FireLink 1394 OHCI
Link Controller

82C881

Preliminary Data Book
CONFIDENTIAL

Revision 1.0

912-2000-031

December 13, 1999

OPTi Inc.

1440 McCarthy Blvd.

Milpitas, CA 95035

Tel: (408) 486-8000

Fax: (408) 486-8001

WWW: http://www.opti.com

TABLE OF CONTENTS

1.0 Features

The OPTi 82C881 1394 OHCI Link Controller is a PCI-based host controller with the following features.

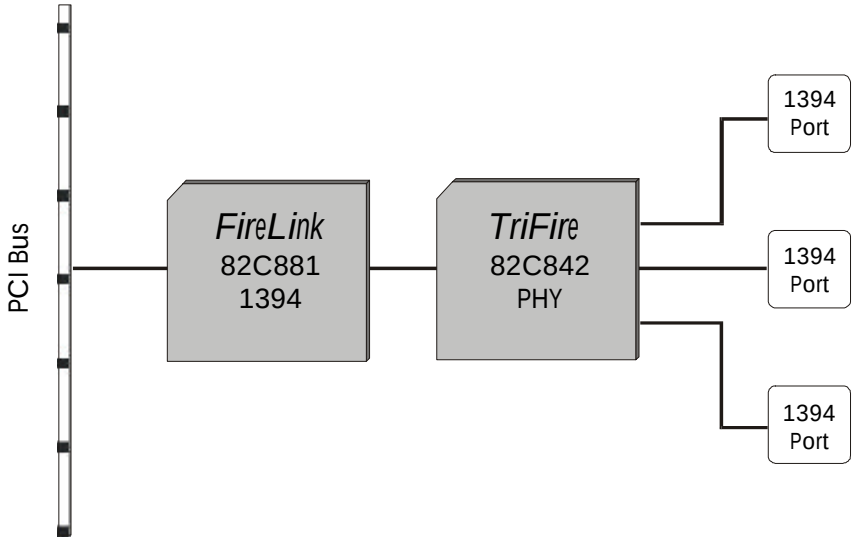
- Compliant with PCI Local Bus Specification 2.1
- Compliant with P1394a Draft 2.0 Standard for a High-performance Serial Bus
- Interfaces to 33MHz, 32-bit PCI bus
- PnP (Plug and Play) compatible per PCI Local Bus Specification rev. 2.1
- Implements IEEE1212-based control and status registers that can be mapped to both I/O and memory space
- Incorporates independent DMA controllers for isochronous and asynchronous operations
- Supports four isochronous transmit and isochronous receive contexts
- Supports burst transactions on the PCI bus interface
- Offers direct access to the physical address space of the host
- Assigns priority for DMA per 1394 OHCI specification 1.00
- Supports four transmit and three receive configurable FIFOs
- Offers both packet per-buffer and buffer-fill modes of operation for the isochronous receive context
- Incorporates two wire industry-standard Serial EEPROM interface
- Functions as a 1394 cycle master
- Supports asynchronous and isochronous transfers at 100, 200 and 400 Mbps
- Implements a fully Bus Manager Capable node including an Isochronous Resource Manager
- Interfaces to PHYs that conform to the Link-PHY interface described in Chapter 5 of the P1394a Draft 2.0 specification
- Interfaces to 1394-1995 compliant PHYs
- Offers selective disabling of 1394a features (controllable by software) for interfacing to a partially P1394a compliant PHY
- Supports posting of physical write request packets
- Complies with the PCI Power Management specification rev. 1.1, supporting ACPI states D0 and $D3_{hot}$ and PME# generation
- Incorporates CLKRUN# support
- Implements comprehensive Debug Registers
- Implements physical upper bound register.

2.0 Overview

This document describes the OPTi FireLink 1394 OHCI Link Controller (82C881). It details:

- Signal Definitions
- Strap Selectable Options
- Functionality
- Register Descriptions.

FireLink 1394 System Block Diagram



3.0 Signal Definitions

3.1 Terminology/Nomenclature Conventions

The "#" symbol at the end of a signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When "#" is not present after the signal name, the signal is asserted when at the high voltage level.

The terms assertion and negation are used extensively. This is done to avoid confusion when working with a mixture of active low and active high signals. The term assert, or assertion indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term negate, or negation indicates that a signal is inactive.

The tables in this section use several common abbreviations. Table 3-1 lists the mnemonics and their meanings. Note that TTL/CMOS/Schmitt-trigger levels pertain to inputs only. Outputs are driven at CMOS levels.

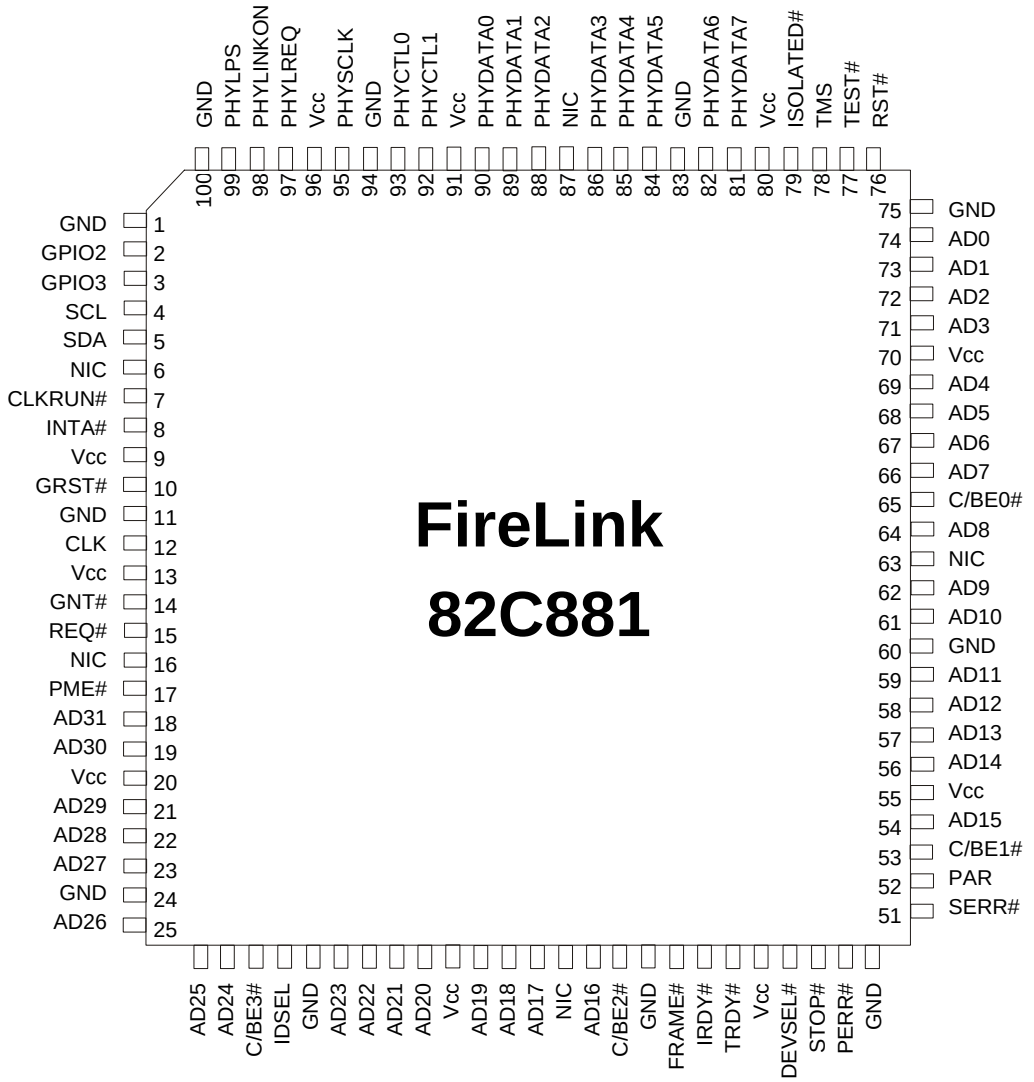
Table 1. Signal Definitions Legend

Mnemonic	Description
Analog	Analog-level compatible
CMOS	CMOS-level compatible
Dcdr	Decoder
Ext	External
G	Ground
I	Input
Int	Internal
I/O	Input/Output
Mux	Multiplexer
NIC	No Internal Connection
O	Output
OD	Open drain
P	Power
PD	Pull-down resistor
PU	Pull-up resistor
S	Schmitt-trigger
S/T/S	Sustain Tristate
TTL	TTL-level compatible

Figure 3-1 LQFP Pin Diagram (Note)

Note: Figure 3-1 shows a pin diagram of the 82C881 packaged in an LQFP (Low-profile Quad Flat Pack, square). The device is also available in a QFP (Quad Flat Pack, rectangular). The pin assignment remains the same.

Refer to Section 5, "Mechanical Package" for details regarding packaging.



