1394 High Performance Serial Bus:

The Digital Interface for ATV

Adam J. Kunzman, Alan T. Wetzel

Texas Instruments

1394 High Performance Serial Bus:

The Digital Interface for ATV

Adam J. Kunzman, Alan T. Wetzel

Texas Instruments

ABSTRACT

The advent of the digital video revolution has created the need for a high-speed digital interface between consumer electronic devices in the home. Examples of this include the interface between a set-top box, a television, and a DVCR or camcorder using MPEG-2 transport streams, and the control and data between interactive games, computers and other devices. Clearly, an interface that supports the general transfer of data is required. The IEEE 1394 standard provides an optimum solution. This paper discusses our proposed implementation of a 1394 based solution and discusses its capabilities and operation in the interactive consumer ATV environment.

INTRODUCTION

The physical topology of the IEEE 1394 High Performance Serial Bus cable environment is a simple tree or daisy-chain network. Up to 63 devices may be connected in one network. Physical connections between nodes are made with a single easy to install cable that carries data in both directions. Devices provide one or more cable connection ports, and may be connected in any order. Configuration of the network is determined automatically during initialization. Devices having multiple cable ports act as signal repeaters to simulate a single logical bus. Each device on the bus consists of terminators and transceivers for each port, plus logic for arbitration, packet formatting, and data transfer control.

The combination of the IEEE 1394 Serial Bus with a Common Isochronous Packet (CIP) application interface layer is being referred to as HyperLynx [2]. The HyperLynx protocol stack consists of the layers shown in Figure 1.
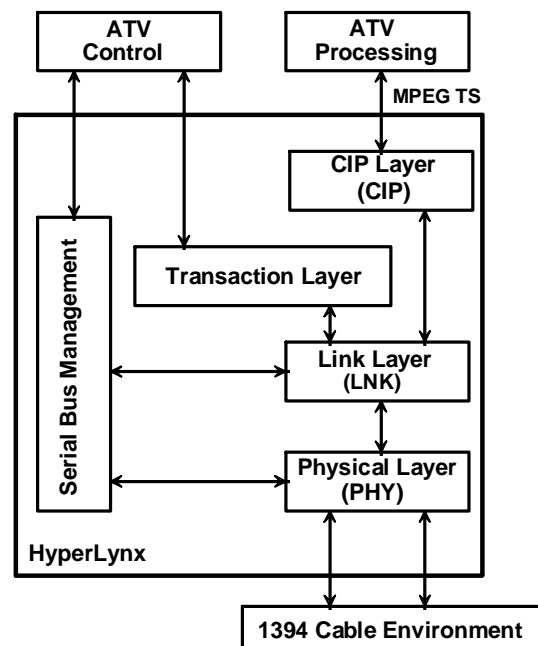

Figure 1 HyperLynx Protocol Layers

Physical Layer

IEEE 1394 defines both a backplane environment and a cable environment. Both environments operate in a similar fashion and can be easily bridged together. We shall discuss only the cable environment.

The Physical layer, or PHY, consists of the physical signaling circuits and logic that are responsible for power-up initialization, arbitration, bus-reset sensing, and data signaling. Two shielded low-voltage differential signal pairs, plus a power pair are defined in IEEE 1394 cable. For the consumer environment it is unnecessary to couple power between devices. This is permitted in the IEEE 1394 standard as long as each multi-port device in the cluster always powers its PHY circuits even when the user thinks the device is 'off'. Keeping the PHY circuits powered ensures the repeating function requirement of IEEE 1394 is met.

Signaling is done by using Data-Strobe bit-level encoding, developed by SGS-Thompson. The use of D-S encoding approximately doubles the jitter tolerance, when compared to unencoded data-clock signaling. Because of the high reliability of the IEEE 1394 cable links, the error recovery features of more complex encoding schemes such as 8B-10B are unnecessary. The D-S encoding and decoding circuitry is very simple to implement, particularly when compared to other encoding schemes such as 8B-10B or Manchester.

