PowerPC™ Microprocessor Common Hardware Reference Platform: I/O Device Reference

LICENSE INFORMATION

To the extent that Apple Computer, Inc., International Business Machines Corporation, and Motorola, Inc. (referred to as "the creators") own licensable copyrights in the *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference* (including accompanying source code samples), the creators grant you a copyright license to copy and distribute portions of this document (including accompanying source code samples) in any form, without payment to the creators, for the purpose of developing original documents, code, or equipment (except integrated circuit processors) which conform to the requirements in this document and for the purpose of using, reproducing, marketing, and distributing such code or equipment. This authorization applies to the content of this specification only and not to the referenced material. This authorization does not give you the right to copy and distribute this document in its entirety.

In consideration you agree to include for each reproduction of any portion of these documents or any derivative works the copyright notice as displayed below.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization. If you fail to comply with the above terms, your authorization terminates.

The creators and others may have patents or pending patent applications, or other intellectual property rights covering the subject matter described herein. This document neither grants or implies a license or immunity under any of the creators or third party patents, patent applications or other intellectual property rights other than as expressly provided in the above copyright license. The creators assume no responsibility for any infringement of third party rights resulting from your use of the subject matter disclosed in, or from the manufacturing, use, lease, or sale of products described in, this document.

Licenses under utility patents of IBM® in the field of information handling systems are available on reasonable and non-discriminatory terms. IBM does not grant licenses to its appearance design patents. Direct your licensing inquiries in writing to the IBM Director of Licensing, International Business Machines Corporation, 500 Columbus Avenue, Thornwood, NY 10594.

Licenses under utility patents of Apple Computer, Inc., that are necessary to implement the specification set forth in this document are available on reasonable and non-discriminatory terms. Apple Computer, Inc. does not grant licenses to its appearance design patents. Direct your licensing inquires in writing to Mac OS Licensing Department, Apple Computer, Inc., 1 Infinite Loop, MS 38-LG, Cupertino, CA 95014.

NOTICES

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law. In such countries, the minimum country warranties will apply.

THE CREATORS PROVIDE THIS DOCUMENT (INCLUDING ACCOMPANYING SOURCE CODE EXAMPLES) “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. THE DISCLAIMER OF WARRANTY APPLIES NOT ONLY TO THE DOCUMENT (INCLUDING ACCOMPANYING SOURCE CODE EXAMPLES) BUT ALSO TO ANY COMBINATIONS, INCORPORATIONS, OR OTHER USES OF THE DOCUMENT (INCLUDING ACCOMPANYING SOURCE CODE EXAMPLES) UPON WHICH A CLAIM COULD BE BASED.

Some states do not allow disclaimers of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

These materials could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the document. The creators may make improvements and/or changes in the product(s) and/or the program(s) described in, or accompanying, this document at any time.

It is possible that this document may contain reference to, or information about, products (machines and programs), programming, or services of the creators that are not announced in your country. Such reference or information must not be construed to mean that the creators intend to announce such products, programming, or services in your country.

Requests for copies of this document or for technical information about products described herein should be directed to the creators. Refer to “Obtaining Additional Information” on page 263 for a description of information available and telephone numbers.

Version 1.0 (May 1996)

TRADEMARKS AND SERVICE MARKS

Trademarks or service marks in the United States or other countries are denoted by a registered symbol (®) or a trademark symbol(™) on their first occurrence in this document. See “Trademark Information” on page 259 for a complete listing of all referenced trademarks and the companies that own them.

PowerPC™ Microprocessor Common Hardware Reference Platform: I/O Device Reference (Draft Version 0.9)

Developed by Apple Computer, Inc., International Business Machines Corporation, and Motorola, Inc.

Contents

Figures

Tables

About this Document

This document is an extension of the I/O Device chapter in the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*. The purpose of this document is to define architecture and minimum system requirements for I/O devices. These requirements will govern devices which may be used in CHRP systems. These requirements are intended to be precise enough to assure application software compatibility for several operating system environments, broad enough to cover portables through server platforms in single or multiprocessor configurations, and forward-looking enough to allow evolution, including 64-bit addressing.

These requirements were developed by Apple Computer, Inc., International Business Machines Corporation, and Motorola, Inc. to define a system which is intended to become the pervasive open industry standard for single-user portable through multi-user server configurations. Systems built to these requirements will have PowerPC microprocessor(s) and will share components with the Apple® Macintosh® family and IBM compatible personal computers. These systems will be capable of running various native operating systems. The set of operating systems anticipated to be available includes Apple Mac™ OS, IBM AIX™, and PowerPC™ editions of Microsoft Windows NT™ Workstation and SunSoft Solaris™. Most applications for these operating environments may be run on these systems either in native mode or through some emulation capability. With the appropriate operating system and x86 emulation support, these systems will be capable of running Windows™ and DOS applications.

Within the context of this document, “architecture” is defined as the specification of the interface between the hardware platform and the operating systems and applications. Device drivers also use this architecture, but require additional definition of the operating system interfaces.

To the extent that firmware abstracts the hardware interface, it becomes part of the hardware. The firmware to operating system interface is defined in this architecture. Two types of firmware are discussed here. Open Firmware is the initialization or boot code that controls the platform prior to the transfer of control to the operating system. Run-Time Abstraction Services (RTAS) is the run-time firmware which provides abstractions to the executing operating system. Interfaces within the software or within the hardware are not defined in this document. Where necessary, reference will be made to documents where those definitions can be found.

Within the context of this document the term “the architecture” or “the CHRP architecture” is used to refer to the requirements contained in the document, *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1] and this extension, *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference* [11].

Goals of this Specification

The specific goals of this specification are as follows:

- To document the architecture at a register level for native I/O devices. Some devices are documented by manufacturer neutral documents and these documents will be referred to as the source of this architectural definition. Other devices are only defined by current implementations. Where a single commonly used

implementation exists we will document it. When several competing implementations are in current use we will document a subset which provides some component vendor choice.

- To create an open industry standard to be used for the implementation of PowerPC based systems. The architecture document is available to the industry and can be used by any hardware or software vendor to develop compliant products.
- To allow compatible differentiation through the use of abstracted hardware interfaces, defined minimum hardware, and extension mechanisms.
- To leverage existing and future industry-standard buses and interfaces. Existing bus architectures have a proven level of performance and functionality. Established industry-standard interfaces—for example SCSI, IDE, LocalTalk®, Ethernet™, etc.— and newer bus architectures, interfaces and protocols —for example PCI, PC Card, IrDA, etc.— provide higher levels of performance or utility not achievable by the older standards. The architecture allows platform and system designers to determine which buses, interfaces, and protocols best suit their target environment.
- To provide an architecture which can evolve as technology changes. The creators of the architecture invite industry participation in evolving future versions of it.

Audience for this Document

This document defines the architecture and system requirements for I/O devices to be used in building PowerPC platforms which comply with the Common Hardware Reference Platform (CHRP™). This document is the source of information that a hardware platform, operating system, or hardware component developer would need to create compatible products. Additional requirements are defined by the industry standards referenced in this document.

This document must be used by those building CHRP systems. The document describes the hardware to operating system interface which must be provided for I/O devices in these platforms. Platform designers must assemble components and firmware which match this interface. Also, the document defines minimum I/O device configuration requirements. Platform designers must meet or exceed these minimums to build a CHRP system. The operating systems which are expected to support this architecture are listed in an appendix for the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1]. Each operating system has configuration requirements which are described in documents maintained by the owners of the operating systems. Using these requirements, a hardware platform designer may target a configuration to support multiple operating systems.

This document must be used by those building compatible software including operating systems, device drivers, boot software, or firmware. If a function is supported, software developers must provide support for the interfaces described in this document. This software must provide the mandatory functions and capabilities as described in the requirements in this document. However, this document does not limit this software from going beyond the specification through software tailored to specific hardware.

Component manufacturers will use this information to produce compatible I/O chips and adaptors to use on these platforms.

Organization of this Document

The following chapters provide the detailed requirements for I/O devices to be used in compliant systems. Where appropriate, references are made to other industry standards or readily available reference documentation for the required architecture information.

Following is a summary and brief description of the chapters and appendices of this document:

- Chapter 1, “General Requirements and Information,” on page 1 gives some requirements which apply to all devices and acknowledges the source of some of this material.
- Chapter 2, “ISA DMA Controller,” on page 3 defines the ISA direct memory access controller requirements and programming model.
- Chapter 3, “Floppy Disk Controller,” on page 31 gives the requirements for the floppy disk controller.
- Chapter 4, “Legacy Interrupt Controller,” on page 61 sets the requirements for the interrupt controllers which are a legacy from IBM compatible PCs.
- Chapter 5, “Parallel Port Controller,” on page 79 defines the requirements for the parallel port.
- Chapter 6, “UART Controller,” on page 93 sets the requirements for the serial communications controller which is a legacy from IBM compatible PCs.
- Chapter 7, “ISA Keyboard/Mouse Controller,” on page 105 specifies the requirements for the PS/2 style keyboard and mouse interface.
- Chapter 8, “Audio,” on page 113 defines configuration retirements for the audio controller.
- Chapter 9, “ESCC,” on page 115 defines requirements for the enhanced serial communications controller which is the Apple style serial port.
- Chapter 10, “Apple Desktop Bus Controller,” on page 129 defines the requirements for the Apple Desktop Bus™ controller which provides the Apple style mouse and keyboard interface.
- Section 11, “IDE Drive Controller,” on page 141 sets the requirements for the integrated device electronics controller.
- Chapter 12, “SCSI,” on page 147 describes the small computer system interface requirements.
- Chapter 13, “Graphics,” on page 161 gives the requirements for the graphics subsystem.
- Chapter 14, “Versatile Interface Adapter (VIA),” on page 165 describes the requirements for this legacy Apple device
- Chapter 15, “Descriptor-Based DMA,” on page 171 defines the DBDMA requirements for this Apple legacy controller.
- Appendix A, “VGA Programming Model,” on page 195 is a description of the programming model for the video graphics array which is an IBM PC legacy device.
- Appendix B, “Requirements Summary,” on page 237 is a complete list of all requirements statements imbedded in a chapter level table of contents. This appendix is a useful summary or check list for those implementing CHRP systems.

Suggested Reading

The Bibliography provides a full list of references and ordering information for these references. Within this document, the number of the reference in the bibliography is placed after the citation in brackets, “[nn]”.