3.2 Numerical Pin Cross-Reference List

Pin No.	Signal Name	Power Plane
1	GND	VccCORE
2	GPIO2	
3	GPIO3	
4	SCL	
5	SDA	
6	NIC	
7	CLKRUN#	
8	INTA#	
9	Vcc	
10	GRST#	
11	GND	
12	CLK	
13	Vcc	
14	GNT#	
15	REQ#	
16	NIC	
17	PME#	
18	AD31	
19	AD30	
20	Vcc	
21	AD29	
22	AD28	
23	AD27	
24	GND	
25	AD26	
26	AD25	
27	AD24	
28	C/BE3#	
29	IDSEL	
30	GND	
31	AD23	
32	AD22	
33	AD21	
34	AD20	

Pin No.	Signal Name	Power Plane
35	Vcc	VccCORE
36	AD19	
37	AD18	
38	AD17	
39	NIC	
40	AD16	
41	C/BE2#	
42	GND	
43	FRAME#	
44	IRDY#	
45	TRDY#	
46	Vcc	
47	DEVSEL#	
48	STOP#	
49	PERR#	
50	GND	
51	SERR#	
52	PAR	
53	C/BE1#	
54	AD15	
55	Vcc	
56	AD14	
57	AD13	
58	AD12	
59	AD11	
60	GND	
61	AD10	
62	AD9	
63	NIC	
64	AD8	
65	C/BE0#	
66	AD7	
67	AD6	
68	AD5	

Pin No.	Signal Name	Power Plane
69	AD4	VccCORE
70	Vcc	
71	AD3	
72	AD2	
73	AD1	
74	AD0	
75	GND	
76	RST#	
77	TEST#	
78	TMS#	
79	ISOLATED#	
80	Vcc	
81	PHYDATA7	
82	PHYDATA6	
83	GND	
84	PHYDATA5	
85	PHYDATA4	
86	PHYDATA3	
87	NIC	
88	PHYDATA2	
89	PHYDATA1	
90	PHYDATA0	
91	Vcc	
92	PHYCTL1	
93	PHYCTL0	
94	GND	
95	PHYSCLK	
96	Vcc	
97	PHYLREQ	
98	PHYLINKON	
99	PHYLPS	
100	GND	

3.3 Strapping Options

3.3.1 Test Mode Selection (Pins 77, 78)

Pin 78, TMS#	Pin 77, TEST#	Function
Pulled up	Don't care	Normal Operation
Pulled down	Pulled down	Scan Chain Test mode
Pulled down	Pulled up	NAND Tree Test mode

3.3.2 Serial EEPROM Presence Detect (Pin 5)

Pin 5, SDA	Serial EEPROM	Function
Pulled down	Not present	Load default values on PCI registers.
Pulled up	Present	Load PCI values from EEPROM

3.4 Signal Descriptions

3.4.1 PCI Bus Interface Signals

Signal Name	Pin No.	Pin Type	Signal Description
AD[31:0]	Refer to Pin Diagram	I/O	**Address and Data Lines 31 through 0:** This bus carries the address and/or data during a PCI bus cycle. A PCI bus cycle has two phases - an address phase which is followed by one or more data phases. During the initial clock of the bus cycle, the AD bus contains a 32-bit physical byte address. AD[7:0] is the least significant byte (LSB) and AD[31:24] is the most significant byte (MSB). After the first clock of the cycle, the AD bus contains data. When the 82C881 is the target, AD[31:0] are inputs during the address phase. For the data phase(s) that follow, the 82C881 may supply data on AD[31:0] in the case of a read or accept data in the case of a write. When the 82C881 is the master, it drives a valid address on AD[31:2] during the address phase, and drives write or accepts read data on AD[31:0] during the data phase. As a master, the 82C881 always drives AD[1:0] low.
C/BE[3:0]#	28, 41, 53, 65	I/O	**Bus Command and Byte Enables 3 through 0:** These signals provide the command type information during the address phase and carry the byte enable information during the data phase. C/BE0# corresponds to byte 0, C/BE1# to byte 1, C/BE2# to byte 2, and C/BE3# to byte 3. If the 82C881 is the initiator of a PCI bus cycle, it drives C/BE[3:0]#. When it is the target, it samples C/BE[3:0]#.
PAR	52	O	**Even Parity:** The 82C881 calculates PAR for both the address and data phases of PCI cycles. PAR is valid one PCI clock after the associated address or data phase, but may or may not be valid for subsequent clocks. It is calculated based on 36 bits - AD[31:0] plus C/BE[3:0]#. "Even" parity means that the sum of the 36 bit values plus PAR is always an even number, even if one or more bits of C/BE[3:0]# indicate invalid data.
FRAME#	43	I/O (s/t/s)	**Cycle Frame:** This signal is driven by the current PCI bus master to indicate the beginning and duration of an access. The master asserts FRAME# at the beginning of a bus cycle, sustains the assertion during data transfers, and then negates FRAME# in the final data phase. FRAME# is an input when the 82C881 is the target and an output when it is the initiator. FRAME# is tristated from the leading edge of RESET# and remains tristated until driven as either a master or slave by the 82C881.

Signal Name	Pin No.	Pin Type	Signal Description
IRDY#	44	I/O (s/t/s)	**Initiator Ready:** IRDY#, along with TRDY#, indicates whether the 82C881 is able to complete the current data phase of the cycle. IRDY# and TRDY# are both asserted when a data phase is completed. During a write, the 82C881 asserts IRDY# to indicate that it has valid data on AD[31:0]. During a read, the 82C881 asserts IRDY# to indicate that it is prepared to accept data. IRDY# is an input when the 82C881 is a target and an output when it is the initiator. IRDY# is tristated from the leading edge of RESET# and remains tristated until driven as either a master or a slave by the 82C881.
TRDY#	45	I/O (s/t/s)	**Target Ready:** TRDY#, along with IRDY#, indicates whether the 82C881 is able to complete the current data phase of the cycle. TRDY# and IRDY# are both asserted when a data phase is completed. When the 82C881 is acting as the target during read and write cycles, it performs in the following manner: 1. During a read, the 82C881 asserts TRDY# to indicate that it has placed valid data on AD[31:0]. 2. During a write, the 82C881 asserts TRDY# to indicate that is prepared to accept data. TRDY# is an input when the 82C881 is the initiator and an output when it is the target. TRDY# is tristated from the leading edge of RESET# and remains so until driven as either a master or a slave by the 82C881.
STOP#	48	I/O (s/t/s)	**Stop:** STOP# is an output when the 82C881 is the target and an input when it is the initiator. As the target, the 82C881 asserts STOP# to request that the master stop the current cycle. As the master, the assertion of STOP# by a target forces the 82C881 to stop the current cycle. STOP# is tristated from the leading edge of RESET# and remains so until driven by the 82C881 acting as a slave.
DEVSEL#	47	I/O (s/t/s)	**Device Select:** The 82C881 claims a PCI cycle via positive decoding by asserting DEVSEL#. As an output, the 82C881 drives DEVSEL# for two different reasons: 1. If the 82C881 samples IDSEL active in configuration cycles, DEVSEL# is asserted. 2. When the 82C881 decodes an internal address or when it subtractively decodes a cycle, DEVSEL# is asserted When DEVSEL# is an input, it indicates the target's response to an 82C881 master-initiated cycle. DEVSEL# is tristated from the leading edge of RESET# and remains so until driven by the 82C881 acting as a slave.

Signal Name	Pin No.	Pin Type	Signal Description
IDSEL	29	I	**Initialization Device Select:** This signal is the "chip select" during configuration read and write cycles. IDSEL is sampled by the 82C881 during the address phase of a cycle. If IDSEL is found to be active and the bus command is a configuration read or write, the 82C881 claims the cycle with DEVSEL#.
PERR#	49	I/O	**Parity Error:** The 82C8681 uses this line to report data parity errors during any PCI cycle except a Special Cycle.
SERR#	51	I	**System Error:** The 82C881 uses this line to report address parity errors and data parity errors on the Special Cycle command, or any other system error where the result will be catastrophic.
REQ#	15	O	**Bus Request:** REQ# is asserted by the 82C881 to request ownership of the PCI bus.
GNT#	14	I	**Bus Grant:** GNT# is sampled by the 82C881 for an active low assertion, which indicates that it has been granted use of the PCI bus.
CLKRUN#	7	I/O	**Clock Run:** The CLKRUN# function is available on this pin and can be used to reduce chip power consumption during idle periods. It is an I/O sustained tristate signal and follows the PCI 2.1 defined protocol.
INTA#	8	I/O	**Interrupt:** The INTA# pin function is available on this pin and can be used to reduce chip power consumption during idle periods. It is an I/O sustained tristate signal and follows the PCI 2.1 defined protocol.

3.4.2 PHY-Link Interface Signal Set

Signal Name	Pin No.	Pin Type	Signal Description
PHYLREQ	97	O	**Link Request:** This signal is used by the 82C881 to request access to the serial bus and to read or write PHY registers.
PHYCTL[1:0]	92, 93	I/O	**Link-PHY Control Bus:** These two signals are used by both the Link and PHY devices to transfer control information to and from each other.
PHY DATA[7:0]	Refer to Pin Diagram	I/O	**Link-PHY Data Bus:** Data is transferred between the Link and PHY devices over DATA[7:0].
PHYSCLK	95	I	**System Clock:** This clock is generated by the PHY to the Link when PHYLPS is high, and is used to synchronize data transfer between the Link and PHY. Its specified frequency is 49.152MHz.
PHYLPS	99	O	**Link Power Status:** Indicates to the PHY whether or not link activity is required.
PHYLINKON	98	I/O	**Link On:** Wakeup indication.
ISOLATED#	79	I	**"Isolated" Indicator from PHY:** The PHY sets this signal active to indicate that it has electrically isolated itself from the 82C881 Link Controller.