The base data rate defined for the IEEE 1394 cable environment is 98.304 Mbit/s. Compatible signaling rates of 196.608 Mbit/sec (2X) and 393.216 Mbit/s (4X) are also defined. Devices with different data rate capabilities may be freely interconnected and communications will automatically be performed at the highest rate supported by the lower rate devices. In the HyperLynx environment the 2X rate of 196.608 Mbit/s is the maximum supported. The 2X rate provides adequate capacity while allowing use of longer and less expensive cable links.

Link Layer

Data is formatted into packets in the Link layer. Two classes of data communication between devices are supported: asynchronous and isochronous. Asynchronous communications can be characterized as always acknowledged, whereas isochronous communications can be

characterized as always on time. The timely nature of the isochronous communications is provided by having isochronous cycles at a guaranteed rate of 8000 times per second. The isochronous cycles take priority over asynchronous traffic.

Asynchronous transactions provide a fully acknowledged datagram transfer between devices. Asynchronous transfers can take place any time the bus is free of isochronous traffic. The protocol guarantees that a minimum portion of the bus time is reserved for asynchronous data transfers. A short acknowledge packet is returned to the sender for every received packet indicating whether or not the packet was received and acted upon without error.

Isochronous transfers provide a real-time data transfer mechanism. An ongoing isochronous communication between one or more devices is referred to as a channel. Once a channel has been established, the requesting device is guaranteed to have the requested amount of bus time for that channel every isochronous cycle. Only one device may send data on a particular channel, but any number of devices may receive the data on a channel. A single device may have multiple channels allocated, and additional channels may be added as long as isochronous capacity is available for allocation. Figure 2 illustrates the general 1394 isochronous packet format.
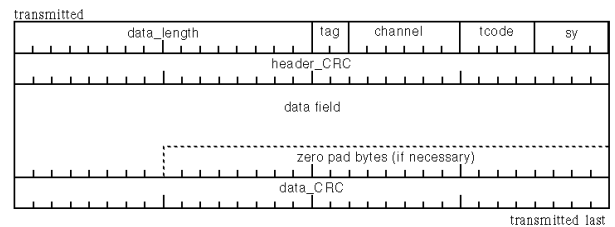

Figure 2 1394 Isochronous Packet

Every valid isochronous packet is a sequence of aligned quadlets. The length of an isochronous packet is at least two quadlets. An isochronous packet consists of a packet header, and may also include a data block.

Only one type of isochronous packet is defined for the IEEE 1394 Serial Bus.

The following summarizes the isochronous packet components and abbreviations.

Packet component	Abbreviation	Size (bits)	Comment
data length	data_length	16	All data-block packets
tag	tag	2	Data format, 01 for HyperLynx
isochronous channel	channel	6	Isochronous data-block packets
transaction code	tcode	4	All isochronous packets
synchronization code	sy	4	Isochronous data-block packets
header CRC	header_CRC	32	All isochronous packets
data-block payload	data_field	---	All data-block packets
data-block CRC	data_CRC	32	All data-block packets

Transaction Layer

The transaction layer defines a complete request-response protocol to perform bus transactions. These support the IEEE 1212 Control and Status Register (CSR) Architecture [3] with the operations of read, write, and lock. Although the transaction layer does not add any services for isochronous data transfers, it does provide a path for management of the resources needed for isochronous services. This is through reads and writes to the isochronous control CSR's.

Asynchronous data is transferred between nodes on the IEEE 1394 Serial Bus by three different types of transactions:

- Read: Data is retrieved from an address in a different node.

- Write: Data is transferred to an address in a different node.

- Lock: Data is sent to a different node, operated on by the other node, and then returned to the original node.

In addition the transaction layer defines a retry mechanism to handle situations where resources are busy and unable to respond. Both the requests and the responses may be retried.

Bus Management

Bus Management consists of the protocols, services, and operating procedures whereby one node is selected and may then exercise management level control over the operation of the remaining nodes on the bus. Several levels of management capability are defined by IEEE 1394. The minimum level, Isochronous Capable, is required for any devices which can source or receive isochronous packets. Further, at least one node in a network with isochronous traffic must be available as the Isochronous Resource Manager. Also, there may optionally be implemented the higher Bus Manager level of control.

