Table 202. Line Status Register Bit Definition

Bit	Description
7	**Error in Receiver FIFO Register:** In FIFO mode, this bit indicates that a parity error, framing error, or break occurred. This bit is cleared when the Line Status register is read in the FIFO mode. It is set to 0 in the Character mode.
6	**Transmitter Shift Register Empty:** This bit is set to 1 to indicate the Transmitter Holding register and the Transmitter Shift register are both empty. This bit is set to 0 when either register contains a data character. In FIFO mode, this bit is set to 1 when the Transmitter FIFO register and the Transmitter Shift register are both empty.
5	**Transmitter Holding Register Empty:** This bit indicates that the controller is ready to accept the next character for transmission. This bit is set to 1 to indicate that a character was transferred from the Transmitter Holding register to the Transmitter Shift register. This bit is set to 0 when a character is written to the Transmitter Holding register. This bit also causes the controller to issue an interrupt if the interrupt is enabled. In FIFO mode, this bit is set to 1 when the Transmitter FIFO register is empty. It is set to 0 when at least one byte is written to the Transmitter FIFO register.
4	**Break Interrupt:** This bit is set to 1 to indicate the received data input is held in the Spacing (0) state for longer than a full-word transmission time (the total time of start bit +data bits + parity + stop bits). This bit is reset to 0 when the Line Status register is read. In FIFO mode, this error is associated with a particular character in the FIFO and is revealed only when that character is at the top of the FIFO. Bits 1 through 4 are the error conditions that produce a receiver-line-status interrupt whenever any of the corresponding conditions are detected and the interrupt is enabled.
3	**Framing Error:** This bit is set to 1 when the stop bit, following the last data bit or parity bit, is at a spacing level. This indicates that the received character did not have a valid stop bit (framing error). This bit is reset to 0 when the Line Status register is read. In FIFO mode, this error is associated with a particular character in the FIFO and is revealed only when that character is at the top of the FIFO.
2	**Parity Error:** This bit is set to 1 to indicate a parity error.
1	**Overrun Error:** When set to 1, this bit indicates that data in the Receiver Buffer register was not read before the next character was transferred into the Receiver Buffer register, destroying the previous character. This bit is reset to 0 when the Line Status register is read. If the FIFO mode data continues to fill the Receiver FIFO register beyond the trigger level, an overrun error occurs. The overrun occurs only after the Receiver FIFO register is full and the next character is completely received in the Receiver Shift register. An overrun error is indicated to the system microprocessor when it happens. The character in the Receiver Shift register is overwritten, but it is not transferred to the Receiver FIFO register.
0	**Data Ready:** This bit is the receiver data-ready indicator. It is set to 1 when a complete incoming character has been received and transferred into the Receiver Buffer register or the Receiver FIFO register. This bit is reset to 0 by reading the Receiver Buffer register or by reading all of the data in the Receiver FIFO register.

6.3.9 Modem Status Register (MSR)

This 8-bit register is used to monitor the current state of the control lines from the modem (or external device). Also, bits 3 through 0 indicate change information.

Table 203. Modem Status Register (MSR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initialized Reset Value
1 byte	Direct	irba[0]	0x06	Read/Write	b'xxxx0000'	N/A

Table 204. Modem Status Register Bit Definition

Bit	Description
7	**Data-Carrier-Detect:** This bit is the inverted '-data carrier detect' signal (-DCD) modem control input. If bit 4 of the Modem Control Register is set to 1, this bit is equivalent to bit 3 in the Modem Control Register.
6	**Ring Indicator:** This bit is the inverted '-ring indicator' signal (-RI) modem control input. If bit 4 of the Modem Control Register is set to 1, this bit is equivalent to bit 2 in the Modem Control Register.
5	**Data-Set-Ready:** This bit is the inverted '-data set ready' signal (-DSR) modem control input. If bit 4 of the Modem Control Register is set to 1, this bit is equivalent to bit 0 in the Modem Control Register.
4	**Clear-to-Send:** This bit is the inverted '-clear to send' signal (-CTS) modem control input. If bit 4 of the Modem Control Register is set to 1, this bit is equivalent to bit 1 in the Modem Control Register.
3	**Delta-Data-Carrier-Detect:** When set to 1, this bit indicates that the '-data carrier detect' signal (-DCD) modem control input has changed state since the last time it was read by the system microprocessor. Whenever bit 0, 1, 2, or 3 is set to 1, a modem status interrupt is generated.
2	**Trailing Edge Ring Indicator:** When set to 1, this bit indicates that the '-ring indicator' signal (-RI) modem control input has changed from an active condition to an inactive condition.
1	**Delta-Data-Set-Ready:** When set to 1, this bit indicates that the '-data set ready' signal (-DSR) modem control input has changed state since the last time it was read by the system microprocessor.
0	**Delta-Clear-to-Send:** When set to 1, this bit indicates that the '-clear to send' signal (-CTS) modem control input has changed state since the last time it was read by the system microprocessor.

6.3.10 Scratch Register (SCR)

This register can be used by the system microprocessor as a temporary buffer or work area. It is not used by the UART in any way.

Table 205. Scratch Register (SCR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initialized Reset Value
1 byte	Direct	irba[0]	0x07	Read/Write	Undefined	N/A

6.3.11 Baud Rate Divisor Register (DLL)

The DLL register is the low order byte of the Baud Rate Divisor. The Divisor Latch register is used to program the baud-rate generator. The value in this register forms the divisor of the clock input, which establishes the desired baud-rate.

Table 206. Baud Rate Divisor Register (DLL) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initialized Reset Value
1 byte	Direct, LCR(7)=1	irba[0]	0x00	Write Only	Undefined	N/A

6.3.12 Baud Rate Divisor Register (DLH)

The DLH register is the high order byte of the Baud Rate Divisor. The Divisor Latch register is used to program the baud-rate generator. The value in this register forms the divisor of the clock input, which establishes the desired baud-rate.

Table 207. Baud Rate Divisor Register (DLH) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initialized Reset Value
1 byte	Direct, LCR(7)=1	irba[0]	0x01	Write Only	Undefined	N/A

Requirements:

6–10. The UART serial device must contain a baud rate generator that generates baud rates (within acceptable RS232 tolerances) according to the following relationship: (baud rate x 16) = (frequency input) / DLH|DLL(decimal).

6–11. The UART serial device must have the "**`clock-frequency`**" property specified in its device node in the Open Firmware device tree. This property value corresponds to the "frequency input" term in Requirement 6–10, and is not necessarily the oscillator input to the chip that implements the serial device.

6.4 References

1. *National Semiconductor PC87308VUL Super I/O Enhanced Sidewinder Lite Plug and Play Compatible Chip, with a Floppy Disk Controller, a Keyboard Controller, a Real-Time Clock, Two UARTs, Full Infrared Support and an IEEE1283 Parallel Port* [24]
2. *National Semiconductor PC87332VLJ (3.3V/5V) and PC87332VLJ-5 (5V) (Super I/O III Premium Green) with a Floppy Disk Controller, Dual UARTs, IEEE 1284 Parallel Port, and IDE Interface* [26]
3. *IEEE Std 1275-1994, IEEE Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices,* published by IEEE [27]
4. *CHRP ISA Serial Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [28]

ISA Keyboard/Mouse Controller

7

7.1 Overview and General Requirements

Table 208 on page 105 is an excerpt from the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* showing the PS/2 keyboard interface is optional support for alphanumeric input devices and pointing devices.

Table 208. Minimum Requirements for Alphanumeric Input Device and Pointing Device

Subsystem	Specification	Portable	Personal	Server	Description
Alphanumeric Input Device		R	R	O	Servers do not require a keyboard for normal operation, however a means to attach an A/N Input Device must be provided; an ASCII terminal is a example of such a device. Keyboards must be capable of generating at least 101 scan codes.
	PS/2 Keyboard interface	R*	R*	O	
	ADB	R*	R*	O	
	Terminal	O	O	O	
Pointing Device		R	R	O	If a platform includes a keyboard, it must also include a pointing device with the functionality of at least a 2-button mouse
	2 buttons	R	R	O	
	PS/2 interface (see note 1)	R*	R*	O	
	ADB	R*	R*	O	

The keyboard/mouse controller is actually:

- A microcontroller (commonly based on an 8042 microcontroller)
- Firmware for controlling the microcontroller
- Peripheral logic for accepting data, commands and status from the keyboard, mouse or other auxiliary device and system

Requirements:

7–1. All requirements given in this chapter are specifically for the device represented by the Open Firmware properties “name=8042, device_type=8042, compatible = “chrp,8042”.

7–2. The keyboard and the mouse/auxiliary device are assigned separate interrupts for indicating to the system when user input is available. These interrupts must be separate edge-triggered interrupts.

7–3. A controller that is compatible with the controller described herein must be represented in the device tree as specified in the *CHRP ISA Bus Binding to: IEEE Standard 1275-1994* [15].

7–4. This device must be addressed purely with ISA I/O addresses which can be aliased or non-aliased.

7.2 Keyboard/Mouse Controller Register Table

Requirements:

7–5. The registers shown in Table 209 must all be present in the device defined in this chapter. The Open Firmware “reg” property information given in Table 209 must be provided in the Open Firmware device tree for this device.

7–6. All the reg property fields specified in Table 209 must comply with the format given in the *CHRP ISA Bus Binding to: IEEE Standard 1275-1994* [15].

The positions in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 209. The format is described in *CHRP ISA Bus Binding to: IEEE Standard 1275-1994* [15].

Table 209. Keyboard/Mouse Controller Registers

Offset	ISA Address	Register		Open Firmware Reg Property
		Read	Write	
0	0x0060	Output Register	Input Register	reg[1] ,size=1
0	0x0064	Status	Control	reg[2],size=1

7.3 Keyboard Controller Registers

Requirements:

7–7. For the device described herein, all the registers defined in sections 7.3.1 through 7.3.4 must be implemented precisely as described herein.