3.4.3 Miscellaneous Signals

Signal Name	Pin No.	Pin Type	Signal Description
TMS	78	I	**Test Mode Select:** The 82C881 logic can be strapped into Test Mode for two types of tests: NAND tree and Boundary Scan. Details are provided in the Test Modes section of this document. This pin must be strapped high for Normal Operation.
TEST#	77	I	**Test Scan Enable:** This pin is used solely for Test Mode operations and should be strapped either high or low for Normal Operation.
RST#	76	I	**PCI Reset:** The PCI interface and standard register set of the 82C881 Link Controller is reset to its default state by this line, which must be held active for at least **??????** PCI clocks to be effective. PCI Power Management registers and Vendor ID / Subsystem ID registers are not reset. All PCI outputs are tri-stated when this line is asserted.
G_RST#	10	I	**Global Reset:** The 82C881 Link Controller is completely reset to its default state by this line, which must be held active for at least **??????** PCI clocks to be effective.
SCL	4	I/O	**Serial Clock:** This clock signal synchronizes data from an industry-standard serial EEPROM.
SDA	5	I/O	**Serial Data:** This data line is used to send commands to and read data from an industry-standard serial EEPROM. If no ROM is used, this line should be tied to ground.
GPIO2-3	2, 3	I/O	**General Purpose I/O:** The GPIO pins can be used by software and hardware for any predefined purpose. They are controlled and monitored as inputs or outputs through PCICFG 4Ch.

4.0 Functional Description

The OPTi 82C881 1394 OHCI Link Controller sits between the PCI bus as host bus on one side, and the P1394-compliant PHY on the other side. On the host bus side, it interfaces to a 33MHz, 32-bit PCI bus.

The 82C881 can initiate read/write transactions on the PCI bus. It supports burst accesses for both read and write transactions with zero wait states.

The logic core supports three register spaces:

- PCI Configuration register set, mapped through standard PCI configuration cycles
- OHCI register set, mapped through either memory or I/O space
- FIFO Configuration register set, also mapped through either memory or I/O space.

When acting as a target of the PCI bus, the 82C881 supports byte aligned accesses for PCI Configuration Registers and quadlet (32-bit word) aligned accesses for the OHCI and FIFO Configuration registers.

Any packet coming from the 1394 serial bus is processed by the PHY, which then moves the data to the 82C881 per the 1394 packet format specification. According to the speed code received, data from data lines is read and converted to 32 bit quadlets. The received packet is passed through several checks such as header and data CRCs, packet type, and packet addressing with respect to physical and asynchronous request filters. The 82C881 converts the incoming packets in 1394 format to OHCI format and puts them on the PCI bus by through a write operation.

When a packet has to be transmitted, the 82C881 gets the required data from the host Memory through the PCI bus in OHCI format by a read operation through the host interface. The 82C881 converts the data to 1394 packet format and passes it on to the PHY, which in turn puts it out on the 1394 serial bus. For transmission, a specific packet is chosen based on its phase, isochronous or asynchronous, and priority among asynchronous packets as defined in 1394 OHCI specification1.0. Depending on the packet to be sent, the 82C881 sends different requests to the PHY through LReq patterns. It transmits the packet when the PHY wins the arbitration. Data will be transmitted on the data lines as appropriate for the PHY speed.

If the 82C881 is the root node, cycle start packets are formed and sent out at every cycle sync event. Bus reset packets are written into the Receive FIFO of the 82C881 whenever there is a bus reset indication.

Software has to program the necessary configuration registers to enable the software context. Once the context is active, the system is ready to receive and transmit packets.

The 82C881 has an interface to an external serial EEPROM. This memory device should contain the GUID of the 82C881 and may contain initialization information. The GUID is read only once after host power reset by an autonomous load operation from the EEPROM, if present, by the 82C881.

4.1 Functional Block Description

A logic block diagram is provided on the following page. The key sub elements shown in the diagram are described below.

4.1.1 PCI Interface

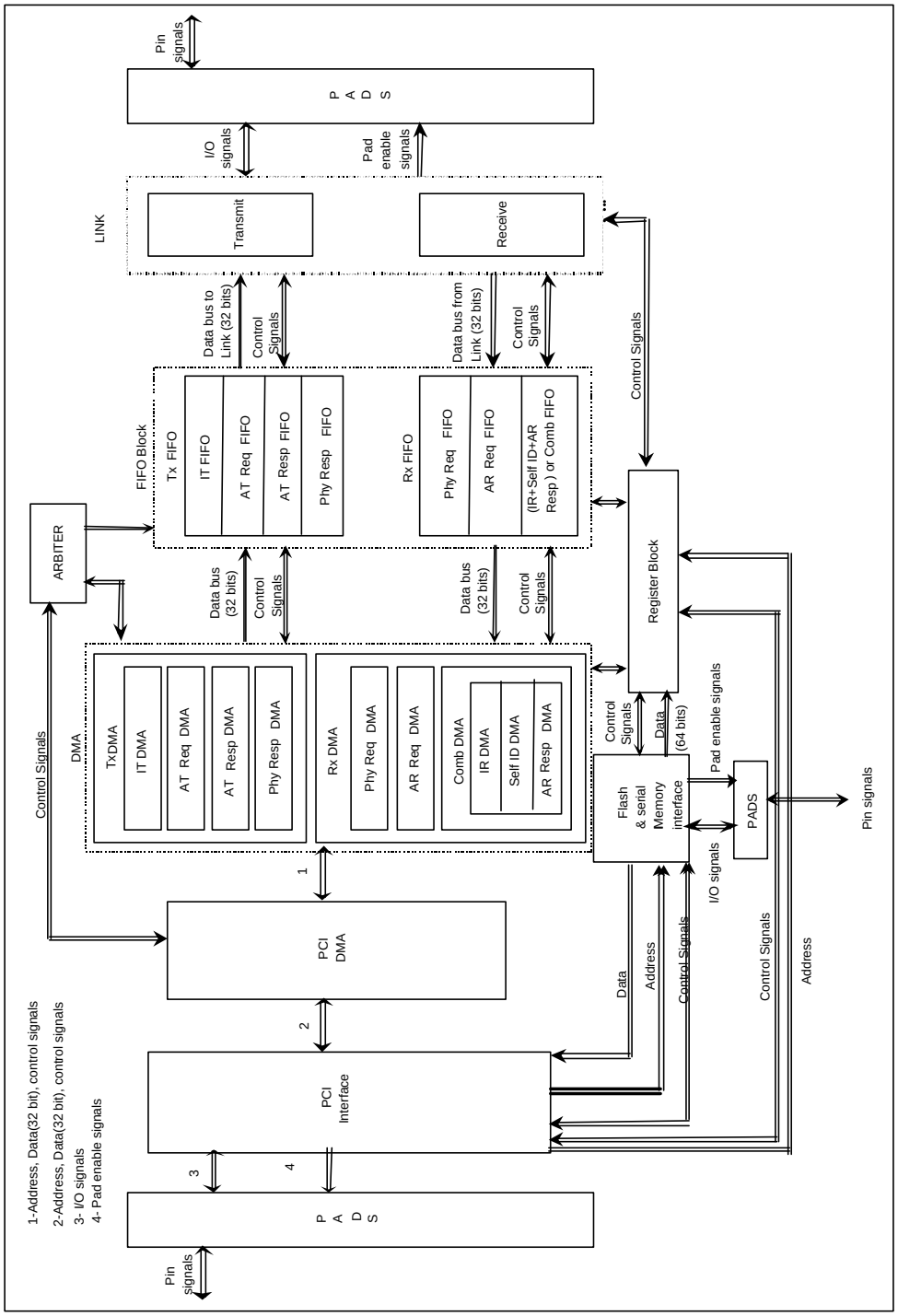
The PCI Interface block is the interface to the PCI bus; it performs bus master functions and can act as a target as well. As a target, it responds to I/O and Memory transactions. Lock transactions are not supported.

Target and master aborts occurring during a transaction are reported to the DMA control block. The PCI interface can perform burst transfers both as a target and as a master.

Data Parity checking and reporting at the PCI bus interface is according to the PCI Local Bus Specifications 2.1. In case of a data parity error during a transaction initiated by the 82C881, the transaction is terminated normally and error is reported to the DMA control block. As a target, if an address parity error is detected, a target abort is generated.

PCI Power Management and PCI CLKRUN# functionality are supported as described elsewhere in this document.

Block Diagram



4.1.1.1 PCI-DMA Block

The PCI-DMA block interfaces the DMA Control block to the PCI Interface block. This arrangement enables the functioning of the DMA blocks to be independent of the Host bus that it is interfaced to. A DMA transaction is a transfer of four or fewer bytes of data.

The PCI-DMA block translates transactions on the DMA interface to transactions on the PCI interface, aligns data to be written out on the PCI bus, merges data fetched, and restarts transactions in case of a disconnect. It also reports errors occurring on the PCI bus to the DMA Control block.

4.1.2 DMA Control Block

The DMA control block supports seven types of DMA. All of the seven DMA controllers can be broadly classified as TxDMA and RxDMA.

4.1.2.1 TxDMA

The TxDMA block is made up of four sub-blocks.

- Isochronous Transmit (IT)
- Asynchronous Transmit Response (AT Resp)
- Asynchronous Transmit Request (AT Req)
- Physical Response (Phy Resp)

Each of these blocks controls the transfer of packets from the host bus to the respective Tx FIFO. Host side software sets up the DMA “contexts,” which are essentially programmed environments used by the respective DMA sub blocks for processing and managing movement of data. Each context consists of a programmed environment and a set of registers. The programmed environment directs the DMA controller in the assembly of packets for transmission.

4.1.2.2 RxDMA

The RxDMA block is made up of three sub-blocks.

- Combined (Comb), consisting of:
  - Isochronous Receive (IR)
  - Self ID
  - Asynchronous Receive Response (AR Resp)
- Asynchronous Receive (AR Req)
- Physical Receive Request (Phy Req)

This block picks out packets from the Rx FIFO, and puts them into the host memory. Host side software sets up contexts to enable logical storage of the received packets. The registers pertaining to these contexts are programmed to achieve this. On receiving a packet, the Rx FIFO gives an indication to the respective DMA context and the DMA context starts processing the packet.

