(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 January 2005 (06.01.2005)

PCT

(10) International Publication Number
WO 2005/001702 A2

(51) International Patent Classification[7]: G06F 15/16, 15/173

(21) International Application Number: PCT/US2004/018659

(22) International Filing Date: 10 June 2004 (10.06.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/478,336 13 June 2003 (13.06.2003) US
10/746,283 23 December 2003 (23.12.2003) US

(71) Applicant *(for all designated States except US)*: **APPLE COMPUTER, INC.** [US/US]; 1 Infinite Loop, MS 3-PAT, Cupertino, CA 95014 (US).

(72) Inventors; and

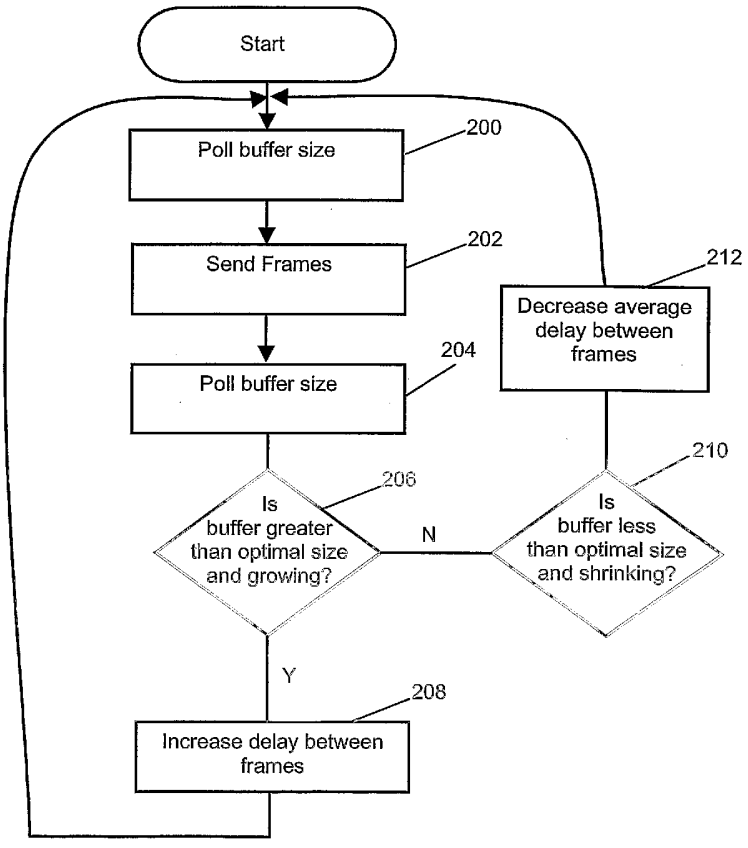
(75) Inventors/Applicants *(for US only)*: **AGNOLI, Giovanni, M.** [US/US]; 1740 Lexington Avenue, San Mateo, CA 94402 (US). **YANOWITZ, Andrew** [US/US]; 373 Bahr Drive, Ben Lomond, CA 95005 (US). **ABT, John, O.** [US/US]; 190 Ventana Sierra Drive, Grass Valley, CA 95945 (US). **BOWMAN, Samuel, R.** [US/US]; 1040 Golden Oak Court, Colfax, CA 95713 (US). **DELWICHE, James, A.** [US/US]; 15087 Lorie Drive, Grass Valley, CA 95949 (US). **DILLON, Jeffrey, C.** [US/US]; 1712 Stone Way, Auburn, CA 95603 (US).

(74) Agent: **CROSBY, John, W.**; Sierra Patent Group, LTD, P.O. Box 6149, Stateline, NV 89449 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE,

[Continued on next page]

(54) Title: SYNCHRONIZED TRANSMISSION OF AUDIO AND VIDEO DATA FROM A COMPUTER TO A CLIENT VIA AN INTERFACE



(57) Abstract: A method for controlling data transmission between a computer and a video client via an interface, the method comprising: the computer polling the interface a first time to determine the site of the buffer on the interface; receiving a first buffer size value from the interface; sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame; the computer polling the interface a second time to determine buffer site after the frames are sent to the interface; receiving a second buffer size value from the interface; and modifying the amount of time between the transmission of frames.

WO 2005/001702 A2

KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYNCHRONIZED TRANSMISSION OF AUDIO AND VIDEO DATA FROM A COMPUTER TO A CLIENT VIA AN INTERFACE

FIELD OF THE INVENTION

The present invention relates broadly to devices in communication over a network. Specifically, the present invention relates to data flow management between devices transmitting and receiving data at different transmission rates. More specifically, the present invention relates to controlling data flow through a buffer by monitoring the buffer and adjusting data transmission based on buffer conditions.

BACKGROUND OF THE INVENTION

A "bus" is a collection of signals interconnecting two or more electrical devices that permits one device to transmit information to one or more other devices. There are many different types of busses used in computers and computer-related products. Examples include the Peripheral Component Interconnect ("PCI") bus, the Industry Standard Architecture ("ISA") bus and Universal Serial Bus ("USB"), to name a few. The operation of a bus is usually defined by a standard which specifies various concerns such as the electrical characteristics of the bus, how data is to be transmitted over the bus, how requests for data are acknowledged, and the like. Using a bus to perform an activity, such as transmitting data, requesting data, etc., is generally called running a "cycle." Standardizing a bus protocol helps to ensure effective communication between devices connected to the bus, even if such devices are made by different manufacturers. Any company wishing to make and sell a device to be used on a particular bus, provides that device with an interface unique to the bus to which the device will connect. Designing a device to particular bus standard ensures that device will be able to communicate properly with all other devices connected to the same bus, even if such other devices are made by different manufacturers. Thus, for example, an internal fax/modem (i.e., internal to a personal computer) designed for operation on a PCI bus will be able to transmit and receive data to and from other devices on the PCI bus, even if each device on the PCI bus is made by a different manufacturer.

Currently, there is a market push to incorporate various types of consumer electronic equipment with a bus interface that permits such equipment to be connected to other equipment with a corresponding bus interface. For example, digital cameras, digital video recorders, digital video disks ("DVDs"), printers are becoming available with an IEEE 1394 bus interface. The IEEE ("Institute of Electrical and Electronics Engineers") 1394 bus, for example, permits a digital camera to be connected to a printer or computer so that an image acquired by the camera can be printed on the printer or stored electronically in the computer. Further, digital televisions can be coupled to a computer or computer network via an IEEE 1394 bus.

However, many devices exist without any sort of IEEE 1394 interface. This presents a problem as such devices are unable to be to be connected with other devices as described above. There is a heartfelt need to overcome this problem to provide connectivity to devices that otherwise cannot be connected to an IEEE 1394 bus.

SUMMARY OF THE INVENTION

The present invention controls the transmission of data from a computer to a video client via an interface device that buffers the data frames sent and communicates to the computer and the video client using different protocols. In an embodiment, the present invention provides a method of performing data transmission flow control by polling the interface a first time to determine the size of the buffer on the interface; receiving a first buffer size value from the interface; sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame; polling the interface a second time to determine buffer size after the frames are sent to the interface; and receiving a second buffer size value from the interface. If the second buffer size value is larger than the optimal size, and larger than the first buffer size value, then the delay period between transmission of frames from the computer to the interface is increased.

In another embodiment, the present invention provides a method of performing data transmission flow control, by polling the interface a first time to determine the size of the buffer on the interface; receiving a first buffer size value from the interface; sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame; polling the interface a second time to determine buffer size after the frames are sent to the interface; and receiving a second buffer size value from the interface.

If the second buffer size value is smaller than optimal size, and smaller than the first buffer size value, then the delay period between transmission of frames from the computer to the interface is decreased.