Figure 4 on page 107 gives a block diagram that shows the registers in the keyboard/mouse controller and the output and input ports to the actual keyboard and mouse it controls.

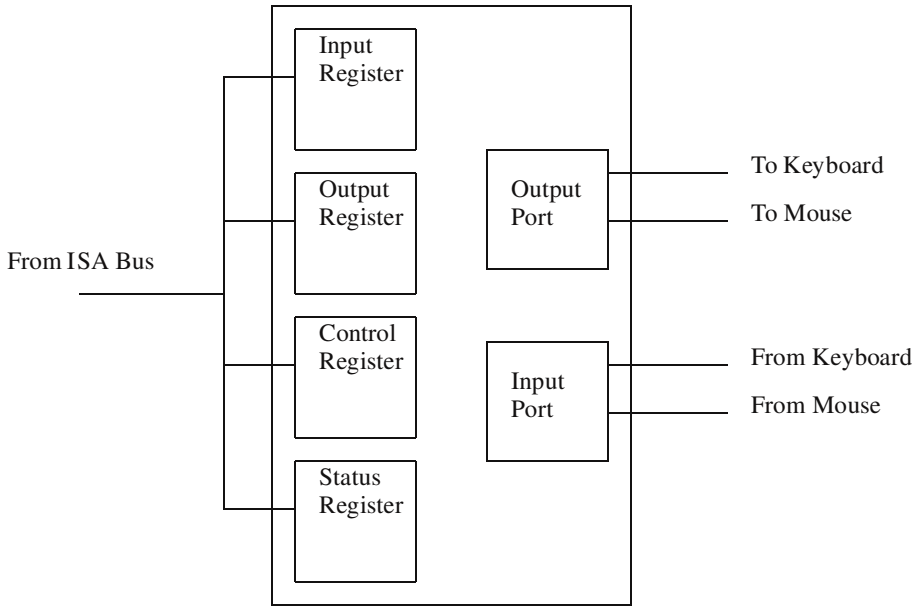

Figure 4. Block Diagram of Keyboard/Mouse Controller Register

7.3.1 Output Register (OUT)

Table 210. Output Register (OUT) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x00	Read	Undefined	N/A

Table 211 on page 107 defines the bits in the Output Register.

Table 211. Output Register (OUT)

Bit	Function
7-0	**(Data):** This is used by the system to read data from an attached device or the keyboard controller

7.3.2 Input Register (IN)

Table 212. Input Register (IN) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x00	Write	Undefined	N/A

Table 211 on page 107 defines the bits in the Input Register.

Table 213. Input Register (IN)

Bit	Function
7- 0	**(Data):** Data from the system to the keyboard controller or an attached device

7.3.3 Status Register (STA)

Table 214. Status Register (STA) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[2]	0x00	Read	Undefined	N/A

Table 211 on page 107 defines the bits in the Status Register.

Table 215. Status Register (STA)

Bit	Normal Function	Special Function
7-6	00: No-op 01: The byte was clocked out of the device, but a response was not received within the time limit specifiied. 10: Undefined 11: The byte was clocked out of the device, but a response indicates a parity error	See the Self-Test command in Table 218 on page 109 for a description of these bits if an error is detected on self-test.
5	**Auxiliary Byte (AUXB):** When this bit is a 1 and OUTB is a 1, the OUT register contains data from the auxiliary device (mouse). When it is a 0 and OUTB is a 1, it contains keyboard data or command controller response data from the 8042.	
4	**Keyboard Lock (KEYL):** When this bit is a 1, the keyboard is locked and the password state is active. When this bit is a 0, the keyboard is free.**Keyboard Lock Status**	
3	**Command/Data (C/D):** When this bit is a 1, it indicates a command has or will be written to the IN register. When this bit is a 0, it indicates that data has or will been written to the IN register C/D (Command/Data)	N/A

Table 215. Status Register (STA) ***(Continued)***

Bit	Normal Function	Special Function
2	System Flag (SYSF): This bit is set to a 1 or 0 by writing to the system flag bit (bit 2) in the Controller Command byte.	N/A
1	**Input Byte (INPB):** When this bit is a 1, the system has put data in the In register for the keyboard	N/A
0	**Output Byte (OUTB):** When this bit is a 1, the keyboard/mouse has put data in the OUT register. If this bit is a 0, the OUT register is empty (the system has read the last byte presented in the OUT register).	N/A

7.3.4 Control Register (CTL)

Table 216. Keyboard Control Reguster CTL) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[2]	0x00	Write	Undefined	N/A

The ways in which these bits are used are described in the next sections.

Table 217. Control Register (CTL)

Bit	Function
7-0	**(Command):** Table 218 gives the commands that the system sends to the keyboard/mouse controller.

.

Table 218. Controller Commands

Code	Command	Description
0x20-0x3F	Read Controller RAM	Bits 5-0 of this command specifies the internal address that thekeyboard/ mouse controller will use to address data to be returned in the output register. Internal address 0x00 is assigned as the Controller Command Byte (See Table 219 on page 111 for details on the Controller Command Byte).
0x60-0x7F	Write to Controller RAM	Bits 5-0 of this command specifies the internal address that the next byte written into the Control Register will be written to . (The address is written to this register followed by the data to be written to the address). Internal address 0x00 is assigned as the Controller Command Byte. (See Table 219 on page 111 for details on the Controller Command Byte).
0xA4	Reserved	
0xA5	Reserved	
0xA6	Reserved	
0xA7	Disable Auxiliary Device (Mouse)	This command sets bit 5 of the Controller Command byte to 1. This disables the auxiliary device interface by driving the clock line low. Data is not received while the interface is disabled..

Table 218. Controller Commands ***(Continued)***

Code	Command	Description
0xA8	Enable Auxiliary Device (Mouse)	This command sets bit 5 of the Controller Command byte to 0, releasing the auxiliary device interface.
0xA9	Check Interface to Auxiliary Device (Mouse)	Checks the interface to the auxiliary device and stores the check code in the output Register: 0x00=no error 0x01=clock line stuck low 0x02=clock line stuck high 0x03=data line stuck low 0x04=data line stuck high
0xAA	Self-Test	The keyboard controller executes a self-test and writes 0x55 into the output Register if no error is detected.. Bit 0 of the Controller Status Register is set to a 1 upon completion of the self-test. The system should allow one second for the self-test to complete before assuming an error occurred and checking the Controller Status Register bits 7-4 for t he error indication. Bits 7-4 of Controller Status Register if error occurs: 0x1 - Valid 0xAA self-test command check 0x2 - Controller instruciton processing tests 0x3 - RAM data and addressing tests 0x4 - ROM data checksum and addressing tests 0x5 - Timer and interrupt handling tests 0x6 - Initialization routines and checks 0x8 - Self-test complete and 0x55 placed in the output buffer
0xAB	Check Keyboard Interface	Checks the interface to the keyboard device and stores the check code in the output Register: 0x00=no error 0x01=clock line stuck low 0x02=clock line stuck high 0x03=data line stuck low 0x04=data line stuck high
0xAD	Disable keyboard	Disables the keyboard. and sets bit 4 of the Controller Command byte to 1.
0xAE	Enable Keyboard	Enables the keyboard. and sets bit 4 of the Controller Command byte to a 0.
0xC0	Read Input Port	Reads the input port to the keyboard/mouse and transfers the data into the output Register.
0xC2	Poll Input Port (high)	Reads bits 7-4 of the input port from the device and transfers them into bits 7-4 of the status register.
0xC3	Poll Input Port (low)	Reads bits 3-0 of the input port from the device and transfers them into bits 7-4 of the status register.
0xD0	Read Output Port	Reads the ouput port from the keyboard/mouse and transfers the data from the output port to the output register.
0xD1	Write Output Port	Writes the following data byte in to the ouput port.
0xD2	Write Keyboard Output Register	Writes the following data byte into the output register and clears AUXB in the status register.
0xD3	Write Ouput Register to Auxiliary Device (Mouse)	Writes the following data byte into the output register and sets AUXB in the status register. An interrupt occurs if the interrupt is enabled in the Controller Command Byte.
0xD4	Write Auxiliary Device	Writes the following byte into the auxiliary device. An interrupt occurs if the interrupt is enabled in the Controller Command Byte.
0xE0	Read Test Input Port	This command causes the keyboard/mouse controller to read its test inputs and place the results in the output buffer. Test 0 (T0) is connected to the keyboard clock line, and test 1 (T1) is connected to the auxiliary device clock line. Data bit 0 representss T0, and data bit 1 represents T1.
0xF0-0xFF	Send Pulses to Output Port	This command pulses selected bits in the keyboard/mouse controller ouput port (to the device) for approximately 6 microseconds. Bits 3-0 of this command select the respective bits in the output port .

Table 219. Controller Command Byte

Bit	Description
7	Keyboard Data Out : Reflects state of this data line to the keyboard.
6	Keyboard Clock Out : Reflects state of this clock line to the keyboard.
5	Auxiliary Device Interrupt (When this is 1 an interrupt has been generated by the auxiliary device in the output register)
4	Keyboard Interrupt (When this is 1 an interrupt has been generated by the keyboard device putting data in the output register or any connected device posting a command to the output buffer.
3	Auxiliary Clock Out:: Reflects state of this data line to the auxiliary device.
2	Auxiliary Data Out:: Reflects state of this clock line to the auxiliary device.
1	Gate Address Line A20
0	Reset System: When this line is set to a 0 the keyboard controller holds the system in a reset state until this bit is set to a 1.