The isochronous resource manager provides a common location for the other nodes to check on the availability of channels and bandwidth, and to register their new allocations. The location of the isochronous resource manager is determined and becomes known by all nodes during the Self-ID portion of the reset and initialization process.

In the HyperLynx environment devices which can only receive isochronous packets need only implement the Isochronous Capable level. Devices which can source isochronous packets must implement at least the Isochronous Resource Manager Capable level of management.

Device Control

The primary mechanism used for controlling the resources in an IEEE 1394 bus is through reads and writes to a standardized set of memory mapped registers. All nodes contain a minimum set of registers (CSR's) for information unique to that particular device. During initialization, one or two nodes in the network will be selected as the home location for additional CSR's that

contain information relevant to the operation of the bus as a whole. Note that all nodes capable of sourcing isochronous packets are required to contain these CSR's, so no special devices are needed in a network to act as the master reference location. The CSR's provide information about the channel numbers assigned or available, the bus time available per isochronous cycle, and other parameters .

Additional sets of isochronous resource CSR's are being defined for the consumer ATV environment by members of the 1394 Trade Association. One set, referred to as plug control registers (PCR's), provides an alternative virtual plug and jack mechanism for controlling the connections between devices. Another proposal provides a standardized location for information about the control languages and protocols supported by a device. A third set of proposed CSR's gives details of a device's interface characteristics such as maximum packet size and packet rate.

The CSR's defined by 1394 are alligned on quadlet, or 32 bit, boundaries. Operations on these registers will be in multiples of four bytes. The minimum asynchronous packet payload that must be received by any node is therefore one quadlet, or 4 bytes. For the purposes of control, a minimum of 8 quadlets, or 32 bytes is specified. This is sufficient for all CSR operations, and is also sufficient for the control language formats that will be used in this environment.

Common Isochronous Packet (CIP) Layer

IEEE 1394 defines a basic mechanism for real-time data transport, but does not establish the protocols needed for specific application requirements such as sending MPEG-2 transport streams, SD or HD DVCR data, or ATM cells. The CIP specification, as defined by the 1394 Trade Association, is a general protocol to format application specific packet data into 1394 isochronous packets. Specific applications using CIP are being formally specified by the 1394 Trade Association.

A generic A/V CIP format has been defined to accept incoming packets, break them up if necessary into smaller data blocks to send them efficiently over the bus, then recreate the packet stream timing at the receiving device. This process is depicted in Figure 3.
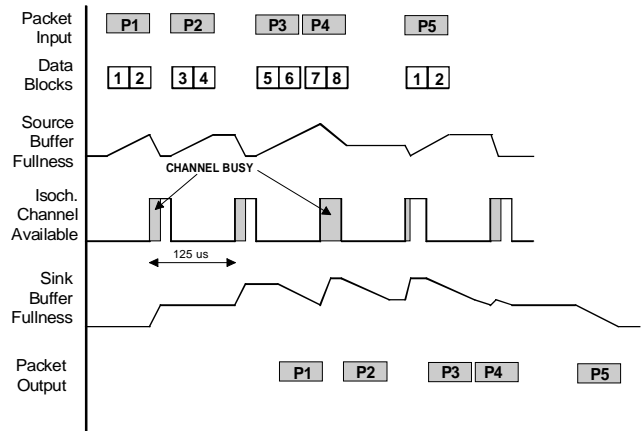

Figure 3 Packet transmission using CIP

Incoming packets may be divided (by a factor of 2, 4 or 8) to form smaller data blocks or left whole. Small data blocks allows for more efficient use of the bus capacity. Each isochronous cycle, a set of data blocks are extracted from the FIFO (up to the maximum capacity of the isochronous channel) and broadcast on the bus. The number of data blocks that are sent each cycle is variable depending on FIFO fullness. To detect missing data blocks, each data block is counted. The count value (index) of the initial data block in each CIP packet is placed in the header.

