ISOCHRONOUS AUDIO NETWORK SOFTWARE INTERFACE

Inventor: Paul Benjamin Buckwalter, Goshen, IN (US)

Correspondence Address:
INDIANAPOLIS OFFICE 27879
BRINKS HOFER GILSON & LIONE
ONE INDIANA SQUARE, SUITE 1600
INDIANAPOLIS, IN 46204-2033 (US)

Appl. No.: 10/704,164
Filed: Nov. 7, 2003

Publication Classification

Int. Cl. H04L 12/28
U.S. Cl. 370/395.52

ABSTRACT

A computer system includes a network interface that transmits and/or receives isochronous audio packets and data packets. The computer system may have isochronous audio software to extract audio data from the isochronous audio packets, or isochronous audio software to format audio data into isochronous audio packets. The software may extract data from the data packets.
RECEIVE PACKET

402

TRANSMISSION IN PROGRESS?

NO

YES

404

SYNC PACKET?

NO

YES

406

BUFFER READY?

NO

YES

408

PREPARE PACKET; INSERT PACKET HEADERS

410

ADD BUNDLE NUMBER TO PACKET

412

ADD AUDIO TO PACKET

414

TRANSMIT PACKET

END

Fig. 4
Fig. 5
ISOCHRONOUS AUDIO NETWORK SOFTWARE INTERFACE

BACKGROUND OF THE INVENTION

[0001] 1. Technical Field

[0002] This invention relates to an interface, and more particularly, to a software interface that communicates uncompressed isochronous audio.

[0003] 2. Related Art

[0004] The transmission and reception of digital audio signals are becoming more common. Digital audio signals may be transmitted in a compressed format, or in an uncompressed format. While signal compression reduces the bandwidth needed to store and transmit digital audio signals, signal compression can cause perceptible delays and signal loss in some audio systems.

[0005] Some audio systems communicate uncompressed data across real-time, digital audio networks. These networks are considered isochronous when audio devices on the network respond to a clock signal in a manner similar to a synchronous serial communication system. The networks rely on dedicated hardware to interface audio devices.

[0006] One known proprietary system uses proprietary interfaces to control timing and other low-level processing tasks. These proprietary interfaces cannot communicate in a Transmission Control Protocol/Internet Protocol (“TCP/IP”) used by the Internet. A second interface is needed. Because multiple interfaces are required to transmit and receive digital audio signals and provide access to the Internet, proprietary systems may be expensive.

[0007] Therefore, there is a need for a system that can transmit or receive isochronous audio using a standard network interface.

SUMMARY

[0008] This invention provides a system that includes a network interface. The network interface is compatible with known operating systems and may receive isochronous audio packets and other data. The system may include software that extracts audio data from isochronous audio packets.

[0009] The system may also use software that formats audio data into isochronous audio packets. The isochronous audio packets may be formatted to many protocols, including a CobraNet specification.

[0010] Other systems, methods, features and advantages of the invention will be, or will become, apparent to one with skill in the art upon examination of the following figures and detailed description. It is intended that all such additional systems, methods, features and advantages be included within this description, be within the scope of the invention, and be protected by the following claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The invention can be better understood with reference to the following drawings and description. The components in the figures are not necessarily to scale, emphasis instead being placed upon illustrating the principles of the invention. Moreover, in the figures, like referenced numerals designate corresponding parts throughout the different views.

[0012] FIG. 1 is a first isochronous audio controller.

[0013] FIG. 2 is a second isochronous audio controller.

[0014] FIG. 3 is a third isochronous audio controller.

[0015] FIG. 4 is a transmission flow chart for an isochronous audio driver.

[0016] FIG. 5 is a reception flow chart for an isochronous audio driver.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0017] First Isochronous Audio Controller

[0018] A first illustrative isochronous audio controller 100 is shown in FIG. 1. The isochronous audio controller 100 may include a network interface 102, a memory 104, a processor 106, and a user interface 108. The network interface 102 connects the isochronous audio controller 100 to another device and may be a card or a socket that transfers information from one place to another.