This document assumes the reader has an understanding of computer architecture, the PowerPC microprocessor architecture, the current PowerPC processors, Apple and IBM compatible personal computers, Peripheral Component Interconnect (PCI) local bus, and Open Firmware. Some understanding of the current personal computer and workstation architectures is also useful. A list of suggested background reading includes:

- *Computer Architecture: A Quantitative Approach* [3]
- *The PowerPC Architecture: A Specification for a New Family of RISC Processors* [2]. Note that this specification is referred to as *The PowerPC Architecture* in the body of this document.
- *PowerPC 603 RISC Microprocessor User's Manual* [4]
- *PowerPC 603 RISC Microprocessor Technical Summary* [5]
- *PowerPC 604 RISC Microprocessor User's Manual* [6]
- *PowerPC 604 RISC Microprocessor Technical Summary* [7]
- *Technical Introduction to the Macintosh Family* [8]
- *PowerPC Reference Platform Specification*, Version 1.1 [12]
- *PCI Local Bus Specification*, Revision 2.1 [10]
- IEEE 1275, *IEEE Standard for Boot (Initialization Configuration) Firmware, Core Requirements and Practices* [9] and relevant bindings

Conventions Used in this Document

Within the body of this document lists of requirements are clearly defined. The convention used is to head each requirements list with the word "**Requirements:**" in bold face type. Following this heading are one or more requirements. These requirements may point to other standards documents, or figures or tables which conveniently show the requirement. The referenced material becomes part of the requirements in this document. Users of this document must comply with these requirements to build a standard platform. Other material in this document is supportive description of these requirements, architecture notes, or implementation notes. Architecture or implementation notes are flagged with a descriptive phrase—for example, "Hardware Implementation Note"— and followed by indented paragraphs. The descriptive material and notes provide no additional requirements and may be used for their information content.

Big-Endian numbering of bytes is used in this document, unless indicated otherwise.

Numbering of bits starts at zero for the least significant bit and continues to the most significant bit. Diagrams show the most significant bit at the left.

Typographical conventions used in this document are described in Table i on page xxvi.

Table i. Typographical Conventions

Text Element	Description of Use
Italics	Used for emphasis such as the first time a new term is used. Indicates a book title. Indicates PowerPC instruction mnemonics. Indicates Open Firmware properties, methods, and configuration variables. Indicates RTAS function and parameter names
0xnnnn	Prefix to denote hexadecimal numbers.
0bnnnn	Prefix to denote binary numbers.
nnnnn	Numbers without a prefix are decimal numbers.
0xF. . . FFF100	This hexadecimal notation represents a replication of the hexadecimal character to the right of the ellipsis to fill out the field width. For example, the address 0xF . . . FFF100 would be 0xFFFFF100 on a processor with a 32-bit address bus or 0xFFFFFFFFFFFFF100 on a processor with a 64-bit address bus.
0:9	Ranges of bits are specified by two numbers separated by a colon. The range includes the first number, all numbers in between, and the last number.

Table i. Typographical Conventions ***(Continued)***

Text Element	Description of Use
0xm-0xn	A range of addresses or values within the document is always inclusive, from m up to and including n.

Within this document each register is described using a table in the format of Table ii on page xxvii. Within this table the first column gives the size in bytes of the register. The second column gives the type of addressing either “Direct”, “Indirect”, “Sequenced”, or “Mode Selected”. The third column gives the Open Firmware “reg” property which gives the base address of a set of registers. The fourth column, “Offset Address” is the register offset from the base address given by the reg property. The fifth column gives the register read and write characteristics as viewed by the software. The sixth column gives the initial value in the register when Open Firmware passes control to the operating system. The last column gives the value to which the register is reset, if software invokes a reset of that device. For those devices which can not be reset by software, this column is “N/A”

Table ii. Register 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]	0x00	Read/Write	0x00	0x00

Each chapter has a table similar to Table iii on page xxvii which indicates that a particular device is required or optional in a configuration for different platform types. The table does not indicate that the architecture is optional. If a device is present, then the architecture must be as described in this CHRP I/O document. This short table in each chapter is excerpted from Table 2 on page 19 in the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1]. An explanation of this table and the symbols used in it is given in Section 2.5.1, “Table Description,” on page 17 in the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1] document. For convenience, the legend used in the table is repeated below:

- R: Required.
- R*: Entries categorized as R* denote specifications from which platform vendors may *choose* one or more to satisfy the requirement. All R* specifications are supported by portable and personal CHRP operating systems.
- O: Optional (refer to the definition of optional given in Section 2.6, “Options and Extensions,” on page 20).
- M: Required for systems designed to run Apple Mac OS.

Table iii. 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	

Table iii. Minimum Requirements for Alphanumeric Input Device and Pointing Device ***(Continued)***

Subsystem	Specification	Portable	Personal	Server	Description
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	

Acknowledgments

This document has come together through the work of many people in several companies. The primary responsibility for writing sections fell on employees of Apple, IBM, and Motorola. Industry review was solicited and comments were received from such companies as Advanced Micro Devices, Canon, FirePower, National Semiconductor, Toshiba and Winbond System Laboratory.

Not everyone who worked on this document can be mentioned here due to space limitations. The contributions of some key individuals are worth mentioning. The document was brought together by a team of engineers who merged existing material with new material and spent countless hours reviewing and discussing this document. This team included Art Adkins, Tim Andersen, Richard Arndt, Clinton Bauder, Mike Bell, Mark Bellon, Matt Braun, Steve Bunch, Dick Bush, Bill Coffee, Matt Denman, Simon Douglas, Brad Frey, Scott Geranen, Dan Gillen, Brian Hansche, Ron Hochsprung, Sitian Jin, John Kingman, Tom Kriz, L. LaMar, Mike Manning, Doug Maxey, Bruce Merritt, Todd Moore, Dan Neal, Kuldip Oberoi, John O'Quin, Mike Paczan, Byron Pang, Ray Pedersen, Craig Prouse, Paul Resch, Carl Runaurer, Karl Rusnock, Rick Sulack, Steve Thurber, Dave Tjon, George Towner, Wally Tuten, Mike Wiese, Keith Williamson, and Lee Wilson.

Comments on this Document

Comments on this document are welcome. Because of time and resource constraints, we will not always be able to provide responses to general industry comments. We will review your comments for possible incorporation in future versions of the specification. All comments become the property of Apple, IBM, and Motorola and may be used for any purpose whatsoever. For this reason, comments must not contain any proprietary data. Please preface your comments with the statement "These comments do not contain confidential or proprietary information and may be used for any purpose". Comments may be addressed to:

info-io@austin.ibm.com

General Requirements and Information

1

This chapter contains any requirements which apply to all devices contained in this document.

1.1 General Requirements

Requirements:

1–1. The interfaces define in this document must not be accessed by RTAS or marked as *used-by-rtas* with two exception as follows:

a. Any RTAS service that does not return to the operating system may access any interface.

b. The interfaces in Chapter 13, “Graphics,” on page 161 may be accessed by the RTAS *display-character* function, if the graphics interface in the Open Firmware device tree node does not contain the *used-by-rtas* property, but is the device indicated by the *rtas-display-device* property.

1–2. If the implementation of the devices in this document have additional registers which are not part of the programming model given in this document, then these registers must not be required for correct operation of the device by the operating system and they must all be declared in the Open Firmware device tree to prevent other devices from being assigned to their addresses.

1–3. Any controller not contained on standard adaptor cards for their I/O bus must have the *builtin* property.

1.2 Credit for Material

Some material in this specification has been copied from documentation describing existing components. Credit for this material is given in the following section.

1.2.1 From National Semiconductor Corporation

Material in some chapters was copied from *National Semiconductor PC87308VUL SuperI/O*^{TM} *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 IEEE1284 Parallel Port* [24]. and was reprinted with permission of National Semiconductor Corporation.

National Semiconductor asked that the following Life Support Policy be included.

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION.

As used herein:

1. Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

ISA DMA Controller

2

2.1 Minimum System Requirements

The Direct Memory Access (DMA) controller allows I/O devices to transfer data directly to and from memory. The minimum system requirements are shown in Table 1 on page 3, which is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* [1].

Table 1. Summary of Minimum Requirements for ISA DMA (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Direct Memory Access (DMA)	ISA	R	R	R	[20] ISA DMA must be 82378ZB compatible.

Logically, the DMA Controller is actually two controllers designated DMA Controller 1 and DMA Controller 2 as shown in Figure 1 on page 3. When DMA Controller 1 is controlling DMA transfers, channels 0-3 are capable of 8 bit count by byte transfers. When DMA Controller 2 is controlling DMA transfers, channels 5-7 are capable of 16 bit count by word (2 byte) transfers. (ISA bus masters are not affected by those restrictions.) Channel 4 is used for cascading DMA Controller 1 to DMA Controller 2 and is unusable for data transfer with an ISA slave device.

There are two modes of operation for the DMA transfers: Scatter/Gather mode and DMA mode. In DMA mode, all the control information necessary for the DMA controller to make a transfer is programmed into the DMA controller's registers. This means that a single buffer can be transferred on a given DMA channel before host processor service is required. In Scatter/Gather mode (abbreviated S/G mode), software builds a list of Scatter/Gather Descriptors (SGD's) in system memory and provides a pointer to the DMA controller that gives the start of the list. Software then issues commands to start and stop the processing of the SGD list by the DMA controller.

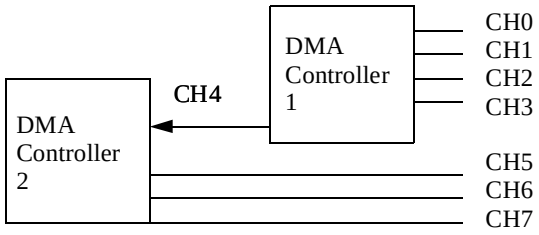


Figure 1. Cascaded DMA Controllers

Requirements:

2–1. An ISA DMA Controller must be compatible with the Intel 82378ZB System I/O (SIO), except as noted in the following requirements in this chapter.

2–2. This device (ISA DMA Controller) must be addressed only with ISA I/O addresses.

2.2 References

3. *Intel 82378ZB System I/O (SIO) Specification* [20]
4. *SL82C565 System I/O Controller With PCI Arbiter*, from Winbond Systems Laboratory [21]
5. *CHRP ISA DMA Controller Device Binding to IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [22]

2.3 ISA DMA Controller Open Firmware Properties

Requirements:

2–3. This device must be represented by the Open Firmware properties “name=dma-controller, device_type=dma-controller, compatible= chrp,dma”.

2–4. This device must have the Open Firmware properties defined in *CHRP ISA DMA Controller Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*.

2–5. The registers shown in Table 2 on page 5, Register Map for ISA DMA Device, must all be present in the device defined in this chapter.

2–6. All the reg property fields specified in Table 2 on page 5 must comply with the format given in the *ISA Bus Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*.

2–7. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

The positions in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 2 on page 5. The format is described in the referenced Open Firmware bindings and not repeated in this document.

The ISA DMA Controller register map and Reg properties are shown below in Table 2 on page 5.