4.1.3 Serial EEPROM Interface

The Serial EEPROM interface is a two-wire industry-standard interface to the Serial EEPROM. This interface is capable of performing reads as well as writes to the Serial EEPROM. Information like GUID, Device ID, Vendor ID, Class Code, Revision ID, Subsystem ID, and Subsystem Vendor ID can be stored in the Serial PROM. These are then loaded from the Serial EEPROM to the corresponding PCICFG registers at each power-on reset.

4.1.4 Arbiter

Since there are seven DMA controllers, the 82C881 prioritizes the DMA controllers. To enable this functionality, the bus request signals from all the DMA contexts are fed into the arbiter, which in turn, based on the priorities specified in 1394 OHCI specification, decides on which DMA controller should get the grant.

4.1.5 Register Block

The Register block includes PCI bus management control and status registers (PCICFG), OHCI Registers as specified in 1394 OHCI specification, and FIFO Configuration registers. The OHCI Registers include control registers for both DMA and the Link, and individual DMA context registers.

All registers (OHCI, PCICFG, and FIFO Configuration) are implemented in the PCICLK domain except for the Isochronous Cycle Timer Register, which is implemented in the PHYSCLK clock domain.

Configuration of the various register blocks is described below.

4.1.5.1 OHCI DMA and Link Registers

The OHCI registers provide the standard 1394-OHCI functionality as noted in the 1394a specification.

These registers should be programmed in the sequence specified in the specification. Programming the registers in a random order will lead to unspecified behavior.

4.1.5.2 FIFO Configuration Registers

The FIFO Configuration registers allow sizing and control of the FIFOs and provide the ability to selectively disable features of the 1394a Link.

Programming of these registers is described elsewhere in this document and is detailed in Appendix B.

4.1.5.3 PCI Interface Registers

The logic implements the required PCI register interface for 1394 OHCI Link controllers.

PCI configuration registers are configured per PCI Local Bus Specification Revision 2.1.

4.1.6 FIFO Block

Primarily the FIFO block is used as a temporary storage place of data between DMA block on one side and the Link block on the other. This is required as data transfer rates on the host bus interface and Link sides are different. Therefore data available on one side, intended for transfer to the other side, is temporarily stored in the FIFO.

The DMA and Link blocks operate at two different frequencies. Therefore the FIFO block also acts as a synchronizer, synchronizing signals from one frequency domain to other.

4.1.6.1 Tx FIFO Block

The Tx FIFO is used to store data for transmission from the transmit DMA block to the Link block, and is logically divided into sub-FIFOs in the following order.

- Isochronous Transmit (IT)
- Asynchronous Transmit Response (AT Resp)
- Asynchronous Transmit Request (AT Req)
- Physical Response (Phy Resp)

The sub-FIFOs are configurable. Software can modify their size, Read Watermark value and Write Watermark value.

Each sub-FIFO has separate controller logic. Handshaking signals are exchanged between the corresponding Transmit-DMA-to-Tx FIFO and Tx FIFO-to-Link before the actual transaction starts.

4.1.6.2 Rx FIFO Block

The Rx FIFO aids in transmitting data from the Link to the Rx DMA block. The Rx FIFO has been logically sub divided into sub-FIFOs in the following order:

- Combined (Comb)
- Isochronous Receive (IR)
- Self ID
- Asynchronous Receive Response (AR Resp)
- Asynchronous Receive (AR Req)
- Physical Receive Request (Phy Req)

Each sub-FIFO has separate controller logic. Handshaking signals are exchanged between the corresponding Receive-DMA-to-Rx FIFO and Rx FIFO-to-Link before the actual transaction starts.

4.1.7 Link Block

The Link block formats the final packets from OHCI to 1394 for transmission, and also checks the received packets from the PHY for errors and translates them back to OHCI format. While transmitting, the Link block reads the packets from different Tx FIFOs, reformats and appends CRC information, and then gives the packets to the PHY. While receiving, after checking the CRC information and format for errors, the Link block writes the received packets to the Rx FIFO.

The Link logic has two main blocks.

- Receive block
- Transmit block.

4.1.7.1 Receive Block

The Receive block deals with the packets in 1394 format, coming from the PHY to the 82C881. It takes data from the serial data lines, converts it into parallel 32-bit data, performs required checks on the data, converts the data to OHCI format and puts it into the appropriate FIFO.

The Receive Block checks the following.

- The tcode, to determine the packet type.
- The destination ID, if the packet is asynchronous. The Receive block will receive the packet only if the destination ID matches with ID of this node.
- The request filter register to determine if this node can accept the request, if the packet is a request packet.
- The physical request filter bit to determine if this node should accept the physical request, if the packet offset address lies within the physical range.
- The retry code, to ensure that the packet is acceptable per the dual phase retry protocol.
- The header and data CRC.
- The data length of the packet, if the packet is a block request packet. If it is greater than the maximum accepted value, the packet is rejected.

The Receive block also generates the ack code to be sent in response to a packet and the event code in response to the packet sent.

4.1.7.2 Transmit Block

The Transmit block deals with the packets intended for transfer from the 82C881 to the PHY. The Transmit block receives data quadlets in OHCI format, converts them to 1394 packet format, adds CRC information and converts the packets for transferring to the PHY.

Following is the list of main functions performed by this block.

- Chooses packet types for transmitting to the PHY, corresponding to whether the state of this period is isochronous or asynchronous.
- Adds header CRC and data CRC information to the packet to be transmitted. The CRC is calculated per clause 6.4, page 171 of IEEE 1394-1995.
- Converts 32-bit parallel data into two, four, or eight bits of serial data depending on the speed of transmission.
- Decodes the different fields of the status transfer from the PHY.
- Sends appropriate patterns to the PHY on the PHYLREQ line.
- Generates PHYLPS signal to the PHY.
- Implements the out-bound single-phase retry logic.

4.2 Serial EEPROM Interface

This interface is capable of performing reads as well as writes onto a 256 byte two wire Serial EEPROM (AT24C01 or similar). Information like GUID, Subsystem ID, Subsystem Vendor ID, some PCI configuration register values and a few OHCI register values can be stored in the Serial PROM. These values will be loaded from the Serial EEPROM after a power on reset if the 82C881 detects its presence through a pull up on the SDA pin. If the EEPROM is not present (SDA sensed LOW after reset), default values are loaded instead.

PCICFG registers initialized from EEPROM data:

1. Subsystem ID
2. Subsystem vendor ID
3. PCI maximum latency, PCI minimum grant

OHCI registers initialized from EEPROM data:

1. Link enhancements control register
2. Host Control register
3. GUID

4.2.1 Read Operations from the Serial EEPROM

Data from the serial EEPROM can be read through a PCI read cycle at any time other than in the power-on read phase. The sequence for reading data out of the EEPROM is as follows

1. Write the desired byte count (up to 8) of the READ operation in PCICFG 53h[6:3] and the eight bit start address on PCICFG 50h[7:0].
2. Set PCICFG 53h[0]=1 to start the EEPROM read state machine.
3. When PCICFG 53h[0] reads back 0, the operation is complete and and PCICFG 53h[2] contains the operation status: 0 for OK, 1 for ERROR.
4. If there is no error, data can be read from PCICFG 54h[31:0] (EEPROM_DATA1), and also PCICFG 58h[31:0] (EEPROM_DATA2) if the read byte count is greater than 4.

If an error occurs during a power-on reset EEPROM configuration read, the default values are loaded.

4.2.2 Write operations to the Serial EEPROM

Data can be written to the serial EEPROM through PCI write cycles after the power-on read is completed.

1. Write the data in registers EEPROM_DATA1 and EEPROM_DATA2.
2. Write the byte count in PCICFG 53h[6:3].

3. Write the start address in PCICFG 50h[7:0].
4. Set PCICFG 53h[1]=1 to start the write operation.
5. When PCICFG 53h[1] reads back 0, the operation is complete and and PCICFG 53h[2] contains the operation status: 0 for OK, 1 for ERROR.

If an error occurs, the software should clear PCICFG 53h[2] before re-trying EEPROM operations.

4.2.3 Related PCICFG Registers

7	6	5	4	3	2	1	0
PCICFG 50h	**EEPROM Start Address Register** Starting address for transfer to or from EEPROM						**Default = 00h**
PCICFG 51h	**Reserved**						**Default = 00h**
PCICFG 52h	**Reserved**						**Default = 00h**
PCICFG 53h	**EEPROM Control Register**						**Default = 00h**
Reserved	Byte count - Must be loaded with value from 1 to 8 prior to issuing read or write command				ERROR 0: operation successful 1: error This bit must be cleared by software	Writing 1 initiates Write. Read: 1: EEPROM write in progress 0: Write complete	Writing 1 initiates Read. Read: 1: EEPROM read in progress 0: Read complete
PCICFG 54h	**EEPROM Data**						**Default = 00h**
	Byte 0						
PCICFG 55h	Byte 1						**Default = 00h**
PCICFG 56h	Byte 2						**Default = 00h**
PCICFG 57h	Byte 3						**Default = 00h**
PCICFG 58h	Byte 4						**Default = 00h**
PCICFG 59h	Byte 5						**Default = 00h**
PCICFG 5Ah	Byte 6						**Default = 00h**
PCICFG 5Bh	Byte 7						**Default = 00h**

4.2.4 Serial EEPROM MAP

Byte Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00	MIN GNT[7:0]							
01	MAX LAT[7:0]							
02	SUBYSTEM VENDOR ID [7:0]							
03	SUBSYSTEM VENDOR ID [15:8]							
04	SUBYSTEM ID [7:0]							
05	SUBSYSTEM ID [15:8]							
06	RSVD	Program Phy 1=Enable	ClkRun 1=Enable	Interface Reset Detection 1=Enable	Multi Speed Concate-nation 1=Enable	Drive Idle Before Transmit 1=Enable	Accelera-tion Control 1=Enable	RSVD
07	1394 GUID HI [7:0]							
08	1394 GUID HI [15:8]							
09	1394 GUID HI [23:16]							
0A	1394 GUID HI [31:24]							
0B	1394 GUID LOI [7:0]							
0C	1394 GUID LO [15:8]							
0D	1394 GUID LO [23:16]							
0E	1394 GUID LO [31:24]							

5.0 Register Descriptions

The 82C881 register set implements the OHCI registers, the FIFO Configuration registers, and the PCI bus configuration registers.