Many other features and advantages of the present invention will be realized by reading the following detailed description, when considered in conjunction with the accompanying drawings, in which:

BRIEF DESCRIPTION OF THE DRAWINGS

FIG.1 illustrates in block diagram form major components used in connection with embodiments of the present invention;

FIG. 2 illustrates the format of a frame in accordance with embodiments of the present invention;

FIGS. 3A and 3B illustrate the format of the first data packet and following data packet, respectively;

FIGS. 4A and 4B illustrate the organization of video data within data packets in accordance with the embodiments of the present invention;

FIGS. 5A and 5B illustrate the organization of audio data within data packets in accordance with the embodiments of the present invention;

FIGS. 6 and 7 illustrate elements of a header included in the frame in accordance with embodiments of the present invention;

FIG. 8 illustrates a collection of packets that combine to form a frame in accordance with embodiments of the present invention;

FIGS. 9A-9D illustrates an alternative embodiment of the present invention in which variations of SDTI frames are used in accordance with embodiments of the present invention;

FIG. 9E illustrates an alternative embodiment in which the transmitter divides the SDTI stream across multiple channels;

FIG. 10 illustrates in flow chart form acts performed to provide external clocking between a computer and a hardware interface in accordance with embodiments of the present invention;

FIG. 11 illustrates the register memory map for the interface device in accordance with embodiments of the present invention;

FIG. 12 illustrates organization of A/V global registers contained within the interface

of the present invention;

FIG. 13 illustrates organization of global status registers contained within the interface device of the present invention;

FIG. 14 illustrates the isochronous control register contained in the interface device of the present invention;

FIG. 15 illustrates the organization of the flow control register contained in the interface device of the present invention; and

FIG. 16 illustrates the organization of the isochronous channel register contained in the interface device of the present invention.

DETAILED DESCRIPTION

Directing attention to FIG. 1, there is shown in block diagram form components connected to transmit audio and video data between a computer 100 and client 102, connected by bus 104 to interface 106. Computer 100 in the preferred embodiment is a computing device capable of processing and video and audio data and displaying it in a recognizable form to a user. Such devices include desktop, laptop, and palmtop computers. Client 102 as referred to herein is a video consumer or video producer, and includes such devices as digital cameras, and video storage devices, such as linear and random access devices. Bus 104, as referred to herein, includes a physical connection between computer 100 and interface 106, as well as the serial protocol adhered to by devices communicating over bus 104. In the preferred embodiment, bus 104 utilizes the IEEE 1394 serial bus protocol known as Firewire. Interface 106 accepts from client 102 both analog and digital inputs, and converts the input to scanned lines that can be used by an audio/video player executed on computer 100. In an alternative embodiment, interface 106 accepts from client 102 a digital compressed/uncompressed signal and transmits the entire signal or subsets of that signal. In an embodiment, interface 106 divides the input into frames 108 them over bus 104 to computer 100.

The format of frame 108 is illustrated in FIG. 2. Frame 108 includes a frame header 110, video block 112, audio block 114, and optionally an audio header 116. Audio data in audio block 114 is sampled with respect to the video data in video block 112. The audio sample count per frame varies in accordance with the number defined in the ANSI/SMPTE 272M specification, incorporated herein by reference in its entirety. The audio sample count cadence is necessary to divide the integer number of samples per second across the NTSC

frame rate (29.97 fps). Similarly, the size of frame 108 can vary to accommodate various video formats such as PAL or NTSC, and 8 or 10 bit video data, and audio formats such as 48Khz and 96Khz 16 and 24 bit etc. Similarly, the frame size of compressed data can vary to accommodate the compressed format. In an embodiment, video block 112 and audio block or compressed block are of a predetermined size, to make parsing frame 108 simple and requiring little processing overhead by applications such as direct memory access programs. In the event that not all of video block 112 or audio block 114 is not completely full of data, the remaining portions of blocks 112, 114 can be filled with zeros. In one embodiment, data contained in video block 112 and audio block 114 is not compressed, further reducing processing overhead on interface 106, as well as processing overhead required by decompression programs running on computer 100.

Interface 106, upon converting the input received from client 102 and converting it to scan lines and organizing it into frames 108, sends a frame at each vertical blanking interval to provide synchronization with computer 100. Computer 100 can derive the vertical blanking interval from the frequency of frames received and synchronize itself with the audio and video data of the incoming frames 108 received from interface 106. In this manner, processing resources are preserved, as there is no need to perform synchronization on each frame as it is received, thus providing higher quality performance of audio and video display on computer 100.

FIGS. 3A and 3B illustrate the format of the first data packet and following data packet, respectively.

FIGS. 4A and 4B illustrate the organization of video data within data packets. FIGS. 5A and 5B illustrate the organization of audio data within data packets.

FIG. 6 illustrates the contents of frame header 110. Included are format flags 130, which indicate how many bits per sample, SMPTE time code 132, incrementing frame counter 134, audio cycle count 136, audio sample count 138, channel count 140, block size byte count 142, audio format flags 144, and video format flags 146. Audio sample count 138 indicates a number of samples, which is in accordance with a cadence. The value in audio cycle count 136 indicates location within the cadence. A cadence of frames form a cycling pattern. In an alternative embodiment, some of the contents of frame header 110 can be moved or copied to optional audio header 116. An alternative view of frame header 110 is shown in FIG. 7, showing byte count, data length, and a frame bit.

As illustrated in FIG. 8, frame 108 is constructed from a plurality of packets 150 of a

predetermined size. Associated with each packet is a 1394 isochronous packet header. Data transmission in accordance with the present invention takes advantage of a synchronization bit to find the beginning of a frame. The first packet in frame 108 is marked with the synchronization bit. This allows the stream of data to be identified by computer 100 as it is received, further reducing processing overhead by allowing computer 100 to synchronize the flow of frames received from interface 106.

In an alternative embodiment of the present invention, frames adhering to the serial digital interface (SDI) standard can be utilized as illustrated in FIGS, 9A through 9E. In these embodiments, bus 104 adheres to the IEEE 1394B serial bus protocol to accommodate data rate restrictions set forth by the SDI standard. As described above, interface 106 forms frames from received input by creating scanned lines, performing deinterlacing, packetizing, and creating fixed-size SDTI frames of audio and video data. Various modifications can be made to SDTI frames, depending on the processing resources available on computer 100, interface 106, client 102, or other device. As described above, the transmission of SDTI frames sent over bus 104 are synchronized to the vertical blanking interval of the accepted signal.

As shown in FIG. 9A, SDTI frame 160 generally has two components: vertical blanking portion 162 and horizontal retrace 164. Alternatively, in another embodiment (FIG. 9B), SDI frame header 166, a header having a synchronization bit and a frame count, is added to SDTI frame 160 for further synchronization and fault detection purposes, such as recovering from data lost in transmission or the occurrence of a bus reset. In this embodiment, a frame count synchronization bit is included in SDTI frame header 166 and SDTI frame header 166 is synchronized with vertical blanking portion 162. For example, in an application where interface 106 is unable to read compressed data, or excessive upgrades to interface 106 would be required, SDTI frame 160 can be transmitted to computer 100, where processing on the SDTI stream is performed by software in a non-realtime manner. Alternatively, as shown in FIG. 9C, SDTI frame 160 can be constructed without horizontal retrace 164 to further reduce processing overhead. An SDTI frame constructed without a horizontal retrace but having header 166, can also be utilized in an embodiment, as shown in FIG. 9D. In yet another embodiment, as shown in FIG. 9E, the SDTI frame can be split between multiple channels and also include SDTI frame header 166. In this embodiment, the transmitter splits the SDTI stream in half, with half of the lines being transmitted across channel A, the other half being transmitted across channel B. An attached header for each

partial frame can be used to assist in re-combining frame data.