CIP defines two parts of the isochronous packet. First, the *tag* field is defined to have the value of 0x1. This allows a device to quickly determine if the isochronous packet follows the CIP syntax. Second, the data block is defined to have the CIP header information followed by the application packet data.

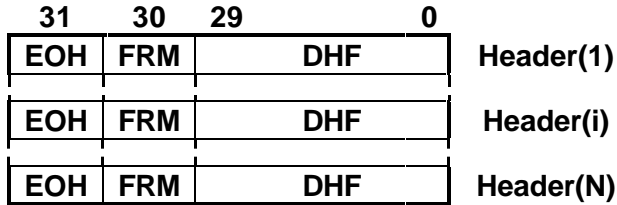The proposed CIP header syntax is illustrated in Figure 4.

```
 31    30   29              0
┌─────┬─────┬──────────────┐
│ EOH │ FRM │     DHF      │   Header(1)
├─────┼─────┼──────────────┤
│ EOH │ FRM │     DHF      │   Header(i)
├─────┼─────┼──────────────┤
│ EOH │ FRM │     DHF      │   Header(N)
└─────┴─────┴──────────────┘
```

Figure 4 CIP Header Format

The definitions of the fields are given as follows:

- EOH (End of Header): A bit that identifies if this is the last header quadlet:

   0: Another quadlet will follow.
   1: The last quadlet of the header.

- FRM: A bit used in combination with the EOH bit, which identifies the code definition of the DHF.

- DHF (Data Header Field): The data header field of the ith quadlet. The code definition of DHF depends on i, EOH, and FRM.

A/V CIP Format

The generic format for consumer A/V transmission has a 2 quadlet CIP header defined. The structure of the two quadlet header is shown below.
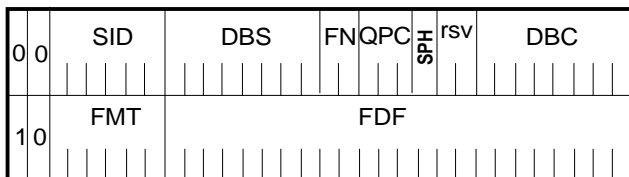
```
┌─┬─┬───────┬──────┬──┬───┬─┬───┬────────┐
│0│0│  SID  │ DBS  │FN│QPC│S│rsv│  DBC   │
│ │ │       │      │  │   │P│   │        │
│ │ │       │      │  │   │H│   │        │
├─┼─┼───────┴──────┴──┴───┴─┴───┴────────┤
│1│0│ FMT   │         FDF                │
│ │ │       │                            │
└─┴─┴───────┴────────────────────────────┘
```

Figure 5 Two Quadlet CIP Data Header

The fields are defined as follows.

- SID: Source node ID (Node ID of transmitter).

- DBS: Data block size in quadlets. The DBS field is 8 bits because 256 quadlets is the maximum size allowed for an isochronous packet at the IEEE 1394 S100 base rate of 98.304 Mbit/sec. Multiple data blocks may be put into a bus packet, if higher bandwidth is required at the S200 or S400 rates.

- FN: Fraction number. Source packet subdivision is optional and is used to improve bus time utilization. The number of data blocks into which a source packet may be split is shown below. When packet division is used, the lower bits of the DBC field indicate the offset (in data blocks) from the beginning the source packet. The bits to be utilized depends on the FN value, as shown below:

FN	Packet division	Offset
00	Not divided.	N/A
01	Divided by 2	Lowest bit
10	Divided by 4	Lowest 2 bits
11	Divided by 8	Lowest 3 bits

- QPC: Quadlet Padding Count (0 quadlet to 7 quadlets). Used only when the FN does not indicate 00. In the case where FN is 00, QPC shall be 000.

- SPH: Source packet header. A flag to indicate the source packet has its own header.

- rsv: Reserved for future extension

- DBC: Data Block Count. A continuity counter is incremented every data block. When packet division is used, multiple data blocks are sent in each CIP packet and this field also contains the offset value of the first data block in the CIP packet.

