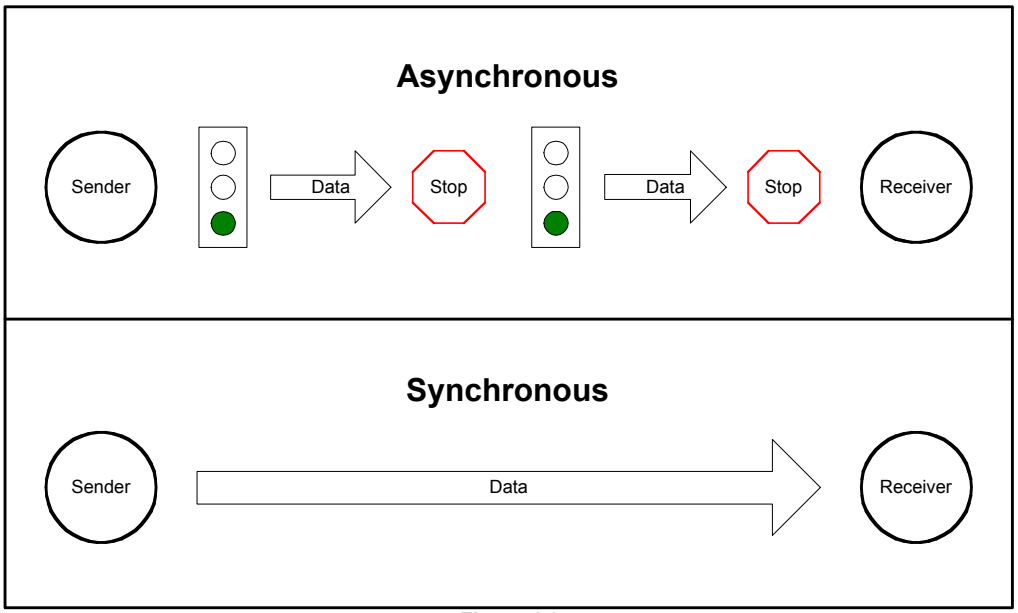


Figure 1.3

Figure 1.3 Asynchronous vs. Synchronous Communication (created in Adobe Illustrator)

Isochronous communication makes use of a fast, steady uninterrupted data stream. Unlike asynchronous and synchronous communication, which both involve elaborate error checking mechanisms, isochronous communication deals with a steady speed of data transfer. Isochronous clocking information is derived from or included in the data stream, and the delay factor is dependent on a channel's characteristics and can be logically determined. Communication can be disrupted if the transmitter does not maintain a constant transfer rate, or if the receiver has an insufficient buffer to store data at that rate. In effect, that data must be held until it can be processed by software. [Data01] The fast, steady uninterrupted flow of data, provided by isochronous communication, is best suited for time-critical applications such as audio/video. Audio/video are time-critical because they consist of continuous streams of information. When audio/video information is transmitted live, it must continue without interruption from beginning to end. Isochronous communication provides the means for an uninterrupted transmission of data.

1.2 The Main Competitors: USB and FireWire

The top two current communication ports for the connection of external peripheral devices and data transfer are USB and FireWire. The USB and FireWIre formats differ from their predecessors in that they do not send raw data. Unlike previous serial and parallel technologies, USB and FireWire send packets of information between devices. In this fashion, a large number of devices can be linked together sharing the same bus and new devices can be added while others are running.

USB released an upgrade, USB 2.0, in 2001. FireWire is in the process of being upgraded and both these technologies seem to be headed for continuing future development. At first glance, USB and FireWire seem very similar as shown by the following summarization of their technical characteristics.

USB 2.0	**FireWire**
1.5, 12 and 480 Mb/s supported	100, 200 and 400 Mb/s supported
USB controller is required to control the bus and data transfer	Works without control, devices communicate peer-to-peer
Cable up to 5 m.	Cable up to 4.5 m.
Up to 127 devices supported	Up to 63 devices supported
Power supply to external devices is 500mA/5V (max.)	Power supply to external devices is 1.25A/12V (max.)

Both USB 2.0 and FireWire are up to the task of handling the transfer of digital audio. However, FireWire is better suited for dealing with professional audio/video applications in which there is a demand for high quality. This is due to the following factors:

USB, 1.0 and 2.0, is designed to move chunks of data as fast as possible without regard for the integrity of the stream, thus is best suited for handling burst delivery of data, data that can be transferred as quickly as the connected devices will allow. USB is more like a standard network connection, but between dumber devices. USB is a first-rate protocol for peripherals like printers, scanners and even MIDI, but is not suitable for use as a professional audio or video transport, unless bandwidth needs are extremely light. It is possible for audio/video to run over USB, just not nearly as much of it will run before data gets dropped.

Although USB is supposed to be a plug-and-play protocol, it has a few quirks. The first is that USB is master-reliant, not peer-to-peer like MIDI or FireWire. This means that although a USB MIDI keyboard can be connected to a computer, that keyboard cannot be directly connected to a USB sound module. The computer, or host, must play traffic cop. In fact, it isn't physically possible to connect two USB peripherals, because USB cables have a different connector on each end to ensure that you can't make such circular hookups. [Batt02]

While FireWire transfers data every 125 microseconds, USB adds a delay because all data must be transferred to the host on the USB transport's time scale, usually 1 ms intervals. This along with the time needed for an ADC (analog-to-digital converter) to make a conversion can create more than 1.33 ms before an application is notified that data is ready.

Firewire differs in that it was designed specifically with audio/video streaming in mind. FireWires' architecture is built around its' isochronous channel, synchronization and device handshaking as defined in the spec. These features, which make FireWire ideal for audio/video applications, will be discussed in depth in chapters 2 (FireWire Technology) and 3, (Handling Audio With FireWire).

In conclusion, USB is well suited for connecting low-bandwidth devices, however it is a poor interface for connecting faster devices or devices that inherently involve large numbers of transactions. FireWire is well suited for time-based media applications such as digital audio/video. Also, unlike USB, FireWire does not require host intervention with every transaction.

2 FireWire Technology

FireWire is the name of Apple's trademark high speed, high performance, serial input/output (I/O) bus. Technologists at Apple Computer conceived FireWire in 1986 as a way of replacing a variety of interface ports on the back of their computers. The name 'FireWire' was chosen in reference to its high speeds of operation and the thermal "noise" created in the connections when the highest speeds were implemented. The first specification for FireWire technology was completed in 1987. FireWire technology conveniently makes use of a high speed, high performance, serial bus via a small sized cable which plugs into small inserts. The serial bus interconnect, provided by FireWire, allows for a wide range of high performance devices to be attached. [Ande99] These devices include, but are not limited to, cameras, video equipment and audio interfaces.

There are many applications that benefit from FireWire technology. The low overhead, high data rates, ability to mix real-time and asynchronous data on a single connection, and the ability to mix low speed and high speed devices on the same network provides a truly universal connection for almost any consumer, computer, or peripheral application.

2.1 Technical Characteristics

FireWire operates through the implementation of special integrated circuit chips. Like Ethernet and other high-speed digital data transmission systems, FireWire is a layered transport system. The FireWire standard defines four protocol layers to simplify the implementation of hardware and software. The protocol layers consist of the: Bus management layer, Transaction layer, Link layer, and Physical layer.

Following is a brief description of each layer:

- Bus management layer — supports bus configuration and management activities for each node.

- Transaction layer — provides the protocol for asynchronous transfers and can perform bus configuration and management.

- Link layer — The basic function of the link is to translate data from the application into packets and send them across the FireWire bus via the physical layer, in addition to decoding packets received from the bus via the physical layer and forwarding them to the application.

- Physical layer — provides electrical and mechanical interface to the bus allowing packet transfers and bus arbitration. [Phil02]

The FireWire connector comes in 4 pin and 6 pin versions. The 4 pins, found in all connectors, allow for signaling. Signaling is performed using two separately-shielded twisted pair transmission lines: one pair for data transmission and another for synchronization. [Ande99] The two twisted pairs are crossed in each cable assembly to create a transmit-receive connection. The additional 2 pins in the 6-pin version allow for bus power (from 8vdc to 40vdc, 1.5 a max.), or rather allow for a device to draw off the power of its host device (see fig.2.1).