[0019] In FIG. 1, the network interface 102 may include a PCMCIA card, an ISA card, a PCI card, a networking circuit or another device. These devices may provide network connectivity in accordance with a standard, such as an Institute of Electrical and Electronics Engineers (“IEEE”) 802 standard, an Ethernet standard, an IEEE 1394 standard, a wireless Ethernet standard, a Fiber Distributed Data Interface (“FDDI”) standard, or a similar standard. Preferably, the network interface 102 operates at a frequency of about 100 Mbps or higher. Likewise, the network 116 may conform to one of these standards.

[0020] The memory 104 is a medium that preserves data for retrieval. It may include a hard disk drive, a compact disc drive, a digital versatile disc drive, a Random Access Memory (“RAM”), a Read Only Memory (“ROM”), a Programmable Read-Only Memory (“PROM”), an Electrically Erasable Programmable Read-Only Memory (“EEPROM”), a flash memory, or any other digital storage device. The processor 106 may include a Central Processing Unit (“CPU”) such as an Intel Pentium® microprocessor, a Sun SPARC® microprocessor, a Motorola microprocessor, an International Business Machines (IBM) processor, or any other processor.

[0021] The user interface 108 may include software or hardware that enables a user to access an application. The user interface 108 may include an absolute pointing device, a relative pointing device, or a physical pointing device. Other cursor control devices may also be used including a keyboard and a touch screen system.

[0022] The memory 104 may include network drivers 110, isochronous audio software 114, and applications 112. The network drivers 110 may provide low-level communications with network interface 102. Additionally, the network drivers 110 may support protocols such as a TCP/IP, IPX/SPX, NetBIOS, ATM, NetBEUI, AppleTalk, and Token-Ring. The network drivers 110 may also provide a software interface for the applications 112. The network drivers 110 may be drivers available from the manufacturer of the network.
interface 102, or from a third party. Illustratively, the isochronous audio controller 100 may include an operating system where the network drivers 100 implement the Open System Interconnection (OSI) model. The OSI model is a networking framework for implementing network protocols in seven layers. Packets or datagrams are passed from one layer to the next.

[0023] The isochronous audio software 114 may conform to an isochronous audio specification to facilitate communication with other isochronous audio devices. The isochronous audio software 114 may process isochronous audio data, play audio data, and/or save audio data in a recordable medium. The isochronous audio software 114 may also receive audio data from a medium, such as a hard disk, or another source, such as a microphone, and create and send audio data to the network drivers 110 for transmission over the network 116.

[0024] Second Isochronous Audio Controller

[0025] A second illustrative isochronous audio controller 200 is shown in FIG. 2. The isochronous audio controller 200 may include a network interface 202, a memory 204, a processor 206, and a user interface 208. The network interface 202, the memory 204, the processor 206, and the user interface 208 may have the same structure and functionality as the network interface 102, the memory 104, the processor 106, and the user interface 108 described above.

[0026] The memory 204 may include layers of network software that provide network support. The memory 204 may include a network interface driver 210, a multiple protocol driver 212, a protocol stack 214, an Application Program Interface ("API") 216, and one or more applications 218. The memory 204 may also include isochronous audio software 220.

[0027] The network interface driver 210 may be a hardware driver provided by the manufacturer of the network interface 202, a hardware driver provided by an operating system vendor, or a hardware driver provided by a third party. The multiple protocol drivers 212 may be a software driver provided by the supplier of an operating system used with the isochronous audio controller 300, or a similar abstraction layer provided by another party.

[0028] The protocol stack 214 comprises a set of protocols that work together on different levels to enable communication on a network. The protocol stack 214 may be a protocol suite available from the simulator of an operating system, or may be a protocol stack provided by a third party. Similarly, the API 216 may be an API available from the supplier of the operating system, or may be an API provided by a third party. (The isochronous audio software 220 may include its own protocol stack and API.) The applications 218 may include many other user interfaces and applications, such as a browser, news client, e-mail client, file server, or other servers or clients.