7.4 Programming Devices Attached to the Keyboard/Mouse Controller

The translation of the data and commands that pass through the Input Register (IN) and the Output Register (OUT) are device dependent and as such are not addressed herein

7.5 References

1. *National Semiconductor PC87308VUL SuperI/O Enhanced Sidewinder LitePlug and Play Compatible Chip, with a Floppy Disk Controller, a Keyboard Controller, a Real-Time Clock, Two UARTs, Full Infrared Support and an IEEE1284 Parallel Port* [22]

2. *National Semiconductor PC87332VLJ (3.3V/5V) and PC87332VLJ-5 (5V) (SuperI/O III Premium Green) Floppy Disk Controller, Dual UARTs, IEEE 1284 Parallel Port, and IDE Interface* [24]

3. *Personal Computer Standard V2.0 (SurePath BIOS)* [36]. This can be obtained on the World Wide Web at http://www.surepath.ibm.com/documents/pcs/othrtoc.html.

Audio

8

8.1 Overview

Because of the lack of maturity of audio subsystem technology and its disparate goals in the spectrum of system configurations, the creators have chosen not to limit systems to a single audio subsystem architecture. Platform designers may implement any audio subsystem which meets the requirements below and those in *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1], and are responsible for securing device driver support from the operating systems that will be part of their system offering.

Table 220 on page 113 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements for the audio subsystem of CHRP platforms.

Table 220. Summary of Minimum Platform Requirements (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Audio	16bit Stereo, 22.05 and 44.1 KHz, full duplex.	R	R	O	Servers do not require high quality audio. The programming model for this basic audio capability is specified in *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference*.

Requirements:

8–1. The audio subsystem must provide one stereo analog audio input, which may typically be a line level or microphone input attached to an external jack.

8–2. The audio subsystem must provide one stereo analog audio output, which may typically be a jack for the attachment of speakers or a headphone.

8–3. An analog-to-digital converter (ADC) must be provided to digitize the stereo analog audio input for delivery at the stereo digital audio output.

8–4. An digital-to-analog converter (DAC) must be provided to convert the stereo digital audio input for delivery at the stereo analog audio output.

8–5. The audio subsystem must be capable of simultaneous recording and playback with 16 bit sample resolution at frequencies of 22.05 and 44.1 KHz.

8–6. The audio subsystem must be configured to a passive state (inputs muted, interrupt and DMA requests disabled) when control is passed to the operating system. System resets must also return the audio subsystem to a passive state.

ESCC

9

The Extended Serial Communications Controller (ESCC) provides support for asynchronous (i.e. UART), synchronous (used by synchronous modems and GeoPort™) and HDLC/SDLC (used by LocalTalk) protocols. The ESCC is an enhanced version of the Zilog 85C30. Refer to the 85C30 chapter in the *Zilog Serial Communications Controllers: Product Specifications Data Book* [39] for a description of this part.

The ESCC includes 2 channels (called Channel A and Channel B). Each channel's control, mode and status registers are accessed via indirect accesses to that channel's Command Register. A separate Data Register is allocated to each channel for direct access to its data FIFO. The ESCC also has its own Enhancement Register.

The ESCC has additional registers for simplifying the implementation of LocalTalk, controlling access timing, etc. These registers are shared between channels.

The ESCC also includes separate DBDMA channels for transmit and receive for each channel to provide support for high-speed, full-duplex operation. DBDMA is described in Chapter 15, "Descriptor-Based DMA," on page 169

The ESCC has two alternate address decodings for its Command, Data and Enhancement Registers. The two decodings use different base addresses. The preferred address decoding is shown in Table 222 on page 117. The "legacy" decoding is shown in Table 223 on page 118. Note that the offsets of the shared registers are identical between decodings. Only the offsets of the Command, Data and Enhancement Registers are different.

In order to communicate the different decodings, redundant Open Firmware nodes for the ESCC will be present. The preferred decoding will be the one used by Open Firmware for its serial device driver support. The legacy decoding is present only for completeness.

Each of the channels for the "escc" and "escc-legacy" nodes will be represented by a child node in the Open Firmware tree.

9.1 Overview and General Requirements

Table 221 on page 115 is an excerpt from the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* showing the configuration requirements for the ESCC controller.

Table 221. Summary of Minimum Platform Requirements(excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Serial Port		O	R	O	It is acceptable to provide the PC style interface or the Apple style interface by means of a plug-in card to achieve certification.
	16550	O	R	O	
	SCC	O	R	O	

Requirements:

9–1. When control of the ESCC Controller defined herein is passed from Open Firmware to the Operating System, it must be configured as described herein.

9–2. In the Open Firmware device tree “escc” and “escc-legacy” device nodes, Open Firmware must define the “clock-frequency” property to indicate the frequency of the configured ESCC PCLK. This clock source must be selectable from either the RTxC input or the ESCC’s PCLK. When it is selected from the RTxC input it must have a nominal frequency of 3.684 MHz.

9–3. A controller that is compatible with the controller described herein must be represented in the device tree as specified in the CHRP ESCC Device Binding.

9.2 ESCC Controller Open Firmware Properties

Requirements:

9–4. The registers shown in Table 222 on page 117 must all be present in the device with name= “escc”, device_type= “escc” and compatible= “chrp,es0”.

9–5. The Open Firmware “reg” property information given in Table 222 on page 117 must be provided in the Open Firmware device tree for this device.

9–6. The “escc” device node must have a “ranges” property, along with corresponding “#address-cells”=1 and “#size-cells”=1 properties consistent with the representation of its parent bus. The “ranges” must map the address ranges described by the child node’s “reg” entries.

9–7. All the reg property fields specified in Table 222 on page 117 must comply with the format for the parent bus as defined by the *Mac I/O* [14] binding .

The positions in the reg property (denoted by reg[0], reg[1], reg[2], etc.) and the value of the size fields are stated in Table 222 on page 117. The format is described in referenced Open Firmware binding and not repeated here.

Table 222. ESCC Registers

Offset	Register	Open Firmware Reg Property
0x00	Command B	reg[1],size=256
0x10	Data B	
0x20	Command A	
0x30	Data A	
0x40	Channel B Enhancement	
0x50	Channel A Enhancement	
0x60	SCC Recovery Count	
0x70	LTPC Start A	
0x70	LTPC Start B	
0x80	LTPC Detect AB	
0x90	Timer	
0xA0	Special Character 1	
0xB0	Special Character 2	
0xC0	Special Character 3	
0xD0	Special Detect	
0xE0	Receive Mask	
See Chapter 15, “Descriptor-Based DMA,” on page 169	Channel A DBDMA Tx Registers	reg[2],size=
	Channel A DBDMA Rx Registers	reg[3],size=
	Channel B DBDMA Tx Registers	reg[4],size=
	Channel B DBDMA Rx Registers	reg[5],size=

9.3 Legacy ESCC Controller Open Firmware Properties

Requirements:

9–8. The registers shown in Table 223 on page 118 must all be present in the device with name= “escc-legacy”, device-type= “escc-legacy” and compatible= “chrp,es1”.

9–9. The Open Firmware “reg” property information given in Table 223 on page 118 must be provided in the Open Firmware device tree for this device.

9–10. The “escc-legacy” device node must have a “ranges” property, along with corresponding “#address-cells”=1 and “#size-cells”=1 properties consistent with the representation of its parent bus. The “ranges” must map the address ranges described by the child node’s “reg” entries.

9–11. All the reg property fields specified in Table 223 on page 118 must comply with the format for the parent bus as defined by the *Mac I/O* [14] binding for Open Firmware.

The positions in the reg property (denoted by reg[0], reg[1], reg[2], etc.) and the value of the size fields are stated in Table 223 on page 118. The format is described in theOpen Firmware binding and not repeated here.

Table 223. ESCC Legacy Registers

Offset	Register	Open Firmware Reg Property
0x00	Channel B Command	reg[1],size=256
0x02	Channel A Command	
0x04	Channel B Data	
0x06	Channel A Data	
0x08	Channel B Enhancement	
0x0A	Channel A Enhancement	
0x60	SCC Recovery Count	
0x70	LTPC Start A	
0x80	LTPC Start B	
0x90	LTPC Detect AB	
0xA0	Timer	
0xB0	Special Character 1	
0xC0	Special Character 2	
0xD0	Special Character 3	
0xE0	Special Detect	
0xF0	Receive Mask	
See Chapter 15, “Descriptor-Based DMA,” on page 169	Channel A DBDMA Tx Registers	reg[2],size=
	Channel A DBDMA Rx Registers	reg[3],size=
	Channel B DBDMA Tx Registers	reg[4],size=
	Channel B DBDMA Rx Registers	reg[5],size=

9.4 Child Nodes for ESCC Open Firmware Properties

Requirements:

9–12. The registers shown in Table 224 on page 118 must all be present in the device with name= “ch-a”, device_type= “serial”, and compatible= “chrp,es2”.

9–13. The registers shown in Table 224 on page 118 must all be present in the device with name= “ch-b”, device_type= “serial”, and compatible= “chrp,es3”.

9–14. The Open Firmware “reg” property information given in Table 224 on page 118 must be provided in the Open Firmware device tree for this device.

9–15. The “reg” property fields for the child nodes of the “escc” device must conform to the order described in Table 224 on page 118.

Table 224. ESCC Child Node Registers

Offset	Register	Open Firmware Reg Property
0x00	Channel B Command	reg[0]
0x00	Channel B Data	reg[1]

Table 224. ESCC Child Node Registers *(Continued)*

Offset	Register	Open Firmware Reg Property
0x00	Channel B Enhancement	reg[2]
See Chapter 15, “Descriptor-Based DMA,” on page 169	DBDMA Tx Registers	reg[3]
	DBDMA Rx Registers	reg[4]

9.5 Child Nodes for ESCC-Legacy Open Firmware Properties

Requirements:

9–16. The registers shown in Table 225 on page 119 must all be present in the device with name= “ch-a”, device-type= “legacy” , and compatible=”chrp,es4”.