The ability for bus power allows for the ease of such things as field recording with a battery powered laptop computer. The reason for this is that a battery in a laptop computer has enough energy to power both a laptop computer and most bus powered devices simultaneously. However, the additional energy charge will drain the battery faster, especially if multiple bus-powered devices are in use.

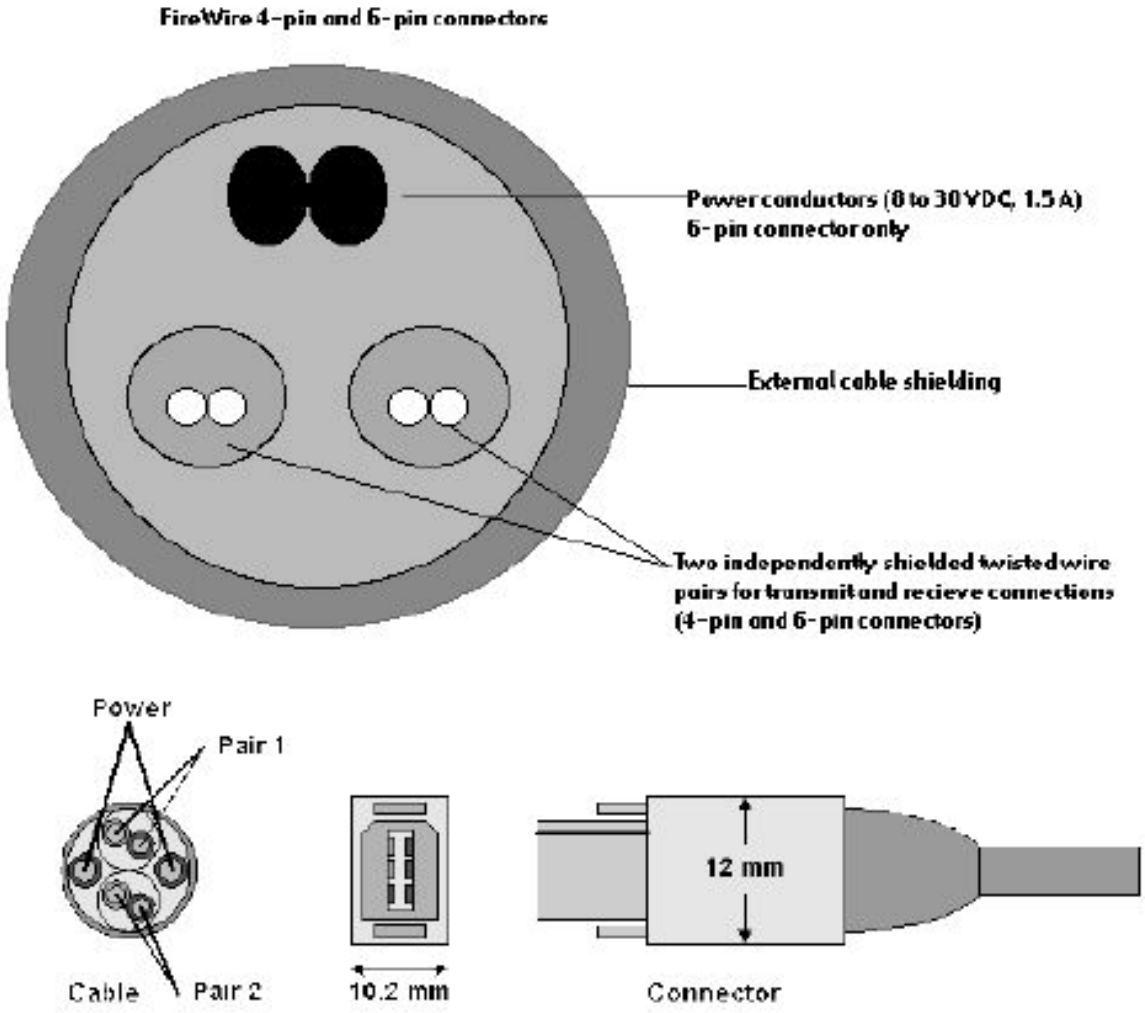


Figure 2.1

FireWire devices support operations of anywhere between 100 - 400 Mb/s These high speeds make FireWire an optimum candidate for the recording and transfer of audio in real time and allow for multi tracking which is an important aspect of audio recording.

According to the standard, the FireWire cable is good for 400 Mb/s over 4.5 meters. Longer cable runs can be achieved by using thicker cable or by lowering the bit rate. Although way outside of the spec, several people have reported successful 100 Mb/s transmissions over more than 20 meters using standard cable. There are also reports of thicker cables being used to span lengths of 30 meters or more at 100 Mb/s.

Figure 2.1 A FireWire Connector
(top half created in Adobe Illustrator, bottom half from “Fire on the Wire: The IEEE 1394 High Performance Serial Bus” by Roger Jennings)

FireWire supports 64 node addresses (0-63) on a single serial bus implementation. [Ande99] This means that up to 63 devices can be daisy chained on a FireWire bus. These devices do not share bus speed, therefore multiple devices can be connected to a FireWire bus with no reduction to the speed of data transfer. Each of the 64 nodes (0-63) has 256TB of address space, making the total address space 16 petabytes.

FireWire allows for hot plugging, also known as plug and play. This allows for the addition/removal of devices to/from a computer while the computer is running. All devices on a FireWire bus will be automatically configured without the need for device IDs or terminators. The addition/removal of devices is automatically recognized by the operating system.

FireWire provides a flexible bus management system that provides connectivity between a wide range of devices. Bus management involves the following three services:

- A cycle master that broadcasts cycle start packets and initiates the start of isochronous transactions on regular 125μs intervals.

- An isochronous resource manager tracks the number of isochronous channels available and the amount of isochronous bus bandwidth available and allocates these resources when requested by serial bus nodes.

- A bus manager publishes maps that specify the topology of the serial bus and that indicate the maximum packet speed support by each cable segment in the system. Also, it performs a variety of bus management functions associated with bus power bandwidth reservation for asynchronous transactions and bus optimization. [Ande99]

Bus reset forces all nodes into their initialization state and starts the configuration process. On bus reset, the structure of the bus is determined, node IDs (physical addresses) are assigned to each node, and arbitration for cycle master, isochronous resource manager and bus master nodes occurs (see fig.2.2). [Jenn98]

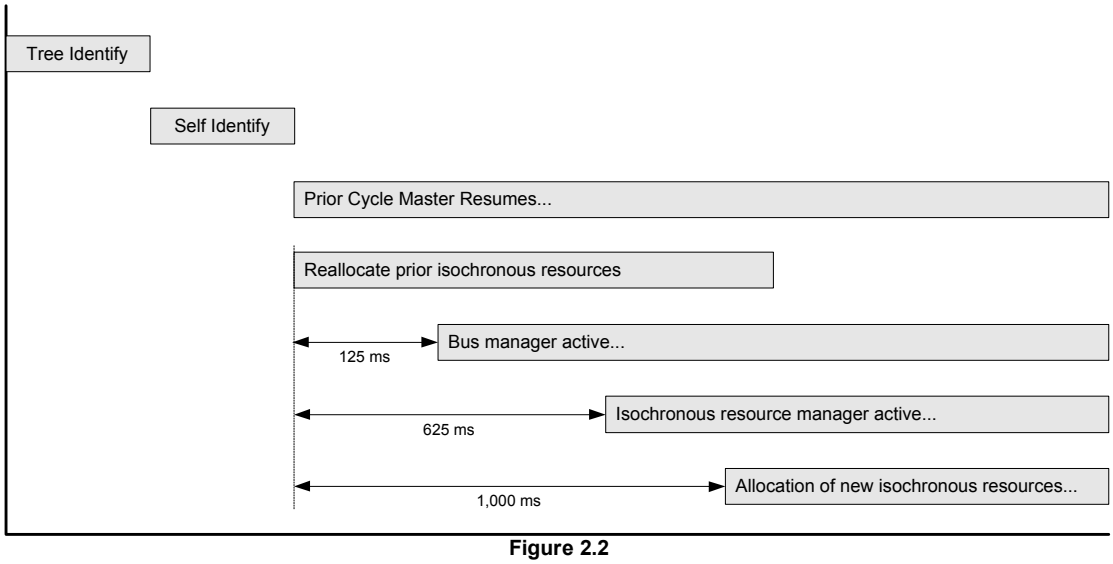


Figure 2.2

2.2 Technical Characteristics Pertaining to Audio

FireWire supports two data transfer types: asynchronous and isochronous. Asynchronous transfers make use of error checking through confirmation of data delivery. Asynchronous data is transferred to an explicit address, and is used by non-time critical applications. Isochronous transfers do not require confirmation of data delivery. Isochronous data is transmitted across channels, which have guaranteed bandwidth and latency. FireWire implements arbitration to ensure that isochronous applications (time critical) are guaranteed a constant bus bandwidth, while asynchronous applications (non-time critical) are permitted access to the bus on a fairness algorithm. [Ande99] Isochronous transfer is important for the transfer of audio data, a time dependent application.