[0029] The isochronous audio software 212 may process isochronous audio data, play audio data, and/or save audio data in a recordable medium. The isochronous audio software 212 may also receive audio data from a medium, such as a compact disc, or another source, such as a tape deck or a radio receiver, and create and send audio data to the multiple protocol drivers 204 for transmission over the network 116.

[0030] Third Isochronous Audio Controller

[0031] A third illustrative isochronous audio controller 300 is shown in FIG. 3. The isochronous audio controller 300 may include a network interface 302, a memory 304, a processor 306, and a user interface 308.

[0032] The memory 304 may include layers of network software that provide network support for the applications 318. The memory 304 may include a network interface driver 310, a Network Device Interface Specification (NDIS) driver 312, a TCP/IP stack 314, a socket API 316, and one or more applications 318. The isochronous audio controller 300 may also include an isochronous audio driver 320 and an isochronous audio application 322. Alternatively, the isochronous audio driver 320 and an isochronous audio application 322 may be integrated into a single isochronous audio software package 324.

[0033] The network interface driver 310 may be a hardware driver provided by an operating system vendor, a hardware driver provided by the manufacturer of the network interface 302, or a hardware driver provided by a third party. The NDIS driver 312 may be a driver provided by the supplier of the operating system used with the isochronous audio controller 300, or a similar network layer provided by another party.

[0034] The network interface driver 310 and NDIS driver 312 may be drivers available from the manufacturer of the network interface 302, from a supplier of an operating system used in the isochronous audio controller 300, or from a third party. The TCP/IP stack 314 may be a protocol stack available from the supplier of the operating system, or may be a TCP/IP stack included with a firewall application or may be a stand-alone protocol stack. Similarly, the socket API 316 may be an API available from the supplier of the operating system, or may be an API provided by a third party. In addition to or in place of the TCP/IP stack 314 and the socket API 316, protocol stacks and APIs for other protocols such as IPX/SPX, NetBIOS, NetBEUI, AppleTalk, ATM and/or Token-Ring may also be used. For simplicity, the TCP/IP stack 314 and the socket API 316 are discussed. In FIG. 3, the isochronous audio driver 320 includes its own protocol stack and API, and does not rely upon another protocol stack.

[0035] The applications 318 may include many other user interfaces and applications, such as a browser, content providing clients, news clients, e-mail clients, file servers, or other servers or clients. The isochronous audio driver 320 and an isochronous audio application 322, or, alternatively, the isochronous audio software package 324, may process isochronous audio data, play audio data, save audio data in a recordable medium, receive audio data from a medium, and/or create and send audio data to the NDIS driver 312 for transmission over the network 116.

[0036] Standard TCP/IP Processing Example

[0037] As shown in FIG. 3, the applications 318 do not communicate directly with the network interface 302. Rather, the isochronous audio controller 300 includes a high-level network interface through the socket API 316. The applications 318 send and receive network requests and responses to and from the socket API 316, which communicates with the next lower layer.
[0038] When the socket API 316 receives a request from one of the applications 318, the socket API 316 formats the request and forwards the request to the TCP/IP stack 314. The TCP/IP stack 314 reformats the request as one or more data packets. The term “data packet” refers to a variable or fixed length transmission unit that includes binary digits representing information other than isochronous audio data. The units of information may contain both data and a header including an identification number, source and destination addresses (including multicast addresses), and sometimes error control data. Data packets may include IP packets, UDP packets, Ethernet packets, and IPX packets. The term “isochronous audio packet” describes packets of information that contain isochronous audio data. The data may be formatted to be transmitted over a network, and may contain a header having an identification number, source and destination addresses (including multicast addresses), and sometimes error control data.

[0039] The TCP/IP stack 314 forwards data packets to the NDIS driver 312. The NDIS driver 312 is a software layer that allows a plurality of protocol stacks to share one or more network interfaces 302. The NDIS driver 312 forwards data packets to the network interface driver 310. In this example, the network interface driver 310 is the lowest layer of the software. The network interface driver 310 sends data packets directly to the network interface 302.