9–17. The registers shown in Table 225 on page 119 must all be present in the device with name= “ch-b”, device-type= “legacy” , and compatible=”chrp,es5”.

9–18. The Open Firmware “reg” property information given in Table 225 on page 119 must be provided in the Open Firmware device tree for this device.

9–19. The “reg” property fields for the child nodes of the “escc-legacy” device must conform to the order described in Table 225 on page 119.

Table 225. ESCC Legacy Child Note Registers

Offset	Register	Open Firmware Reg Property
0x00	Channel B Command	reg[0]
0x00	Channel B Data	reg[1]
0x00	Channel B Enhancement	reg[2]

9.6 LocalTalk Support

Apple LocalTalk is a family of hardware interconnection products for local area networking. LocalTalk support is provided by mapping a section of logic called the the LTPC Logic module onto the ESCC cell’s address space. For information about LocalTalk, see *Technical Introduction to the Macintosh Family,* second edition [8].

9.7 ESCC Channel Specific Registers

The ESCC has a set of three registers which are assigned independently to each channel.

Requirements:

9–20. For the device described herein, all the registers defined in sections 9.7.1, “Command Register,” on page 120 through Section 9.7.3, “Enhancement Register,” on page 120 must be implemented precisely as described herein.

9.7.1 Command Register

Table 226. Command Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	parent =reg[1] child=reg[0]	See Table 222 and Table 223	Read/Write	Undefined	N/A

The Command Register is used to pass byte wide commands to and from devices on the ESCC channel.

9.7.2 Data Register

Table 227. Data Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	parent =reg[1] child=reg[1]	See Table 222 and Table 223	Read/Write	Undefined	N/A

The Data Register is used by the DMA engine and by system software to send and receive bytes on the ESCC port.

9.7.3 Enhancement Register

Table 228. Enhnacement Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	parent =reg[1] child=reg[2]]	See Table 222 and Table 223	Read/Write	Undefined	N/A

This register is needed for GeoPort and is described in Table 229.

Table 229. Enhancement Register

Bit	Description
7-5	**Reserved**
4	This line is used by Geoport. When it is a 1, the CTSB interface line is connected to the GPIOB line from the serial port, and the DCDB interface line is connected to ground. When it is a 0, CTSB is connected to TRXCBIn_I and DCDB is connected to GPIOB.
3-0	Reserved

9.8 ESCC Shared Registers

The ESCC has a set of shared registers which are used by both the A and B channels.

Requirements:

9–21. For the device described herein, all the registers defined in sections 9.8.1, “SCC Recovery Count Register,” on page 121 through Section 9.8.10, “Receive Mask,” on page 126 must be implemented precisely as described herein.

9.8.1 SCC Recovery Count Register

Table 230. SCC Recovery Count Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x60	Read/Write	Undefined	N/A

The ESCC has timing processes that require a read/write Initial Recovery Count register in the ESCC I/O module.This register is shown in Table 233 on page 122. The value loaded into the 6-bit Initial Recovery Count register (IRC) defines the delay between accesses to the ESCC. The I/O module logic loads this count value into an internal timer when an ESCC access occurs and starts counting down. The ESCC I/O module holds off a second access to ESCC until the timer has reached 0.

Table 231. SCC Recovery Count

Bit	Function
7-6	Reserved
5-0	Initial Recovery Count

9.8.2 LTPC Start A

Table 232. LTPC Start Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x70	Read/Write	Undefined	N/A

When bit 0 is set, the LTPC takes control of the RTSA interface line to the serial port and begins looking for the mark bits at the end of a LocalTalk packet. After approximately 15 mark bits, the LTPC will tristate the serial port and set the corresponding detect bit in the DBDMA S-bits and the LTPC Detect AB Register.

Table 233. LTPC Start A Register

Bit	Function
7-1	Reserved
0	Start A

9.8.3 LTPC Start B

Table 234. LTPC Start B Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x80	Read/Write	Undefined	N/A

When bit 0 is set, the LTPC takes control of the RTSA interface line to the serial port and begins looking for the mark bits at the end of a LocalTalk packet. After approximately 15 mark bits, the LTPC will tristate the serial port and set the corresponding detect bit in the DBDMA S-bits and the LTPC Detect Register.

Table 235. LTPC Start B Register

Bit	Function
7-1	Reserved
0	Start B

9.8.4 LTPC Detect AB Register

Table 236. LTPC Detect Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0x90	Read/Write	Undefined	N/A

The 1-bit read-only register shown in Table 233 consists of a single 1-bit register. It indicates whether the LTPC has detected the abort sequence for the corresponding transmit channels. The register is cleared on reset and is set when the LTPC logic has been armed (by setting the Start register) and has then detected the abort sequence. It stays set until the Start register is cleared by software.

Table 237. LTPC Detect AB Register

Bit	Function
7-2	Reserved
1	Detect A
0	Detect B

9.8.5 Timer Register

Table 238. Timer Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
4 byte	Direct	reg[1]	0xA0	Read/Write	Undefined	N/A

Table 233 on page 122 gives the format of this register.

Table 239. Timer Register

Bit	Function
31-17	Reserved
16	Timer Clock Select: When this bit is a 1, the timer is decremented on the rising edge of the external clock input to the SCC. When this bit is 0, the timer is decremented on the rising edge of a 921.6 Khz clock, giving approximately 1 microsecond resolution to the counter.
15-8	Reserved

Table 239. Timer Register

Bit	Function
7-0	Timer Value: This is where the initial timer value is placed. These bits are the state of the decrementing counter when Timer Clock Select is a 1. When this counter goes to zero, it sets Sbit 7 in the Transmit DMA Status Register.

9.8.6 Special Character 1 Register

Table 240. Special Character 1 Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
2 byte	Direct	reg[1]	0xB0	Read/Write	Undefined	N/A

Table 233 on page 122 gives the format of this register.

Table 241. Special Character 1 Register

Bit	Function
15-9	Reserved. Set to 0.
8	Special Character Enable: When this bit is a 1, bits 7-0 contains a character that this ESCC controller searches for in the received data stream.
7-0	Character: This is the character the ESCC controller searches for when bit 8=1.

9.8.7 Special Character 2 Register

Table 242. Special Character 2 Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
2 byte	Direct	reg[1]	0xC0	Read/Write	Undefined	N/A

Table 233 on page 122 gives the format of this register.

Table 243. Special Character 2 Register

Bit	Function
15-9	Reserved. Set to 0.

Table 243. Special Character 2 Register

Bit	Function
8	Special Character Enable: When this bit is a 1, bits 7-0 contains a character that this ESCC controller searches for in the received data stream.
7-0	Character: This is the character the ESCC controller searches for when bit 8=1.

9.8.8 Special Character 3 Register

Table 244. Special Character 3 Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
2 byte	Direct	reg[1]	0xD0	Read/Write	Undefined	N/A

Table 233 on page 122 gives the format of this register.

Table 245. Special Character 3 Register

Bit	Function
15-9	Reserved. Set to 0.
8	Special Character Enable: When this bit is a 1, bits 7-0 contains a character that this ESCC controller searches for in the received data stream.
7-0	Character: This is the character the ESCC controller searches for when bit 8=1.

9.8.9 Special Character Detect Register

Table 246. Special Character Detect Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
2 byte	Direct	reg[1]	0xE0	Read/Write	Undefined	N/A

When bits 0, 1 or 2 are set as desribed in Table 233 on page 122 an interrupt to the system is generated.

Table 247. Special Character Detect Register

Bit	Function
7-3	Reserved. Set to 0.
2	Special Character Register 3's character has been detected when this bit is a 1.
1	Special Character Register 2's character has been detected when this bit is a 1.
0	Special Character Register 1's character has been detected when this bit is a 1.

9.8.10 Receive Mask

Table 248. Receive Mask Register Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	S/W Initiated Reset Value
1 byte	Direct	reg[1]	0xF0	Read/Write	Undefined	N/A

This register is used to mask out bits from the incoming data stream from the ESCC.

Table 249. Receive Mask Register

Bit	Function
7	Masks bit 7 of the incoming data stream from the ESCC channel when this bit is a 1.
6	Masks bit 6 of the incoming data stream from the ESCC channel when this bit is a 1.
5	Masks bit 5 of the incoming data stream from the ESCC channel when this bit is a 1.
4	Masks bit 4 of the incoming data stream from the ESCC channel when this bit is a 1.
3	Masks bit 3 of the incoming data stream from the ESCC channel when this bit is a 1.
2	Masks bit27 of the incoming data stream from the ESCC channel when this bit is a 1.
1	Masks bit 1 of the incoming data stream from the ESCC channel when this bit is a 1.
0	Masks bit 0 of the incoming data stream from the ESCC channel when this bit is a 1.

9.9 ESCC DBDMA Channel Status Register Usage

The interpretation of the DBDMA Channel Status Registers is device dependent. For the ESCC, their interpretation is given here.

9.9.1 DBDMA Transmit Channel Status Register Usage

The general purpose status bits (ChannelStatus.s7..s0) are allocated as indicated in Table 250.

Table 250. ESCC DBDMA transmit channel status bits

Bit	Meaning
s7..s6	not implemented
s5	LTPC detect
s4..s1	not implemented
s0	Wait (externally controlled)

9.9.2 DBDMA Receive Channel Status Register Usage

The general purpose status bits (ChannelStatus.s7..s0) are allocated as indicated in Table 251.

Table 251. ESCC DBDMA receive channel status bits

Bit	Meaning
s7..s1	not implemented
s0	Wait (externally controlled)

9.10 Conditional Interrupt, Branch and Wait Generation

This section details how Interrupt, Branch and Wait conditions are generated.