In another aspect of the present invention, external clocking can be utilized to synchronize data transmission between computer 100, interface 106 and client 102. In an embodiment, client 102 includes a high-quality reference clock 180 (FIG.1) that can be used to synchronize clock 182 on interface 106 and prevent overflow of buffer 184 on interface 106. In this embodiment, the value of reference clock 180 on client 102 is derived on interface 106 from the frequency at which data is transmitted from computer 102 to interface 106. To perform flow control, cycles are skipped between transmission of frames. A skipped cycle increases the amount of time between transmissions of frames, to slow the data rate of the frame transmission. Directing attention to FIG. 10, at reference numeral 200, computer polls interface 106 to read the size of buffer 184. While for exemplary purposes the buffer is referred to in terms such as "bigger" and "smaller," it is to be understood that in the case of a fixed-size buffer bigger and smaller refer to fullness of the buffer. At reference numeral 202, computer 100 then sends a plurality of frames to interface 106. At reference numeral 204, computer 100 again polls interface 106 to determine the size of buffer 184. If buffer 184 has grown in size from the last poll of its size (decision reference numeral 206), control proceeds to reference numeral 208, where computer 100 increases the delay between frames it is sending to interface 106. In an embodiment, the delay between frames sent is 125 milliseconds. In another embodiment a fractional delay is attained by modulating the delay over a number of frames. For instance if a delay between frames of 2.5 times 1.25microseconds is required, alternating frame delays of 2 and 3 cycles (of 125microseconds) are interspersed. Control then returns to reference numeral 202, where the frames are sent to interface 106 with the additional delay between frames. However, returning to decision reference numeral 206, if buffer 184 has not grown in size since the last polling of its size, control transitions to decision reference numeral 210. At decision reference numeral 210, if buffer 206 has decreased in size, control transitions to reference numeral 212, where the delay between frames sent from computer 100 to interface 106 is decreased. In an embodiment, the amount of this decrease is also 125Ms. Control then transitions to reference numeral 202, where the frames are sent from computer 100 to interface 106 with the reduced delay between frames. Returning to decision reference numeral 210, if the size of buffer 184 has not reduced since the last polling of the size of buffer 184, then no adjustment to the delay between frames is necessary, and control transitions to reference numeral 202.

Interface 106 includes a serial unit 300 for enabling communication across bus 104. Serial unit 300 includes a unit directory 302 as shown in Table 1.

Name	Key	Value
Unit_Spec_ID	0x12	0x000a27
Unit_SW_Version	0x13	0x000022
Unit_Register_Location	0x54	Csr_offset to registers
Unit_Signals_Supported	0x55	Supported RS232 signals

TABLE 1

The Unit_Spec_ID value specifies the organization responsible for the architectural definition of serial unit 300. The Unit_SW_Version value, in combination with Unit_Spec_ID value, specifies the software interface of the unit. The Unit_Register_location value specifies the offset in the target device's initial address space of the serial unit registers. The Unit_Signals_Supported value specifies which RS-232 signals are supported, as shown in the Table 2. If this entry is omitted from the serial unit directory 302, then none of these signals are supported.

Field	Bit	Description
Ready to Send (RTS)	0	Set if RTS/RFR is supported
Clear to Send (CTS)	1	Set if CTS is supported
Data Set ready (DSR)	2	Set if DSR is supported
Data Transmit Ready (DTR)	3	Set if DTR is supported
Ring Indicator (RI)	4	Set if RI supported
Carrier (CAR)	5	Set if CAR/DCD is supported
Reserved	[31..6]	Reserved

TABLE 2

Also included in serial unit 300 is a serial unit register map 304 that references registers contained in serial unit 300. The organization of serial unit register map 304 is shown in Table 3.

Hex Offset	Name	Access	Size(quads)	Value
0x0	Login	W	2	Address of initiator's serial

				registers
0x8	Logout	W	1	Any value
0xc	Reconnect	W	1	Initiator's node ID
0x10	TxFIFO Size	R	1	Size in bytes of Tx FIFO
0x14	RxFIFO Size	R	1	Size in bytes of Rx FIFO
0x18	Status	R	1	CTS/DSR/RI/CAR
0x1c	Control	W	1	DTR/RTS
0x20	Flush TxFIFO	W	1	Any value
0x24	Flush RxFIFO	W	1	Any value
0x28	Send Break	W	1	Any value
0x2c	Set Baud Rate	W	1	Baud rate 300->230400
0x30	Set Char Size	W	1	7 or 8 bit characters
0x34	Set Stop Size	W	1	1, 1.5 or 2 bits
0x38	Set Parity	W	1	None, odd or even parity
0x3c	Set Flow Control	W	1	None, RTS/CTS or Xon/Xoff
0x40	Reserved	-	4	Reserved
0x50	Send Data	W	TxFIFO size	Bytes to transmit

TABLE 3

Serial unit register map 304 references a login register. A device attempting to communicate with serial unit 300, is referred to herein as an initiator. For example, an initiator can be computer 100, or other nodes connected on a network via a high-speed serial bus and in communication with interface 106. The initiator writes the 64 bit address of the base of its serial register map to the login register to log into serial unit 300. If another initiator is already logged in, serial unit 300 returns a conflict error response message. The high 32 bits of the address are written to the Login address, the lower 32 bits to Login+4. The serial unit register map also references a logout register. The initiator writes any value to this register to log out of the serial unit. After every bus reset the initiator must write its

(possibly changed) nodeID to the reconnect register. If the initiator fails to do so within one second after the bus reset it is automatically logged out. The 16-bit nodeID is written to the bottom 16 bits of this register, the top 16 bits should be written as zero. A read of the TxFIFOSize register returns the size in bytes of the serial unit's transmit FIFO. A read of the RxFIFOSize register returns the size in bytes of serial unit 300's receive FIFO. A read of the status register returns the current state of CTS/DSR/RI/CAR (if supported). The status register is organized as shown in Table 4.

Field	Bit	Description
CTS	0	1 if CTS is high, else 0
DSR	1	1 if DSR is high, else 0
RI	2	1 if RI is high, else 0
CAR	3	1 if CAR is high, else 0
Reserved	[31..4]	Always 0

TABLE 4

A write to the control register sets the state of DTR and RTS (if supported). The organization of the control register is shown in Table 5.

Field	Bit	Description
RTS	0	If 1 set RTS high, else set RTS low
DTR	1	If 1 set DTR high, else set DTR low
Reserved	[31..2]	Always 0

TABLE 5

A write of any value to the FlushTxFIFO register causes serial unit 300 to flush its transmit FIFO, discarding any bytes currently in it. A write of any value to the FlushRxFIFO register causes the serial unit to flush its receive FIFO, discarding any bytes currently in it. A write of any value to the send break register causes serial unit 300 to set a break condition on its serial port, after transmitting the current contents of the TxFIFO. A write to the set baud rate register sets serial unit 300's serial port's baud rate. The set baud rate register is organized as shown in Table 6.

Value written	Baud Rate
0	300
1	600

2	1200
3	2400
4	4800
5	9600
6	19200
7	38400
8	57600
9	115200
10	230400

TABLE 6

The set char size register sets the bit size of the characters sent and received. The organization of the set char size register is shown in Table 7. 7 bit characters are padded to 8 bits by adding a pad bit as the most significant bit.

Value written	Character bit size
0	7 bits
1	8 bits

TABLE 7

The set stop size register designates the number of stop bits. The set stop size register is organized as shown in Table 8.

Value written	Stop bits
0	1 bit
1	1.5 bits
2	2 bits

TABLE 8

The set parity register sets the serial port parity. The organization of the set parity register is shown in Table 9.

Value written	Parity
0	No Parity bit
1	Even parity

2	Odd parity

TABLE 9

The set flow control register sets the type of flow control used by the serial port. The organization of the set flow register is shown in Table 10.

Value written	Flow Control
0	None
1	CTS/RTS
2	XOn/Xoff

TABLE 10

The send data register is used when the initiator sends block write requests to this register to write characters into the transmit FIFO. Block writes must not be larger than the transmit FIFO size specified by the TxFIFOSize register. If there isn't enough room in the Tx FIFO for the whole block write, then a conflict error response message is returned and no characters are copied into the FIFO.

Also included in serial unit 300 is an initiator register map having a plurality of registers, organized as shown in Table 11.

Hex Offset	Name	Access	Size (quads)	Value
0x0	Break	W	1	Any value
0x4	Framing Error	W	1	Received character
0x8	Parity Error	W	1	Received character
0xc	RxFIFO overflow	W	1	Any value
0x10	Status change	W	1	CTS/DSR/RI/CAR
0x14	Reserved	-	3	Reserved
0x20	Received Data	W	RxFIFO size	Bytes received

TABLE 11

When serial unit 300 detects a break condition on its serial port, it writes an arbitrary value to this register. When serial unit 300 detects a framing error on its serial port, it writes the received character to the framing register. When serial unit 300 detects a parity error on

its serial port, it writes the received character to the parity error register. When serial unit 300's receive FIFO overflows, serial unit 300 writes an arbitrary value to the RxFIFO overflow register. When serial unit 300 detects a change in state of any of CTS/DSR/RI/CAR it writes to the status change register indicating the new serial port signal state. The organization of the status register is shown in table 12.