Table 2. Register Map for ISA DMA Device

Offset	Traditional ISA I/O Address	Register		Reg Property
		Read	Write	
0x00	0x0000	Ch0 Current Addr Reg (CA0)	Ch0 Current Addr Reg (CA0)	
			Ch0 Base Addr Reg (BA0)	
0x01	0x0001	Ch0 Current Cnt Reg (CC0)	Ch0 Current Cnt Reg (CC0)	
			Ch0 Base Cnt Reg (BC0)	
0x02	0x0002	Ch1 Current Addr Reg (CA1)	Ch1 Current Addr Reg (CA1)	
			Ch1 Base Addr Reg (BA1)	
0x03	0x0003	Ch1 Current Cnt Reg (CC1)	Ch1 Current Cnt Reg (CC1)	
			Ch1 Base Cnt Reg (BC1)	
0x04	0x0004	Ch2 Current Addr Reg (CA2)	Ch2 Current Addr Reg (CA2)	
			Ch2 Base Addr Reg (BA2)	
0x05	0x0005	Ch2 Current Cnt Reg (CC2)	Ch2 Current Cnt Reg (CC2)	
			Ch2 Base Cnt Reg (BC2)	
0x06	0x0006	Ch3 Current Addr Reg (CA3)	Ch3 Current Addr Reg (CA3)	
			Ch3 Base Addr Reg (BA3)	
0x07	0x0007	Ch3 Current Cnt Reg (CC3)	Ch3 Current Cnt Reg (CC3)	
			Ch3 Base Cnt Reg (BC3)	
0x08	0x0008	DMA 1 Status Register (DS1)	DMA 1 Command Register (DCOM1)	reg[1],Size=16 bytes
0x09	0x0009		DMA 1 Request Register (DR1)	
0x0A	0x000A		DMA 1 Write Single Bit Mask Register (WSM1)	
0x0B	0x000B		DMA 1 Channel Mode Register (DCM1)	
0x0C	0x000C		DMA 1 Clear Byte Pointer Register (CBP1)	
0x0D	0x000D		DMA 1 Master Clear (DMC1)	
0x0E	0x000E		DMA 1 Clear Mask Register (DCLM1)	
0x0F	0x000F		DMA 1 Write All Mask Register (WAM1)	
0x00	0x0080	Reserved		
0x01	0x0081	CH2 Current Low Page Register (CLOP2	CH2 Current Low Page Register (CLOP2)	
			CH2 Base Low Page Register (BLOP2)	
0x02	0x0082	CH3 Current Low Page Register (CLOP3)	CH3 Current Low Page Register (CLOP3)	
			CH3 Base Low Page Register (BLOP3)	
0x03	0x0083	CH1 Current Low Page Register (CLOP1)	CH1 Current Low Page Register (CLOP1)	
			CH1 Base Low Page Register (BLOP1)	

Table 2. Register Map for ISA DMA Device ***(Continued)***

Offset	Traditional ISA I/O Address	Register		Reg Property
		Read	Write	
0x04	0x0084	Reserved		
0x05	0x0085	Reserved		reg[2],Size=16 bytes
0x06	0x0086	Reserved		
0x07	0x0087	CH0 Current Low Page Register (CLOP0)	CH0 Current Low Page Register (CLOP0)	
			CH0 Base Low Page Register (BLOP0)	
0x08	0x0088	Reserved		
0x09	0x0089	CH6 Current Low Page Register (CLOP6)	CH6 Current Low Page Register (CLOP6)	
			CH6 Base Low Page Register (BLOP6)	
0x0A	0x008A	CH7 Current Low Page Register (CLOP7)	CH7 Current Low Page Register (CLOP7)	
			CH7 Base Low Page Register (BLOP7)	
0x0B	0x008B	CH5 Current Low Page Register (CLOP5)	CH5 Current Low Page Register (CLOP5)	
			CH5 Base Low Page Register (BLOP5)	
0x0C-0x0E	0x008C-0x008E	Reserved		
0x0F	0x008F	Reserved		
0x00	0x00C0	Reserved		
0x01	0x00C1	Reserved		
0x02	0x00C2	Reserved		
0x03	0x00C3	Reserved		
0x04	0x00C4	Ch5 Current Addr Reg (CA5)	Ch5 Current Addr Reg (CA5)	
			Ch5 Base Addr Reg (BA5)	
0x05	0x00C5	Reserved		
0x06	0x00C6	Ch5 Current Cnt Reg (CC5)	Ch5 Current Cnt Reg (CC5)	
			Ch5 Base Cnt Reg (BC5)	
0x07	0x00C7	Reserved		
0x08	0x00C8	Ch6 Current Addr Reg (CA6)	Ch6 Current Addr Reg (CA6)	
			Ch6 Base Addr Reg (BA6)	
0x09	0x00C9	Reserved		
0x0A	0x00CA	Ch6 Current Cnt Reg (CC6)	Ch6 Current Cnt Reg (CC6)	
			Ch6 Base Cnt Reg (BC6)	
0x0B	0x00CB	Reserved		reg[3], Size=32 bytes
0x0C	0x00CC	Ch7 Current Addr Reg (CA7)	Ch7 Current Addr Reg (CA7)	
			Ch7 Base Addr Reg (BA7)	
0x0D	0x00CD	Reserved		

Table 2. Register Map for ISA DMA Device ***(Continued)***

Offset	Traditional ISA I/O Address	Register		Reg Property
		Read	Write	
0x0E	0x00CE	Ch7 Current Cnt Reg (CC7)	Ch7 Current Cnt Reg (CC7)	
			Ch7 Base Cnt Reg (BC7)	
0x0F	0x00CF	Reserved		
0x10	0x00D0	DMA 2 Status Register (DS2)	DMA 2 Command Register (DCOM2)	
0x11	0x00D1	Reserved		
0x12	0x00D2		DMA 2 Request Reg (DR2)	
0x13	0x00D3	Reserved		
0x14	0x00D4		DMA 2 Write Single Bit Mask Register (WSM2)	
0x15	0x00D5	Reserved		
0x16	0x00D6		DMA 2 Channel Mode Register (DCM2)	
0x17	0x00D7	Reserved		
0x18	0x00D8		DMA 2 Clear Byte Pointer Reg (CBP2)	
0x19	0x00D9	Reserved		
0x1A	0x00DA		DMA 2 Master Clear Register (DMC2)	
0x1B	0x00DB	Reserved		
0x1C	0x00DC		DMA 2 Clear Mask (DCLM2)	
0x1D	0x00DD	Reserved		
0x1E	0x00DE		DMA 2 Write All Mask (WAM2)	
0x1F	0x00DF	Reserved		
0x00	0x040A	Scatter/Gather Interrupt Status Register (SGIS)		reg[4], Size=1 byte
0x00	0x040B		DMA 1 Extended Mode Register (DCEM1)	reg[5], Size=1 byte
0x00	0x0410		CH0 Scatter/Gather Command (SGC0)	
0x01	0x0411		CH1 Scatter/Gather Command (SGC1)	
0x02	0x0412		CH2 Scatter/Gather Command (SGC2)	
0x03	0x0413		CH3 Scatter/Gather Command (SGC3)	
0x04	0x0414	Reserved		
0x05	0x0415		CH5 Scatter/Gather Command (SGC5)	
0x06	0x0416		CH6 Scatter/Gather Command (SGC6)	
0x07	0x0417		CH7 Scatter/Gather Command (SGC7)	
0x08	0x0418	CH0 Scatter/Gather Status (SGS0)		

Table 2. Register Map for ISA DMA Device ***(Continued)***

Offset	Traditional ISA I/O Address	Register		Reg Property
		Read	Write	
0x09	0x0419	CH1 Scatter/Gather Status (SGS1)		reg[6], Size=48 bytes
0x0A	0x041A	CH2 Scatter/Gather Status (SGS2)		
0x0B	0x041B	CH3 Scatter/Gather Status (SGS3)		
0x0C	0x041C	Reserved		
0x0D	0x041D	CH5 Scatter/Gather Status (SGS5)		
0x0E	0x041E	CH6 Scatter/Gather Status (SGS6)		
0x0F	0x041F	CH7 Scatter/Gather Status (SGS7)		
0x10-0x13	0x0420-0x0423	CH0 Scatter/Gather Descriptor Table Pointer (SGPTR0)		
0x14-0x17	0x0424-0x0427	CH1 Scatter/Gather Descriptor Table Pointer (SGPTR1)		
0x18-0x1B	0x0428-0x042B	CH2 Scatter/Gather Descriptor Table Pointer (SGPTR2)		
0x1C-0x1F	0x042C-0x042F	CH3 Scatter/Gather Descriptor Table Pointer (SGPTR3		
0x20-0x23	0x0430-0x0433	Reserved		
0x24-0x27	0x0434-0x0437	CH5 Scatter/Gather Descriptor Table Pointer (SGPTR5)		
0x28-0x2B	0x0438-0x043B	CH6 Scatter/Gather Descriptor Table Pointer (SGPTR6)		
0x2C-0x2F	0x043C-0x043F	CH7 Scatter/Gather Descriptor Table Pointer (SGPTR7)		
0x00	0x481	CH 2 Current High Page Register (CHIP2)	CH 2 Current High Page Register (CHIP2)	reg[7], Size=11 bytes
			CH 2 Base High Page Register (BHIP2)	
0x01	0x482	CH 3 Current High Page Register (CHIP3)	CH 3 Current High Page Register (CHIP3)	
			CH 3 Base High Page Register (BHIP3)	
0x02	0x483	CH 1Current High Page Register (CHIP1)	CH 1 Current High Page Register (CHIP1)	
			CH 1 Base High Page Register (BHIP1)	
0x03-0x05	0x484-0x486	Reserved		
0x06	0x487	CH 0 Current High Page Register (CHIP0)	CH 0 Current High Page Register (CHIP0)	
			CH 0 Base High Page Register (BHIP0)	
0x07	0x488	Reserved		

Table 2. Register Map for ISA DMA Device ***(Continued)***

Offset	Traditional ISA I/O Address	Register		Reg Property
		Read	Write	
0x08	0x489	CH 6 Current High Page Register (CHIP6)	CH 6 Current High Page Register (CHIP6)	
			CH 6 Base High Page Register (BHIP6)	
0x09	0x48A	CH 7 Current High Page Register (CHIP7)	CH 7 Current High Page Register (CHIP7)	
			CH 7 Base High Page Register (BHIP7)	
0x0A	0x48B	CH 5 Current High Page Register (CHIP5)	CH 5 Current High Page Register (CHIP5)	
			CH 5 Base High Page Register (BHIP5)	
0x00	0x4D6		DMA 2 Extended Mode Register (DCEM2)	reg[8], Size=1 byte

2.4 DMA Channel Registers

2.4.1 Current Address Register 0 (CA0)

Table 3. Current Address Register 0 (CA0) Characteristics

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

DMA Channel 0 has a 16 bit Base Address Register, BA0, and a 16 bit Current Address Register, CA0. The CA0 is described here. CA0 contains the address the DMA channel is currently using. It is used for both scatter/gather and DMA operations. Before writing to this register, the DMA Clear Byte Pointer Register (Section 2.5.15, “DMA 1 Clear Byte Pointer Register (CBP1),” on page 23) must be written to reset an internal flip flop that prepares this 16 bit register for two successive byte writes. CA0 and BA0 are located at the same address. CA0 is incremented or decremented with each transfer depending on the DMA channel 0 mode settings. BA0 stores the initial value of CA0 (Section 2.4.3, “Base Address Register 0 (BA0),” on page 10). CA0 is autoinitialized back to its starting value after a terminal count, TC, if autoinitialize mode is set in the DMA Channel Mode Register 1 (Section 2.5.3, “DMA 1 Channel Mode Register (DCM1),” on page 16.)