- FMT: Format ID (such as DVCR or MPEG). Code allocation is illustrated below:

FMT(binary)	Packet Type
000000	DVCR
000001	MPEG-2 TP
000010-111100	Reserved
111110	Vendor Unique
111111	No Data

Code Allocation of FMT

NOTE: In case FMT = 111111 (no data), the fields for DBS, FN, QPC, SPH and DBC are ignored and no data blocks are transmitted.

- FDF: Format dependent field. This field is defined for each format ID. For example, in the case of DVCR data, the SD/HD flag, the 60/50 flag, the unreliable data flag, and synchronization time stamp is defined.

Bus Operation

The 1394 bus sequences through three general phases: a cycle initiation phase, an isochronous phase, and an asynchronous phase.

The 1394 bus operation is divided into time segments that nominally last 125 µs (8 kHz). Each segment is called an isochronous cycle. The cycle begins when the bus cycle master (determined during bus initialization) arbitrates for the bus and transmits a special asynchronous packet called a cycle start packet. Within this packet is the value of the cycle master's clock counter. Each device on the bus receives this value and updates their own local clock counter value. This guarantees that all devices on the bus have a common time-base which is essential for removing time-jitter from MPEG-2 transport packets.

At the completion of the cycle start packet, isochronous packet transfers are enabled. Devices that have obtained an isochronous channel arbitrate for the bus. The order of arbitration is not guaranteed from cycle to cycle. However, a node is guaranteed to have access to the bus for the requested amount of time. This allows the worst-case delay time to be bounded and subsequently buffer sizes may be determined.

The bus management protocol guarantees that no more than 80% of the nominal isochronous cycle (125 µs) is allocated for all of the isochronous channels. This ensures that some asynchronous transfers can take place even if the network is fully loaded with isochronous traffic. The asynchronous phase concludes when the bus cycle master issues a new cycle start packet.

MPEG-2 Transport

A common view of the system timing is essential if multiple devices and services are to interoperate over the 1394 bus. Refer to the Figure 6. A 1394 system consists of source and a sink nodes. The HyperLynx model assumes that a constant average rate of data is flowing between nodes during a single session. A session is defined to be the time between an initiation of an isochronous channel transfer until a new isochronous channel is initiated on the bus or when one is terminated.
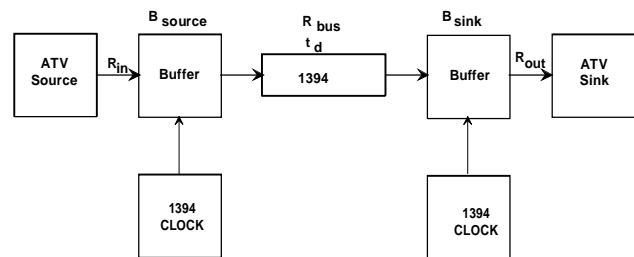

Figure 6 System Timing Model

An ATV data stream is composed of 188 byte long (47 quadlets) MPEG-2 transport packets. Since the packet rate cannot be assumed to be constant, each transport packet has a time stamp prepended to it that is derived from the local 1394 clock. This adds 4 bytes for a total of 192 bytes/packet. Each time-stamped packet is placed into the source FIFO.

A terrestrial (8-VSB) ATV data stream has a bit rate of 19.4 Mbits/sec. This means that the

IEEE 1394 isochronous transport system (at 8000 cycles/sec) will be carrying an average of:

$$\frac{19.4 \bullet 10^6}{8000} = 2425 \text{ bits / cycle}$$

$$= 303.125 \text{ bytes / cycle}$$

$$= 1.6124 \text{ TP / cycle}$$

Assuming the source packets are not subdivided into smaller data blocks, the structure of the complete IEEE 1394 ATV isochronous packet is illustrated below. Approximately 2 out of 3 cycles the packet will contain 2 MPEG-2 transport packets, while during the other cycle the packet will contain only one MPEG-2 transport packet. The number of full packets available in the source buffer will determine how many are transported, the average over time will be the previously calculated 1.6124 MPEG-2 transport packets/cycle.

