1394 Open Host Controller Interface

Michael D. Johas Teener
Chief Technical Officer,
Firefly, Inc.
269 Mt. Herman Rd. #102
Scotts Valley, CA 95066-4000
mike@fireflyinc.com

What it is

- Register set, data structures, and behavior for IEEE 1394 interface hardware
  - 1394 Open Host Controller
- Hardware implementation notes for specific busses
  - PCI/x86 and PCI/PowerMac examples

What it is NOT

- NOT a particular hardware implementation
  - although a good “external” specification
  - vendors are allowed to add extra features if properly done
  - for example, PHY may or may not be integrated
- NOT a software API
  - although it will ease the specification of APIs for different OSs

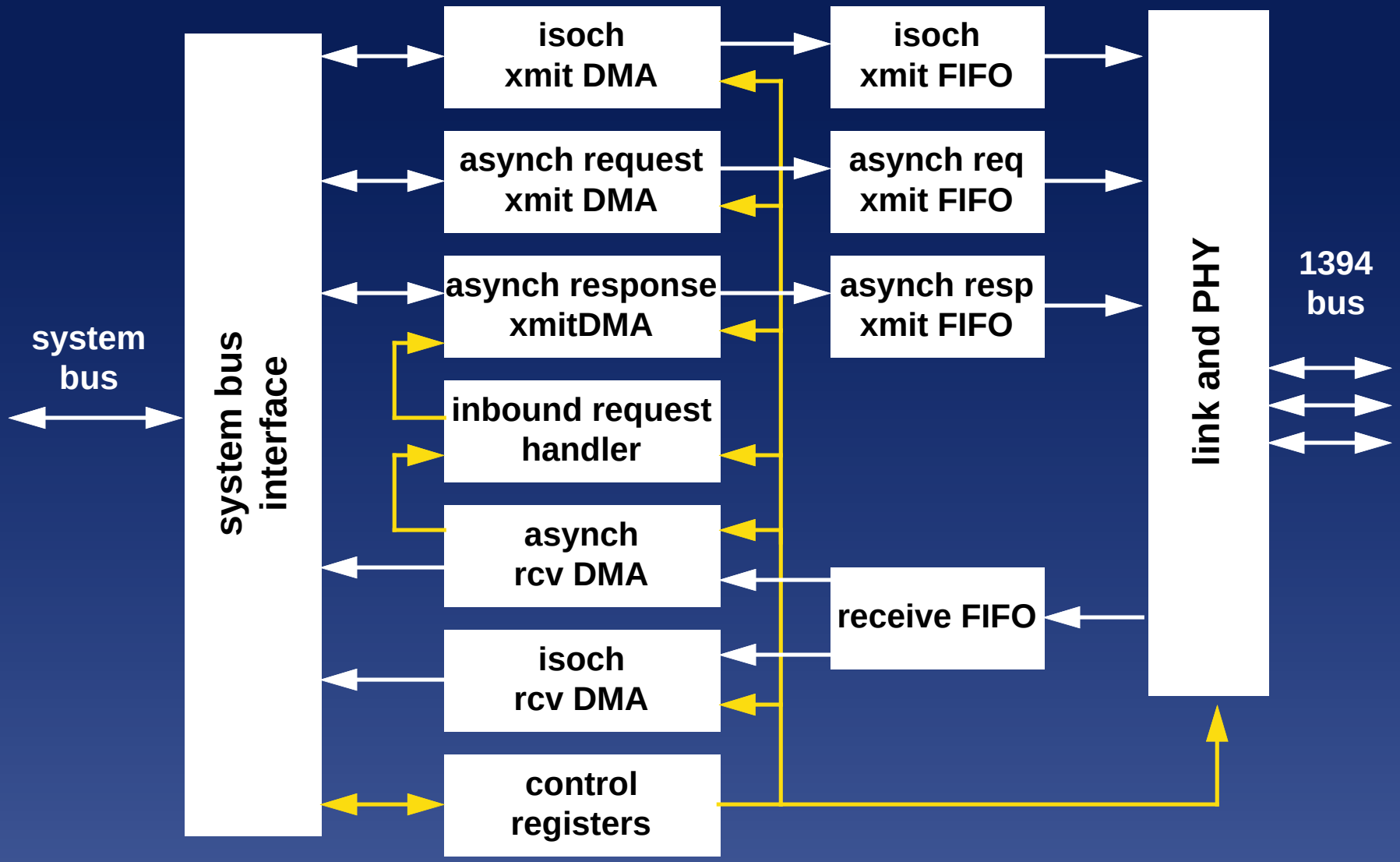
History

- November 1995:PC vendors, Microsoft, Apple, IC vendors meet to discuss common needs
- Winter 1996: Basic requirements set, all agree that a single interface will speed deployment, Open HCI effort starts
- Spring 1996: Apple's "Pele" design used as baseline, extensive modifications and fixes added
- Summer 1996: Open HCI design firm, "1394 Open HCI Promoter's agreement''