Table 4. Current Address Register (CA0)

Bit	Function
15 -8	**High order address byte**: This is the second byte written after a DMA Clear Byte Pointer command.
7-0	**Low order address byte**: This is the first byte written after a DMA Clear Byte Pointer command

2.4.2 Current Address Registers (CA1-CA7)

Current Address Registers 1-7, CA1-CA7, are identical to Current Address Register 0 (Section 2.4.1, “Current Address Register 0 (CA0),” on page 9) with the following exceptions:

1. they are associated with the other DMA channels as indicated by their numeric suffix (i.e. CA5 is the Current Address Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5
2. The address programmed into CA5-CA7 must be an even address that has been shifted right by one bit. The High Page register (Section 2.4.13, “Current High Page Register 0 (CHIP0),” on page 13) and Low Page register (Section 2.4.9, “Current Low Page Register 0 (CLOP0),” on page 12) are not shifted by software. As an example, for address 0x1FE, the address programmed into the Current Address register is 0xFF.

2.4.3 Base Address Register 0 (BA0)

Table 5. Base Address Register 0 (BA0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
2 byte	Indirect	reg[1]	0x00	Write Only	undefined	N/A

DMA Channel 0 has a 16-bit Base Address Register, BA0, which stores the initial value of CA0 for DMA operations. Before writing to this register, the DMA Clear Byte Pointer Register must be written to reset an internal flip flop that prepares this 16 bit register for two successive byte writes (Section 2.5.15, “DMA 1 Clear Byte Pointer Register (CBP1),” on page 23). CA0 and BA0 are located at the same address. For DMA operations, BA0 is used to reload the starting value for CA0 after a terminal count, TC, if autoinitialize mode is set in the DMA Channel Mode Register 1 (Section 2.5.3, “DMA 1 Channel Mode Register (DCM1),” on page 16). During scatter/gather operations, the DMA controller loads a reserve buffer into BA0.

Table 6. Base Address Register (BA0)

Bit	Function
15 -8	**High order address byte**: this is the second byte written after a DMA Clear Byte Pointer command.
7-0	**Low order address byte**: This is the first byte written after a DMA Clear Byte Pointer command

2.4.4 Base Address Registers 1-7 (BA1-BA7)

Base Address Registers 1-7, BA1-BA7, are identical to Base Address Register 0 (Section 2.4.3, “Base Address Register 0 (BA0),” on page 10) with the following exceptions:

1. they are associated with the other DMA channels as indicated by their numeric suffix (i.e. BA5 is the Base Address Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5
2. The address programmed into BA5-BA7 must be an even address that has been shifted right by one bit. The High Page register (1.3.13 on page 12) and Low Page register (1.3.9 on page 11) are not shifted by software. As an example, for address 0x1FE, the address programmed into the Base Address register is 0xFF.

2.4.5 Current Count Register 0 (CC0)

Table 7. Current Count Register 0 (CC0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
2 byte	Indirect	reg[1]	0x01	Read/Write	undefined	N/A

DMA Channel 0 has a 16 bit Current Count Register, CC0, and a 16 bit Base Count Register 0, BC0. CC0 is described here. It contains the “number of bytes to be transferred - 1”. It is used for both scatter/gather and DMA operations. Before writing to this register, the DMA Clear Byte Pointer Register must be written to reset an internal flip flop that prepares this 16 bit register for two successive byte writes (Section 2.5.15, “DMA 1 Clear Byte Pointer Register (CBP1),” on page 23). CC0 and BC0 are located at the same address. CC0 is decremented with each transfer. BC0 is write only and stores the initial value of CC0 (Section 2.4.7, “Base Count Register 0 (BC0),” on page 12). When CC0 goes from 0x0000 to 0xFFFF a terminal count, TC, is generated. If autoinitialize mode is set in the DMA Channel Mode Register 1 (Section 2.5.3, “DMA 1 Channel Mode Register (DCM1),” on page 16) CC0 is autoinitialized to Base Count Register 0 (Section 2.4.7, “Base Count Register 0 (BC0),” on page 12) after a TC. During scatter/gather mode operations, CC0 contains the 16 bits of the current byte count.

Table 8. Current Count Register (CC0)

Bit	Function
15 -8	**High order byte**
7-0	**Low order byte**

2.4.6 Current Count Registers 1-7 (CC1-CC7)

Current Count Registers 1-7, CC1-CC7, are identical to Current Count Register 0 (Section 2.4.5, “Current Count Register 0 (CC0),” on page 11) with the following exceptions:

1. they are associated with the other DMA channels as indicated by their numeric suffix (i.e. CC5 is the Current Count Register for DMA channel 5). They have unique addresses shown in Table 2 on page 5
2. CC1 through CC3 initially contain the number of bytes to be transferred -1, and decrement with each byte transferred. Note: some hardware designs may allow CC1 through CC3 to count word (16-bit) transfers. The architecture only requires them to count byte transfers.
3. CC5 through CC7 initially contain the number of words (16 bits) to be transferred - 1, and decrement with each word transferred. Note: some hardware designs may allow CC5 through CC7 to count byte transfers. The architecture only requires them to count word transfers.
4. DMA Channel Mode Register 2 controls autoinitialize for channels 5-7.

2.4.7 Base Count Register 0 (BC0)

Table 9. Base Count Register (BC0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
2 byte	Indirect	reg[1]	0x01	Write Only	undefined	N/A

DMA Channel 0 has a 16 bit Base Count Register 0, BC0. The “number of bytes to be transferred-1” is programmed into BC0. It is used for both DMA and scatter/gather operations. Before writing to this register, the DMA Clear Byte Pointer Register (Section 2.5.15, “DMA 1 Clear Byte Pointer Register (CBP1),” on page 23) must be written to reset an internal flip flop that prepares this 16 bit register for two successive byte writes. CC0 and BC0 are located at the same address. BC0 is write only and stores the initial value of CC0 (Section 2.4.5, “Current Count Register 0 (CC0),” on page 11). In scatter/gather mode, BC0 is used to store a reserve buffer.

Table 10. Base Count Register (BC0)

Bit	Function
15 -8	**High order byte**
7-0	**Low order byte**

2.4.8 Base Count Registers 1-7 (BC1-BC7)

Base Count Registers 1-7, BC1-BC7, are identical to Base Count Register 0 (Section 2.4.7, “Base Count Register 0 (BC0),” on page 12) with the following exceptions:

1. they are associated with the other DMA channels as indicated by their numeric suffix (i.e. BC5 is the Base Count Register for DMA channel 5). They have unique addresses shown in Table 2 on page 5.
2. BC1 through BC3 contain the number of bytes to be transferred-1. Note: some hardware designs may allow BC1-BC3 to contain the number of words (16 bits) to be transferred. The architecture only requires them to handle bytes.
3. BC5 through BC7 contain the number of words to be transferred. Note: some hardware designs may allow BC5-BC7 to contain the number of bytes to be transferred. The architecture only requires them to handle words.

2.4.9 Current Low Page Register 0 (CLOP0)

Table 11. Current Low Page Register 0 (CLOP0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[2]	0x07	Read/Write	undefined	N/A

Each DMA channel has a Current Low Page Register, Base Low Page Register, Current High Page Register and Base High Page Register. The Current Low Page Register 0, CLOP0, provides the second most significant byte, bits 23-16, of the 32 bit current address for both DMA and Scatter/Gather (S/G) transfers on DMA channel 0. If the Current Address Register 0, CA0, overflows or underflows during a S/G transfer, the CLOP0 register must be incremented or decremented accordingly. CLOP0 need not increment/decrement during a DMA operation, although some hardware designs may allow it. The Current Low Page Register, CLOP0, and the Base Low Page Register, BLOP0 (Section 2.4.11, "Base Low Page Register 0 (BLOP0)," on page 13), are at the same address and are written concurrently.

2.4.10 Current Low Page Register 1-7 (CLOP1-CLOP7)

Current Low Page Registers 1-7, CLOP1-CLOP7, are identical to Current Low Page Register 0 (Section 2.4.9, "Current Low Page Register 0 (CLOP0)," on page 12) with the exception they are associated with the other DMA channels as indicated by their numeric suffix (i.e. CLOP5 is the Current Low Page Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5.

2.4.11 Base Low Page Register 0 (BLOP0)

Table 12. Base Low Page Register 0 (BLOP0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[2]	0x07	Write Only	undefined	N/A

Each DMA channel has a Base Low Page Register which holds the initial value programmed into its Current Low Page Register. Base Low Page Register 0, BLOP0, is the Base Low Page Register for channel 0. If autoinitialize is enabled in the DMA Channel Mode register (Section 2.5.3, "DMA 1 Channel Mode Register (DCM1)," on page 16), BLOP0 is copied into CLOP0 after a Terminal Count (TC). During a scatter/gather operation, BLOP0 is used to hold a reserve buffer.

2.4.12 Base Low Page Register 1-7 (BLOP1-BLOP7)

Base Low Page Registers 1-7, BLOP1-BLOP7, are identical to Base Low Page Register 0 (Section 2.4.11, "Base Low Page Register 0 (BLOP0)," on page 13) with the exception they are associated with the other DMA channels as indicated by their numeric suffix (i.e. BLOP5 is the Base Low Page Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5

2.4.13 Current High Page Register 0 (CHIP0)

Table 13. Current High Page Register 0 (CHIP0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[7]	0x06	Read/Write	undefined	N/A

Each DMA channel has a Current Low Page Register, Base Low Page Register, Current High Page Register and Base High Page Register. The Current High Page Register 0, CHIP0, provides the most significant byte, bits 31-24, of the 32 bit current address for both DMA and Scatter/Gather (S/G) transfers on DMA channel 0. If the Current Low Page Register 0, CLOP0, overflows or underflows during a S/G transfer, the CHIP0 register must be incremented or decremented accordingly. CHIP0 need not increment/decrement during a DMA operation, although some hardware designs may allow it. The Current High Page Register, CHIP0, and the Base High Page Register, BHIP0 (2.4.15 on page 14), are at the same address and are written concurrently.

CHIP0 is set to 0x00 after the Low Page Register (2.4.9 on page 12) is programmed. CHIP0 must be programmed after the Low Page Register.

2.4.14 Current High Page Register 1-7 (CHIP1-CHIP7)

Current High Page Registers 1-7, CHIP1-CHIP7, are identical to Current High Page Register 0 (2.4.13 on page 13) with the exception they are associated with the other DMA channels as indicated by their numeric suffix (i.e. CHIP5 is the Current High Page Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5

2.4.15 Base High Page Register 0 (BHIP0)

Table 14. Base High Page Register 0 (BHIP0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[7]	0x06	Write Only	Undefined	N/A

Each DMA channel has a Base High Page Register which holds the initial value programmed into its Current High Page Register. Base High Page Register 0, BHIP0, is the Base High Page Register for channel 0. If autoinitialize is enabled in the DMA Channel Mode register (2.5.3), BHIP0 is copied into CHIP0 after a Terminal Count (TC). During a scatter/gather operation, BHIP0 is used to hold a reserve buffer.