[0040] When a response is received, the data packets travel in a reverse order of a request. That is, the data packets are first received by the network interface 302, and are processed by the network interface driver 310, the NDIS driver 312, the TCP/IP stack 314, and the socket API 316.

[0041] Isochronous Audio Processing Example

[0042] In FIG. 3, the illustrative isochronous audio controller 300 includes the isochronous audio application 322 and an isochronous audio driver 320. Like the TCP/IP stack 314 and the Socket API 316, the isochronous audio driver 320 provides an interface between the NDIS driver 312 and the isochronous audio application 322. The isochronous audio driver 320 may include, create, or access a transmit buffer 326 and a receive buffer 328 that temporarily store audio data.

[0043] In FIGS. 3 and 4, when an isochronous audio data packet is received from the NDIS driver 312, the isochronous audio driver 320 determines whether the isochronous audio application 322 is transmitting audio data at 402. This determination may assess if the isochronous audio controller 300 has been authorized to transmit isochronous audio packets over the network 116. If the isochronous audio application 322 is not transmitting data, the isochronous audio driver 320 stops and waits for the next packet. At 404, the isochronous audio driver 320 determines whether the received packet is a synchronization packet. An authorized transmitter may only transmit upon reception of a synchronization packet. If the received packet is not a synchronization packet, then the isochronous audio driver 320 stops processing and waits for the next packet.

[0044] If the received packet is a synchronization packet, then at 406 the isochronous audio driver 320 determines whether the audio data in the transmit buffer 326 should be sent. It may be necessary to buffer audio data prior to a transmission so that there is enough audio data to fill an isochronous audio packet when a synchronization packet is received. If the audio data in the transmit buffer 326 is not ready to be sent, then the isochronous audio driver 320 stops until it receives the next packet.

[0045] If audio data within the transmit buffer 326 should be sent, then at 408 the isochronous audio driver 320 prepares an isochronous audio packet and inserts the appropriate headers. The headers may include information such as the resolution of the samples, the frequency of the samples, and how many channels (slots) of audio data are included in the isochronous audio packet. Isochronous audio data may be pulse-code modulated (PCM), although other modulation schemes may be used. PCM resolutions may be about 16 bits, 20 bits, or 24 bits in length. Some sampling frequencies that may be used include 32 KHz, 44 KHz, 48 KHz, and 96 KHz among other acceptable sampling frequencies. In this example, the maximum number of channels per packet is eight.

[0046] At 410, the isochronous audio driver 320 adds a bundle number to the packet. Bundle numbers facilitate communication of multiple channels of audio data. At 412, the isochronous audio driver 320 adds audio data from the transmit buffer 326 to the isochronous audio packet, and transmits the isochronous audio packet at 416. When the isochronous audio packet is received by the NDIS driver 312, it is conveyed to the network 116 through the network interface driver 310 and the network interface 302.

[0047] In FIGS. 4 and 5, when an isochronous audio data packet is received, the isochronous audio driver 320 determines whether the isochronous audio application 322 is currently receiving audio data at 502. If it is not receiving audio data, then the isochronous audio driver 320 waits for the next packet. Otherwise, at 504 the isochronous audio driver 320 determines whether the received packet is an isochronous audio packet. If it is not an isochronous audio packet, the isochronous audio driver 320 stops until it receives the next packet. If the received packet is an isochronous audio packet, then at 506 the isochronous audio driver 320 determines whether the packet’s bundle number is a bundle number that the isochronous audio application 322 is currently processing. If the packet’s bundle number is not a bundle number that the isochronous audio application 322 is currently processing, the isochronous audio driver 320 stops until it receives the next packet.

[0048] When the packet has a bundle number that is currently being processed, at 508 the isochronous audio driver 320 extracts audio data from the packet and separates the audio data into slots or channels. At 510 the isochronous audio driver 320 stores the audio data in the receive buffer 328. At 512, the isochronous audio driver 320 determines whether the audio data in the receive buffer 328 should be sent to the isochronous audio application 322. If it should not be sent to the isochronous audio application 322, then isochronous audio driver 320 stops until it receives the next packet. If the audio data in the receive buffer 328 should be sent to the isochronous audio application 322, then at 514 the isochronous audio driver 320 transmits the audio data to the isochronous audio application 322. The isochronous audio driver 320 may run as a system process, so that it has a higher priority than some applications. The isochronous audio driver 320 may be very “lean,” as shown in FIGS. 3-5.