5.1 OHCI and Bus Management Control and Status Registers

The OHCI Registers are memory mapped to Memory Base Address Register 1 (10h) and I/O mapped to I/O Base Address Register 1 (18h) of PCI Configuration space.

The OHCI Registers and bus management control and status registers are implemented per the 1394 OHCI Specification, Revision 1.0. For the register definitions, bit-mapping into the registers and access requirements for writing and reading the registers, refer to the 1394 OHCI Specification, Revision 1.0.

5.2 FIFO Configuration Registers

As the FIFOs are configurable, the configuration can be changed by programming the FIFO Configuration registers. For each sub-FIFO, there is a 32 bit Register containing the configuration value.

The FIFO Configuration registers are accessed as offsets from a base address. The access can be either through memory or I/O space depending on the PCI base address configuration selected: a memory base address can be programmed in PCICFG 14h, while an I/O base address can be programmed in PCICFG 1Ch.

Note: After programming new FIFO Configuration register values, the FIFO Config Enable bit at OFST 1Ch[0] must be set to actually load the changed values.

There are eight registers in the FIFO Configuration register set:

- Isochronous Tx Configuration
- Asynchronous Tx Response Configuration
- Asynchronous Tx Request Configuration
- Physical Tx Configuration
- Miscellaneous Rx Configuration
- Asynchronous Rx Response Configuration
- Physical Rx Configuration
- Miscellaneous.

The first seven registers hold the FIFO Configuration values, while the Miscellaneous register is used to initiate loading of the values programmed into the first seven registers and to selectively enable/disable specific P1394a features.

5.2.1 Tx FIFO-Related Registers

7	6	5	4	3	2	1	0
OFST 00h			**IsochronousTx Configuration Register**				**Default = 40h**
			Holds configuration data for Isochronous Transmit sub-FIFO				
			Byte 0				
			Write Watermark bits [7:0] - If the number of empty quadlet space is equal to or greater than this value, the corresponding DMA controller can start a burst write into the FIFO.				
OFST 01h			**Byte 1**				**Default = 80h**
						Write Watermark bits [9:8]	
			Read Watermark bits [5:0] - If the number of quadlets in the sub FIFO is greater than or equal to this value, the Link can start reading from that FIFO.				
OFST 02h			**Byte 2**				**Default = 00h**
				Read Watermark bits [9:6]			
			Sub-FIFO Size bits [3:0]				
OFST 03h			**Byte 3**				**Default = 10h**
				Sub-FIFO Size bits [9:4]			
Reserved							
OFST 04h			**Asynchronous Tx Response Configuration Register**				**Default = 40h**
			Holds configuration data for AT Response sub-FIFO				
			Byte 0				
			Write Watermark bits [7:0]				
OFST 05h			**Byte 1**				**Default = 00h**
						Write Watermark bits [9:8]	
			Read Watermark bits [5:0]				
OFST 06h			**Byte 2**				**Default = 02h**
				Read Watermark bits [9:6]			
			Sub-FIFO Size bits [3:0]				
OFST 07h			**Byte 3**				**Default = 10h**
				Sub-FIFO Size bits [9:4]			
Reserved							
OFST 08h			**Asynchronous Tx Request Configuration Register**				**Default = 40h**
			Holds configuration data for AT Request sub FIFO				
			Byte 0				
			Write Watermark bits [7:0]				
OFST 09h			**Byte 1**				**Default = 00h**
						Write Watermark bits [9:8]	
			Read Watermark bits [5:0]				
OFST 0Ah			**Byte 2**				**Default = 02h**
				Read Watermark bits [9:6]			
			Sub-FIFO Size bits [3:0]				
OFST 0Bh			**Byte 3**				**Default = 10h**
				Sub-FIFO Size bits [9:4]			
Reserved							
OFST 0Ch			**Physical Tx Configuration Register**				**Default = 40h**
			Holds configuration data for Physical Transmit sub FIFO				
			Byte 0				
			Write Watermark bits [7:0]				
OFST 0Dh			**Byte 1**				**Default = 00h**
						Write Watermark bits [9:8]	
			Read Watermark bits [5:0]				
OFST 0Eh			**Byte 2**				**Default = 02h**
				Read Watermark bits [9:6]			
			Sub-FIFO Size bits [3:0]				
OFST 0Fh			**Byte 3**				**Default = 10h**
				Sub-FIFO Size bits [9:4]			
Reserved							

5.2.2 Rx FIFO-Related Registers

7	6	5	4	3	2	1	0
OFST 10h			**Miscellaneous Rx Configuration Register**				**Default = 00h**
Holds configuration data for Combined sub-FIFO; contains data quadlets for Iso Receive, AR Response and Self ID packets.							
Byte 0							
Reserved							
OFST 11h			**Byte 1**				**Default = 00h**
Read Watermark bits [5:0] - If the number of quadlets in the sub-FIFO is greater than or equal to this value, the corresponding DMA controller can start a burst read of the data from the sub FIFO.						Reserved	
OFST 12h			**Byte 2**				**Default = 01h**
Sub-FIFO Size bits [3:0]				Read Watermark bits [9:6]			
OFST 13h			**Byte 3**				**Default = 20h**
Reserved		Sub-FIFO Size bits [9:4]					
OFST 14h			**Asynchronous Rx Request Configuration Register**				**Default = 00h**
Holds configuration Data for AR Request sub-FIFO.							
Byte 0							
Reserved							
OFST 15h			**Byte 1**				**Default = 00h**
Read Watermark bits [5:0]						Reserved	
OFST 16h			**Byte 2**				**Default = 01h**
Sub-FIFO Size bits [3:0]				Read Watermark bits [9:6]			
OFST 17h			**Byte 3**				**Default = 10h**
Reserved		Sub-FIFO Size bits [9:4]					
OFST 18h			**Physical Rx Request Configuration Register**				**Default = 00h**
Holds configuration Data for Physical Receive Request sub-FIFO.							
Byte 0							
Reserved							
OFST 19h			**Byte 1**				**Default = 00h**
Read Watermark bits [5:0]						Reserved	
OFST 1Ah			**Byte 2**				**Default = 01h**
Sub-FIFO Size bits [3:0]				Read Watermark bits [9:6]			
OFST 1Bh			**Byte 3**				**Default = 10h**
Reserved		Sub-FIFO Size bits [9:4]					

Table 2. Miscellaneous Register

7	6	5	4	3	2	1	0
OFST 1Ch			**Miscellaneous Register**				**Default = 00h**
Bits in this register are used for effecting FIFO configuration and selective enabling of P1394a features (Note 1).							
Byte 0							
CLKRUN# 0=Enable 1=Disable If CLKRUN# is asserted, PCICLK is not allowed to be stopped for power saving mode.	FIFOTag Bit (RO) When in Debug Mode, tag bit readout from the Transmit /Receive FIFO is written to this bit.	Debug Mode 0=Disable 1=Enable When Debug Mode is enabled, the Receive and Transmit FIFOs can be read/written.	Detect Interface Reset in case of 1394a PHY 0=Enable (if 1394a features are enabled) 1=Disable	Drive Idle Before Xmit on Link-PHY interface 0=Enable (if 1394a features are enabled) 1=Disable	Acceleration Control 0=Enable (if 1394a features are enabled) 1=Disable	Multi-Speed Concatena-tion 0=Enable (if 1394a features are enabled) 1=Disable	FIFO Config Enable (Note 2) 0=Disable 1=Enable
Note 1: All bits in this register default to 1 at Hard Reset and Soft Reset; they are not affected by bus reset. **Note 2:** Software must set this bit to 1 to load register changes to the FIFO Configuration registers. The 82C881 will clear this bit after the operation is complete.							
OFST 1Dh				**Byte 1**			**Default = 00h**
Reserved							
OFST 1Eh				**Byte 2**			**Default = 00h**
Reserved							
OFST 1Fh				**Byte 3**			**Default = 00h**
Reserved							

5.3 PCI Configuration Registers

The PCI Configuration space registers implemented in the 82C881 are listed in the following table. They are not affected by bus reset/soft reset.

The configuration space of the PCI 1394 OHCI controller is accessed through Mechanism #1 as Device #X (Device # depends on which AD line is connected to the IDSEL input), Function #0, hereafter referred to as PCICFG.