2.4.16 Base High Page Register 1-7 (BHIP1-BHIP7)

Base High Page Registers 1-7, BHIP1-BHIP7, are identical to Base High Page Register 0 (2.4.15 on page 14) with the exception they are associated with the other DMA channels as indicated by their numeric suffix (i.e. BHIP5 is the Base High Page Register for DMA channel 5). They have unique addresses as shown in Table 2 on page 5.

2.5 DMA Controller Registers

2.5.1 DMA 1 Command Register (DCOM1)

Table 15. DMA 1 Command Register (DCOM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x08	Write Only	0x00	0x00 See 2.5.17 and 2.5.18

This write-only register controls the configuration of DMA Controller 1 which controls DMA channels 0-3. DMA Controller 1 is cascaded through DMA channel 4 of DMA Controller 2. It is cleared by a hardware reset or a write to the Master Clear register described in Section 2.5.17, “DMA 1 Master Clear Register (DMC1),” on page 24.

Table 16. DMA Command Register

Bit	Function
7	Set to 0. In some hardware implementations, bit 7 is the DACK# assert level. This bit determined the polarity of the DACK# signal presented to the ISA bus.
6	Set to 0. In some hardware implementations, bit 6 is the DREQ assert level. This bit determined how signals from the ISA DREQ line are decoded and recognized by the ISA DMA controller.
5	Reserved, set to 0.
4	**DMA Group Arbitration Priority**: Channel groups are assigned either fixed or rotating arbitration priority. When PCIRST# is asserted, this bit assumes a value of 0 and the channel group controlled by DMA controller 1 (channels 0-3) is initialized to fixed priority. When this bit is set to 1, this channel group is assigned rotating priority.
3	Reserved, set to 0.
2	**DMA Channel Group Enable**: When a 1 is written to this bit, the channel group including DMA channels 0-3 is disabled.
1-0	Reserved, set to 0

Requirements:

2–8. For ISA DMA, the DACK# signal must be active low.

2–9. For ISA DMA, the DREQ signal must be active high.

2.5.2 DMA 2 Command Register (DCOM2)

Table 17. DMA 2 Command Register (DCOM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x10	Write Only	0x00	0x00 See 2.5.17 and 2.5.18

DMA Command Register 2, DCOM2, is identical to the DMA Command Register 1, DCOM1, with the following differences:

1. DCOM2 controls channels 4-7whereas DCOM1 controls channels 0-3.
2. Setting bit 2 in DCOM2, DMA Channel Group Enable, will disable DMA controller 1 which is controlled by DCOM1. Setting bit 2 in DCOM2 will disable DMA channels 0-3 and 4-7.

2.5.3 DMA 1 Channel Mode Register (DCM1)

Table 18. DMA 1 Channel Mode Register (DCM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[1]	0x0B	Write Only	Undefined	N/A

Each DMA channel has a DMA Channel Mode Register. The channel mode registers for channels 0-3 are accessed through DCM1. Bits 1-0 of DCM1 select one of four channels. The value of the DMA Channel Select, bits 1-0, determines which channel mode register bits 7-2 are written to. Writing to DCM1 sets the DMA transfer type, transfer mode, address increment/decrement and autoinitialization for the selected channel.

Table 19. DMA Channel Mode Register

Bit	Function
7-6	**DMA Transfer Mode**: 00 = Demand Mode 01=Single Mode 10=Block Mode 11=Cascade Mode
5	**Address Increment/Decrement Select**. 0=increment, 1=decrement.
4	**Autoinitialize Enable**: When this bit is 1, the DMA controller loads the current low page, current high page, current address and current count information from their respective base registers following a terminal count, TC. When this bit is 0, these registers will not be restored after a TC.
3-2	**DMA Transfer Type** 00=Verify Transfer 01=Write Transfer 10=Read Transfer 11=Illegal

Table 19. DMA Channel Mode Register ***(Continued)***

Bit	Function
1-0	**DMA Channel Select** 00=Channel 0 (4) 01=Channel 1 (5) 10=Channel 2 (6) 11=Channel 3 (7)

2.5.4 DMA 2 Channel Mode Register (DCM2)

Table 20. DMA 2 Channel Mode Register (DCM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[3]	0x16	Write Only	Undefined	N/A

This register is identical to the DCM1 with the following two differences:

1. DCM2 controls channels 4-7 whereas DCM1 controls channels 0-3.
2. The DMA Channel Mode Register for channel 4 (indirectly addressed through DCM2) must be set to cascade mode since this channel is used as the cascade for channels 0-3.

2.5.5 DMA 1 Extended Mode Register (DCEM1)

Table 21. DMA 1 Extended Mode Register (DCEM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[5]	0x00	Write Only	0b000000XX	N/A

Each channel has a DMA Extended Mode Register. This register is used to program DMA timing modes. Bits [1:0] select the appropriate Channel Extended Mode Register. DCEM1 is used for channels 0-3. DMA 2 Channel Extended Mode Register (DCEM2) described in Section 2.5.6, “DMA 2 Extended Mode Register (DCEM2),” on page 19 is used for channels 5-7.

Table 22. DMA Extended Mode Register

Bit	Function
7	Must be 0.
6	Must be 0. Was TC (EOP) Input/Output Selection in some hardware designs**:** The architecture only requires TC to be an output of the ISA DMA controller.
5-4	**DMA Cycle Timing Mode**. See Table 23 on page 18

Table 22. DMA Extended Mode Register ***(Continued)***

Bit	Function
3-2	**Addressing Mode** Must be set to 0b00 for channels 0-3. (8-bit count by byte addressing mode) Must be set to 0b01 for channels 5-7 (16-bit count by word addressing mode)
1-0	**DMA Channel Select** 00=Channel 0 01=Channel 1 (5) 10=Channel 2 (6) 11=Channel 3 (7)

2.5.5.1 Terminal Count

Requirements:

2–10. For ISA DMA, software must assume the terminal count (TC) is an output from the DMA controller. This means some ISA devices may not work if they depend on TC being an input to the DMA controller.

2–11. For ISA DMA, a hardware design must not produce an error if 0b0 is written to bit 6 of DCEM1 or DCEM2.

2.5.5.2 DMA Timing Modes

Four DMA transfer timings are selectable: ISA compatible, type “A”, type “B”, and type “F”. The timings given assume an ISA SYSCLK of 8.33 Mhz

.

Table 23. DMA Timing Modes

Bits 5-4	DMA Timing Mode
0b00	**ISA-Compatible Timing:** Compatible timing is provided for DMA slave devices that cannot support the higher speed modes (A, B or F). It uses 9 ISA SYSCLKs during a single cycle transfer (1080ns/single cycle). During the repeated portions of BLOCK and DEMAND mode transfers it uses 8 SYSCLKs (960ns/cycle).
0b01	**Type “A” Timing:** The I/O setup portion of the cycle is the same as with compatible timing (9 SYSCLKS - 1080ns/single cycle). During the repeated portions of BLOCK and DEMAND mode transfers type “A” timing uses 6 SYSCLKs (720ns/cycle).
0b10	**Type “B” Timing:** The I/O setup portion of the cycle is the same as with compatible timing (9 SYSCLKS - 1080ns/single cycle). During the repeated portions of BLOCK and DEMAND mode transfers type “B” timing uses 5 SYSCLKs (600ns/cycle).
0b11	**Type “F” Timing:** The I/O setup portion of the cycle is the same as with compatible timing (9 SYSCLKS - 1080ns/single cycle). During the repeated portions of BLOCK and DEMAND mode transfers type “F” timing uses 3 SYSCLKs (360ns/cycle). This is the fastest mode providing 8.33 Mbytes/sec of data transfer.

2.5.5.3 DMA Addressing Modes

DMA Controller 1 is only required to support 8-bit count by byte mode. DMA Controller 2 is only required to support 16-bit count by word (2-byte) mode. The ISA DMA Controller is not required to support 16-bit count by bytes addressing mode.

Requirements:

2–12. For operations involving ISA DMA slave devices, software must assume the hardware only allows addressing modes as follows:

a. Channels 0-3 are 8-bit count by bytes addressing mode. The DMA address, count, and page registers must be programmed accordingly.

b. Channels 5-7 are 16-bit count by word (2 bytes) addressing mode. The DMA address, count, and page registers must be programmed accordingly.

c. 16-bit count by bytes addressing mode is not supported.

2–13. For ISA DMA, a hardware design must not produce an error if 0b00 is written to bits [3:2] of DCEM1, or if 0b01 is written to bits [3:2] of DCEM2.

2.5.6 DMA 2 Extended Mode Register (DCEM2)

Table 24. DMA 2 Extended Mode Register (DCEM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[8]	0x00	Write Only	0b000001XX	N/A

This register is identical to the DCEM1 with the following differences:

1. DCEM2 controls channels 5-7 whereas DCEM1 controls channels 0-3. Bits [1:0] must not be set to 0b00.
2. Bits [3:2] must be written with 0b01 to indicate 16-bit count by word transfers.

2.5.7 DMA 1 Request Register (DR1)

Table 25. DMA 1 Request Register (DR1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[1]	0x09	Write Only	0b000000XX	0b000000XX See 2.5.17 and 2.5.18

The host processor can initiate a DMA request on a DMA channel 0-3 by identifying the desired DMA channel in bits 1-0, DMA Channel Select, of this register while setting bit 2, DMA Channel Service Request, to a 1. These requests are treated as if the ISA DREQ signal was asserted for the DMA channel. These host processor initiated requests are non-maskable and subject to prioritization by the priority encoder network.

Table 26. DMA Request Register

Bit	Function
7-3	Reserved (Must be 0)
2	**DMA Channel Service Request**: Writing a zero to this bit resets the system initiated DMA channel request. Writing a one to this bit initiates a DMA channel request on the channel indicated by bits 1-0, DMA Channel Select.
1-0	**DMA Channel Select** 00=Channel 0 01=Channel 1 (5) 10=Channel 2 (6) 11=Channel 3 (7)

2.5.8 DMA 2 Request Register (DR2)

Table 27. DMA 2 Request Register (DR2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x12	Write Only	0b000000XX	0b000000XX See 2.5.17 and 2.5.18

This register is identical to DR1 with the following difference:

1. DR2 initiates DMA requests for channels 5-7 as indicated in Table 23. Bits [1:0] must not be set to 0b00.

2.5.9 DMA 1 Write Single Bit Mask Register (WSM1)

Table 28. DMA 1 Write Single Bit Mask Register (WSM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[1]	0x0A	Write Only	0b000001XX	N/A

Incoming DMA channel service requests (DREQs) for channels 0-3 can be individually enabled and disabled using this register. Upon hardware reset and OF transfer to the OS, all channels are masked.

Table 29. DMA 1 Write Single Bit Mask Register

Bit	Function
7-3	Reserved (Must be 0)
2	**Channel Mask Select**: When this bit is a 1, the DMA channel selected by bits 1-0, DMA Channel Select, will have its DREQ line masked off. The DMA channel will be disabled. When this bit is a 0, the DMA channel selected by the DMA Channel Select field will be enabled.