1394 isochronous packet header
header CRC
CIP header 1
CIP header 2
timestamp for MPEG TP #1
1st quadlet MPEG TP #1
additional TP #1 quadlets
47th quadlet MPEG TP #1
timestamp for MPEG TP #2
1st quadlet MPEG TP #2
additional TP #2 quadlets
47th quadlet MPEG TP #2
data CRC

The complete isochronous packet has a total size of:

$$Packet\_size = 2 + 2 + 2 \bullet (1 + 47) + 1$$

$$= 101 \text{ quadlets}$$

$$= 404 \text{ bytes.}$$

The source buffer must account for the maximum delay that may occur between packet transmissions. This includes the time between cycles, plus the delay that may be induced by asynchronous bus traffic (78 μs), plus the maximum delay that may occur due to other isochronous bus traffic. This latter value is determined by the maximum time available for isochronous packets minus the time required for the the packet itself (100 μs - maximum channel time).

The longest channel time needed for an ATV packet is set by two MPEG-2 packets, plus the timestamp and header data (as shown above) which equal 404 bytes. Channel time is then:

$$t_c = \frac{Packet\_size}{Bit\_rate}$$

$$= \frac{404 \bullet 8}{98.304 \bullet 10^6}$$

$$= 32.9 \mu s$$

which leads to a maximum delay of:

$$t_d = t_{cycle} + t_{asynch} + \left(t_{isoch} - t_c\right)$$

$$= 125 + 78 + \left(\frac{8}{10} \bullet 125 - 32.9\right)$$

$$= 280 \mu s.$$

The minimum buffer size is calculated by:

$$B_{source} = t_d \bullet R$$

$$= 280 \bullet 10^{-6} \bullet \frac{19.4 \bullet 10^6}{8}$$

$$= 679 \text{bytes}$$

Therefore, for terrestial ATV bitstreams, a minimum source buffer size of 679 bytes is required.

The 1394 system clock and packet time-stamps may be used to effectively cancel the received packet delivery time variations. The accuracy with which this may be accomplished is related to three parameters:

1. The clock granularity at the source and sink nodes, based on the 1394 system clock at 24.576 MHz;

2. The maximum slip or skew between the source and sink node clocks between updates, based on a maximum of 100 ppm over the 100 µsec isochronous cycle;

3. The maximum variation in delivery delays through the network of the cycle start packet clock update, based on a 1 clock resolution at 98.304 MHz in each of 16 retiming nodes.

This calculates as:

$$t_{de} = t_g + t_{cs} + t_{dd}$$

$$= 2 \bullet 40.69 + 2 \bullet \left(100 \bullet 10^{-6}\right)^2 + 16 \bullet 10.17$$

$$= 81.38\text{ns} + 20.00\text{ns} + 162.67\text{ns}$$

$$= 264.14\text{ns}$$

This is believed to be significantly better than what is required for most known or anticipated isochronous applications, including MPEG-2 transport streams.

SUMMARY

The IEEE 1394 High Performance Serial Bus provides an ideal mechanism for connecting digital consumer audio/video equipment. It combines high bandwidth with guaranteed delivery of time critical data through isochronous channels. Easy to use cable assemblies, plug and play auto configuration, and unconstrained topologies relieve the user from the headaches of today's one-per-signal wiring. A simple Common Isochronous Header and A/V specific CSR's are added to the basic IEEE 1394 serial bus. The HyperLynx combination can effectively carry the MPEG-2 transport packet streams and control packets unique to the consumer audio/video devices.

REFERENCES

[1] IEEE Standard of a High Performance Serial Bus, P1394/Draft 7.1v1, August 5, 1994.

[2] HyperLynx, Proposal for EIA Standard ATV Baseband Digital Interface, February 8, 1995.

[3] ISO/IEC 13213 (ANSI/IEEE 1212-1991) CSR Architecture

Additional information and updates concerning IEEE 1394 and the 1394 Trade Association may be found on the world Wide Web page at: *html:\\www.skipstone.com.*