Synchronization of devices is very important during isochronous transactions because there is no confirmation of data to verify if data is being sent correctly. There are two attributes of isochronous transactions that relate to synchronization. The first is that there is a Cycle Start Packet broadcast by one of the devices on the bus 8000 times per second that keeps all of the devices locked together and counting at the same rate. Another is that isochronous packets have priority on the bus over other types of packets such that devices sending them are guaranteed bandwidth. In other words, if an audio device is using, say, 4Mb/s of bus bandwidth and a disk drive starts doing things on the bus too, that audio device will continue to operate without glitches because its isochronous packets have priority over the asynchronous packets of the disk.

Figure 2.2 Bus Reset Timeline
(from “Fire on the Wire: The IEEE 1394 High Performance Serial Bus” by Roger Jennings)

Another aspect of synchronization is that FireWire has the capability of identifying streams, whether they be audio or video, and guaranteeing bandwidth for each stream to be delivered to the destination device within a defined window - that window being the destination devices' buffering requirement to maintain full playback integrity, i.e.: synchronization.

FireWire makes use of peer-to-peer technology which supports peer-to-peer transfers. Peer-to-peer transfers allow for data transfer without intervention from a host system. This enables high throughput between devices without adversely affecting performance of the computer system. [Ande99] For example, a DAT (digital audio tape) machine can set up a transfer between itself and an external hard drive, when both reside on the same bus.

FireWire also allows for a specific type of peer-to-peer communication called "device handshaking". The control channel built into FireWire lets two devices cooperatively monitor the health of a real-time data stream. If the destination device gets bogged down or needs another shot at a segment of data, it can make requests of the source device, and the two will conspire to "open up the pipe" long enough to transfer the extra data, but *without interrupting the synchronous delivery of the main stream*.
It may seem like a subtle difference from standard file transfer and networking processes, but when running a lot of real-time audio or video streams (i.e. multi-tracking) it becomes indispensable.

The isochronous transfer of data, synchronization and device handshaking in the FireWire specifications seem to imply that it was designed specifically with audio and video streaming in mind.

A summarization of FireWires' current technical characteristics follows:

1. Speeds of 100, 200 and 400 Mb/s supported.

2. Peer-to-peer transfer allows for the ability of bus devices to perform transactions between themselves, without the intervention of a host CPU.

3. Cable up to 4.5 m.

4. Up to 63 devices supported on a single serial bus.

5. Power supply to external devices.

6. Hot plugging/plug and play allows for devices to be attached or removed from the bus without having to power the system down.

7. FireWire supports both asynchronous and isochronous transfers.

2.3 Standardization and Recognition

Apple opened their FireWire technology for standardization under the auspices of the Institute of Electrical and Electronics Engineers, Inc. (IEEE), an international organization that promotes standards in the field of electronics. Apple did this in order to create an international open standard for their FireWire technology. FireWire was standardized as IEEE 1394 in 1995 and promoted for open licensing in the industry. An international, open standard enables everything FireWire-equipped to talk to each other irrespective of the manufacturer. [Elen02] This is a tremendous asset for the audio world, since there are so many audio devices in the market and an open standard allows for open communication between those devices.

The open standard of FireWire extends beyond the ability to connect a range of compatible peripherals to a computer. FireWire is a powerful networking system that can carry almost any kind of digital data and control signals. This includes the connection of FireWire with different pieces of digital audio/video equipment without the need for a central computer system. The international open standard and networking abilities of FireWire makes it very powerful and attractive to manufacturers of all types of digital devices.

On August 22nd, 2001 Apple Computer received a 2001 Primetime Emmy Engineering Award given by the Academy of Television Arts & Sciences for creating FireWire technology. The award was given because the high-speed connection provided by FireWire has become a key technology in the television industry for moving video on and off computers. [Ples01]

3 Handling Audio With FireWire

The transfer/conversion of audio to/in the digital domain must be done using methods which address the peculiarities of audio information. Audio consists of a stream of information made up of waveforms which may or may not change over time. These waveforms have various characteristics such as varying frequencies and amplitudes. Audio becomes digitized when numbers are used to represent the frequencies and amplitudes that a vibrating object makes to create sound. Those numbers are either converted from an analog form to a digital form or from a digital form to an analog form. In both cases transfers involve additional conversions into voltage levels. These voltage levels, analog representations of waveforms, can then be converted to digital information by sampling the waveforms at discrete intervals. Conversions are done through the use of an ADC (analog-to-digital converter) and a DAC (digital-to-analog converter).

FireWire is well constructed to handle audio information. It does this through its' ability to take a large number of streams and transfer them at a fast and steady pace, without interruption. This is extremely important being that all audio in an audio stream is desired in order to gain an accurate representation of a sound wave. If interruptions occur during transfer, there becomes a possibility for dropouts in which information will not be captured. FireWire deals with this through its' architecture, which incorporates transfer methods, speeds of transfer and additional features such as bus power which are conveniently set up for audio applications. The FireWire architecture is also set up to handle copy protection of audio information. Being that digital information lends itself for exact replication of material copy protection becomes extremely important for audio information in terms of being an important commodity for many artists, companies and industries.

3.1 Analog and Digital Conversion

The process of moving audio through the digital domain can occur in three different ways: digital-to-digital transfers, digital-to-analog transfers and analog-to-digital transfers.

In the case of digital-to-digital transfers, exact replications are made. This process can be used for multiple situations such as the transfer of digital audio between different types of media, such as, from a DAT (digital audio tape) to a computer hard drive. The data must be read and written as fast as it is sent. Digital conversions, which involve changing the sampling rate/bit rate, will slow down the reading/writing process. However, this is not common practice being that these types of digital conversions are usually done after the original data has been transferred into the desired system.

Digital-to-analog conversions are necessary in order to hear the result of a digital audio file. This is because humans hear in the analog, not the digital, domain. In order to make a digital to analog conversion, a digital file must go through a DAC (digital-to-analog converter). A DAC will take a digital file and convert its number values into corresponding voltages which in turn will be sent to a speaker which will vibrate at the rate of the corresponding voltage values.

Analog-to-digital conversions are required in order to record audio from an analog source and convert it to digital audio. This process is done through an ADC (analog-to-digital converter). The process of converting analog audio to digital audio is more difficult and requires additional steps. Three important aspects of audio are considered in order to make an effective digital recording:

1. Frequency

A specific sampling rate must be used to convert an analog audio signal to a digital signal. During the sampling process, the value of a signal is taken over evenly spaced moments in time. The sampling rate is the frequency at which samples are taken and must be at least twice the rate of the analog frequency that is captured. This is because any frequencies above the sampling rate become folded over and when foldover occurs, aliasing (the generation of a false frequency along with the correct one) occurs. Aliasing produces an undesirable buzzing sound. To solve this problem the sampling rate must make use of the sampling theorem, which is:

2 * Sr (sampling rate) per second

[There has been much debate over what sampling rate should be for high quality recordings. Many people believe that the human range of hearing, approximately 20 Hz - 20 kHz, should determine an appropriate sampling rate. Other people believe that a higher sampling rate is more desirable because a low-pass filter (a filter that removes any frequency components of the signal exceeding one-half the sampling rate) would not need to be set so close to the range of human hearing, creating the possibility for audible discrepancies. Still other people believe that audio can be perceived in other ways aside from hearing and that the sampling rate should be set as high as possible in order to keep all the information. Despite the debate, a higher sampling rate will create a larger digital file and take up more space, yet is desirable because more information will be captured.]