[0049] The direct communication between the isochronous audio driver 320 and the isochronous audio application
322 facilitates high-speed operation, so that the isochronous audio driver 320 is able to process (transmit and receive) 750 isochronous audio packets per second.

[0050] While various embodiments of the invention have been described, it will be apparent to those of ordinary skill in the art that many more embodiments and implementations are possible within the scope of the invention. Accordingly, the invention is not to be restricted except in light of the attached claims and their equivalents.

What is claimed is:

1. A computer system, comprising:
   a network interface operable to receive isochronous audio packets and data packets from a network; and
   isochronous audio software that is executable to extract audio data from the isochronous audio packets received from a network.

2. The computer system of claim 1 further comprising network software executable to extract data from the data packets received from the network.

3. The computer system of claim 1 where the isochronous audio software comprises an isochronous audio driver.

4. The computer system of claim 1 where the isochronous audio software comprises an isochronous audio application.

5. The computer system of claim 1 where the data packets are formatted to a transmission control protocol/Internet protocol.

6. The computer system of claim 1 where the network is an Ethernet network and the network interface is an Ethernet network interface.

7. The computer system of claim 1 where the network interface conforms to an IEEE 802.3 standard.

8. The computer system of claim 1 where the network interface is adapted to transmit the isochronous audio packets to the network.

9. The computer system of claim 8 where the isochronous audio software is executable to format audio data into the isochronous audio packets.

10. The computer system of claim 1 where the isochronous audio packets are formatted in accordance with a CobraNet specification.

11. A computer system, comprising:
   a network interface adapted to transmit isochronous audio packets to a network and configured to receive data packets from a network; and
   isochronous audio software executable to format audio data into isochronous audio packets.

12. The computer system of claim 11 where the isochronous audio packets are formatted in accordance with a CobraNet specification.

13. The computer system of claim 11 comprising network software that is executable to extract data from the data packets received from the network.

14. The computer system of claim 11 where the isochronous audio software comprises an isochronous audio driver.

15. The computer system of claim 11 where the isochronous audio software comprises an isochronous audio application.

16. The computer system of claim 11 where the data packets are formatted to a transmission control protocol/Internet protocol.

17. The computer system of claim 11 where the network is an Ethernet network and the network interface is an Ethernet network interface.

18. A computer system, comprising:
   a network interface;
   an operating system that is executable to communicate with a network via the network interface according to a plurality of network protocols; and
   isochronous audio software executable to communicate isochronous audio packets with the network via the network interface.

19. The computer system of claim 18 where one protocol of the plurality comprises a transmission control protocol/Internet protocol.

20. The computer system of claim 18 where the isochronous audio packets are formatted to a CobraNet specification.

21. A method of communicating isochronous audio packets over a network,
   comprising:
   receiving a packet with a network interface;
   processing the packet with a network interface driver;
   processing the packet with an NDIS driver; and
   processing the packet with an isochronous audio driver.

22. The method of claim 21 where processing the packet with an NDIS driver includes determining whether the packet is an isochronous audio packet.

23. The method of claim 21 where processing the packet with an isochronous audio driver includes extracting audio data from the packet only if the packet is an isochronous audio packet.

24. The method of claim 22 where processing the packet with an isochronous audio driver comprises processing the audio data with an isochronous audio application.

25. An isochronous audio driver, comprising:
   code executable to receive an isochronous audio packet from an operating system;
   code executable to extract audio data from the isochronous audio packet; and
   code executable to transmit the audio data to an isochronous audio application.

26. The isochronous audio driver of claim 25 further comprising:
   code executable to receive audio data from an isochronous audio application;
   code executable to format the audio data into an isochronous audio packet; and
   code executable to send an isochronous audio packet to an operating system.

* * * * *