Table 29. DMA 1 Write Single Bit Mask Register ***(Continued)***

Bit	Function
1-0	**DMA Channel Select** 00=Channel 0 (4) 01=Channel 1 (5) 10=Channel 2 (6) 11=Channel 3 (7)

2.5.10 DMA 2 Write Single Bit Mask Register (WSM2)

Table 30. DMA 2 Write Single Bit Mask Register (WSM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Indirect	reg[3]	0x14	Write Only	0b000001XX	N/A

This register is identical to WSM1 with the following differences.

1. For DMA Controller 2, WSM2 controls enabling and disabling DMA channels 4-7.
2. For DMA Controller 2, disabling DMA channel 4 will disable channels 0-3 since they are cascaded through channel 4.

Note: Upon hardware reset and OF transfer to the OS, all channels are masked

2.5.11 DMA 1 Write All Mask Bits Register (WAM1)

Table 31. DMA 1 Write All Mask Register (WAM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x0F	Write Only	0x0F	0x0F See 2.5.17 and 2.5.18

The difference between this register and the DMA 1 Write Single Bit Mask Register, WSM1, is that four DMA channels can be enabled or disabled selectively in one write with the WAM1 register. With WSM1, only one DMA channel can be enabled or disabled at a time.

Table 32. DMA 1 Write all Mask Bits Register

Bit	Function
7-4	Reserved (Must be 0)
3-0	**Channel Mask Select:** For WSM1, bit 0 enables/disables channel 0. (1 disables a channel. 0 enables a channel.) Bit 1 enables/disables channel 1. Bit 2 enables/disables channel 2. Bit 3 enables/disables channel 3.

Requirements:

2–14. Software must assume the DMA Write All Mask Bits Registers (WAM1, WAM2) are write-only.

Note: It is recommended that hardware designs allow this register to be read/write, to facilitate software debug. If it is Read/Write, hardware returns the current value of this register, not the last value written by software (they could be different).

2.5.12 DMA 2 Write All Mask Bits Register (WAM2)

Table 33. DMA 2 Write All Mask Bits Register (WAM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x1E	Write Only	0x0F	0x0F See 2.5.17 and 2.5.18

This register is identical to WAM1 with the following differences:

1. For DMA Controller 2, bit 0 of the WSM2 enables/disables channel 4. (1 disables a channel. 0 enables a channel.) Bit 1 enables/disables channel 5. Bit 2 enables/disables channel 6. Bit 3 enables/disables channel 7.
2. For DMA Controller 2, disabling DMA channel 4 will disable channels 0-3 since they are cascaded through channel 4.

2.5.13 DMA 1 Status Register (DS1)

Table 34. DMA 1 Status Register (DS1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x08	Read Only	0x00	0x00 See 2.5.17 and 2.5.18

Each DMA controller has a read-only DMA Status Register that indicates which DMA channels have reached terminal count and which DMA channels have a pending DMA request. Bits 3-0 are set to indicate that their corresponding channels have reached terminal count, TC. These four bits are reset when the DMA Status Register is read.

Table 35. DMA 1 Status Register

Bit	Function
7-4	**Channel Request Status:** These bits are set when there is a corresponding DMA request active on the ISA bus. When a valid DMA request is pending for a channel (on the ISA DREQ signal line for the channel), the appropriate bit is set here. For DMA Controller 1, DREQ0 set bits 4, DREQ1 sets bit 5, DREQ2 sets bit 6 and DREQ3 sets bit 7.

Table 35. DMA 1 Status Register ***(Continued)***

Bit	Function
3-0	**Channel Terminal Count Status**: When a channel reaches its terminal count, TC, its corresponding bit in this field is set. For DMA Controller 1, channel 0's TC status is on bit 0. Channel 1's TC status is on bit 1. Channel 2's TC status is on bit 2. Channel 3's TC status is on bit 3.

2.5.14 DMA 2 Status Register (DS2)

Table 36. DMA 2 Status Register (DS2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x10	Read Only	0x00	0x00 See 2.5.17 and 2.5.18

This register is identical to DS1 with the following differences:

1. For DMA Controller 2, DREQ5 sets bit 5, DREQ6 sets bit 6 and DREQ7 sets bit 7. DMA channel 4 is the cascade and therefore bit 4 is always 0 for DS2.
2. For DMA Controller 2, Channel 5's TC status is on bit 1. Channel 6's TC status is on bit 2. Channel 7's TC status is on bit 3. Since Channel 4 is the cascade, bit 0 is always 0.

2.5.15 DMA 1 Clear Byte Pointer Register (CBP1)

Table 37. DMA 1 Clear Byte Pointer Register (CBP1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x0C	Write Only	Undefined	N/A

A clear byte pointer command is executed prior to writing new address or count information to the DMA controller or reading current address or count information from the DMA controller. After doing a clear byte pointer command, a read or write to the address or count registers will access the least significant byte. The second access to the same address will give the most significant byte of the address or count. CBP1 is for DMA channels 0-3

Bit 7-0 (Clear Byte Pointer): The content of this register is meaningless. The act of writing to its address accomplishes the clear byte pointer command.

2.5.16 DMA 2 Clear Byte Pointer Register (CBP2)

Table 38. DMA 2 Clear Byte Pointer Register (CBP2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x18	Write Only	Undefined	N/A

This register accomplishes the same purpose as CBP1 with the following exceptions:

1. CBP2 performs the clear byte pointer command for DMA controller 2 which controls channels 5-7.

2.5.17 DMA 1 Master Clear Register (DMC1)

Table 39. DMA 1 Master Clear Register (DMC1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x0D	Write Only	Undefined	N/A

The DMA Master Clear Register 1, DMC1, allows software to perform the equivalent of a hardware reset of DMA Controller 1 (Channels 0-3) under software control. Writing to DMC1resets the Command, Status, Request, and Internal First/Last flip-flop, and sets the Mask Register.

Bit 7-0 (Master Clear): The contents of this register have no meaning. Writing to this register accomplishes a soft reset which is logically equivalent to a hardware reset for DMA controller 1.

2.5.18 DMA 2 Master Clear Register (DMC2)

Table 40. DMA 2 Master Clear Register (DMC2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x1A	Write Only	Undefined	N/A

This register, is identical to DMC1 with the following differences:

1. For DMA Controller 2 (Channels 4-7), DMC2 performs the equivalent of a hardware reset.

2.5.19 DMA 1 Clear Mask Register (DCLM1)

Table 41. DMA 1 Clear Mask Register1 (DCLM1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[1]	0x0E	Write Only	Undefined	N/A

Writing to this register clears any masks which may be in place for DMA channels 0-3. The value of the data for the write is irrelevant.

2.5.20 DMA 2 Clear Mask Register (DCLM2)

Table 42. DMA 2 Clear Mask Register (DCLM2) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[3]	0x1C	Write Only	Undefined	N/A

Writing to this register clears any masks which may be in place for DMA channels 4-7. The value of the data for the write is irrelevant.

2.6 Scatter/Gather Registers

Requirements:

2–15. During Scatter/Gather operations, the High Page Registers (described in Section 2.4.13, “Current High Page Register 0 (CHIP0),” on page 13 through Section 2.4.14, “Current High Page Register 1-7 (CHIP1-CHIP7),” on page 14) and Low Page Registers (described in Section 2.4.9, “Current Low Page Register 0 (CLOP0),” on page 12 through Section 2.4.10, “Current Low Page Register 1-7 (CLOP1-CLOP7),” on page 13) must increment appropriately as part of the full 32 bit transfer address.

2.6.1 The Scatter/Gather Descriptor

Figure 2 on page 26 gives the format of the Scatter/Gather Descriptor (abbreviated as SGD). A contiguous list of SGD’s in real memory makes up a SGD Table. A SGD must be aligned on a 4-byte boundary.

DWORD0 of a SGD contains the address of a buffer in memory that is to be transferred to or from an ISA I/O device. If the transfer mode of the operation is 16-bit count by word, (i.e. one of channels 5-7 is used), then the address of that buffer must be even (the lsb of the address is 0).

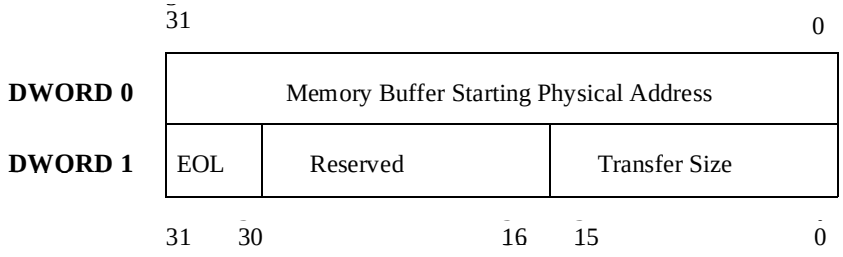
Requirements:

2–16. Scatter/Gather Descriptor: The Memory Buffer Starting Physical Address field in the SGD must be programmed as follows:

a. For 8-bit count by byte operations, this field is programmed with the byte address of the buffer to be transferred. The address may be even or odd.

b. For 16-bit count by word operations, this field is programmed with the *even* byte address of the buffer to be transferred. Thus, bit 0 in this field will be 0.

The Transfer Size field gives the number of bytes or 16 bit words to be transferred. The last SGD in a scatter/gather program must have the EOL field set to 1 to signal that it is the last entry.



Entries in system memory are in LE format.

Figure 2. Scatter Gather Descriptor Table Entry

2.6.2 Scatter/Gather Interrupt Status Register (SGIS)

Table 43. Scatter/Gather Interrupt Status Register (SGIS) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[4]	0x00	Read Only	0x00	N/A

The Scatter/Gather Interrupt Status Register indicates the DMA channel which is the source of a DMA Scatter/Gather interrupt on IRQ13 of the Legacy Interrupt Controller (see Chapter 4, “Legacy Interrupt Controller,” on page 61). The DMA controller drives IRQ13 active after it reaches a terminal count during a Scatter/Gather DMA operation, if bits [7:6] of the Scatter/Gather Command Register are 0b01 (Section 2.6.3, “Scatter/Gather Command Register 0 (SGC0),” on page 27).

Requirements:

2–17. IRQ 13 must be reserved for the ISA DMA device.

Table 44. Scatter/Gather Interrupt Status Register (SGIS)

Bit	Function
7	1=interrupt pending on channel 7. 0=no interrupt pending.
6	1=interrupt pending on channel 6. 0=no interrupt pending.
5	1=interrupt pending on channel 5. 0=no interrupt pending.
4	Reserved (Must be 0)
3	1=interrupt pending on channel 3. 0=no interrupt pending.
2	1=interrupt pending on channel 2. 0=no interrupt pending.
1	1=interrupt pending on channel 1. 0=no interrupt pending.
0	1=interrupt pending on channel 0. 0=no interrupt pending.

2.6.3 Scatter/Gather Command Register 0 (SGC0)

Table 45. Scatter/Gather Command Register 0 (SGC0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[6]	0x00	Write Only	0x00	N/A

The Scatter/Gather Command Register 0 (SGC0) provides commands to start and halt scatter/gather operations on DMA channel 0 and TC to set the method of how the end of the scatter/gather operation will be signalled.