2. Amplitude

The dynamic range/amplitude of audio must also be considered during the digitization of audio. This is accomplished by using a bit-rate which represents a quantization interval/the dynamic range of an audio sample. The higher the bit rate, the greater the dynamic range. For example, a 1-bit system can only represent two levels of amplitude (off = 0, or on = 1), while a two bit system can represent four levels of amplitude (off = 00, soft = 01, medium = 10, or loud = 11). Dynamic range climbs dramatically as the bit rate increases. For example a 16 bit system can represent 65,536 levels of amplitude/96 dB, while a 32 bit system can represent 1,048,576 levels of amplitude/192 dB. When the amplitude of a signal being sampled falls between a quantization interval, the input signal is truncated. The result is a square wave for each instance where the digital device cannot reconcile the difference. These square waves leave digital artifacts that do not represent any frequency in the analog waveform. This is known as quantization error. To correct quantization error a process called dithering is used in which broadband noise is added to the signal in order to force the lower level amplitude values up to the next threshold level.

3. Multi-Tracking

Audio is commonly recorded as several tracks/streams to allow for the ability to isolate tracks for mix down. Audio can also be played back from several tracks to create a kind of surround sound environment, as used in many modern theaters. To do this, all audio tracks must be lined up and synchronized so that all tracks start and stop at specific times. This is not difficult to do in the digital domain as long as the digital processor can handle the load and there is no perceptible latency between the various tracks of audio.

These issues must be addressed for the professional transfer of audio in/out of the digital domain. FireWire addresses these issues through its' specifications and comes up with solutions for effectively transferring audio in/out of the digital domain.

3.2 The Speed Issue

Unlike opening a stagnant digital file, such as a picture or text document, audio, like video, consists of a steady stream which is constantly moving and changing. Therefore an audio file has a certain speed associated with it which must be considered. The rate of this speed is directly proportional to the size of the file. Therefore, the sampling rate/bit rate will change the rate of speed in which the file is read/played.

To effectively capture audio, a recording device must be able to read/record every bit of information at least as fast as it is sent. Due to the extended signal path, transfer speed is especially important when recording from a third party device to a computer system. As fast as audio is played it must be converted to electrical voltage and then to digital information and sent to a computer to be processed. This must all happen quickly enough so that every moment of audio is captured.

There are a number of speeds associated with the transfer of digital audio, all of which can be realized using a simple equation:

SR (sample rate) * BR (bit rate)/8 = Bytes per Second

The key here is 8 bits in a byte. For 'X' samples per second 'Y' bits per sample are being recorded. Multiplying the number of bits by the sample rate gives the number of bits per second. Dividing by 8 gives the number of bytes per second.

So for
44.1 kHz @ 16 bit
Gives
44100 * 16/8 = 88200 bytes/sec.
This translates to
88.2 KB/S or .0882 MB/S

Or for
96 kHz @ 24 bit
Gives
96000 * 24/8 = 288000 bytes/sec.
This translates to
288 KB/S or .288 MB/S

Since every channel of audio is a separate file, the result of the equation must be multiplied by the amount of channels being recorded. Therefore, this number can become very high depending on the amount of channels that are being simultaneously recorded onto.

FireWire provides a very large number of multi-tracking to occur, even at extremely high sampling and bit rates. If, for instance, a recording session required 200 separate channels to be recorded with a sampling rate of 192 kHz at 24 bits there would still be bandwidth to spare, as shown by the following equation:

192000 *24/8 = 576000 bytes/sec.
Therefore
1 channel = 576000 bytes/sec.
X 200 channels = 115200000 bytes/sec.
This translates to
115200 KB/S or 115.2 MB/S

FireWire supports speeds of 100, 200 and 400 Mb/s so even at its' mid speed of operation FireWire could handle this load.

(Additional factors worth considering are the amount of hard drive space, which would add up considerably over time, and the ability for a computer system/software to continually process such a large amount of information.)

Following is a chart of file sizes/speed of the various combinations of sampling rates and bit rates which are commonly used:

A 1-minute digital audio track yields these approximate file sizes:

44.1 mono:

16-bit: 5.1 MB
24-bit: 7.6 MB

44.1 stereo:

16-bit: 10.2 MB
24-bit: 15.2 MB

48 mono:

16-bit: 5.5 MB
24-bit: 8.3 MB

48 stereo:

16-bit: 11.0 MB
24-bit: 16.6 MB

88.2 mono:

16-bit: 10.2 MB
24-bit: 15.2 MB

88.2 stereo:

16-bit: 20.4 MB
24-bit: 30.4 MB

96 mono:

16-bit: 11.0 MB
24-bit: 16.6 MB

96 stereo:

16-bit: 22.0 MB
24-bit: 33.3 MB

3.3 Isochronous Transfers

The FireWire bus allows for isochronous transfers. An isochronous transfer does not require confirmation of data delivery and transactions consist only of a request transaction. (see fig. 3.1) Therefore, isochronous transactions allow data to be delivered at a constant rate of transfer. This fast, steady, uninterrupted data stream, which is the driving force behind isochronous communication, is an extremely important factor in the recording of multitrack high quality digital audio. [Data01] The isochronous data transport of the FireWire bus provides the guaranteed bandwidth and latency required for high-speed data transfer over multiple channels.

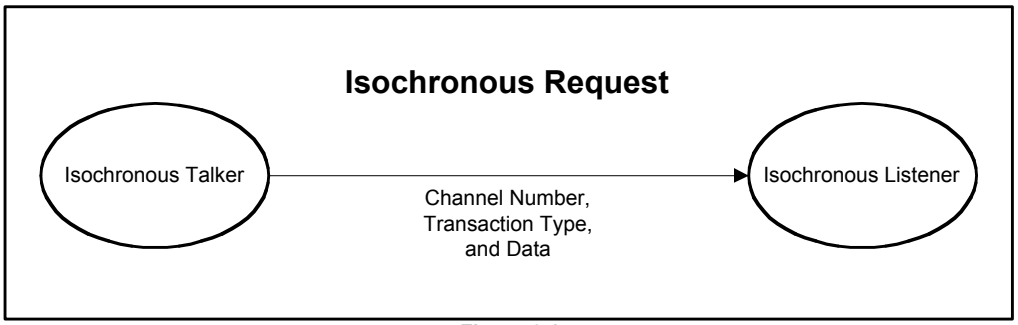


Figure 3.1

Isochronous transactions act by sending data to a target device at regular intervals of 125µs. These transactions occur without feedback from the receiving node. Each transaction makes use of a requester function/isochronous talker and a responder function/isochronous listener. [Ande99]

Isochronous transfers allow for a guaranteed bus bandwidth. To ensure that the bandwidth is available, nodes wishing to perform isochronous transfers must request the needed bandwidth from the node that performs the role of isochronous resource manager. The isochronous resource manager is a single node residing on the FireWire bus that provides services for all isochronous capable nodes. Two registers located within the isochronous resource manager are defined for this purpose. [Ande99]

1. CHANNELS_AVAILABLE
2. BANDWIDTH_AVAILABLE

Figure 3.1 Isochronous Transaction
(created in Adobe Illustrator)

The isochronous resource manager assigns a channel number (0 to 63) to nodes that request isochronous bandwidth based on values in the manager's CHANNELS_AVAILABLE register. The assigned channel number identifies all isochronous packets. When a node no longer requires isochronous resources, it is expected to release its bandwidth and channel number. [Jenn98] For example, the isochronous resource manager would send signals to cause an ADC (analog to digital converter) to commence talking on its channel and a digital recording device (i.e. a computer sequencer) to commence listening on its channel for audio data from the isochronous resource manager application.