9.10.1 Transmit Channel

Only bits s0 and s5 may be used to generate the Interrupt, Branch and Wait conditions. These are the only bits implemented in the InterruptSelect, BranchSelect, and WaitSelect registers for these channels.

9.10.2 Receive Channel

Only bit s0 may be used to generate the Interrupt, Branch and Wait conditions. This is the only bit implemented in the InterruptSelect, BranchSelect, and WaitSelect registers for these channels.

9.10.3 Sending Packets Using the LTPC

The following steps list the sequence that should be followed to send packets using the LTPC module.

1. Disable interrupts through software.
2. Program the ESCC to flag idle.
3. Ensure that the Start register is cleared. This will cause RTS_ to be passed through the LTPC without any modification.
4. Turn on the RTS_ bit in the ESCC.
5. Wait for greater than 1 bit time to guarantee that an edge is generated on the line.
6. Turn off the RTS_ bit in the ESCC.
7. Wait 1.5 bit times to generate the missing clock.
8. Enable the transmitter and turn on the RTS_ bit in the ESCC.
9. Program the ESCC to mark idle.
10. Start the DBDMA transmit channel program.
11. Set the Start register. This will cause it to start monitoring the TxData line looking for the abort sequence. RTS_ will remain in pass-through mode until the abort sequence has finished.
12. SW may now reenable interrupts.
13. Poll the Detect register to determine when the abort sequence has finished. When the LTPC has detected 16 ones in a row it will drive the RTS_ line high, terminating the abort sequence.
14. Turn off the RTS_ bit in the ESCC.
15. Reset the missing clocks flag.
16. Clear the Start register. This will cause RTS_ to be passed through without any modification.

9.11 References

1. *Zilog Serial Communications Controllers: Product Specifications Data Book* Document number: DC8316-01 [39]

Apple Desktop Bus Controller 10

The Apple Desktop Bus (ADB) is a low-speed bus for I/O devices such as keyboards, pointing devices, and tablets. The ADB controller relieves the microprocessor from performing the ADB control functions. It performs all ADB master functions, including autopolling, service request (SRQ) autopolling with error recovery, and keyboard-invoked reset and NMI interrupt actions.

10.1 CHRP Requirements

Table 252 on page 129 is an excerpt from the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* showing the ADB is optional support for alphanumeric input devices and pointing devices.

Table 252. Summary of Minimum Platform Requirements (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Alphanumeric Input Device		R	R	O	Servers do not require a keyboard for normal operation, however a means to attach an A/N Input Device must be provided; an ASCII terminal is a example of such a device. Keyboards must be capable of generating at least 101 scan codes.
	PS/2 Keyboard interface	R*	R*	O	
	ADB	R*	R*	O	
	Terminal	O	O	O	
Pointing Device		R	R	O	If a platform includes a keyboard, it must also include a pointing device with the functionality of at least a 2-button mouse
	2 buttons	R	R	O	
	PS/2 interface	R*	R*	O	
	ADB	R*	R*	O	

Requirements:

10–1. If an ADB is present in the system, it must perform as defined in this chapter.

10.2 ADB Open Firmware Properties

Requirements:

10–2. The ADB controller device must be represented by the Open Firmware properties “name=adb, device_type=adb, compatible= chrp-adb0”.

10–3. The registers shown in Table 253 on page 130 must all be present in the device defined in this chapter.

10–4. The Open Firmware “reg” property information given in Table 253 on page 130 must be provided in the Open Firmware device tree for this device.

10–5. The interrupt property must be provided for this device in the Open Firmware device tree.

10–6. The Open Firmware device tree node for the ADB serial device described in this chapter must follow the specification given in the Open Firmware *Mac I/O* [14] binding.

The position in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 253 on page 130. The format is described in the referenced Open Firmware bindings and not repeated here.

Table 253. ADB status and control registers

Name	Offset	Reg Properties for Open Firmware (base address, size)
Interrupt Status	0x0000	(reg[1] ,size = 273)
Command	0x0010	
Data1	0x0020	
Data2	0x0030	
Data3	0x0040	
Data4	0x0050	
Data5	0x0060	
Data6	0x0070	
Data7	0x0080	
Data8	0x0090	
Interrupt Source Enable	0x00A0	
Data Type/Count	0x00B0	
Error Status	0x00C0	
Control	0x00D0	
Autopoll Control	0x00E0	
Active Device Address Low	0x00F0	
Active Device Address High	0x0100	
Reserved	0x0110	

10.3 ADB Registers

Requirements:

10–7. The ADB controller must be implemented with all the registers as defined in :

- **a.** Table 254, “Interrupt Status Register (IST) Characteristics,” on page 131
- **b.** Table 255, “Interrupt Status Register,” on page 131
- **c.** Table 256, “Command/Data Register File (CDRF) Characteristics,” on page 132
- **d.** Table 257, “Interrupt Source Enable (ISE) Characteristics,” on page 132
- **e.** Table 258, “Interrupt Source Enable,” on page 132

f. Table 259, “Data Type Count (DTC) Characteristics,” on page 132

g. Table 260, “Data Type Count Register,” on page 133

h. Table 261, “Error Status Register (ESR) Characteristics,” on page 133

i. Table 262, “Error Status Register,” on page 133

j. Table 263, “Control (CTL) Characteristics,” on page 134

k. Table 264, “Control Register,” on page 134

l. Table 265, “Autopoll Control (ACL) Characteristics,” on page 134

m. Table 266, “Autopoll Control Register,” on page 134

n. Table 267, “Active Device Address low and High (ACD) Characteristics,” on page 135

o. Table 268, “Active Device High and Low Registers,” on page 135

10.3.1 Interrupt Status (IST)

The IST register contains the two bits indicating the source of the interrupt.

Table 254. Interrupt Status Register (IST) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x0000	Read/Write	0x00	N/A

.

Table 255. Interrupt Status Register

Bit	Function
7-2	Reserved
1	**TAG:** The TAG bit determines whether or not a transfer may take place. The ADB controller must set this bit to 1 when system software is granted access to write a command and data. System software must only write the command and data when Tag is set to 1.
0	**DFB:** The DFB bit indicates whether or not data from the ADB is present. The ADB controller must set this bit to 1 when a command and data are available for system software.

10.3.2 Command/Data Register File (CDRF)

The CDRF is used for both sending and receiving data. The commands and data formats are described in the ADB chapter of the book *Inside Macintosh: Devices* [15]

Table 256. Command/Data Register File (CDRF) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
9 bytes	Direct	reg[1]	See Table 253 on page 130	Read/Write	0x00	N/A

10.3.3 Interrupt Source Enable (ISE)

The ISE register masks or unmasks the two conditions for an interrupt from the ADB.

Table 257. Interrupt Source Enable (ISE) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x00A0	Read/Write	0x00	N/A

Table 258. Interrupt Source Enable

Bit	Function
7 -2	Unused
1	**TAG_IS_EN:** System software must set the TAG_IS_EN bit to enable ADB controller to issue an interrupt when write access to the CDRF is granted (that is, when the TAG bit is set).
0	**DFB_IS_EN:** System software must set the DFB_IS_EN bit to enable the ADB controller to issue an interrupt when data from the ADB is present (that is when the DFB bit is set).

10.3.4 Data Type Count (DTC)

The DTC register has a bit for indicating when data was received from the ADB as a result of an autopoll sequence and a count to indicate the number of valid command and data bytes that have been stored in the CDRF.

Table 259. Data Type Count (DTC) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x00B0	Read/Write	0x00	N/A

Table 260. Data Type Count Register

Bit	Function
7-5	Unused
4	**APD:** The ADB controller must set APD bit when the data that is posted to the CDRF was obtained as the result of an autopoll sequence. The ADB controller must clear the APD bit at the start of any autopoll sequence. System software must only read this bit.
3-0	**HMB:** The four HMB bits are used to indicate the number of valid command and data bytes that have been stored in the CDRF. The ADB controller must set these bits when the data is being transferred from the ADB to the system software. System software must sets these bytes when data is being transferred from the system software to the ADB. These bits must contain a 4-bit binary number from 0 to 9.

10.3.5 Error Status Register (ESR)

The ESR register has bits for indicating when a data byte has failed to complete on the ADB interface and when a device has failed to respond to a Talk command.

Table 261. Error Status Register (ESR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x00C0	Read/Write	0x00	N/A

Table 262. Error Status Register

Bit	Function
7-3	Reserved
2	SRQ: The service request bit indicates another device on the bus was requesting service during the last transaction. If this was set as a result of an autopoll, then autopoll must seek out the device asserting SRQ in the next autopoll cycle.
1	**DLE:** The ADB controller must set the DLE bit when it was in process of reading data from the bus, at least 1 bit but less than 8 bits have been received, and no additional edges are detected within $t_{cyc\ max}$ (period time of a bit cell, worst case). The DFB bit must also be set.
0	**NRE:** The ADB controller must set the NRE bit, if in response to a command from the system software, it has issued a Talk command to a device that is known to be on the bus and there is no response from the device. In other words, no start bit is detected within the $T_{lt\ max}$ time (stop-to-start timing, worst case). The DFB bit must also be set.

10.3.6 Control (CTL)

The CTL register contains bits which control the ADB operation.