Table 46. Scatter/Gather Command Register 0 (SGC0)

Bit	Function
7	**IRQ13/TC Select:** This bit selects whether Terminal Count (TC) or IRQ13 is asserted when the last buffer in a scatter/gather list has been completed or the last buffer in the DMA has been completed after the DMA has been suspended (a terminal count has occurred). A 1 selects TC as the signaling mechanism. A 0 selects IRQ13 as the signaling mechanism. This bit is only valid if bit 6, IRQ13/TC Programming Enable, is a 1.
6	**IRQ13/TC Programming Enable:** When this bit is a 0, bit 7 does not affect the current state of whether TC or IRQ13 is being used to signal the end of a scatter/gather operation as noted under bit 7. When bit 6 is a 1, bit 7 is enabled to change the state of how the DMA controller signals a terminal count.
5-2	Reserved (Must be set to 0)
1-0	**Scatter/Gather Commands.** See Table 47 on page 27.

Table 47. Scatter/Gather Commands

Bits 1-0	Scatter/Gather Commands
00	**No S/G Operation:** No Scatter/Gather operation is performed. This is typically used when it is desired to change bits 6 and 7.

Table 47. Scatter/Gather Commands ***(Continued)***

Bits 1-0	Scatter/Gather Commands
01	**Start S/G Command:** This command initiates the scatter/gather DMA operation. The DMA controller will begin fetching entries from the Scatter/Gather Descriptor table immediately upon the setting of this bit. When this command is used, all base address, base word, current address and current word registers should be empty (unused).
10	**Stop S/G Command:** This command halts a scatter/gather transfer immediately. Bits 0, 2, 3, and 5 in the Scatter/Gather Status Register (2.6.5)are cleared to 0 and the DMA channel mask bit in the WAM0 or WAM1 register is set to 1. (WAM0 for channels 0-3, WAM1 for channels 5-7)
11	Reserved

2.6.4 Scatter/Gather Command Registers 1-7 (SGC2-SGC7)

Scatter/Gather Command Registers 1-7, SGC1-SGC7, are identical to Scatter/Gather Command Register 0, SGC0, with the exception they control different DMA channels as indicated by the suffixes on the register names and abbreviations. They have unique addresses as shown in Table 2 on page 5

2.6.5 Scatter/Gather Status Register 0 (SGS0)

Table 48. Scatter/Gather Status Register 0 (SGS0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
1 byte	Direct	reg[6]	0x08	Read Only	0x00	0bX0000000 See 2.6.3

This register provides status information on the Scatter/Gather unit for a given DMA channel, in this case, channel 0.

Table 49. Scatter/Gather Status Register 0 (SGS0)

Bit	Function
7	**Next Link Null Indicator:** If the next SGD fetched has an EOL value of 1, it is the last SGD in the scatter/gather list in memory for this scatter/gather operation. When EOL is a 1 in the next SGD, bit 7 is set to a 1.
6	Reserved
5	Reserved.
4	Reserved
3	**Scatter/Gather Base Register Status**: When this bit is a 0, the DMA Base Address Register is empty. When it is a 1, a buffer is pre-loaded (the next link in the SGD table) in the DMA Base Address Register for this channel.
2	**Scatter/Gather Current Register Status**: When this bit is a 0, the DMA Current Address Register is empty. When it is a 1, a buffer is loaded (the current link in the SGD table) in the DMA Current Address Register for this channel.
1	Reserved

Table 49. Scatter/Gather Status Register 0 (SGS0) ***(Continued)***

Bit	Function
0	**Scatter/Gather Active:** This bit indicates the status of a current scatter/gather DMA operation. When this bit is a 1, a scatter/gather start command has been issued. When this bit is set to 0, no operation is in progress, a terminal count on the last buffer in the SGD list has occurred or a scatter/gather stop command has been issued.

2.6.6 Scatter/Gather Status Register 1-7 (SGS1-SGS7)

Scatter/Gather Status Registers 1-7, SGS1-SGS7, are identical to Scatter/Gather Status Register 0, SGS0, with the exception they present status for different DMA channels as indicated by the suffixes on the register names and abbreviations. They have unique addresses as shown in Table 2 on page 5.

2.6.7 Scatter/Gather Descriptor Table Pointer Register 0 (SGPTR0)

Table 50. Scatter/Gather Descriptor Table Pointer Register 0 (SGPTR0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Write Characteristics	Value in Register When OF Passes Control to OS	Software Initiated Reset Value
4 byte	Direct	reg[6]	0x10-0x13	Read/Write	Undefined	N/A

This register contains a 32-bit address for scatter/gather channel 0 that points to the first Scatter/Gather Descriptor, SGD, in a list in system memory. This register is programmable using a single 32 bit write. The SGD list and this register must be programmed (in addition to other things) prior to the issue of the scatter/gather start command.

Bits [31:0] of this register correspond to bits [31:0] on the PCI bus.

Requirements:

2–18. The Scatter/Gather Descriptor Table Pointer Registers must be programmed with a single 32-bit write.

2.6.8 Scatter/Gather Descriptor Table Pointer Registers 1-7 (SGPTR1-SGPTR7)

Scatter/Gather Descriptor Table Pointer Registers 1-7, SGPTR1-SGPTR7, are identical to Scatter/Gather Descriptor Pointer Register 0, SGPTR0, with the exception they service different DMA channels as indicated by the suffixes on the register names and abbreviations. They have unique addresses as shown in Table 2 on page 5.

2.7 Support for ISA Bus masters

ISA bus masters may operate using any of the seven available DMA channels, regardless of transfer width. (ISA bus masters provide their own address and count information.)

Requirements:

2–19. The ISA DMA Controller must not preclude an ISA bus master from using any of the seven available DMA channels, regardless of addressing mode.

2–20. The ISA DMA hardware must provide at least 24-bit addressing to system memory for ISA bus masters.

When programming for an ISA bus master operation, only two registers are used in the ISA DMA controller,

1. Channel Mode register (DCM1, DCM2). This puts the desired channel into cascade mode.
2. Write Single Bit Mask register (WSM1, WSM2). This is used to unmask the desired channel.

Note: some hardware designs allow the High Page Register to form the high-order address bits of a 32-bit ISA bus master PCI address.

During an ISA bus master operation, many of the DMA and all of the S/G registers described in previous sections are not used. These include:

1. Current Address Registers
2. Base Address Registers
3. Current Count Registers
4. Base Count Registers
5. Current Low Page Register
6. Base Low Page Register
7. Extended Mode Register
8. Clear Byte Pointer Register

Floppy Disk Controller

3

3.1 General Requirements

Table 51 on page 31 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements of the floppy disk controller on CHRP platforms.

Table 51. Summary of Minimum Platform Requirements

Subsystem	Specification	Portable	Personal	Server	Description
Floppy		O	O	O	Not required, but a means to attach for software installation must be provided. This may be through a provided connector or over a network. Media sense: Implementations must allow polling of the drive up to 100x per second to determine the presence of media in the drive. A method for manual ejection of floppies is required.
	3.5” 1.44 MB MFM	R	R	R	
	Media sense	R	R	R	
	Auto eject	R	R	M	
	Manual eject	R	R	R	

Requirements:

3–1. A Floppy Disk Controller that is compliant with the interfaces as defined in this section must be represented in the Open Firmware device tree for the auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=chrp,fdc. For the non-auto-eject version of the device, it is represented by the following properties; name=fdc, device_type=fdc, and compatible=pnpPNP,700.

3–2. The firmware must configure the Floppy Disk Controller in the PS/2™ mode, with enhanced auto-eject or without auto-eject.

3–3. This device must be addressed purely with ISA I/O addresses which can be aliased or non-aliased.

3–4. DMA transfers for the floppy disk controller must be 8-bit DMA slave and must support at least the ISA Compatible Timing mode.

3–5. Interrupts generated by the floppy disk controller must be high true edge sensitive.

Software Implementation Note: The PS/2 mode controllers only support two drives.

Software Implementation Note: The operating system is expected to extract the DMA channel number from the node’s dma property. The above requirement means that the operating system need not extract the DMA transfer size, DMA count width, or master vs slave capability from the node’s dma property. Firmware must set these fields to the values that match the above requirement. Operating systems are recommended to

extract the DMA timing from the node's DMA property, though they are free to default to the ISA Compatibility Timing Mode. Refer to Table 23 on page 18 for information about other timing modes.

Software Implementation Note: The operating system is expected to extract the interrupt number from the node's interrupts property. The above requirement means that the operating system need not extract the interrupt type from the node's interrupts property. Firmware must set this field to the values that match the above requirement.

Hardware Implementation Note: The Floppy interrupts are not tri-stated but left enabled, unlike CHRP requirement 2-1.

3.2 Floppy Disk/Tape Media Supported

Requirements:

3–6. The floppy disk controller must support 3.5 inch media with an unformatted capacity of 2 megabytes and a formatted capacity of 1.44 megabytes.

3–7. The floppy disk controller must support the 3.5 inch media with an unformatted capacity of 4 megabytes and a formatted capacity of 2.88 megabytes for purposes of floppy disk port tape backup units.

It is recommended that the floppy disk controller support 3.5 inch media with an unformatted capacity of 1 megabytes and a formatted capacity of 720 kilobytes to enable the use of legacy media of that density.

3.3 Floppy Disk Controller Open Firmware Properties

Requirements:

3–8. The three sets of registers shown in Table 52 on page 33 must be present in the Open Firmware device tree for the auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=chrp,fdc.

3–9. The first two sets of registers shown in Table 52 on page 33 must be present in the Open Firmware device tree for the non-auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=pnpPNP,700.

3–10. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

3–11. A floppy disk controller must be reported in the Open Firmware device tree as specified in the *CHRP ISA Floppy Device Binding* [16].

Table 52 on page 33 gives the registers used by the floppy disk controller.

Table 52. Floppy Disk Controller Registers

Offset	Register	Open Firmware Reg Property
0	Status Register A (SRA)	(reg[1], size = 6)
1	Status Register B (SRB)	
2	Digital Output Register (DOR)	
3	Tape Drive Register (TDR)	
4	Main Status Register (MSR)/ Data Rate Select Register (DSR)	
5	Data Register (FIFO)	
0	Digital Input Register (DIR)/ Configuration Control Register (CCR)	(reg[2], size = 1)
0	Autoeject Register (AER)	(reg[3], size = 1)

3.4 Diskette Drive Controller Registers

Requirements:

3–12. For the device described herein, all the registers defined in Section 3.4.1, “Status Register A (SRA),” on page 33 through Section 3.4.10, “Autoeject Register (AEJ),” on page 41 must be implemented as described herein.

Two sets of registers are described here, control and status. The control registers each have a unique offset address. Floppy disk operations are split into command, execution and result phases. During the result phase, the status registers described are read through the Data Register FIFO (abbreviated as FIFO).

3.4.1 Status Register A (SRA)

Table 53. Status Register A (SRA) 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[1]	0x00	Read Only	See Table 54 on page 33	See Table 54 on page 33

Status Register A, SRA, monitors the state of the floppy disk interrupt source and some of the disk interface signals.