Bus bandwidth is allocated on a per cycle basis. When bus bandwidth has been acquired for an isochronous transfer, the channel will receive a guaranteed time slice during each 125 µs cycle.

3.3.1 Synchronization

Synchronization of the isochronous clock intervals is important to maintain synchronization during isochronous transactions. Synchronization is achieved through two methods.

The first uses a Cycle Start Packet broadcast by one of the devices on the bus 8000 times per second that keeps all of the devices locked together and counting at the same rate.

The second is that isochronous packets have priority on the bus over other types of packets such that devices sending them are guaranteed bandwidth. In other words, if an audio device is using, say, 4 Mb/s of bus bandwidth and a disk drive starts doing things on the bus too, that audio device will continue to operate without glitches because its isochronous packets have priority over the asynchronous packets of the disk.

FireWire also has the capability of identifying streams, whether they be audio or video, and guaranteeing bandwidth for each stream to be delivered to the destination device within a defined window - that window being the destination devices' buffering requirement to maintain full playback integrity, i.e.: synchronization.

3.4 Peer-to-Peer Transactions

The FireWire protocol is a peer-to-peer network with a point-to-point signaling environment. Nodes on the bus may have several ports on them. Each of these ports acts as a repeater, retransmitting any packets received by other ports within the node. Peer-to-peer transactions allow for data transfer without intervention from a host system. This enables high throughput between devices without adversely affecting performance of the computer system. [Ande99] This also allows for the interconnection of consumer digital audio gear to be connected to a FireWire bus without the need for a bus manager. Therefore any transmitting device can arbitrate for assignment as the isochronous resource manager. For example, a DAT (digital audio tape) machine can set up a transfer between itself and an external hard drive, without the need for a computer system or other device to act as a bus manager, when both reside on the same bus.

3.4.1 Device Handshaking

Device Handshaking is a specific type of peer-to-peer communication. The control channel built into FireWire lets two devices cooperatively monitor the health of a real-time data stream. If the destination device gets bogged down or needs another shot at a segment of data, it can make requests of the source device, and the two will conspire to “open up the pipe” long enough to transfer the extra data. This is done without interrupting the synchronous delivery of the main stream.

It may seem like a subtle difference from standard file transfer and networking processes, but when running a lot of real-time audio or video streams (i.e. multi-tracking) it becomes indispensable.

3.5 Bus Power

Power available from the FireWire bus can be either sourced or sinked by a given node. When sourced, power does not need to be provided to a device and each node reports its ability to source cable power. When sinked, power is provided to a device via the bus and each node reports its requirements for sinking cable power. When sinking cable power, each node transmits a self-ID packet that contains power class information. [Ande99]

When bus power is used in conjunction with a battery-powered computer, such as a laptop, a higher level and potential of audio field recording can be made. This is due to the flexibility that computers can provide vs. specialized field recording devices such as NAGRA, D.A.T.s and Minidisk recorders. The ability to use bus power with any type of computer, based on FireWires' open standard, allows for an even higher level of flexibility.

(As a side note, a computer can be very useful for field recordings on many different levels. Issues such as various sampling rates, bit rates and on the spot editing can be addressed, and, in addition, issues such as organization/archival, extensive documentation, and high level calculations can also be addressed.)

3.6 Copy Protection

Copy protection has become increasingly important in the audio industry due to digital technology, which allows for many things such as making identical copies of digital audio files and having the ability to compress a digital file so that it takes up very little space. FireWire includes a built-in, proven copy-protection system, known as Digital Transmission Copy Protection, or “5C” (after the five industry leaders who developed it: Hitachi, Intel, Matsushita, Sony, and Toshiba). 5C does its job without prohibiting users from copying audio or video for personal use, providing a “copy once,” option, along with a stricter “copy never” prohibition. 5C content protection is implemented in the link layer protocol of FireWire. The main task of the link layer involves collecting and formatting data for transmission on the Firewire bus (for further info. on the link layer see ch. 2.1 - Technical Characteristics). During transmission Full Authentication is required for nodes wishing to transfer copy-never content and Restricted Authentication may be used for other content types. [Phil02] Consumer Electronic devices capable of Full Authentication must be able to perform the functions of Authentication and Key Exchange (AKE) and content stream cipher/decipher. The Digital Transmission Licensing Authority (DTLA) administers the 5C cryptographic standard, licensing the algorithms and issuing the device certificates. 5C copy protection is not intended to be "professional hacker-proof" but rather to achieve the stated goal of the Copy Protection Technical Working Group of "keeping honest people honest". [Phil02]

FireWires’ ability for copy protection makes it a great asset to the audio community by providing for a way of securing information. This issue becomes increasingly important as music and audio continue to be digitized for the purpose of being a commodity.

4 Applications

There are many audio applications that benefit from the use of FireWire. This chapter focuses on audio applications which make use of FireWire and gives an inventory of current high-end FireWire digital I/O audio devices.

4.1 Recording, Playback and Performance

One of the benefits tackles the nuisance of tangled wires/cables. FireWire makes use of a very minimal amount of cables. The first and most noticeable aspect of using FireWire for audio in a recording studio environment would be the lack of cables and cords. This does not, at first, seem to be such an important issue but anyone working in a recording studio environment will be aware of this, often called “spaghetti” (tangled wires/cables). The ability to daisy chain up to 63 devices without having to share bus speed with the connected devices allows for data to be sent/received between multiple devices along the same bus.

Field recording is also an important aspect of the audio world. Field recording allows for the ability to record in environments which needed sounds are sought. Field recording applications include: production sound/on location sound for film/video, specific environmental sounds such as waves crashing on a beach or rain in a forest, wildlife, live music events and ideal acoustical spaces for music such as specific rooms or halls. FireWires’ ability to provide bus power is a perfect complement to the area of field recording. The ability to source power allows for many devices to rely on a single power source. This allows for the ability of a battery-powered device, such as a laptop computer, to provide bus power to any devices attached on the FireWire bus. The more devices using bus power, the higher the drain on the battery. However, field recordists in all disciplines tend to prepare for this by bringing extra batteries to locations.

An advantage of using FireWire to interconnect consumer digital audio gear is that the FireWire bus is operable without a bus manager, and any transmitting device can arbitrate for assignment as the isochronous resource manager (part of the peer-to-peer nature of FireWire, see ch.3). Thus a CD player and a pair of powered loudspeakers can be connected without the need for a computer system or other device to act as a bus manager. The company SoftAcoustiK has developed a FireWire digital audio solution for loudspeaker applications. SoftAcoustik technology repositions the traditional analog loudspeakers in the digital domain while allowing the loudspeakers to act as peripherals. This technology is available as a turnkey solution to loudspeaker manufacturers.

The use/incorporation of a computer system for live performance is becoming more common. However, the output of multiple tracks from a computer system usually requires additional hardware since most computer systems only have a built-in stereo output. FireWire provides a way of connecting an external audio device to a computer, and enabling multiple channels of audio to be sent from a computer.

4.2 Surround Sound

The use of surround sound is constantly being explored as a way to record, playback and perform audio. Surround sound is also used in installation gallery situations, for example, Engine27, a sound gallery located in N.Y.C. The problem that most people face in dealing with surround sound is the amount of equipment involved and potential for compatibility problems between equipment. The use of a modern computer system in a surround sound setup can help make things easier. This is because software conveniently allows for the playback and manipulation of simultaneous multiple tracks in real time. FireWire provides a means of easily transmitting these multiple tracks to a DAC (digital-to-analog converter). The amount of tracks that the DAC can simultaneously input and output will directly correspond to the number of individual tracks that can be independently played back.