Field	Bit	Description
CTS	0	1 if CTS is high, else 0
DSR	1	1 if DSR is high, else 0
RI	2	1 if RI is high, else 0
CAR	3	1 if CAR is high, else 0
Reserved	[31..4]	Always 0

TABLE 12

When serial unit 300 receives characters from its serial port it writes the received characters to the received data register with a block write transaction. It never writes more bytes than the receive FIFO size specified by the RxFIFOSize register. If the initiator cannot receive all the characters sent it responds with a conflict error response message and receives none of the characters sent.

FIG. 11 illustrates the register memory map for the interface device in accordance with embodiments of the present invention. FIG. 12 illustrates organization of A/V global registers contained within the interface of the present invention. FIG. 13 illustrates organization of global status registers contained within the interface device of the present invention. FIG. 14 illustrates the isochronous control register contained in the interface device of the present invention. FIG. 15 illustrates the organization of the flow control register contained in the interface device of the present invention. FIG. 16 illustrates the organization of the isochronous channel register contained in the interface device of the present invention.

In another embodiment of the present invention, a synthesized vertical blanking signal is derived by polling a vertical blanking register on interface 106. The vertical blanking signal invokes code to programs running on computer 100. In an embodiment, timing information may also be provided to programs running on computer 100, either in combination with the invoked code or instead of the invoked code. In an embodiment of the invention, interface 106 contains a register that holds a counter indicating current progress in

the frame, from which the next vertical retrace can be extrapolated or otherwise derived. By deriving boundaries on frame transmission, other data that is within the frame and synchronized to the occurrence of a vertical blanking interval can be located and accessed, such as for sampling operations. Additionally, an embodiment of the present invention derives frame boundaries for locating data that is coincident with the vertical blanking interval but includes no information about the vertical blanking. In an embodiment, the present invention is used to obtain data that is valid for a period after the occurrence of a video blanking interval, such as a time code contained within the frame, can be read, and used in various processing applications. In an embodiment, computer 100 can then schedule an interrupt to fire at this extrapolated time, thus sending out a frame.

What is claimed is:

1. In a system having a computer, a video client, and an interface between the computer and video client that facilitates data transmission between the computer and the video client, the interface having a buffer for storing data frames received from the computer to be sent to the video client, the buffer having a size that varies with the amount of data it stores, the interface having an optimal buffer size, a method of performing data transmission flow control, the method comprising:

the computer polling the interface a first time to determine the size of the buffer on the interface;

receiving a first buffer size value from the interface;

sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

the computer polling the interface a second time to determine buffer size after the frames are sent to the interface;

receiving a second buffer size value from the interface; and

if the second buffer size value is larger than the optimal size, and larger than the first buffer size value, then increasing the delay period between transmission of frames from the computer to the interface.

2. The method of claim 1, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

3. In a system having a computer, a video client, and an interface between the computer and video client that facilitates data transmission between the computer and the video client, the interface having a buffer for storing data frames received from the computer to be sent to the video client, the buffer having a size that varies with the amount of data it stores, the interface having an optimal buffer size, a method of performing data transmission flow control, the method comprising:

the computer polling the interface a first time to determine the size of the buffer on the interface;

receiving a first buffer size value from the interface;

sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

the computer polling the interface a second time to determine buffer size after the frames are sent to the interface;

receiving a second buffer size value from the interface; and

if the second buffer size value is smaller than optimal size, and smaller than the first buffer size value, decreasing the delay period between transmission of frames from the computer to the interface.

4. The method of claim 3, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

5. A computer program product containing instructions which, when executed by a computer, performs data transmission flow control by performing the acts of:

polling for a first time an interface in communication with the computer, the interface having a buffer for storing data frames received from the computer, the frames to be sent to the video client, the buffer having a size that varies with the amount of data contained in the buffer, the buffer having an optimal size;

receiving a first buffer size value from the interface;

sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

polling the interface a second time to determine buffer size after the frames are sent to the interface;

receiving a second buffer size value from the interface; and

if the second buffer size value is larger than the optimal size, and larger than the first buffer size value, then increasing the delay period between transmission of frames from the computer to the interface.

6. The computer program product of claim 5, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

7. A computer program product containing instructions which, when executed by a computer, performs data transmission flow control by performing the acts of:

polling for a first time an interface in communication with the computer, the interface having a buffer for storing data frames received from the computer, the frames to be sent to the video client, the buffer having a size that varies with the amount of data contained in the buffer, the buffer having an optimal size;

receiving a first buffer size value from the interface;

sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

polling the interface a second time to determine buffer size after the frames are sent to the interface;

receiving a second buffer size value from the interface; and

if the second buffer size value is smaller than the optimal size, and smaller than the first buffer size value, then decreasing the delay period between transmission of frames from the computer to the interface.

8. The computer program product of claim 7, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

9. In a system having a computer, a video client, and an interface between the computer and video client that facilitates data transmission between the computer and the video client, the interface having a buffer for storing data frames received from the computer to be sent to the video client, the buffer having a size that varies with the amount of data it stores, the interface having an optimal buffer size, apparatus for performing data transmission flow control, the method comprising:

means for polling the interface a first time to determine the size of the buffer on the interface;

means for receiving a first buffer size value from the interface;

means for sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

means for polling the interface a second time to determine buffer size after the frames are sent to the interface;

means for receiving a second buffer size value from the interface; and

means for, if the second buffer size value is larger than the optimal size, and larger than the first buffer size value, increasing the delay period between transmission of frames from the computer to the interface.

10. The method of claim 9, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

11. In a system having a computer, a video client, and an interface between the computer and video client that facilitates data transmission between the computer and the video client, the interface having a buffer for storing data frames received from the computer to be sent to the video client, the buffer having a size that varies with the amount of data it

stores, the interface having an optimal buffer size, apparatus for performing data transmission flow control, the apparatus comprising:

means for polling the interface a first time to determine the size of the buffer on the interface;

means for receiving a first buffer size value from the interface;

means for sending a plurality of frames of video and audio data to the buffer on the interface such that a delay period exists between the sending of each frame;

means for polling the interface a second time to determine buffer size after the frames are sent to the interface;

means for receiving a second buffer size value from the interface; and

means for, if the second buffer size value is smaller than optimal size, and smaller than the first buffer size value, decreasing the delay period between transmission of frames from the computer to the interface.