Terminology

- IEEE terms used to avoid confusion:
  - “quadlet” is 32-bit word (four bytes)
- Numbers use Verilog notation
  - (number of bits)’(base)(digits)
  - 10’h3FF = 10’b11_1111_1111 = 1023

Architecture



1394 Link and PHY

- Full support for 1394-1995
  - All asynch and isoch packets
  - Automatic assignment of addresses
  - Cycle master capability with internal or external cycle timing
  - Received packets are time-stamped for timing-critical applications
  - Automatic recovery from common errors
- If PHY is separate part, must use standard PHY-Link interface

Some P1394A extensions

- Faster reconfiguration
  - arbitrated reset, possible incremental configuration
- Improved arbitration
  - ACK acceleration (subaction gap not required after ack)
  - First-attempt responses do not require fair access
- Improved PHY management
  - PHY ping for gap count optimization and incremental configuration
  - Support for future 800+ Mbit/sec PHYs

FIFOs

- 4 FIFOs defined
  - Transmit Asynch Request
  - Transmit Aysnch Response
    - not required if hardware guarantees that queued requests do not block new responses
  - Transmit Isoch
    - not required if system bus has very low latency (guaranteed less than 1 μsec or so)
  - Receive
- Size depends on system bus latency
  - Peter Teng of NSC will elaborate

System bus interface

- 32-bit addresses
  - 64-bit in future versions
- Atomic read and write of at least 32-bit quadlet
  - bursts very useful, but do not have to be atomic
- High bandwidth and low latency very useful
  - S400 requires 50 Mbyte/sec average throughput (not peak), 1 μsec latency costs 50 bytes in FIFO
  - S800 will need 100 Mbyte/sec!
- PCI, Power PC and Pentium system busses are examples

Addressing

- Uses 1k byte address space
  - 256 quadlet registers
  - first 128 for general control and status
  - second 128 for DMA control
  - much of address space reserved for vendors or future specification

DMA

- Not just traditional DMA, but also bus master funtions that accelerate transaction and bus management
- Basic DMA uses “Descriptor Based DMA”
  - Transmit asynch requests, software-generated responses
  - Receive isoch data
  - Receive asynch requests and responses not handled by hardware
  - Mike Eneboe of Apple will elaborate

More DMA

- Isoch Transmit DMA uses cycle-time-based array of packet descriptors
  - Dianna Klashman of Sun will elaborate
- Dedicated DMA buffer for received self-ID packets
  - Dianna gets this one, too

Inbound transaction acceleration

- Received read and write requests to first 4 Gbyte of node addresses can map to system bus
  - Filtering by source node ID (accept only reads and writes from particular nodes)
  - Open HC will perform transaction and send appropriate response packet (if any)

Inbound transaction acceleration (cont.)

- Provides inbound bus bridge function
  - Devices provide DMA functionality
    - DMA resources scale with number of devices
    - DMA structures and programming optimized for particular data and control
- Dianna, of course ...

Outbound transaction acceleration

- Received response packets can be handled automatically
- Match on response transaction label and source ID
  - If match, any returned data (from read or lock) is stored at specified address, response code is saved, and optional interrupt generated
- One last time for Dianna ...

Bus management support features

- 1394 Configuration ROM mapping
  - first 5 quadlets of config ROM (bus_info_block, including unique ID) mapped to on-chip read-only registers
    - May be automatically loaded from external ROM
    - If not, boot code must load “write-once” registers before any patches an be activated
  - 1st 1k byte of 1394 config ROM mapped to any 1k byte block in system memory
  - Only quadlet reads supported

More bus management

- Resource management registers implemented in hardware
  - Channels available, bandwidth available, manager ID registers must support compare-swap in timely manner (accessed immediately after bus reset)
  - Compare-swap operation implemented from both the system bus and 1394 sides

Interrupts

- 32 events, individually masked
- Event and mask registers have two addresses: set and clear
  - A “one” written to the “set” address sets the corresponding bit
  - A “one” written to the “clear” address clears the corresponding bit
  - A “zero” written to either address has no effect
- Events include various errors (cycle too long, cycle mismatch, etc) and management conditions (bus reset)

System bus aids

- System bus configuration ROM interface
  - Needed only for PCI and simular slot-based busses
  - PCI needs Plug-and-play ROM for Windows, Open Firmware for Mac/ Solaris/AIX