4.3 mLAN

mLAN is the name for a specification of the communication interface designed for musical instruments and audio equipment. mLAN was developed by Yamaha as a means of transmitting both digital audio and MIDI information between devices via a single FireWire cable. Yamaha believes FireWire is the best selection for mLAN implementation and therefore selected FireWire to transmit digital audio and MIDI. This being the case, one of the most important specifications of mLAN is the transmission protocol of high quality multi channel audio and performance data such as MIDI data over FireWire.

mLAN is advertised as having the ability to transfer music and audio data at speeds of up to 200 Mbps. mLAN allows up to, approximately, 100 channels of digital audio data or up to 256 ports of MIDI data (16 channels x 256 connections) to be transferred via a single cable. (see fig. 4.1)

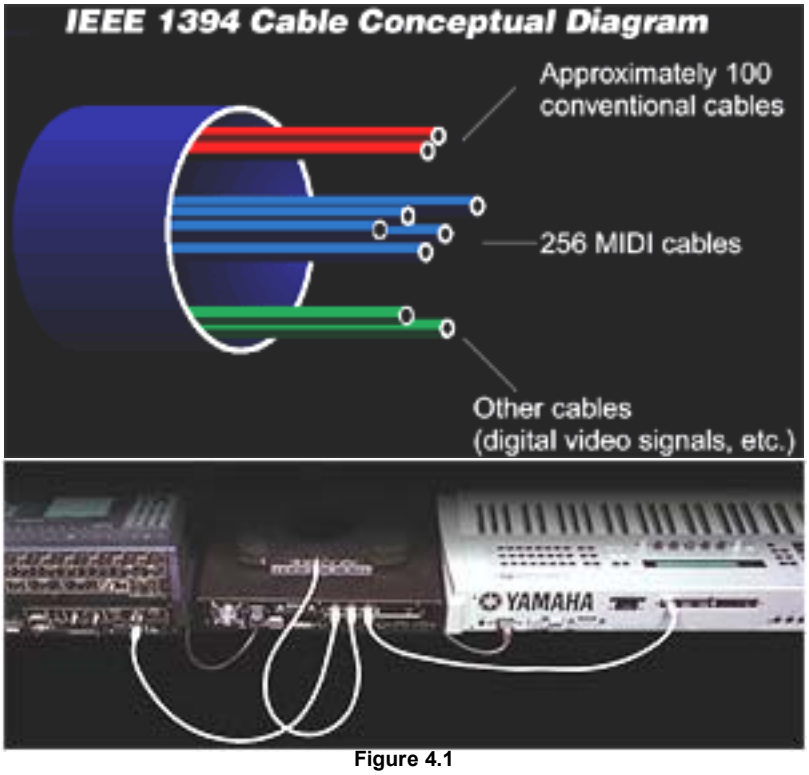


Figure 4.1

Where conventional systems require separate, multiple cables for MIDI, audio, and serial data connections, in an mLAN system all of these functions are handled by a single cable between each device in the network. (see fig. 4.1) Devices can be connected in any order, and mLAN ports are “hot-pluggable” so devices can be plugged in/unplugged without having to power-down or reset the system. [Yama02] mLAN also allows for music and audio devices to be networked without a computer.

The mLAN protocol is Yamaha’s way of capitalizing on FireWire technology. mLAN is a focus on the use of FireWire as an audio interface and merely makes use of FireWire as an audio interface. However, one cannot simply plug in a FireWire cable between two audio devices (or an audio device and a computer) and have mLAN; both units have to understand and use Yamaha’s mLAN protocol. For devices to understand the mLAN protocol they must be outfitted with an mLAN card, which can then be used in conjunction with FireWire.

There are currently some Yamaha and Korg keyboards that support the mLAN protocol, the PreSonus FireStation, and some Yamaha mixers can be outfitted with an mLAN card.

Figure 4.1 Yamahas’ mLAN protocol
(from Yamaha® Pro Audio Products “Simple, Powerful Networking For Music Systems” http://www.yamaha.com/proaudio/products/system_mlan.htm)

4.4 Case Study: Metric Halos' MIO (Mobile Input/Output)

Many companies have bought into the FireWire specs and gone into the manufacturing of FireWire audio interfaces for the purpose of multitrack recording. These companies include Digidesign/ProTools, MOTU, Panasonic, Crest, Yamaha and Metric Halo. The primary function of a digital multitrack recording interface is to provide quality ADCs (analog to digital converters) and DACs (digital to analog converters). However, additional features such as availability of different sampling/bit rates, various types of analog and digital connections, preamps and phantom power are more appealing for professional applications.

Metric Halo has introduced a FireWire audio interface, the Mobile I/O 2882, which has a high potential to be used for professional applications. The Mobile I/O 2882 has all the features that make for a professional audio interface including the ability for complete portability. This makes the device an optimum candidate for a case study of a FireWire audio interface.

The front panel of the MIO shows ten-segment metering for 8 inputs and 8 outputs. As with a conventional mixing board this gives the ability to monitor signals of up to 8 input/output tracks. To the right of the metering the MIO displays 4 sections of information. These sections are: Sample Rate, Clock Source, Status and Digital I.O. These features provide for such things as analog to digital conversion at various sample rates and synchronization between devices through clock information. These are important features to have for the recording of professional audio. There is also a TRS 1/4" output jack for headphones with Mute and Dim buttons which provide access to level control for the headphone output. The Mute button mutes the headphone output while the Dim button attenuates the headphone output by 18 dB. This is mainly used for control room monitoring. (see fig. 4.2)

Mobile I/O Front Panel

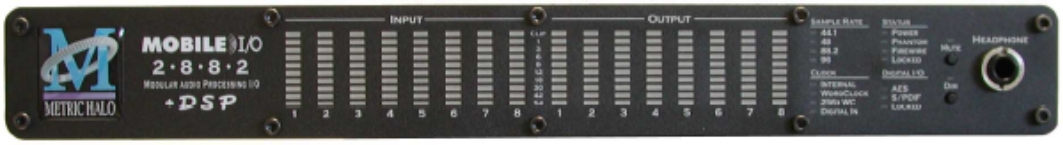


Figure 4.2

The rear panel of the MIO reveals the devices audio digital and analog inputs and outputs along with its FireWire ports. These connections allow for sound to travel to and from a sound source and computer system. There are essentially 7 connection types on the MIO: Analog XLR inputs, Analog TRS inputs and outputs, Optical in and out, Wordclock in and out, SPDIF in and out, AES in and out, and 2 FireWire Ports. The two FireWire ports allow for two FireWire devices to be connected and recognized by all devices on the serial bus. The inputs/outputs on the MIO allow for most of the connections that you would need in a professional environment. (see fig. 4.3)

Mobile I/O Rear Panel

Figure 4.3

Figure 4.2 Mobile I/O Front Panel
Figure 4.3 Mobile I/O Rear Panel
(from Metric Halo's Mobile I/O User's Guide "MobileIOManual.PDF")

The MIO audio transport, as listed in the specs, takes advantage of FireWires' support for isochronous transmission. This allows the MIO to reserve a dedicated amount of bandwidth on the bus for moving audio samples. The MIO manual states that the isochronous model is perfect being that “the audio must be transmitted on a regular basis to ensure continuous playback and recording”.

The MIO makes use of all 6 pins of the FireWire connector by allowing for bus power support. This feature opens up the capabilities of the device by allowing for field recording in conjunction with a laptop or any computer running on battery power. The Mobile I/O allows two options for power DC power or via FireWire bus. The Mobile I/O uses 12 Watts of power thus the device supplying the bus power must be capable of sourcing that much power. This is not a problem for the majority of desktop computers. Most laptop computers also provide enough power, however not enough power for a Mobile I/O and another bus-powered device at the same time. The manual specifically states, “If you are using a Powerbook computer, you should not expect to power both the Mobile I/O and a hard drive from the computer”. The ability to power the MIO via bus power offers a tremendous advantage for field recording. The acoustics of any environment can be captured whether it is the sounds of a secluded forest or of a small musical ensemble in a great hall. And whereas before these types of recordings would take a lot of effort in order to make high quality simultaneous multichannel tracking, the MIO makes it very easy. Being that the MIO is shown as having very high level/professional features the quality of the recordings have very high potentials.

The MIO has effectively taken care of the latency issue. This is done by allowing performers to monitor themselves through the box without having to deal with the latency of the host device. The host device, namely a computer, will most likely not be designed to deal with audio latency in an effective manner and the performer will be subject to the buffer size of the computer. This on top of having the task of processing all incoming audio can potentially create a latency problem. Going through the MIO box the performer gains the MIO feature of virtually zero latency (<2 ms).