5.3.1 PCI Configuration Space (PCICFG 00h to 3Fh)

7	6	5	4	3	2	1	0
PCICFG 00h **PCICFG 01h**			**Vendor Identification Register (RO)**				**Default = 45h** **Default = 10h**
PCICFG 02h **PCICFG 03h**			**Device Identification Register (RO)**				**Default = 81h** **Default = C8h**
PCICFG 04h			**Command Register - Byte 0**				**Default = 00h**
Wait cycle control: Core does not need to insert a wait state between address and data on the AD lines. This bit is always 0.	PERR# (response) detection enable bit: 0 = PERR# not asserted 1 = Core asserts PERR# when receiving data agent and data parity error detected.	VGA palette snooping: This bit is always 0.	Postable memory write command: Not used when core is a master. This bit is always 0.	Special Cycles: Core does not run Special Cycles on PCI. This bit is always 0.	Core can run PCI master cycles: 0 = Disable 1 = Enable	Core responds as a target to memory cycles. 0 = Disable 1 = Enable	Core responds as a target to I/O cycles: 0 = Disable 1 = Enable
PCICFG 05h			**Command Register – Byte 1**				**Default = 00h**
Reserved: These bits are always 0.						Back-to-back enable: Core only acts as a master to a single device, so this functionality is not needed. This bit is always 0.	SERR# (response) detection enable bit: 0 = SERR# not asserted 1 = Core can assert SERR#
PCICFG 06h			**Status Register – Byte 0**				**Default = 80h**
Fast back-to-back capability: Core does not supports fast back-to-back transactions when transactions are not to same agent. This bit is always 0.	Reserved: These bits are always 0.		PCI Power Management Capability (RO) 1 = Yes (always)	Reserved: These bits are always 0.			

7	6	5	4	3	2	1	0
PCICFG 07h			**Status Register - Byte 1**				**Default = 02h**
Detected parity error: This bit is set to 1 whenever the core detects a parity error, even if PCICFG 04h[6] is disabled. Write 1 to clear.	SERR# status: This bit is set to 1 whenever the core detects a PCI address parity error. Write 1 to clear.	Received master abort status: Set to 1 when the core, acting as a PCI master, aborts a PCI bus memory cycle. Write 1 to clear.	Received target abort status: This bit is set to 1 when a core generated PCI cycle (core is the PCI master) is aborted by a PCI target. Write 1 to clear.	Signaled target abort status: This bit is set to 1 when the core signals target abort. Write 1 to clear.	DEVSEL timing (RO): Indicates DEVSEL# timing when performing a positive decode. Since DEVSEL# is asserted to meet the medium timing, these bits are encoded as 01.		Data parity reported: Set to 1 if PCICFG 04h[6] is set and the core detects PERR# asserted while acting as PCI master (whether PERR# was driven by core or not.)
PCICFG 08h			**Revision Identification Register (RO)**				**Default = 00h**
PCICFG 09h **PCICFG 0Ah** **PCICFG 0Bh**			**Class Code Register (RO)** PCI-1394 OHCI Bridge				**Default = 10h** **Default = 00h** **Default = 0Ch**
PCICFG 0Ch			**Cache Line Size Register**				**Default = 00h**
PCICFG 0Dh			**Master Latency Timer Register**				**Default = 00h**
PCICFG 0Eh			**Header Type Register (RO)**				**Default = 00h**
PCICFG 0Fh			**Reserved**				**Default = 00h**
PCICFG 10h-13h			**OHCI Register Set Memory Base Address**				**Default = 00h**
This register maps the OHCI Register Set into system memory space. Bits [10:1] are always 0, requesting a 2048-byte range.							
PCICFG 14h-17h			**FIFO Configuration Register Set Memory Base Address**				**Default = 00h**
This register maps the FIFO Configuration Register Set into system memory space. Bits [4:1] are always 0, requesting a 32-byte range.							
Memory base address registers identify the base address of a contiguous memory space in main memory. Software will write all 1s to this register, then read back the value to determine how big of a memory space is requested. After allocating the requested memory, software will write the upper bytes with the base address. Bits [31:0] correspond to: 10h = [7:0], 11h = [15:8], 12h = [23:16], 13h = [31:24]. - Bit [0] – Indicates that the operational registers are mapped into memory space. Always = 0. - Bits [2:1] – Indicates that the base register is 32 bits wide and can be placed anywhere in 32-bit memory space. Always = 0. - Bit [3] – Indicates no support for prefetchable memory. Always = 0. - Bits [31:4] – Software writes the value of the memory base address to these bits.							
PCICFG 18h-1Bh			**OHCI Register Set I/O Base Address**				**Default = 00h**
This register maps the OHCI Register Set into system I/O space. Bits [10:1] are always 0.							
PCICFG 1Ch-1Fh			**FIFO Configuration Register Set I/O Base Address**				**Default = 00h**
This register maps the FIFO Configuration Register Set into system I/O space. Bits [4:1] are always 0, requesting a 2048-byte range.							
I/O base address registers identify the base address of a contiguous range in system I/O space. Software will write all 1s to this register, then read back the value to determine how big of an I/O range is requested. After allocating the requested space, software will write the upper bytes with the base address. Bits [31:0] correspond to: 10h = [7:0], 11h = [15:8], 12h = [23:16], 13h = [31:24]. - Bit [0] – Indicates that the operational registers are mapped into I/O space. Always = 1. - Bit [1] – Always = 0. - Bits [15:2] – Software writes the I/O base address to these bits. - Bits [31:16] – Always = 0.							
PCICFG 20h-23h			**Reserved**				**Default = 00h**
PCICFG 24h-27h			**Debug Register Memory Base Address** Refer to Appendix B for details				**Default = 00h**

7	6	5	4	3	2	1	0
PCICFG 28h-2Bh			**Reserved**				**Default = 00h**
PCICFG 2Ch-2Dh			**Subsystem Vendor Register (RO)**				**Default = 0000h**
Subsystem Vendor – This register value is loaded from the Serial EEPROM at reset time.							
PCICFG 2Eh-2Fh			**Subsystem ID Register (RO)**				**Default = 0000h**
Subsystem ID – This register value is loaded from the Serial EEPROM at reset time.							
PCICFG 30h-33h			**Reserved**				**Default = 00h**
PCICFG 34h			**Capabilities Pointer (RO)**				**Default = 44h**
Indicates the offset in the PCICFG space for the location of the first item in the Capabilities Linked List. This location is PCICFG 44h.							
PCICFG 35h-3Bh			**Reserved**				**Default = 00h**
PCICFG 3Ch			**Interrupt Line Register**				**Default = FFh**
This register identifies which of the system interrupt controllers the deviceÕs interrupt pin is connected to. The value of this register is used by device drivers and has no direct meaning to the core.							
PCICFG 3Dh			**Interrupt Pin Register (RO)**				**Default = 01h**
This register identifies which interrupt pin a device uses. Since the core uses INTA#, this value is set to 01h.							
PCICFG 3Eh			**Minimum Grant Register (RO)**				**Default = 01h**
PCICFG 3Fh			**Maximum Latency Register (RO)**				**Default = 05h**
PCICFG 40h			**OHCI-Specific Register**				**Default = 00h**
							PCI Global Swap Indicates whether data written to and read from the DMA block should be byte-swapped. 0 = Disable 1 = Enable
PCICFG 41h-43h			**Reserved**				**Default = 00h**

5.4 Power Management Registers

7	6	5	4	3	2	1	0
PCICFG 44h			**CAP_ID Register (RO)**				**Default = 01h**
This register returns a value of 01h to identify the Capabilities list item as being the PCI Power Management Register Block.							
PCICFG 45h			**Next_Item_Ptr Register (RO)**				**Default = 00h**
This register returns a value of 00h to indicate that there are no additional items in the Capabilities list.							
PCICFG 46h			**PMC Register (RO) - Byte 0**				**Default = 0Ah**
Reserved		Device Specific Initialization (DSI): 0 = DSI is not required	Reserved	PME Clock: 1 = PME# clock required to generate PME#	Version: 010 = This function complies with Revision 1.1 of the PCI Power Management Interface Specification.		

7	6	5	4	3	2	1	0
PCICFG 47h			**PMC Register (RO) - Byte 1**				**Default = 40h**
PME Support: 01000 = The Link controller supports PME# generation from $D3_{hot.}$					D2 device state support: 0 = No	D1 device state support: 0 = No	Reserved
PCICFG 48h			**PMCSR_BSE (RO)**				**Default = 00h**
PCICFG 49h			**Data Register (RO)**				**Default = 00h**
PCICFG 4Ah			**PMCSR Register - Byte 0**				**Default = 00h**
Reserved						PowerState (R/W): 00 = D0 01 = D1 (Not Supported) 10 = D2 (Not Supported) 11 = D3hot This field is used both to determine the current power state and to set a new power state. Unsupported states will be ignored when written to.	
PCICFG 4Bh			**PMCSR Register - Byte 1**				**Default = 00h**
PME Status (R/W): This bit is set when a PME event is generated. Write 1 to clear.	Data_Scale (RO): 00 = Data register is not supported		Data_Select (RO): 0000 = Data register is not supported				PME_En (R/W): 0 = PME# assertion is disabled 1 = PME# is asserted when PME_ Status = 1
PCICFG 4Ch			**Miscellaneous Control Register**				**Default = 00h**
Reserved	Reserved	GPIO3 Data Read: Input Mode – Returns value on GPIO3 pin Output Mode – Returns last value written Write: Input Mode – No function Output Mode – Sets value on GPIO3 pin	GPIO3 Mode 0 = Output 1 = Input	GPIO2 Data Read: Input Mode – Returns value on GPIO2 pin Output Mode – Returns last value written Write: Input Mode – No function Output Mode – Sets value on GPIO2 pin	GPIO2 Mode 0 = Output 1 = Input	CLKRUN Support "Enable" indicates that the chip will allow PCICLK to stop per the CLKRUN# protocol 0 = Disable 1 = Enable	Reserved
PCICFG 4Dh - 4Fh			**Reserved**				**Default = 00h**