12. The apparatus of claim 11, wherein buffer size refers to fullness of meaningful data in a fixed-sized buffer.

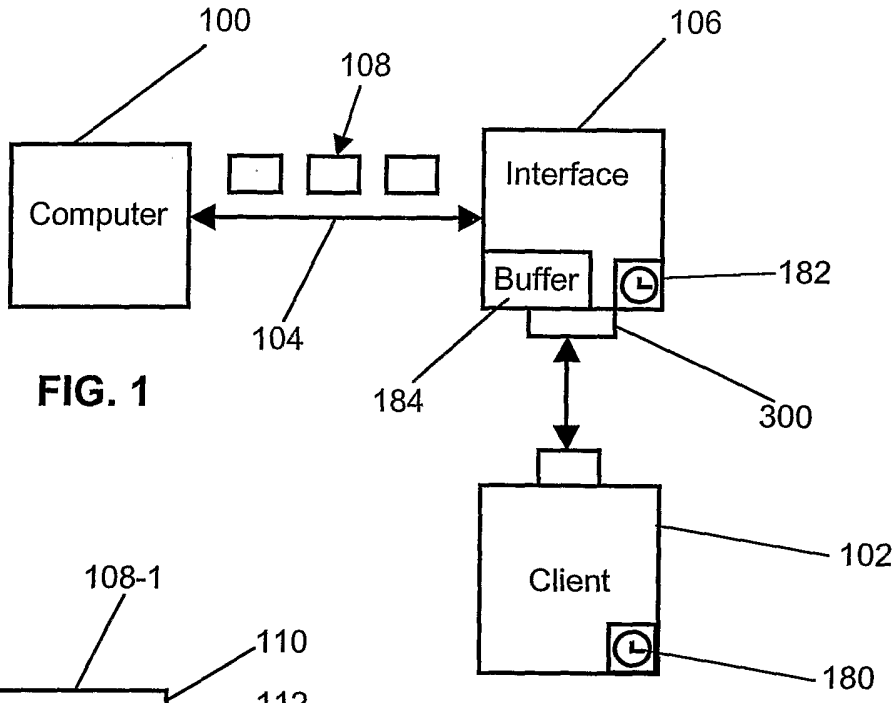

100
108
106
Computer
Interface
Buffer
182
104
184
300
Client
102
180

FIG. 1

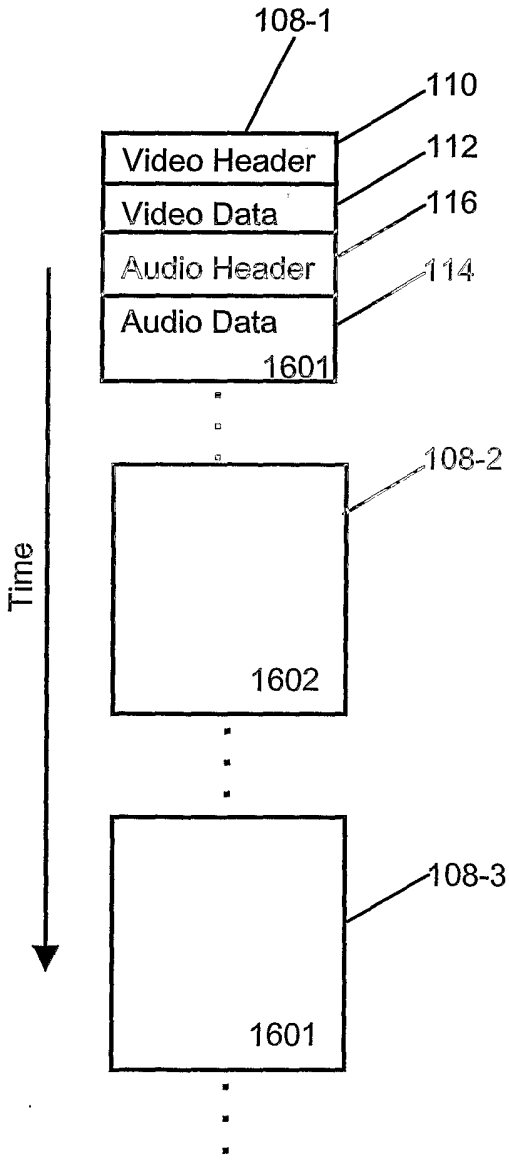

108-1
110
Video Header
112
Video Data
116
Audio Header
114
Audio Data
1601
108-2
1602
108-3
1601
Time

FIG. 2

FIG. 3A

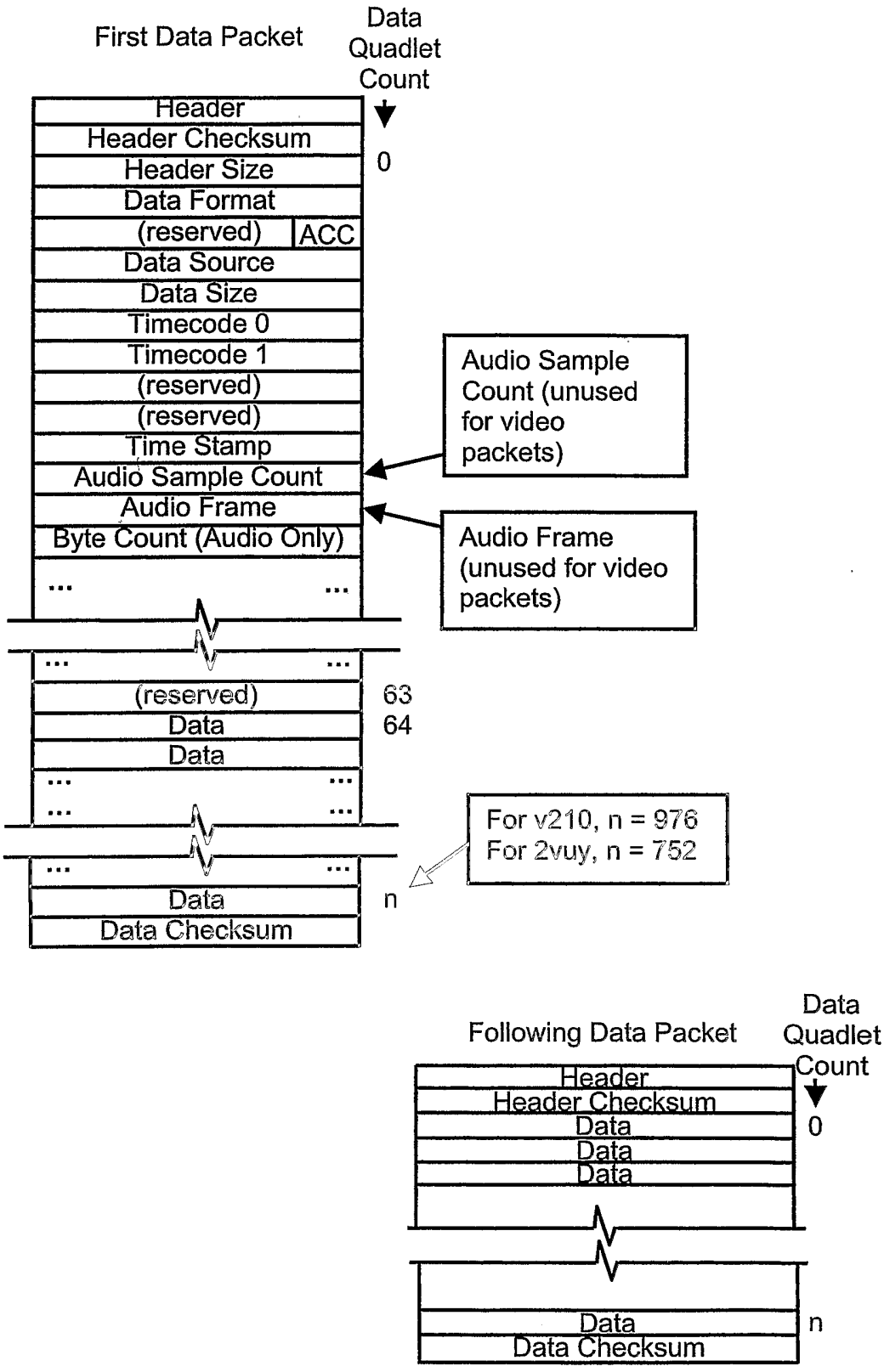

First Data Packet
Data Quadlet Count
Header
Header Checksum
Header Size
Data Format
(reserved)
ACC
Data Source
Data Size
Timecode 0
Timecode 1
(reserved)
(reserved)
Time Stamp
Audio Sample Count
Audio Frame
Byte Count (Audio Only)
0
Audio Sample Count (unused for video packets)
Audio Frame (unused for video packets)
(reserved)
Data
Data
63
64
For v210, n = 976
For 2vuy, n = 752
Data
Data Checksum
n
Following Data Packet
Data Quadlet Count
Header
Header Checksum
Data
Data
Data
0
Data
Data Checksum
n

FIG. 3B

Video Data, v210 (10 Bit $Y/C_b/C_r$)

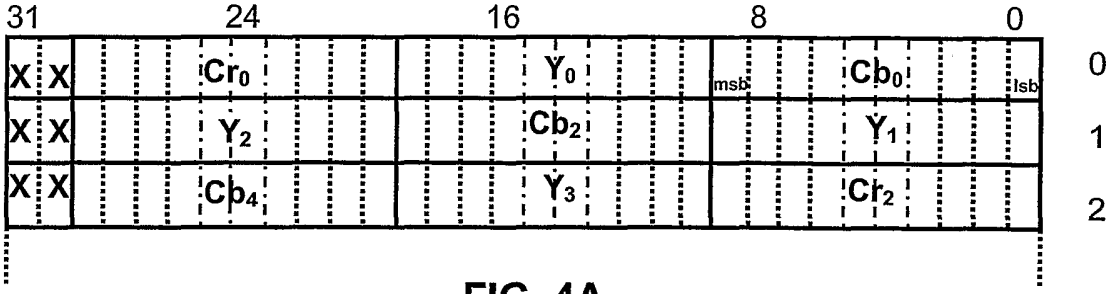


FIG. 4A

Video Data, 2vuy (8 Bit $Y/C_b/C_r$)