4.5 List of Current FireWire Digital I/O Devices

The audio devices that connect to computers for the purposes of recording/playing audio information are in essence analog to digital and digital to analog converters. That is, they take audio waveforms, sampled at regular intervals, and convert these samples into digital numbers. The reverse is done when converting numbers into voltages for playback. The differences between these devices lie in their various features and connection types. One device might feature 4 analog inputs and outputs with sampling rates of up to 44.1 kHz at 24 bit, while another device might feature ADAT lightpipe input and output of up to 8 channels, 2 analog inputs and outputs with sampling rates of up to 96 kHz at 24 bit, and a MIDI input, thru and output. The varieties of devices have been made to cater to a variety of audio needs.

Following is an inventory of current high-end FireWire digital I/O devices for audio applications:

Company	Product	Analog I/O	Digital I/O	Sample Rate	Bit Rate	Additional Features
Crest	Audio FB-88	8 XLR-1/4" combo inputs 8 1/4" TRS outputs	(8/8) ADAT S/PDIF	44.1, 48, 88.2, 96 kHz	24-bit	includes MIDI I/O.
Digidesign	Digi 002	4 XLR inputs 4 1/4" TRS inputs 8 1/4" TRS outputs	(8/8) ADAT (2/2) S/PDIF	44.1, 48, 88.2, 96 kHz	24-bit	8 motorized faders, 4 phantom-powered mic-preamps, MIDI I/O, transport control.
Metric Halo	Mobile I/O 2882	4 XLR inputs 4 1/4" TRS inputs 8 1/4" TRS outputs	(8/8) ADAT lightpipe S/PDIF AES/EBU	44.1, 48, 88.2, 96 kHz	24-bit	8 phantom-powered mic-preamps FireWire bus powerable.
Metric Halo	Mobile I/O 2882 + DSP	4 XLR inputs 4 1/4" TRS inputs 8 1/4" TRS outputs	(8/8) ADAT lightpipe S/PDIF AES/EBU	44.1, 48, 88.2, 96 kHz	24-bit	8 phantom-powered mic-preamps FireWire bus powerable, DSP provides additional processing power.
Metric Halo	ULN	2 XLR-1/4" combo inputs 2 1/4" TRS outputs	S/PDIF AES/EBU	44.1, 48, 88.2, 96 kHz	24-bit	2 mic-preamps.
MOTU	828	6 1/4" TRS inputs 2 XLR-1/4" combo inputs 8 1/4" TRS outputs	(2/2) S/PDIF (optical) or (8/8) ADAT	44.1, 48 kHz	24-bit	2 phantom-powered mic-preamps.
MOTU	896	(8/8) XLR-1/4" combo	(2/2) AES/EBU (8/8) ADAT	44.1, 48, 88.2, 96 kHz	24-bit	8 phantom-powered mic-preamps.
Presonus	FireStation	8 1/4" TRS inputs	10, ADAT, S/PDIF	48 kHz	24-bit	Implements mLAN featuring a built-in 10 channel mixer, 4 preamps, phantom power, defeatable tube circuit, MIDI I/O.
TerraTec	MIC2	8 XLR/TRS line input 2 XLR mic input 8 direct TRS output	ADAT S/PDIF	96 kHz	24-bit	stand-alone A/D-D/A interface.
TerraTec	MIC8	8 XLR-1/4" inputs 8 1/4" TRS outputs	ADAT S/PDIF	96 kHz	24-bit	stand-alone A/D-D/A interface.

Based on the increasing popularity of FireWire for use with audio the number of devices, from the previous list, can be expected to grow very quickly. At the time of this writing many companies have made announcements that they will be releasing high-end FireWire audio devices in the near future.

5 Conclusion and Future Developments

This thesis examined the use of FireWire as a primary interface for the transfer of digital audio. From its' conception FireWire was designed specifically to address audio and video streaming, hence the isochronous channel, synchronization and device handshaking in the spec. FireWire also provides features such as daisy-chaining, bus power and a copy protection scheme, all of which are important in the world of audio production. As FireWire becomes more accepted within the audio community, its' capabilities and advantages will become more evident. Many companies have already begun making FireWire audio devices the main staple in their line of products.

In terms of the marketing of FireWire technology, various names have been pitched including i.Link and IEEE 1394. That multiple identity has interfered with efforts to sell consumers on the benefit of using the connection to link PCs with FireWire devices. In May of 2002 the 1394 Trade Association announced a deal with Apple allowing the group to market and license the FireWire name along with the underlying technology. "Numbers don't work," said 1394 Trade association President James Snider. "Names work and the FireWire name just sticks." [Frie02] The hope is that the name will catch on and gain attention as being an open standard high speed, high performance serial bus with a high level of compatibility between a large number of devices.

On April 2nd 2002 the Institute of Electrical and Electronics Engineers Standards Association (IEEE-SA) Standards Board approved the IEEE Standard 1394b "High-Performance Serial Bus". IEEE 1394b upgrades the prior standards by allowing for gigabit signaling and by extending signaling distance to 100 meters. [Woot02] The IEEE 1394b specs show that the bus will support speeds of 800 Mb/s, 1.6 Gb/s and 3.2 Gb/s. Another solution had been proposed that also increased the serial bus speed. This proposed version, given the name 1394.2, was not backwardly compatible with earlier 1394 versions which caused considerable opposition. [Ande99]

With the future development of the next generation of FireWire technology/IEEE 1394b this interface will become faster and more efficient. Based on these developments, to an interface that already effectively deals with audio, it seems that FireWire will be come the standard for computer-based recording.

Glossary

ADC: (analog-to-digital converter) A device which accepts analog signals and converts them into digital ones.

aliasing: The generation of a false frequency along with the correct one when doing frequency sampling.

asynchronous: A form of data transmission that does not require guaranteed bus bandwidth or delivery at a constant data rate and can tolerate long latencies between data transfers. Asynchronous transfers verify data delivery by error checking through two subactions; bus transaction is initiated by a requester node and received by a responder node that returns a response.

bandwidth: A data transmission rate; the maximum amount of information (bits/second) that can be transmitted along a channel.

bit: (abbreviation for binary unit or binary digit) The smallest amount of digital information. A bit can store or represent only two states, 0 and 1.

bit rate: The number of bits being recorded or transmitted each second.

Bluetooth: A short-range radio technology aimed at simplifying communications among Net devices, between devices and the Internet, and data synchronization between Net devices and other computers.

bus: A collection of wires through which data or power is transmitted. All buses consist of two parts; a data bus and an address bus. The data bus transfers actual data whereas the address bus transfers information about where the data should go.

bus management layer: The FireWire protocol layer which supports bus configuration and management activities for each node.

bus power: Allows for power to be sent from one device to another device(s) chained to the same bus.

byte: A group of eight bits operating together. Usually abbreviated as an uppercase B to distinguish “byte” from “bit” which uses a lowercase b.

CardBus: The trade name for a credit card-sized memory or I/O device that fits into a personal computer, also known as PCMCIA card (see PCMCIA).

coaxial: A type of cable that includes one physical channel that carries the signal surrounded (after a layer of insulation) by another concentric physical channel, both running along the same axis.

compact PCI: A more compact, card version of PCI (see PCI).

cycle master: A root node responsible for initiating each isochronous interval (~125µs) by transmitting a cycle start packet.

cycle start: The beginning of an isochronous interval (initiated by the cycle master node), which notifies all isochronous capable nodes that they may begin arbitration to transmit their isochronous transactions.

DAC: (digital-to-analog converter) A device which accepts digital signals and converts them into analog ones.

daisy chain: A method of connecting devices on a bus in which devices not requesting a signal respond to it by passing it on.

device handshaking: A specific type of peer-to-peer communication which lets two devices cooperatively monitor and make decisions during a real-time data stream.

802.11b: The specification for wireless fidelity/Wi-Fi given by the IEEE. 802.11b operates in the 2.4 GHz range offering data speeds up to 11 megabits per second.

external bus: A bus that connects a computer to peripheral devices.