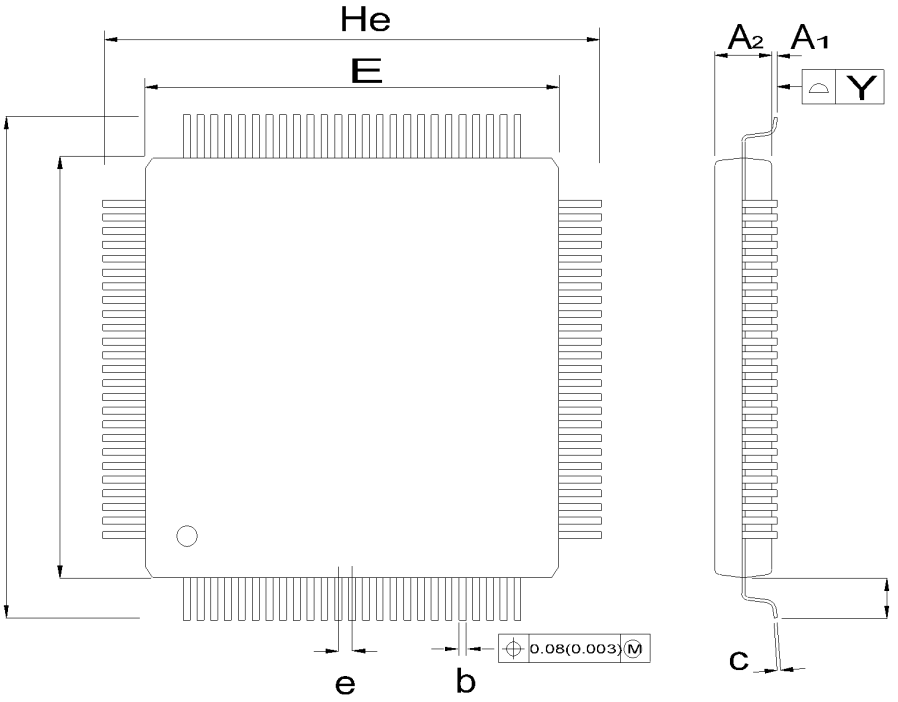
7	6	5	4	3	2	1	0
PCICFG 50h	**EEPROM Start Address Register** Starting address for transfer to or from EEPROM						**Default = 00h**
PCICFG 51h	**Reserved**						**Default = 00h**
PCICFG 52h	**Reserved**						**Default = 00h**
PCICFG 53h	**EEPROM Control Register**						**Default = 00h**
Reserved	Byte count - Must be loaded with value from 1 to 8 prior to issuing read or write command				ERROR 0: operation successful 1: error This bit must be cleared by software	Writing 1 initiates Write. Read: 1: EEPROM write in progress 0: Write complete	Writing 1 initiates Read. Read: 1: EEPROM read in progress 0: Read complete
PCICFG 54h	**EEPROM Data**						**Default = 00h**
	Byte 0						
PCICFG 55h	Byte 1						**Default = 00h**
PCICFG 56h	Byte 2						**Default = 00h**
PCICFG 57h	Byte 3						**Default = 00h**
PCICFG 58h	Byte 4						**Default = 00h**
PCICFG 59h	Byte 5						**Default = 00h**
PCICFG 5Ah	Byte 6						**Default = 00h**
PCICFG 5Bh	Byte 7						**Default = 00h**

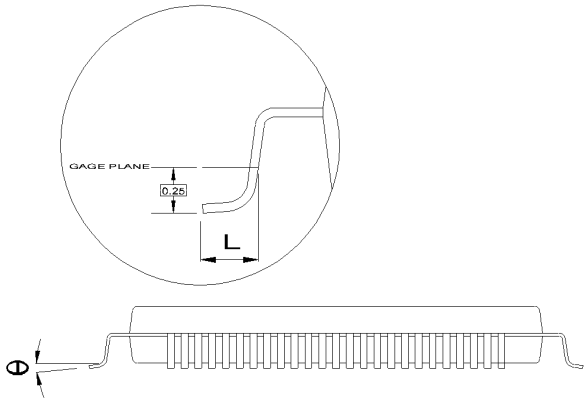
5.5 Timing Information

The timing relations on the 82C881-PCI bus interface are per PCI Local Bus Specification Revision 2.1. The timing relations on the 82C881-PHY interface confirm to timings of Link-PHY interface of chapter 5, P1394a Draft 2.0 specification.

6.0 Electrical Ratings

7.0 Mechanical Package





SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.007	0.009	0.011
c	0.090		0.200	0.004		0.008
D	13.90	14.00	14.10	0.547	0.551	0.555
E	13.90	14.00	14.10	0.547	0.551	0.555
e		0.50			0.020	
Hd	15.90	16.00	16.10	0.626	0.630	0.634
He	15.90	16.00	16.10	0.626	0.630	0.634
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
Y			0.08			0.003
θ	0		7	0		7

Dwg. No.:	AS100TQFP-001	
Dwg. Rev.:	A0	Unit: MM / INCH

8.0 Test Modes

TMS must be high for Normal Operation. To select Test Mode, set TMS low and assert RST#.

In Test Mode, TEST# operates as follows.

0 = Scan Mode

1 = NAND Tree Test Mode

Once Scan Mode has been selected, TEST# is used as the scanEnable signal, selecting between shift and capture cycle during scan operation.

TMS	TEST#	Operation
1	X	Normal mode
0	1	Nand Tree mode (Shift cycle in scan mode)
0	0	Scan Mode(Capture cycle in scan mode)

Signal Name	Pin No.	Signal Type In Scan Mode	Signal Description
TMS+CYCLEIN	78	I	Test Mode Select
CYCLEOUT# + TEST#	77	I	Test Scan Enable
PCICLK + TSCLK_PCI	12	I	Scan Test CLK for 3 PCI Trunks
GPIO2+TSCAN_IN2	2	I	Scan Test Input for PCI Trunk1
GPIO3+TSCAN_IN3	3	I	Scan Test Input for PCI Trunk2
IDSEL+TSCAN_IN4	29	I	Scan Test Input for PCI Trunk3
REQ#+TSCAN_OUT2	15	O	Scan Test Output for PCI Trunk1
PERR#+TSCAN_OUT4	49	O	Scan test Output for PCI Trunk3
SERR#+TSCAN_OUT3	51	O	Scan test output for PCI Trunk2
PHYSCLK+TSCLK_PHY	95	I	Scan Test CLK for PHY Trunk
LINKON+TSCAN_IN1	98	I	Scan Test input for PHY trunk
LINKREQ+TSCANOUT1	97	O	Scan Test Output for PHY Trunk

9.0 Appendix A

9.1 Acronyms and Definitions

IEEE	Institute of Electrical & Electronics Engineers
PCI	Peripheral Component Interconnect.
Mbps	Mega bits per second
PHY	Physical layer device.
CRC	Cyclic Redundancy Check
GUID	Global Unique Identification.
OHCI	Open Host Controller Interface.
FIFO	First In First Out Memory.
DPRAM	Dual Port Random Access Memory.
ROM	Read Only Memory.
CSR	Control and Status Registers.
LINK	1394 OHCI Link IC
IUT	Implementation Under Test

9.2 References

- 1394 Open Host Controller Interface Specification Release 1.00
- IEEE Std 1394-1995, Standard for High Performance Serial Bus.
- P1394a draft 2.0 Standard for a High performance Serial Bus.
- PCI Local Bus Specification Revision 2.1
- IEEE Std 1212, 1994 edition.
- PCI Mobile Design Guide 1.1.
- PCI Power Management Specification 1.0.

10.0 APPENDIX B

10.1 FIFO Programming

FIFO Configuration Registers should be programmed in the following sequence.

- Before setting bit 0 of miscReg, check if its value is equal to zero.
- Write the New Configuration value into the individual Configuration registers.
- Set the bit 0 of miscReg (refer sec 2.4.2)
- As long as the Configuration process is going on, that bit will be one. If that bit becomes 0, it means that Configuration process has ended. (writing into FIFO and reading from FIFO can be done only if the configuration process has ended.)

10.1.1 Programming Notes

Points to be noted while configuring the FIFO Configuration registers:

- By setting FIFOConfigRst (bit 0 of miscReg), FIFO configuration is changed for all the FIFOs according to the FIFO Configuration Register values present. Therefore, unwanted data should not be present in any of the FIFO Configuration Register when the FIFOConfigRst bit gets set. If, for any FIFO, old configuration is to be retained while configuring another FIFO, the old configuration data should be present in that FIFO Configuration Register.
- For each field of Configuration (size, readWatermark, and writeWatermark) 10 bits are allotted. But currently we are using eight bits (The two extra bits are provided so that if the RAM size is increased then a maximum DPRAM of 1K can be supported with the current set-up). So for Configuration, upper 8 bits should be used and lower two bits should be made zero0.
- e.g. If a FIFO size of 64 quadlet has to be programmed, the following bit pattern has to be programmed in bits 29 to 20. Bit 21 and 20 are not required for current implementation of 8-bit size of each field. Hence they are kept as 0.

Bit Position	29	28	27	26	25	24	23	22	21	20
Value	0	1	0	0	0	0	0	0	0	0

- All unused bits should be set to zero for the Configuration Register.
- For configuring mSpdConcatEn_n (bit 1 of miscReg), accControlEn_n (bit 2 of miscReg) drvIdlB4txEn_n (bit 3 of miscReg) software should write into the corresponding bit. The miscReg is written as a 32-bit data bus. Thus care should be taken not to write any unwanted data in the fields, which are of no current writing interest.