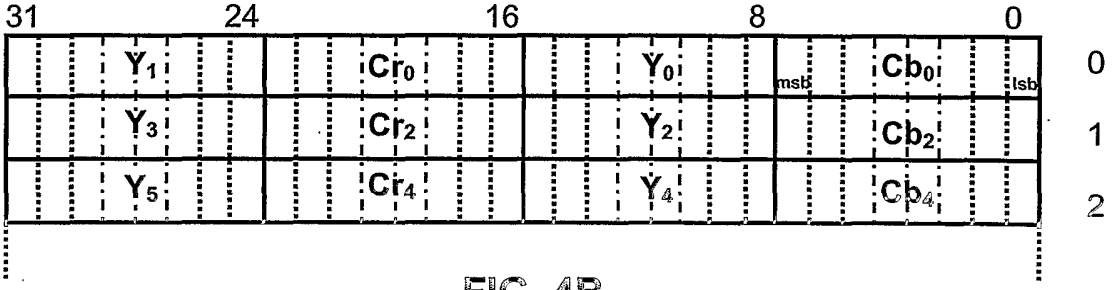


FIG. 4B

Audio Data (24 Bit, Packed)

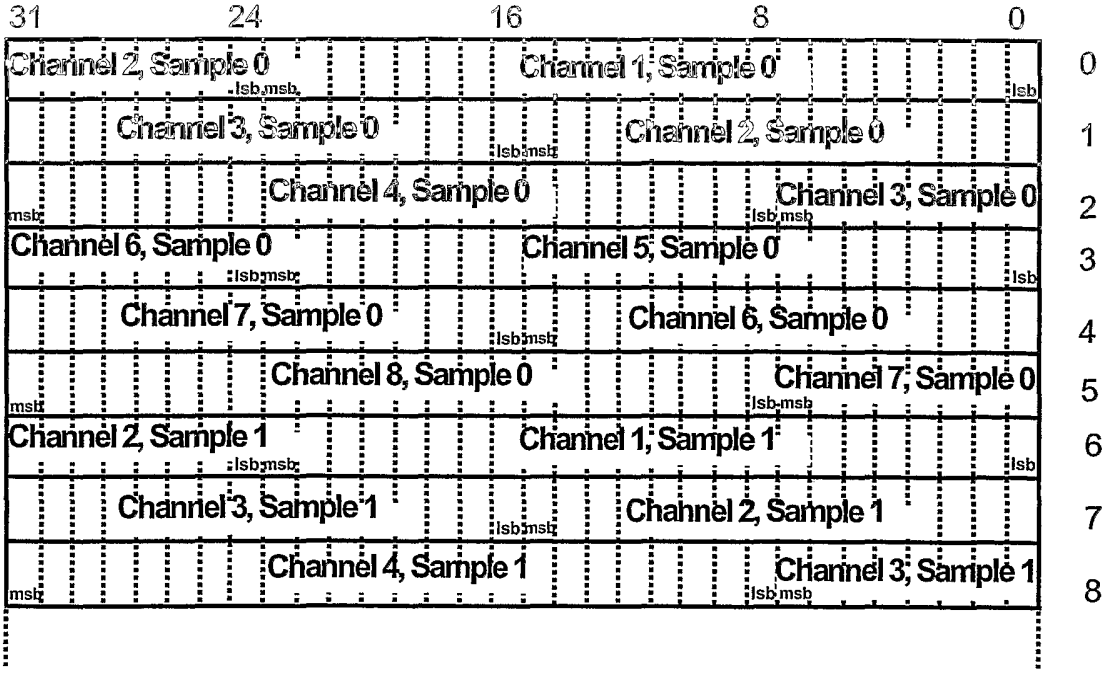


FIG. 5A

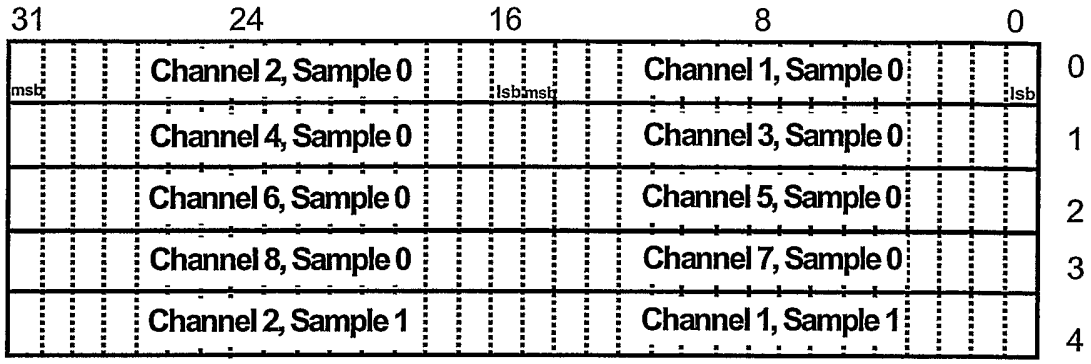

Audio Data (16 Bit)
31
24
16
8
0
msb
Channel 2, Sample 0
lsb
msb
Channel 1, Sample 0
lsb
0
Channel 4, Sample 0
Channel 3, Sample 0
1
Channel 6, Sample 0
Channel 5, Sample 0
2
Channel 8, Sample 0
Channel 7, Sample 0
3
Channel 2, Sample 1
Channel 1, Sample 1
4

FIG. 5B

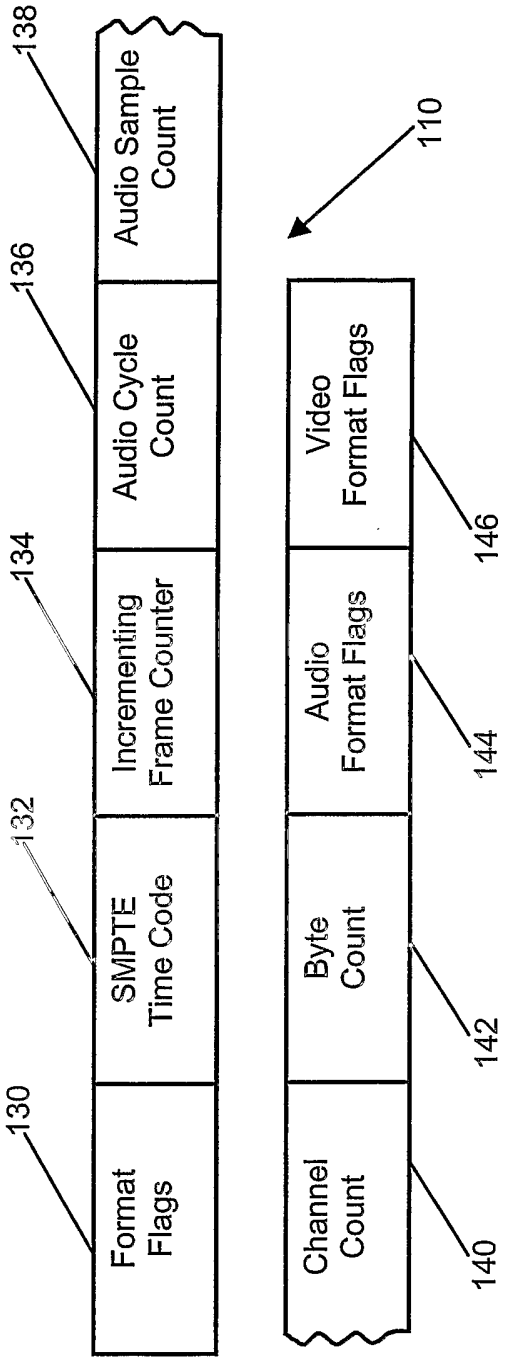

130
Format Flags
132
SMPTE Time Code
134
Incrementing Frame Counter
136
Audio Cycle Count
138
Audio Sample Count
110
140
Channel Count
142
Byte Count
144
Audio Format Flags
146
Video Format Flags