Table 263. Control (CTL) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x00D0	Read/Write	0x00	N/A

Table 264. Control Register

Bit	Function
7-4	Reserved
3	**ADB_RST:** When the system software sets the ADB_RST bit, the ADB controller must issue an ADB Reset. The ADB controller must clear the ADB_RST bit upon completion of the ADB Reset.
2	**CRE:** System software must set the CRE bit to inform the ADB controller that it expects a response from the ADB device that is addressed by the command that is currently in the CDRF. This occurs when the system software is sending a Talk command (performing a read from ADB). When the CRE bit is set, the ADB controller must sample the bus for a response from the device that was addressed. This action may result in an error condition, or may result in data being posted to the CDRF.
1	**DTB:** The value of the DTB bit indicates whether or not data from the system software should be sent to the ADB. System software must set DTB to 1 after it has placed a command and data in the CDRF. The ADB controller must send the command and data to the device and upon completion must clear the DTB bit.
0	**TAR:** System software uses the TAR bit to request access to the bus. System software must set the TAR bit to 1 to inform the ADB controller that it would like to send an explicit command or command and data to the ADB.

10.3.7 Autopoll Control (ACL)

The ACL register is used by the system to initiate an autopoll sequence.

Table 265. Autopoll Control (ACL) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
1 byte	Direct	reg[1]	0x00E0	Read/Write	0x00	N/A

Table 266. Autopoll Control Register

Bit	Function
7-1	Reserved
0	**APE:** System software must sets the APE bit to a 1, when it wants the ADB controller to perform autopolling. The ADB controller must respond by performing autopolling if the APE bit is set.

10.3.8 Active Device Address High and Low (ACD)

The 16 bit ACD register has one bit position for each of sixteen possible devices on the ADB.

Table 267. Active Device Address low and High (ACD) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register when OF passes control to OS	SW Initiated Reset Value
2 bytes	Direct	reg[1]	0x00F0, 0x0100	Read/Write	0x0000	N/A

Table 268. Active Device High and Low Registers

Bit	Function
7-0	**AD_AD_Low[7,6,5,4,3,2,1,0] (at 0x00F0):** The bits in the AD_AD registers indicate which addresses on the ADB contain currently active devices. System software must set these bits on if an active device exists at this position. The ADB controller must use this information during the SRQ autopolling sequence.
15-8	**AD_AD_High[15,14,13,12,11,10,9,8] (at 0x0100):** The bits in the AD_AD registers indicate which addresses on the ADB contain currently active devices. System software must set these bits on if an active device exists at this position. The ADB controller must use this information during the SRQ autopolling sequence.

10.4 CDRF Arbitration

The CDRF is used to queue data between the ADB and the System. Because only one register file is used, it is necessary to ensure that data collisions cannot occur. This section describes the algorithm used to preserve the data integrity of the CDRF.

Requirements:

10–8. When system software wants to transfer data to the ADB, the following process must be used:

a. System software must set the Transfer Access Request (TAR) bit to 1.

b. If TAG_IS_EN is set to 1 polls the ADB controller Interrupt Status register, system software must wait for an interrupt from the ADB controller. Otherwise system software must poll the the Interrupt Status register to determine when the TAG bit is set.

c. When the system software is granted access to the CDRF, the ADB controller must not write to the CDRF.

d. When the TAG bit is set, system software must first write the command byte and then from 0 to 8 bytes of data to the CDRF.

e. Once the command and data bytes are written, system software must set the How Many Bytes (HMB)field of the the Data Type Count register, with the number of bytes written to the command plus data byte registers in the CDRF.

f. If system software expects a response to the command, it must set the Command Response Expected (CRE) bit in the Control register.

g. System software must set the DTB (Data To Bus) bit in the Control register.

h. System software must first clear the TAR bit in the Control register and then clear the TAG bit in the the Interrupt Status register in that order.

i. When the TAG bit is cleared, the ADB controller must sample the DTB bit

j. If the DTB is set, the ADB controller must send the data to the ADB and then clear the DTB.

k. The ADB controller must read any response data and post it to the CDRF.

10–9. When the ADB controller acquires data the following process must be used:

a. The ADB shall fill the 9-byte CDRF with either the command that was sent from the System or the Talk command that caused a response to an autopoll and 0 to 8 data bytes.

b. The ADB controller must set the How Many Bytes, HMB[3:0], bits in the status register to the number of data bytes plus one for the command byte.

c. The ADB controller must set any error status bits (see Section 10.5, “Error Handling,” on page 136) in the Error Status register.

d. The ADB controller must set the DFB interrupt source bit in the Interrupt Status Register.

e. If the data was from an autopoll or SRQ autopoll action, the ADB controller must set the Autopoll Data (APD) bit.

f. System software must clear APD and DFB.

g. The ADB controller must wait until the system software clears the DFB bit before resuming autopolling.

10–10. After processing the commands and data in the CDRF, the ADB controller must resume autopolling, if autopolling is enabled.

Hardware Implementation Note: The *Macintosh Technology in the Common Hardware Reference Platform* [16] describes an implementation of autopolling and SRQ autopolling.

10.5 Error Handling

Requirements:

10–11. The ADB controller must recognize the error conditions listed in Table 269 on page 136.

Table 269. ADB error conditions

Type of error	Description
Data Lost	Additional data was expected but no data was received
No Response	A device was issued a Talk command but no data was received
SRQ Autopolling	An SRQ was detected during autopolling but no device was found to be requesting the bus

Table 269. ADB error conditions *(Continued)*

Type of error	Description
Inactive Device Response	An SRQ was asserted from a device not currently reflected in the Active Device Register

10.5.1 Data Lost Error

A Data Lost error occurs when the ADB controller was in process of reading data from the bus, at least 1 bit but less than 8 bits had been received, and no additional edges were detected within time $t_{cyc\ max}$ (period time of a bit cell, worst case).

Requirements:

10–12. The ADB controller must terminate the read from the bus, append zeros to any undetected bits of an unfinished byte ,post the data that was received and set the DLE error bit in the status register, along with the DFB bit.

10–13. If the interrupt is enabled (DFB_IS_EN), the controller must generate an IRQ to the System.

Software Implementation Note: System software may choose to resend the previous command but should account for the possibility that the error may be repeated.

10.5.2 No Response Error

A No Response error occurs in response to a command from the System when the ADB controller has issued a Talk command to a device that is known to be on the bus and there is no response from the device. No start bit is detected within the $T_{lt\ max}$ time (Stop to Start timing, worst case).

Requirements:

10–14. The ADB controller shall load the command that was sent into the CDRF, shall set the NRE error in the status register, and shall set the DFB bit in the interrupt source register.

Software Implementation Note: The System may choose to resend the previous command, but should account for the possibility that the error may be repeated.

10.5.3 SRQ Autopolling Error

An SRQ autopolling error occurs if an SRQ was detected during autopolling but no device was found to be requesting the bus.

Requirements:

10–15. The ADB controller performing the SRQ autopolling sequence must attempt to resolve SRQ autopolling error.

10–16. If the ADB master cannot resolve the error it must return to the main idle loop and resume autopolling, if autopolling is enabled.

10.5.4 Inactive Device Response Error

An Inactive Device Response Error occurs if an SRQ was detected during autopolling and the SRQ Autopolling and Error Recovery Sequence determined that the SRQ was asserted from a device not currently reflected in the Active Device Register.

Requirements:

10–17. The ADB controller must set the DFB bit to inform the system software that there is data in the CDRF.

10–18. The address of the device must be reflected in the command byte that is stored in the CDRF.

Software Implementation Note: It is up to the system software to decide what to do with this information.

10.6 ADB Controller Special Functions

The ADB controller recognizes some special key combinations and provides a software initiated ADB reset.

10.6.1 Keyboard-Invoked Reset or Interrupt

ADB attached keyboard devices may request a system reset or NMI interrupt. The default operation of the ADB controller is for one of the keyboards to be assigned as logical device two and one of the keyboards to be assigned to that device. Therefore these signals may only be invoked from the keyboard attached as logical device two.

Requirements:

10–19. The ADB controller must present these resets to the platform in manners compliant with the other CHRP requirements.

10–20. The ADB controller must scan incoming commands and data (CDRF) from the keyboard at the default logical device two.

- When the Control (0x0036), Command (0x0037), and the Power (0x7F7F) keys are detected together (with or without other keys) by the ADB controller, the ADB controller must initiate a power-on system reset (called “hard reset” in the PowerPC processor user manuals).
- When the Command (0x0037) and the Power (0x7F7F) keys are detected together (with or without other keys) by the ADB controller, the ADB controller must assert the nonmaskable interrupt (NMI).
- After the NMI is asserted when either the Command (0x0037) or the Power (0x7F7F) key are detected together by the ADB controller as released, the ADB controller must deassert the nonmaskable interrupt (NMI).

Software Implementation Note: The NMI interrupt is used by Mac OS to enter the system debugger. Use of this interrupt by other operating systems is optional.

The NMI interrupt is one of two interrupts generated by the ADB controller.

10.6.2 ADB Bus Reset

System software can instruct the ADB controller to issue an ADB Hard Reset at any time, by setting the ADB_RST bit.

Requirements:

10–21. When the ADB controller finds the ADB_RST bit set, it must perform a bus reset as follows:

- **a.** It must drive the ADB Data Out line low for a minimum of 3 ms, then release the line.
- **b.** It must pull up the open collector output to the deassertion state.
- **c.** It must wait the programmed holdoff time, then clear the ADB_RST bit in the Control register, signalling that the ADB bus reset sequence is complete.

10–22. System software must not attempt to use the ADB controller while the ADB_RST bit in the Control register is set to 1 other than to poll the Control register for a change in the ADB_RST bit to a 0.

Hardware Implementation Note: The ADB controller cell does not interpret the ADB SENDRESET command (0bxxxx0000) to indicate that it should perform an ADB bus reset sequence as described above. The controller sends the SENDRESET command as it does any other ADB command to the system software. If the system software wants a bus reset to be sent to the ADB as a result of the SENDRESET command, it will interpret this command and assert the ADB_RST bit.