10.1.2 Important User Defined Values

Other important values to be programmed:

- The OHCI and FIFO Configuration register sets consist of 32-bit registers. This bus width is defined as: CONFIG_BUS_WIDTH
- The size of each RAM module is 256 quadlets. Hence the address bus width is eight. It should be changed if the DPRAM size changes.
- For Rx FIFO, it is RXFIFO_ADDRBUS_WIDTH.
- For Tx FIFO, it is TXFIFO_ADDRBUS_WIDTH

10.1.3 Current Default Configuration Values

Current Default Configuration Values and the reason for choosing them:

- Each DPRAM is of 256 quadlets, as obtained from the Vendor.
- In Tx FIFO, each sub-FIFO Size is 64 quadlets.
- In Rx FIFO, Combined FIFO is of 128 quadlets. This is because it contains packets for multiple packet types. Other two sub-FIFO's have 64 quadlets each.
- In Tx FIFO, WriteWatermark Value is chosen as 16 quadlets as PCI can support burst up to a maximum of 16 quadlets.
- In Tx FIFO, for IT FIFO, readWatermark value is eight quadlets.
- For other sub-FIFOs in Tx FIFO, readWatermark value is 32 quadlets. This is kept high to avoid frequent hitting of underrun as the Link operates at a higher frequency than the DMA controllers.
- In Rx FIFO, readWatermark value is 16 quadlets for each sub-FIFO. This is also to support burst on the PCI bus, so that the FIFO does not become empty while the DMA controller is continuing a burst transfer onto the PCI bus.

In the above explanation, all values are in quadlet, unless otherwise mentioned.

10.2 Debug Features

10.2.1 Reading/Writing Tx FIFO and Rx FIFO Bypassing DMA / Link Logic

10.2.1.1 Design

The FIFOs are memory mapped to Base Address 5. The access is quadlet aligned. So PCI Address bus bit [1:0] will always be 2'b00. bit[5:2] is used to decode the FIFO accessed. Of which bit[4:2] is directly mapped with BUS_GNT (as defined in commonDefine.h) for all the FIFOs, both for reading and writing.

PCI Data bus is 32 bit and FIFO Data is 33 bits (32 bit Data + 1 bit Tag). bit[5] of PCI address bus is used to write the value of tag bit. If bit[5] is 1, then tag bit corresponding to that data will be 1 and vice versa.

For reading, tag bits are to be read from the FIFOs. So along with Data, tag bit is to be read. Here the 32 bit data is read out as the pciDataOut in one read cycle. The corr. tag bit is simultaneously written in the bit[6] of miscReg in ohciRegTop. So in the first read cycle, read access is done to the FIFO Read and in the next read cycle it is to be done to the miscReg to get the tag bit.

Arbitration Request signal to Link Tx (for Tx FIFO) or RxDMA (for Rx FIFO) is not asserted when the chip is in debug mode.

During debug mode, read / write can only be done through PCI Cycles.

10.2.1.2 Data Format

For all the FIFOs, writing and reading of Tag bit is allowed. But to make a sensible data to be written in the FIFO, so that it can be read as a valid packet in normal mode, the packet format per DMA / Link is to be written.

The following is the format in which the Rx FIFO receives data (32 bit) from link Rx.

10.2.1.2.1 Rx FIFO Receive Data

START DELIMITER
HEADER
DATA
END DELIMITER

10.2.1.2.2 Isochronous Receive Packet Format

START DELIMITER
HEADER
DATA
TRAILER
END DELIMITER

10.2.1.2.3 Other Receive Packet Formats

- **Receive packet start delimiter format**

Field Name	Width	Bit Position	Description
endDelimiter	1	0	0 – Start Delimiter, 1 – End Delimiter
pktType	2	2:1	Packet type (for combined DMA) 00 – Isochronous receive 01 – Asynchronous receive 10 – Self ID receive 11 – Invalid
Reserved	2	4:3	Currently not used
phyPktType	2	6:5	Physical Packet type 00 – Invalid 01 – physical Range 10 – Bus Management resource registers (which are accessible by quadlet Read and quadlet lock transactions) 11 – config ROM header, Bus ID, Bus options and GUID
timeStamp	16	22:7	The time at which this packet was received by the OHCI_LINK.
speed	3	25:23	This is the speed at which the packet was received by the PHY. 000 – 100Mbps, 001 – 200 Mbps, 010 – 400 Mbps
busReset	1	26	This bit if set implies that the packet associated with this delimiter is a bus reset packet
postedWrPkt	1	27	Thid bit is set if the packet associated with this start delimiter is a posted write packet.

Field Name	Width	Bit Position	Description
Reserved	5	31:28	Currently not used

- **Receive packet end delimiter packet**

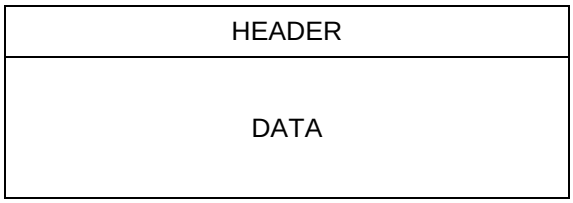
Field Name	Width	Bit Position	Description
endDelimiter	1	0	0 – Start Delimiter, 1 – End Delimiter
error	1	1	error bit, when asserted indicates that the packet had a CRC or data length match error
selfIdIncomplete	1	2	Self ID phase incomplete , This bit if set indicates that the self ID phase is incomplete
Reserved	3	5:3	Currently not used
ackCode	4	9:6	This field gives the ack Codes as defined in Table 3-2 (Packet event codes) of the 1394 OHCI specification 1.0.
FIFOFull	1	10	This field is set to 1 when the FIFO becomes full while receiving a packet.
Reserved	22	31:10	Currently not used

The following is the format in which the Tx FIFO receives data (32 bit) from TxDMA.

10.2.1.2.4 Isochronous Receive Packet Format

START DELIMITER
HEADER
DATA

10.2.1.2.5 Isochronous Transmit Packet Format



10.2.1.2.6 Other Transmit Packet Format

- **Transmit packet start delimiter format**

Field Name	Width	Bit Position	Description
delimiter	1	0	0 – start delimiter 1 – end delimiter
Reserved	7	7:1	Currently not used
C_loadCycCount	13	20:8	Cycle count field of the Isochronous cycle timer register (refer to OHCI specification 1.0 chapter 5,section 5-12)
C_loadCycSecs	3	23:21	Lower 3 bits of the Cycle seconds field of the Isochronous cycle timer register (refer to OHCI specification 1.0 chapter 5,section 5-12)
C_CycMatchEnable	1	24	CycleMatchEnable bit of the IT context control register
Reserved	7	31:25	Currently not used

Header and data are as specified in 1394 OHCI specification 1.0

A tag bit is also specified along with each quadlet of the packet. The tag bit distinguishes between a data and delimiter.

[tag=0 (data), tag=1 (delimiter)]

Special consideration should be taken for Tx FIFOs -

As far as Tx FIFO is concerned, except for IT FIFO, no other FIFO writes Start / End Delimiter (Quadlets corr. to Tag bit = 1'b1) into the DPRAM.

IT FIFO only writes Start Delimiter in the FIFO.

Also except for IT FIFO, in all other Tx FIFOs, at one time only one packet is written into. And the write packet count is reset after seeing the ack for that packet.

Also, while transmitting non-IT packets, although end Delimiter is not written in the FIFO, it is sensed by the FIFO Controller logic to increment the packet count. But this cannot be done in debug mode. Hence while writing in debug mode, intended to be read in normal mode, then the packet size should be greater than the read watermark of the non-IT Tx FIFOs, so that linkArbRequest get triggered after coming back to normal mode.

When arbitrary data quadlets are written into non-IT Tx FIFOs in debug mode, and then read back also in debug mode, then also write packet count may mismatch with respect to read packet count. Hence *ArbReq flag can be triggered when returned to normal mode, even if only 1 quadlet data is written later in the corresponding FIFO. To avoid this, an ack has to be sent for the packet, even if it is not sent out.

10.2.2 Implementation

For Debugging, the chip has to be in debug mode. For this, bit 5 in miscReg of ohciRegVebdor.v is defined. It has to be asserted in order for the chip to be in debug mode.

Base Address Register 5 is assigned to be used for mapping the FIFOs. The identification of this base address Register is to be done during PCI configuration cycle. The address bit [31:6] is used for assigning the memory. Address bit [5:0] is for selecting the FIFO. Base address Reg 5 is memory mapped in the PCI Address space.

10.2.2.1 Writing onto Tx FIFO

For writing into the Tx FIFO, a PCI write cycle has to be started in the debug mode. It will generate a signal called "FIFOWrAcc_c" and assert the correct "grantType" on the "targetAddr" line, to which "noBurst" and "targetReady" will be generated to PCI. Seeing this, it will give the data validated by "targetValidData" signal. "targetValidData" will be MUX-ed to the "wr" pulse of Tx FIFO.

10.2.2.2 Writing onto Rx FIFO

For writing into the Rx FIFO, the implementation is similar as above. But Rx FIFO is written in link clock domain. Hence all the signals are synchronised from host clock domain to link clock domain.

10.2.2.3 Reading from Rx FIFO

For reading from the Rx FIFO, a PCI read cycle has to be started in the debug mode. It will generate a signal called "FIFORdAcc_c" and assert the correct "grantType" on the "targetAddr" line, to which "noBurst" and "targetReady" will be generated, qualifying the dataOut on the PCI Data bus from the Rx FIFO. "noBurst" is considered as "rdIncr" in the debug mode.

10.2.2.4 Reading from Tx FIFO

For reading from Tx FIFO, the scheme is similar as above. But Tx FIFO is read in link clock domain. Hence all signals are synchronised from host clock domain to link clock domain and vice versa.

10.2.3 Direct Read of Internal Signals

10.2.3.1 Design

The signals are clubbed into group of 32 bits and are memory mapped to Base Address 6. The access is quadlet aligned. Hence PCI Address bus bit [1:0] will always be 2'b00. Bit[6:2] is used to map the "group of signals" to the address space.