FIG. 6

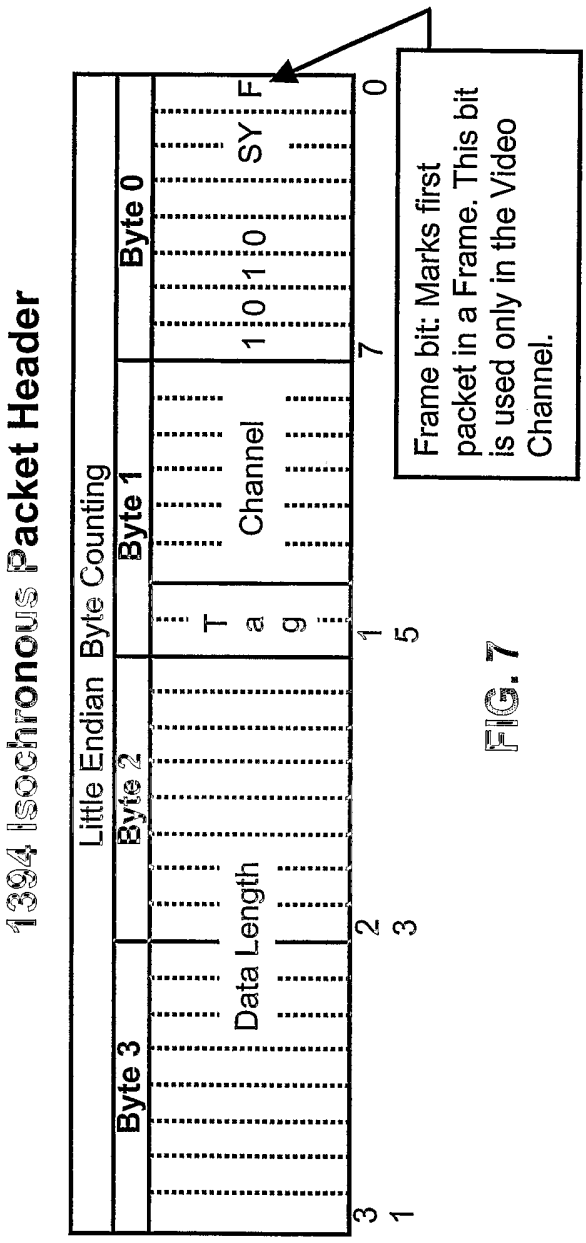

1394 Isochronous Packet Header
Little Endian Byte Counting
Byte 3
Byte 2
Byte 1
Byte 0
Data Length
Tag
Channel
SY
F
Frame bit: Marks first packet in a Frame. This bit is used only in the Video Channel.

FIG. 7

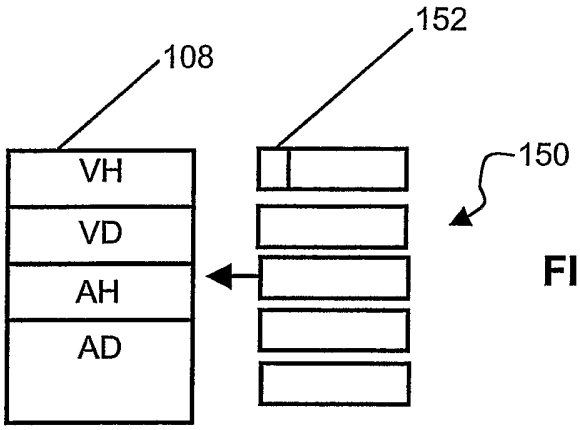


FIG. 8

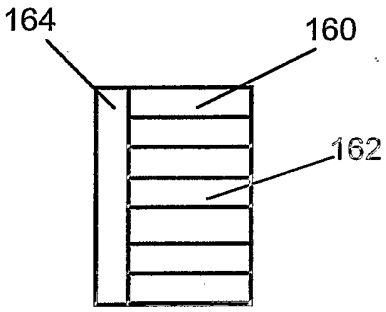


FIG. 9A

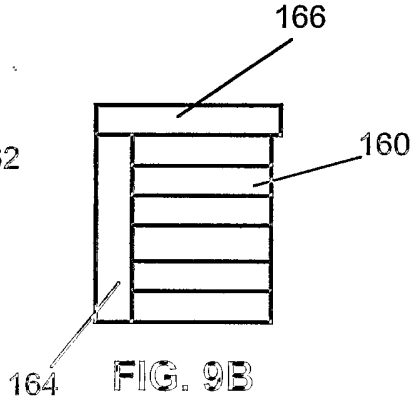


FIG. 9B

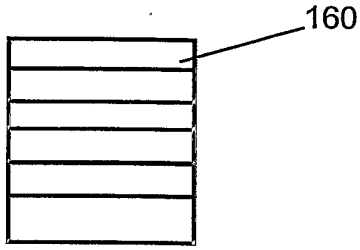


FIG. 9C

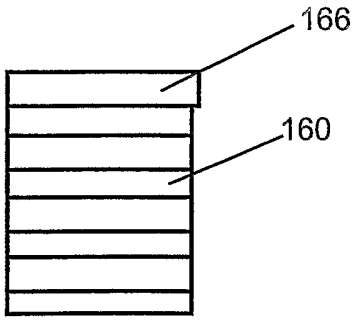
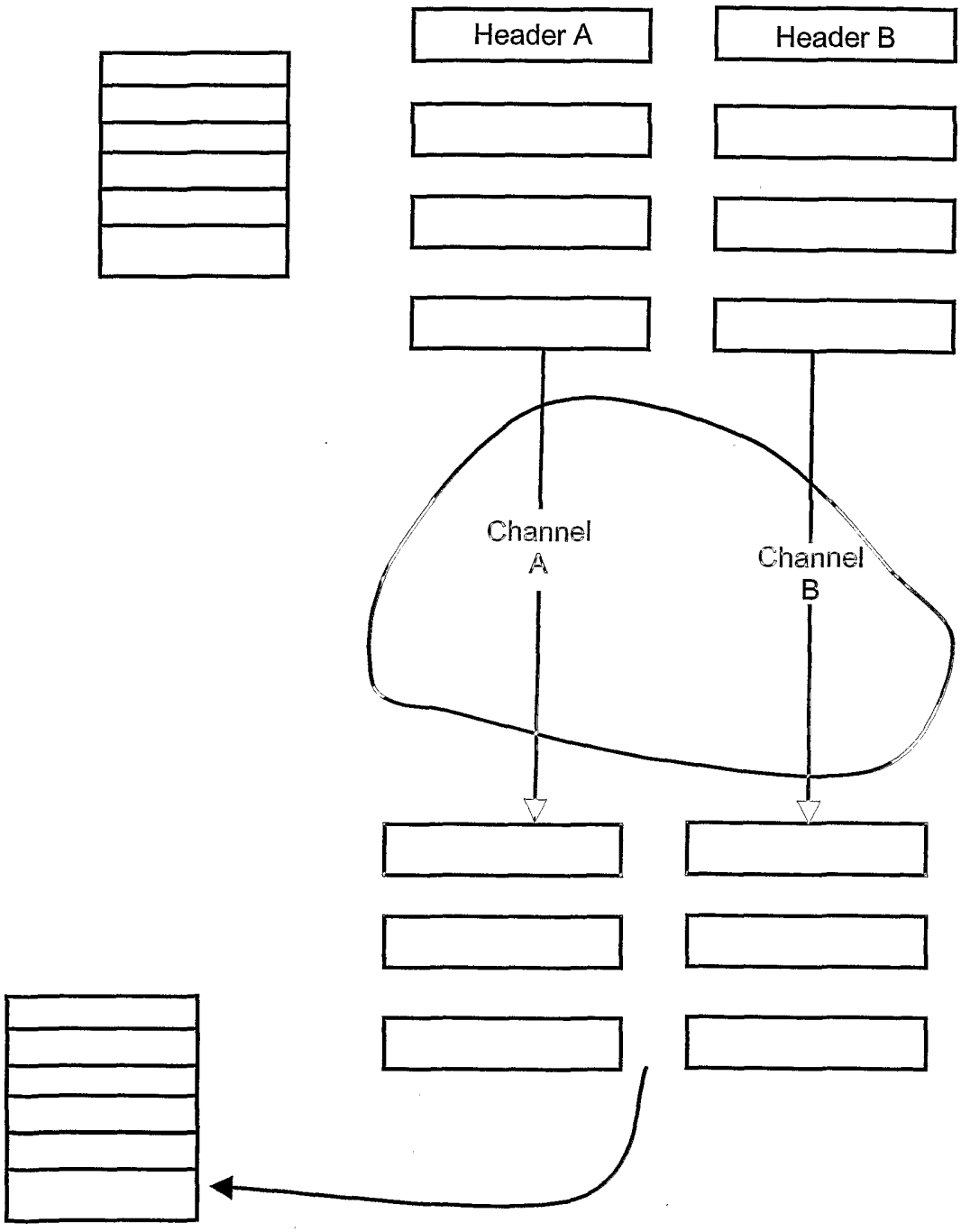