Software Implementation Note: System software may set the APE (Autopoll Enable) bit or the TAR (Transfer Access Request) bit after it sets the ADB_RST bit. The ADB controller shall respond as soon as the reset sequence is completed. There is no need for the ADB controller to notify the System that the reset cycle has completed, since either the TAG bit will be set (in response to TAR) or the DFB bit will be set upon completion of an autopoll sequence, resulting in data received or an error. If the System needs to know when a reset sequence has been concluded it may poll the state of the ADB_RST bit, which the ADB controller shall clear upon completion of the reset sequence.

10.7 Timing Value

This section gives minimum and maximum timing values and recommended default timing values.

Requirements:

10–23. The ADB controller must implement timing values for the ADB as specified in Table 270 on page 139.

Table 270. Timing Values

Register	Maximum time, µs	Minimum time, µs	Default time, µs
Zero low time	67.32	63.24	65.28
One low time	36.72	34.68	34.68
Reset pulse width	8353.8	3000.8	3000.8
Synch pulse width	65.28	63.24	65.28
Attention pulse width	822.12	777.24	801.72
Stop pulse width	71.4	69.36	71.4

Table 270. Timing Values *(Continued)*

Register	Maximum time, µs	Minimum time, µs	Default time, µs
Bit cell period t_0	128.52	71.4	130.56
Stop-to-start pulse width	259.08	140.76	255.31
Autopoll time period	15378	244.1	11288

10.8 References

1. *Mac I/O* [14] Open Firmware binding
2. Refer to the *Inside Macintosh: Devices* [15] for a description of the command and data formats contained in CDRF.
3. Refer to *Macintosh Technology in the Common Hardware Reference Platform* [16] for physical, electrical, and timing information.
4. Refer to *ADB-The Untold Story* [17] for a description of ADB protocols including a mouse.

IDE Drive Controller

11

11.1 General Requirements

Table 271 on page 141 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing that the IDE Controller on CHRP platforms is optional.

Table 271. Minimum Platform Requirements for Hard Disk

Subsystem	Specification	Portable	Personal	Server	Description
Hard Disk	Sized as needed	R	R	R	"Medialess" systems are not covered by this architecture.
	SCSI	O	O	O	
	IDE	O	O	O	
	PC Card	O	O	O	
Infrared	IrDA	O	O	O	

Requirements:

11–1. An IDE Drive Controller that is compliant with the interfaces as defined in this section must be represented in the Open Firmware device tree with a node that has a name property of name=ide, a device_type property of device_type=ide, and a compatible property of compatible=chrp,ide.

11–2. The IDE Bus Master Controller defined herein must supply "Native-PCI" capabilities and PCI bus mastering capabilities concurrently.

11–3. All IDE devices connected to the IDE Bus master Controller defined herein must have DMA capability.

11–4. When control of the IDE Bus master Controller defined herein is passed from Open Firmware to the Operating System, it must be configured as a PCI device as described herein.

11–5. This device must be addressed only with PCI I/O addresses.

Software Implementation Note: In the Native-PCI mode, the registers of the IDE channels are completely relocatable in I/O space.

Software Implementation Note: Bus mastering with PIO IDE devices will not be supported by the device drivers for this device.

11.2 IDE Bus master Controller Open Firmware Properties

Requirements:

11–6. The registers shown in Table 272 must all be present in the device defined in this chapter. The Open Firmware “reg” property information given in Table 272 must be provided in the Open Firmware device tree for this device.

11–7. This device must be represented as an PCI device and a child of a PCI bridge node in the Open Firmware device tree.

11–8. All the reg property fields specified in Table 272 on page 142 must comply with the format given in the ISA (or PCI) bus binding to IEEE Std 1275-1994, as modified by the CHRP binding to IEEE Std 1275-1994.

The IDE Bus master Controller reg properties are listed below in Table 272 on page 142. The positions in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 272 on page 142. The format is described in the referenced Open Firmware bindings.

Table 272. Register Map for PCI Bus master IDE Drive Controller

Offset	Traditional ISA I/O Address	Register		Reservedby Firmware	Open Firmware Reg Property
		Read	Write		
0x00-0x07	PCI - N/A	Primary IDE Registers (Legacy ISA remapped to PCI)		NO	reg[1]; Size=8 bytes
0x00-0x07	PCI - N/A	Secondary IDE Registers (Legacy ISA remapped to PCI)		NO	reg[2]; Size=8 bytes
0x00	PCI - N/A	Alternate Status/Device Control for Secondary IDE (Legacy ISA remapped to PCI)		NO	reg[3]; Size=1 byte
0x00	PCI - N/A	Alternate Status/Device Control for Primary IDE (Legacy ISA remapped to PCI)		NO	reg[4]; Size=1 byte
0x00	PCI - N/A	Primary Command Register		NO	reg[5]; Size=16 bytes
0x01	PCI - N/A	Reserved			
0x02	PCI - N/A	Primary Status Register			
0x03	PCI - N/A	Reserved			
0x04-0x07	PCI - N/A	Primary PRD Table Address Register			
0x08	PCI - N/A	Secondary Command Register			
0x09	PCI - N/A	Reserved			
0x0A	PCI - N/A	Secondary Status Register			
0x0B	PCI - N/A	Reserved			
0x0C-0x0F	PCI - N/A	Secondary PRD Table Address Register			

11.3 “Native-PCI” IDE Register Definition

For information on these registers, IDE modes, etc. see the *ANSI ATA Specification Revision 3.2* [29].

11.4 Physical Region Descriptors for Bus master IDE

Requirements:

11–9. For the device described herein, the Physical Region Descriptor, PRD, must be implemented as defined in this section.

11–10. PRD format in memory must be Little-Endian.

The Physical Region Descriptor, gives a pointer to a single buffer, a byte count for the number of bytes to be transferred into or out of that buffer, and an end of table flag, EOT. The data transfer will proceed until all regions described by the PRD's in the table have been transferred. A PRD entry is shown in Figure 5. Lists of PRD are placed contiguously in memory to create a program for the bus master IDE controller to execute. The last entry in the list of PRD's must have the EOT flag set. The descriptor table must begin on a four byte boundary, and cannot cross a 64 KB boundary in memory.

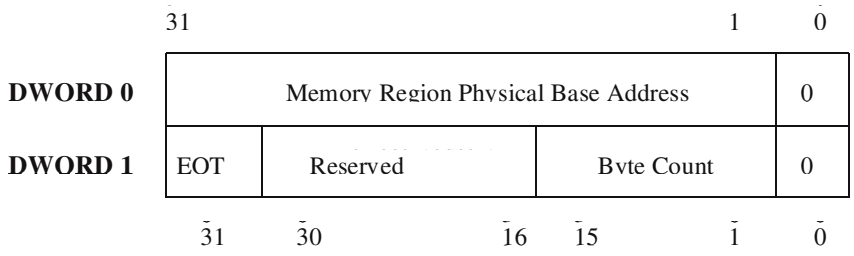

Figure 5. Physical Region Descriptors for Bus master IDE

11.5 Bus Master IDE I/O Registers

Requirements:

11–11. For the device described herein, all the registers defined in sections Table 11.5.1 through Table 11.5.6 must be implemented as described herein.

These registers are designed for use with multi-word DMA devices (devices which are DMA capable).

11.5.1 Primary Command Register (PCR)

Table 273. Primary Command Register (PCR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[5]	0x00	Read/Write	0x00	0x00

This 8-bit, read/write register is used to control DMA data transfers to and from the two IDE devices on the primary IDE when multi-word DMA disk drives are used.

Table 274. Primary Command Register (PCR)

Bit	Description
7-4	Reserved = 0
3	**W/R#:** This bit sets the direction of data transfer for a bus master DMA operation. When 0, data is transferred from the PCI bus to the IDE device. When 1, data is transferred from the IDE device to the PCI bus. This bit must not be changed when the bus master function is active.
2-1	Reserved=0
0	**BMEN:** When this bit is a 1, the IDE bus master controller is set for bus master operation. A bus master operation can be terminated by setting this bit to 0. This is considered to be an abort and any operation that was in progress cannot be resumed. Writing a 1 to this bit will also set the BMEN bit in the Device Control Register.

11.5.2 Secondary Command Register (SCR)

Table 275. Secondary Command Register (SCR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[45	0x08	ReadWrite	0x00	0x00

The bit assignments and definitions for the Secondary Command Register is identical to the Primary Command Register with the difference that this register controls DMA data transfers to/from two multi-word DMA IDE devices on the secondary IDE interface, and it has a different offset address.

11.5.3 Primary Status Register (PSR)

Table 276. Primary Status Register (PSR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[5]	0x02	Read/Write	0x00	0x00

This 8-bit, read/write register is used to control DMA data transfers to and from the two IDE devices on the primary IDE when multi-word DMA disk drives are used.

Table 277. Primary Status Register (PCR)

Bit	Description
7	**Multithread Bit (MT):** The multithread bit is hardwired to a 0b to indicate that both channels operate independently and can be used at the same time.

Table 277. Primary Status Register (PCR)

Bit	Description
6	**Slave Drive DMA Capable (SDC):** The Slave Drive DMA Capable bit is a status bit that is set by system software to indicate that the slave drive on the indicated port is DMA capable and that the IDE bus master controller is initialized for optimum performance.
5	**Master Drive DMA Capable (MDC):** The Master Drive DMA Capable bit is a status bit that is set by system software to indicate that the master drive on the indicated port is DMA capable and that the IDE bus master controller is initialized for optimum performance.
4-3	Reserved
2	**Interrupt Request (IRQ):** This bit is set by the rising edge of the associated IDE port's IDE_IRQ signal. This bit is cleared by writing a one to it. On data transfers from an IDE device to system memory, this bit will be delayed until all data has been transferred to memory. Before attempting to utilize the DMA data written to system memory, software must assure that all of the DMA data has first arrived. Software can do this by satisfying Requirement 5–11 in the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*. Alternatively, the device itself can assure this by satisfying Requirement 11–8 (also described in the same document). This bit is 0 after a reset.
1	**Error (ERR):** This bit is set when the controller encounters an error when transferring data to/from system memory. The specific error conditions are errors that would cause bit 8, 12 or 13 of the Device Status register to become set. This bit is reset by writing a 1 to it.
0	**Active (ACT):** This bit is set when the start bit of the command register is set to a 1. It is cleared when the start bit is set to 0 (abort condition) or when the last transfer for a region is performed where EOT is set in that region descriptor (normal termination).