FireWire: Apple's trademark high speed, high performance, serial bus. Also known as IEEE 1394 and i.link.

hot plugging: The ability to add/remove devices to a computer while the computer is running and have the operating system automatically recognize the change. Also called "plug and play".

IEEE 1394-1995: The IEEE 1394 specification was released in 1995 and was thus given the name IEEE 1394-1995 (see FireWire for further info.).

IEEE 1394a: Different interpretations of the IEEE 1394-1995 specification have led to interoperability problems. Thus, a supplement to the 1995 specification called 1394a was developed. This supplement added additional features and improvements for the purpose of increasing performance/usability. Both IEEE 1394-1995 and IEEE 1394a have been marketed to the general public under the name FireWire, IEEE 1394 and i.link (see FireWire for further info.).

IEEE1394b: This specification, in development, will define serial bus extensions for running the serial bus at speeds into the gigabit per second range.

ISA: (Industry Standard Architecture) A standard bus (computer interconnection) architecture that is associated with the IBM AT motherboard. It allows 16 bits at a time to flow between the motherboard circuitry and an expansion slot card and its associated device(s).

isochronous: A form of data transmission that guarantees bus bandwidth to provide a certain minimum data rate, as required for time-dependent data such as video or audio.

isochronous resource manager: A single node residing on the FireWire bus that provides services for all isochronous capable nodes, including isochronous channel allocation and bus bandwidth allocation.

link layer: The FireWire protocol layer which provides the translation of a transaction layer request or response into a corresponding packet, or subaction, to be delivered over the serial bus. This layer also provides address and channel number decoding for incoming asynchronous or isochronous packets. CRC (cyclic redundancy check) error checking is also performed here.

MIDI: (Musical Instrument Digital Interface) A standard for the interconnection of digital music processing devices (e.g. keyboards, signal processors) and computers together.

network: A series of points connected by communication channels.

node: An addressable device attached to a serial bus/FireWire that fulfills the minimum set of requirements needed for configuration and control. (FireWire allows up to 63 nodes per serial bus).

optical: The medium and the technology associated with the transmission of information as light as light pulses along a glass or plastic wire or fiber.

parallel data transfer: The transmission of data of more than one bit/several bits at a time.

parallel port: A port or interface which can be used for parallel communication.

PCI: (Peripheral Component Interconnect) Designed by Intel as an interconnection system between a microprocessor and attached devices in which expansion slots are spaced closely for high-speed operation.

PCMCIA: (Personal Computer Memory Card International Association). Standards for a credit card-size memory or I/O device that fits into a personal computer, usually a notebook or laptop computer.

peer-to-peer technology: A communications model in which each party has the same capabilities and either party can initiate a communication session and send/receive data.

peripheral device: A computer device that is not part of the essential computer. Peripheral devices can be internal/integrated peripherals or external.

physical layer: The FireWire protocol layer which provides the electrical and mechanical interface required for transmission and reception of data bits (packets) transferred across the serial bus. The physical layer also implements an arbitration process to ensure that only one node at a time transfers data across the bus.

plug and play: (See hot plugging).

port: The entrance to or exit from a network in which a data channel is dedicated to receive/transmit data from/to one or more external remote devices.

RCA: A plug and a jack designed for use with coaxial cable for frequencies ranging from the very lowest up to several megahertz. An RCA connector is sometimes known as a phono plug and jack.

sample rate: The number of samples of a sound that are taken per second in order to create a digital representation.

serial data transfer: The transmission of data one bit at a time.

S/PDIF: (Sony/Philips Digital Interface) A standard audio transfer file format usually found on digital audio equipment such as a DAT machine or audio processing device. Allows for transfer of audio from one file to another without conversion to and from an analog format, which could degrade the signal quality.

stream: Uni-directional data transmission.

synchronous: A form of data transmission similar to asynchronous data transmission, except that data parameters are verified before transmission and not during transmission.

terminal: A point at which information can enter or leave a communication network.

transaction layer: The FireWire protocol layer which supports the request-response protocol for read, write and lock operations related to asynchronous transfers.

USB: (Universal Serial Bus) A plug-and-play serial bus interface. The USB peripheral bus standard was developed by Compaq, IBM, DEC, Intel, Microsoft, NEC, and Northern Telecom.

wired system: A system that allows for communication through wired connections.

wireless system: A system that allows for communication without wired connections.

References

[139402] (2002) The official website for the “1394 Trade Association” http://www.1394ta.org/

[Ande99] Anderson, Don (1999) “FireWire® System Architecture: IEEE1394a / Mindshare, Inc.” 2nd edition, Addison Wesley

[Ande01] Anderson, Don (2001) “Universal Serial Bus System Architecture” 2nd edition Addison Wesley

[Batt02] Battino, David (2002) “Surfin’ USB” Electronic Musician Vol18, No.10 September 2002

[Cano02] Canosa, John “Fundamentals of Firewire” Questra Consulting, Software And Embedded Systems Group written for Questra.com http://www.questra.com/newsroom/article_files/web_services_journal_jan2002.pdf

[Data01] “Communication Overview” www.quatech.com, http://www.quatech.com/figures/com_ov2001.pdf
(I contacted this company about referencing their publication and they replied that their policy is that there is no authorship associated with their websites publishing.)

[Elen02] Elen, Richard (2002) “Fire in the Wire - The Future of Audio Interconnects” written for AudioRevolution.com http://www.audiorevolution.com/news/0702/11.firewire.shtml

[Frie02] Fried, Ian (2002) “What’s in a name? For FireWire, plenty” written for CNET.com http://www.news.com.com/2100-1040-928089.html

[Ieee02] (2002) “IEEE 1394-1995 Serial Bus Specification”

[Jenn98] Jennings, Roger (1998) “Fire on the Wire: The IEEE 1394 High Performance Serial Bus” http://www.oakmusic.com/parkplace/video/DVPapers/FireWire.htm

[Lamb02] Lambert, Denis “1394 Digital Audio Technology White Paper” SoftAcoustik White Paper written for SoftAcoustik.com http://www.softacoustik.com/pdf/white_paper.pdf

[Moor01] Moore, Daniel (2001) “Impact-IEEE 1394.Overview” Skipstone, Inc. written for VXM.com
http://www.vxm.com/21R.49.html

[Phil02] (2002) Philips Semiconductors Technical White Paper: "Integration of 5C with a 1394 Audio/Video Link chip" Chip Center – QuestLink written for ChipCenter.com
http://www.chipcenter.com/networking/appnote005.html

[Ples01] Plessel, Markus (2001) “FireWire® Industry Standard IEEE 1394™ Wins Emmy Award” written for the IEEE
http://standards.ieee.org/announcements/afwemmy.html

[Pohl95] Pohlmann, Ken C (1995) “Principles of Digital Audio” 4th edition McGraw-Hill, Inc.

[Roba02] Robair, Gino (2002) “Field Of Dreams” Electronic Musician Vol.18, No.11 October 2002

[Ross90] Rossing, Thomas D. (1990) "The Science of Sound" 2nd edition, Addison Wesley

[Scha01] Schaefer, Paul (2001) “IEEE 1394 Audio Applications with PDI1394 L41/L40” written for Phillips Semiconductors, Phillips Electronics North America Corporation
www.semiconductors.philips.com/acrobat/applicationnotes/AN2456_1.pdf

[Shep02] Shelepov, Roman (2002) “USB 2.0 vs. FireWire” written for Digit-Life.com
http://www.digit-life.com/articles/usb20vsfirewire/

[Sipp85] Sippl, Charles J. (1985) “Macmillan Dictionary of Data Communications, Second Edition” 2nd edition, The Macmillan Press Ltd

[Woot02] Wooten, David (2002) “IEEE Approves Amendment to IEEE 1394™ Standard for High-Speed Serial Buses Allowing Gigabit Signaling” written for the IEEE
http://standards.ieee.org/announcements/1394bapp.html

[Yama02] (2002) Yamaha® Pro Audio Products “Simple, Powerful Networking For Music Systems” written for Yamaha.com
http://www.yamaha.com/proaudio/products/system_mlan.htm