FIG. 9D

FIG. 9E



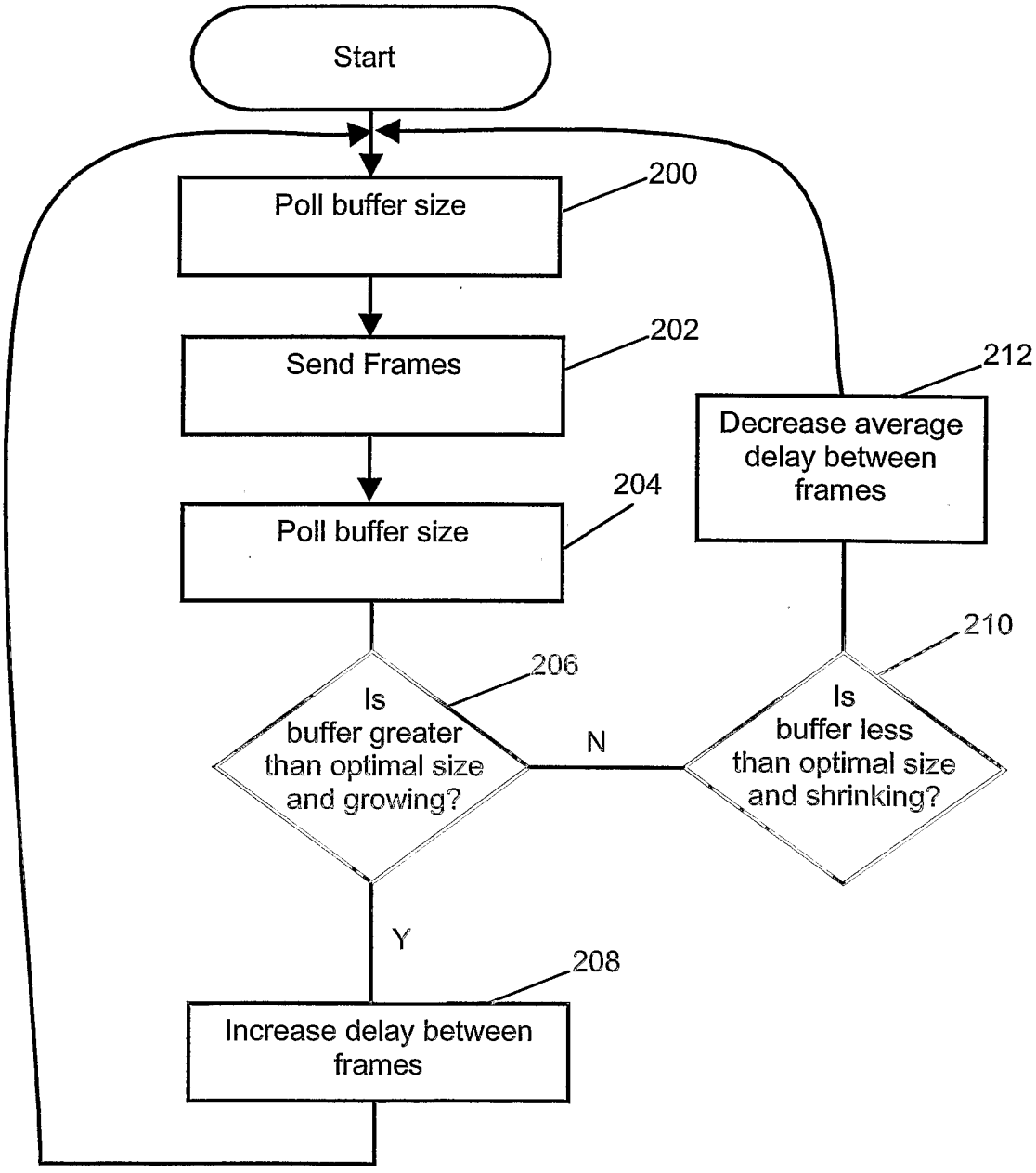

Start
Poll buffer size
200
Send Frames
202
Poll buffer size
204
Is buffer greater than optimal size and growing?
206
N
Y
Increase delay between frames
208
Decrease average delay between frames
212
Is buffer less than optimal size and shrinking?
210

FIG. 10

Register Memory Map

Register Number	Register Name
00	GLOBAL CONTROL
01	GLOBAL STATUS
02	ISOCH CONTROL
03	FLOW CONTROL
04	ISCOH CHAN

FIG. 11

A/V Global Registers

Global Control

ID	Name	Field	Bit	Description
00	Global Control	Frame Rate	[2..0]	000 — Not Valid 001 — 60.00 010 — 59.94 011 — 30.00 100 — 29.97 101 — 25.00 110 — 24.00 111 — 23.98
			[5..3]	Reserved
		Frame Geometry	[9..6]	0000 — 1920x1080 0001 — 1280x720 0010 — 720x486 0011 — 720x576 0110 — 648x486 0111 — 768x576
			[12..10]	Reserved
		Standard	[15..13]	000 — 1080i 001 — 720p 010 — 480i 011 — 576i 100 — 1080p
			[18..16]	Reserved
		Genlock	[20..19]	00 — Reference 01 — Reserved 10 — Reserved 11 — Free Run
		Reference Input Voltage	[21]	0 — 1.0 Volt 1 — 4.0 Volts
		Direction	[22]	0 — Capture 1 — Playback
			[23]	Reserved
		Reference Select	[24]	0 — Video Input 1 — Analog Reference Input
			[31..27]	Reserved

FIG. 12

Global Status

ID	Name	Field	Bit	Description
01	Global Status Register	GenLock State	[0]	0 — No Lock 1 — Locked
		Video Input Present	[1]	0 — Input not present 1 — Input present
			[3..2]	Reserved
		Analog Audio Input1 Unbalanced	[4]	0 — Balanced audio on XLR 1 — Unbalanced audio orphono plug
		Analog Audio Input2 Unbalanced	[5]	0 — Balanced audio on XLR 1 — Unbalanced audio orphono plug
		Analog Audio Input3 Unbalanced	[6]	0 — Balanced audio on XLR 1 — Unbalanced audio orphono plug
		Analog Audio Input4 Unbalanced	[7]	0 — Balanced audio on XLR 1 — Unbalanced audio orphono plug
		Reserved	[15..8]	Reserved
		Format Error	[16]	0 — Ok 1 — Error
		1394 Error[1]	[17]	0 — Ok 1 — Error
		A/V Sync Error[1]	[18]	0 — Ok 1 — Error
			[31..19]	Reserved

FIG. 13

FIG. 14

Isoch Control

ID	Name	Field	Bit(s)	Description
02	Isoch Control	Isoch Enable	[0]	0 — Isoch Channel disabled 1 — Isoch Channel enabled
			[31..1]	Reserved

FIG. 15

Flow Control

ID	Name	Field	Bit(s)	Description
03	Isoch Flow Control	Playback Byte Drift	[31..0]	This unsigned value indicates the number of bytes between the read pointer and the write pointer during playback. A zero indicates that the interface has run out of data for playback.

FIG. 16

Isoch Channel

ID	Name	Field	Bit(s)	Description
04	Isoch Channel	Isoch Channel	[5..0]	Isochronous channel number for A/V transmission
		Isoch Speed	[8..6]	Isochronous channel speed
			[31..9]	Reserved