11.5.4 Secondary Status Register (SSR)

Table 278. Secondary Status Register (SSR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[5]	0x0A	Read/Write	0x00	0x00

The bit assignments and definitions of the Secondary Status Register are identical to the Primary Status Register with the difference that this register is associated with DMA data transfers to/from two multi-word DMA IDE devices on the secondary IDE interface, and it has a different offset address.

11.5.5 Primary PRD Table Address Register (PPRD)

Table 279. Primary PRD Table Address Register (PPRD) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
4 bytes	Direct	reg[5]	0x04-0x07	Read/Write	0x00000000	0x00000000

This 32-bit, read/write register contains the starting address of the first Physical Region Descriptor Table in memory which applies to cases where the IDE controller is functioning as a PCI bus master with one or more multi-word DMA mode disk drives. This register is used to control the Primary IDE channel. The address is in Little-Endian format.

Table 280. Primary Command Register (PCR)

Bit	Description
31-2	**PRD Table Address:** This is the starting address of the first Physical Region Descriptor Table in memory. The 30 bit value is a double word aligned address in memory.
1-0	These bits must be set to 0b00

Bit 31-2 (PRD Table Address):

Bit 1-0: 0.

11.5.6 Secondary PRD Table Address Register (SPRD)

Table 281. Secondary PRD Table Address Register (SPRD) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value when OF passes to SW	Software Reset Value
4 bytes	Direct	reg[5]	0x0C-0x0F	Read/Write	0x00000000	0x00000000

The Secondary PRD Table Address Register is identical to the Primary PRD Table Address Register with the exceptions that it is used to control the Secondary IDE channel and it has a different offset address.

11.6 References

1. *ANSI ATA Specification Revision 3.2* [29]
2. *ANSI ATAPI Specification (Current Revision?)* [30]
3. *Symphony SL82C565 System I/O Controller With PCI Arbiter* [21]
4. *SFF IDE PCI Bus Master Specification??* [31]
5. *PCI Sign IDE PCI Bus Master Specification??* [32]

SCSI

12

12.1 CHRP Requirements

Table 282 on page 147 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements for VIA on CHRP platforms.

Table 282. Summary of Minimum Platform Requirements (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Hard Disk	Sized as needed	R	R	R	"Medialess" systems are not covered by this architecture.
	SCSI	O	O	O	
	IDE	O	O	O	
	PC Card	O	O	O	

12.2 Overview and General Requirements

While the SCSI bus and command set presents a relatively stable hardware interface and conceptual programming model, the programming model presented by various implementations is anything but uniform. This is largely the result of a slight deepening of the device function to include capabilities such as control block queuing and simple command processing which improve performance and off-load the host processor complex. There has been no (successful) attempt by the developers of the SCSI implementations to standardize on a consistent programming model for these capabilities. As a result, it is inappropriate to attempt to choose a programming model for SCSI. Operating systems include drivers for different combinations of implementations, each based on its legacy of supported platforms. In general, the system designer must negotiate device driver support for his choice of interface from the operating systems that will be part of his system offering. The detailed information given in this chapter is focused solely upon MESH (Macintosh Enhanced SCSI Hardware), the Macintosh implementation of the SCSI bus controller. MESH is not a mandated SCSI interface.

MESH SCSI basic architecture is intended to be relatively simple, allowing the maximum latitude in cost, performance, scalability, and flexibility. The Mesh SCSI controller consists of a single DBDMA channel and a set of registers specific to the MESH SCSI controller. The DBDMA function is explained extensively in Chapter 15, "Descriptor-Based DMA," on page 171. DBDMA is a programming model used for MESH SCSI and ESCC today and potentially other devices in the future. Because of the use of the DBDMA function in more than one device, it is documented in a single chapter and mentioned briefly here.

Requirements:

12–1. All requirements given in this chapter are specifically for the device represented by the Open Firmware properties “name=scsi, device_type=scsi, compatible = chrp,mesh0” unless otherwise noted.

12.3 Mesh SCSI Open Firmware Properties

Requirements:

12–2. The registers shown in Table 283 on page 148 must all be present in the device defined in this chapter.

12–3. A MESH SCSI device must make use of properties defined in the *MAC-IO* [14] Open Firmware bindings document.

The Mesh SCSI controller’s properties are listed below in Table 283 on page 148. The Open Firmware “reg” property information given in Table 283 on page 148 must be represented in the Open Firmware device tree for this device.

Table 283. SCSI controller registers

Offset	Name	Reg Properties for Open Firmware
0x00	XferCount0	reg[1]
0x10	XferCount1	
0x20	FIFO	
0x30	Sequence	
0x40	BusStatus0	
0x50	BusStatus1	
0x60	FIFOCount	
0x70	Exception	
0x80	Error	
0x90	InterruptMask	
0xA0	Interrupt	
0xB0	SourceID	
0xC0	DestID	
0xD0	SyncParms	
0xE0	MESHId	
0xF0	SelTO	
0x0	Reserved	reserved
0x4	DBDMA Transmit Channel Control	reg[2]
0xC	DBDMA Transmit Channel Status	
0x10	DBDMA Transmit Interrupt Select	
0x14	DBDMA Transmit Branch Select	
0x18	DBDMA Transmit Wait Select	

12.4 Interrupt Assignment

Requirements:

12–4. The interrupt from the MESH SCSI section of the controller and the DBDMA section of the controller must be wire-OR'ed together so that one interrupt is presented to the system for the complete MESH SCSI controller defined herein.

12.5 MESH SCSI Register Definitions

This section defines the non-DBDMA registers for the MESH SCSI controller.

Requirements:

12–5. For the device described herein, all the registers defined in Section 12.5.1, "Transfer Count 0 Register 0 (XferCount0)," on page 149 through Section 12.5.16, "Selection TimeOut Register (SelTO)," on page 158 must be implemented precisely as described.

12.5.1 Transfer Count 0 Register 0 (XferCount0)

Table 284. Transfer Count Register 0High (XferCount0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x00	R/W	0x00	0x00

The Transfer Count registers, XferCount0 and XferCount1, are used to indicate the number of bytes to be transferred in the subsequent information transfer phase. XferCount0 contains the least significant byte of the count; XferCount1 contains the most significant byte. See Section 12.5.4, "Sequence Register (Sequence)," on page 150 for more details on the use of the Transfer Count registers.

If a phase's Sequence command exhausts the count before a phase change or error occurs, then the CmdDone signal must go active without an Exception interrupt. This allows for simple sequencing of phases as long as the interrupt signal is inactive.

The transfer counter (TC) must be decremented whenever a byte is written into the FIFO. For data transfers to the SCSI bus, the TC is decremented every time the processor writes to the FIFO or a DMA word is transferred. This behavior must be independent of whether or not a command is pending. When the TC reaches 0 and the FIFO becomes empty, the CmdDone bit must be set, indicating the completion of the command. This is true except for synchronous data in, where CD is set when TC reaches 0. On transfers into the MESH controller, the TC must be decremented every time a byte is brought into MESH from the SCSI bus (under hardware control only).

The order of any data transfer should start with setting up the TC, issuing the command, then transferring the data. Technically, the data could be put in the FIFO before the command is issued, but this limits the general model to transferring no more than 16 bytes before issuing the command.

Transfer counter behavior is subject to these special cases:

- In Synchronous DataIn, the TC decrements on the DMA transfer, not on the SCSI bus write action into the FIFO.

- If all 16 bits of the TC are zeros at the beginning of a command, 64 KB must be transferred.
- FIFO must not be preloaded before TC.

There can been a problem in some programming cases where, for instance, the programmer loads the FIFO with the Command Descriptor Block (CDB), loads the TC with the CDB size (say 6), and then sends the CommandPhase command to the MESH controller. The CmdDone bit will never be set because the TC never goes down to zero because no bytes are loaded into the FIFO after setting the TC.

12.5.2 Transfer Count 1 Register (XferCount1)

Table 285. Transfer Count Register 0High (XferCount0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x01	R/W	0x00	0x00

XferCount0 contains the least significant byte of the transfer count; XferCount1 contains the most significant byte. See Section 12.5.4, “Sequence Register (Sequence),” on page 150 for more details on the use of the Transfer Count registers.

12.5.3 Bus FIFO Register (FIFO)

Table 286. Bus FIFO Register (FIFO) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x02	R/W	0x00	0x00

The FIFO register accesses the SCSI bus using programmed I/O, or as the buffer in DMA mode. The current number of bytes in the FIFO is indicated in the FIFO Count register. The FIFO depth is 16 bytes.

If the MESH controller is reselected as a host (or selected as a target), the FIFO register must contain the image of the selection phase data. This means that after the reselection, both the ithe SourceID and the Destination ID will be present in the FIFO (see ref [1]).

12.5.4 Sequence Register (Sequence)

Table 287. Sequence Register (Sequence) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x03	R/W	0x00	0x00

The Sequence register is used to direct the controller into a SCSI bus phase. In the Initiator mode, the requested sequence is the one that the controller expects the Target will enter into. In the Target mode, the phase is the one which the controller will enter into next. The bits in the Sequence register have the functions shown in Table 288 on page 151.

The ways in which these bits are used are described in the next sections.