Table 54. Status Register A (SRA)

Bit	Function	Value when OF passes to SW	Value After Software Reset
7	**Interrupt Pending:** This active high bit reflects the state of the interrupt source from the floppy disk controller.	0	0

Table 54. Status Register A (SRA)

Bit	Function	Value when OF passes to SW	Value After Software Reset
6	**Reserved**	x	x
5	**Step:** Active high bit reflecting the state of the STEP disk interface output.	0	0
4	**-Track 0:** Active low bit reflecting the status of the TRK0 disk interface input.	x	x
3	**Head Select:** Active high bit reflecting the status of the HDSEL disk interface output.	0	0
2	**-Index:** Active low bit reflecting the INDEX disk interface input.	x	x
1	**-Write Protect:** Active low bit reflecting the WP disk interface input.	x	x
0	**Direction:** Active high bit reflecting the DIR disk interface output.	0	0

3.4.2 Status Register B (SRB)

Table 55. Status Register B (SRB) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[1]	0x01	Read Only	See Table 56 on page 34	See Table 56 on page 34

This read-only register shows the status of signals on the diskette drive interface. The write-data and read-data bits change state for each positive transition of the ‘-write data’ or ‘-read data’ signals.

Table 56. Status Register B

Bit	Function	Value when OF passes to SW	Value After Software Reset
7,6	**Reserved**	x	x
5	**Drive Select 0:** Reflects the status of the Drive Select 0 bit in the DOR, Digital Output Register (address2, bit 0). It is cleared after a hardware reset, not a software reset.	x	x
4	**Write Data:** Every inactive edge transition of the WDATA disk interface output causes this bit to change state.	0	0
3	**Read Data:** Every inactive edge transition of the RDATA disk interface output causes this bit to change states.	0	0
2	**Write Gate:** Active high bit reflecting the WGATE interface output.	0	0
1	**Motor Enable 1:** Active high bit reflecting the status of the MTR1 disk interface output.	0	x
0	**Motor Enable 0:** Active high bit reflecting the status of the MTR0 disk interface output.	0	x

3.4.3 Digital Output Register (DOR)

Table 57. Digital Output Register (DOR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[1]	0x02	Read/Write	See Table 58 on page 35	See Table 58 on page 35

The Digital Output Register, DOR, controls the drive select and motor enable disk interface outputs, enables the DMA logic, and contains a software reset bit.

Table 58. Drive Control Register

Bit	Function	Value when OF passes to SW	Value After Software Reset
7	**Motor Enable 3:** This bit controls the MTR3 disk interface output. A 1 in this bit causes the MTR3 pin to go active.	0	x
6	**Motor Enable 2:** This bit controls the MTR2 disk interface output. A 1 in this bit causes the MTR2 pin to go active.	0	x
5	**Motor Enable 1:** This bit controls the MTR1 disk interface output. A 1 in this bit causes the MTR1 pin to go active.	0	x
4	**Motor Enable 0):** This bit controls the MTR0 disk interface output. A 1 in this bit causes the MTR0 pin to go active.	0	x
3	**Reserved**	0	x
2	**-Reset Controller):** Writing a 0 to this bit resets the controller. It remains in the reset condition until a 1 is written to this bit. Software resets do not affect the DSR, CCR and other bits of the DOR. This bit must be 0 for at least 100ns before a 1 can be written to it.	0	x
1-0	**Drive Select:** These two bits are a binary encoding of the four drive selects DR0-DR3. This insures that only one drive select output can be active at a time.	0	x

N/A as the value after soft reset means that the bit value is not affected.

3.4.4 Tape Drive Register (TDR)

Table 59. Tape Drive Register (TDR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[1]	0x03	Read/Write	See Table 60 on page 36	See Table 60 on page 36

The TDR register is the Tape Drive Register and the floppy disk controller media and drive type register. Only what has been called the “enhanced” mode of operation is supported and documented here.

Table 60. Drive Status Register

Bit	Function	Value when OF passes to SW	Value After Software Reset
7	**Extra Density:** When bit 5 is 0, this media id bit is used with bit 6 as described in Table 61 on page 36 to indicate the type of media currently in the floppy drive	x	x
6	**High Density:** When bit 5 is 0, this media id bit is used with bit 7 as described in Table 61 on page 36 to indicate the type of media currently in the floppy drive.	x	x
5	**Drive ID 1 Information:** The state of this bit is determined by the state of bits 1 and 0 in the Drive ID register. If this bit is 0, there is valid media ID sense data in bits 7 and 6 of this register. This bit holds the value of bit 1 of the Drive ID register, when drive 0 is accessed and media sense is configured. It holds the value of bit 3 of the Drive ID register, when drive 1 is accessed and media sense is configured. Otherwise, it is set to 1 to indicate that media information is not available. Valid data should be used only when accessing drives 0 and/or 1. Table 61 on page 36 shows what the decoded interpretations of bits 7-5 mean.	x	x
4	**Drive ID 0 Information:** This bit reflects the value of bit 0 in the Drive ID Register if floppy disk device 0 is accessed. It reflects the value of bit 2 in the Drive ID Register if floppy disk device 1 is accessed	x	x
3-2	**Logical Drive Exchange:** These bits are reserved, and must be set to zero.	0	0
1-0	**Tape Select 1,0:** These bits assign a logical drive number to a tape drive. Drive 0 must remain as a boot drive and cannot be assigned as a tape drive. Table 62 on page 36 shows how these bits assign logical drive numbers to tape drives	0	0

Table 61. Media ID Bit Functions

Bit 7, 6,5	Media Type
X X 1	Invalid Data
0 0 0	5.25 in
0 1 0	2.88 Megabyte (formatted)
1 0 0	1.44 Megabyte (formatted)
1 1 0	720 Kilobyte (formatted)

Software Implementation Note: The Drive ID register in Table 60 on page 36, is not part of the Floppy Disk Controller register set. It is a register where the chipset involved has stored the Drive IDs for drive 0 and drive 1 on the floppy disk interface. It then makes this information available to the Floppy Disk Controller.

Table 62. Tape Drive Assignment Values

Bit 1,0	Drive Selected
00	None
01	1
10	2
11	3

3.4.5 Main Status Register (MSR)

Table 63. Main Status Register (MSR) 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[1]	0x04	Read Only	0x00	0x00

This read-only register facilitates the transfer of data between the system microprocessor and the controller.

Table 64. Diskette Drive Controller Status Register

Bit	Function	Value when OF passes to SW	Value on Software Reset
7	**Request for Master:** When this bit is 1, the Data register is ready to transfer data with the system microprocessor	0	0
6	**Data Input/Output:** This bit indicates the direction of data transfer between the diskette drive controller and the system microprocessor. When this bit is 1, the transfer is to the system microprocessor; when the bit is 0, the transfer is to the controller	0	0
5	**Non-DMA Mode:** When this bit is 1, the controller is in the non-DMA mode	0	0
4	**Command in Progress:** When this bit is 1, command is being processed.	0	0
3	**Drive 3 Busy:** When this bit is 1, diskette drive 3 is in the seek mode.	0	0
2	**Drive 2 Busy:** When this bit is 1, diskette drive 2 is in the seek mode	0	0
1	**Drive 1 Busy:** When this bit is 1, diskette drive 1 is in the seek mode	0	0
0	**Drive 0 Busy:** When this bit is 1, diskette drive 0 is in the seek mode	0	0

3.4.6 Data Rate Select Register (DRS)

Table 65. Data Rate Select Register (DRS) 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[1]	0x04	Write Only	SeeTable 66 on page 38	See Table 66 on page 38

This write-only register is used to select the data rate and precompensation value for each data rate.

Table 66. Precompensation Select Register

Bit	Function	Value when OF passes to SW	Software Reset Value
7	**S/W Reset:** This bit functions the same as the POR Reset bit, except it is self clearing.	0	Not effected
6	**Low Power:** This bit puts the floppy disk controller into low power mode. What low power means can vary from manufacturer to manufacturer. The floppy disk controller comes out of low power mode after a software reset, or access to the Data Register or Main Status Register.	0	Not effected
5	**Reserved:** Must be Set to 0.	0	Not effected
4-2	**Precompensation Select:** These bits select the amount of write precompensation the floppy disk controller uses when writing bits to the floppy disk drive. Table 67 on page 38 shows the pre-compensation values for each bit encoding. Table 69 on page 39 shows the default precompensa-tion values for each data rate.	0	Not effected
1-0	**Data Rate Select:** These bits set the data rate of the floppy disk controller. Table 69 on page 39 shows how data rates for the floppy disk controller are encoded	0	Not effected

Table 67. Precompensation Values

Bits 4 3 2	Precompensation Delay
000	Default
001	41.7 ns
010	83.3 ns
011	125.0 ns
100	166.7 ns
101	208.3 ns
110	250 ns
111	0.0 ns

Table 68. Default Precompensation Values

Transfer Rate	Default Precompensation
500Kbps	125ns
300Kbps	125ns
250Kbps	125ns
1000Kbps	41.7ns

Table 69. Data Rate Select Encodings

Bits 1 0	Data Rate	
	MFM	FM
00	500Kbps	250 Kbps
01	300Kbps	150 Kbps
10	250 Kbps	125 Kbps
11	1 Mbps	Illegal

3.4.7 Data Register (FIFO)

Table 70. Data Register (FIFO) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[1]	0x05	Read/Write	0x00	0x00

The FIFO (read/write) is used to transfer all commands, data and status between the system and the floppy disk controller. During the command phase, the system writes the command bytes into the FIFO after polling the RQM and DIO bits in the MSR. During the result phase, the system reads the result bytes from the FIFO after polling the RQM and DIO bits in the MSR.

Enabling the FIFO, and setting the FIFO threshold, is done via the Configure command. If the FIFO is enabled, only the execution phase byte transfers use the 16 byte FIFO. The FIFO is always disabled during the command and result phases of a controller operation. If the FIFO is enabled, it is not disabled after a software reset if the LOCK bit is set in the lock command. After a hardware reset, the FIFO is disabled to maintain compatibility with PC-AT systems.

The 16-byte FIFO can be used for DMA, Interrupt, or software polling type transfers during the execution of a read, write, format, or scan command. In addition, the FIFO can be put into a burst or non-burst mode with the mode command. In the burst mode, DRQ or IRQ remains active until all of the bytes have been transferred to or from the FIFO. In the non-burst mode, DRQ or IRQ is deasserted for 350 ns to allow higher priority transfer requests to be serviced. The mode command can also disable the FIFO for either reads or writes separately. The FIFO allows the system a larger latency without causing a disk overrun/underrun error. Typical uses of the FIFO are at the 1 Mbit/s data rate, or with multi-tasking operating systems. The default state of the FIFO is disabled, with a threshold of zero. The default state is entered after a hardware reset.

During the execution phase of a command involving data transfer to/from the FIFO, the software must respond to a data transfer service request based on the following formula:

$$(THRESH + 1) \times 8 \times TDRP - (16 \times TICP)$$

This formula is good for all data rates with the FIFO enabled or disabled. THRESH is a four bit value programmed in the configure command, which sets the FIFO threshold. If the FIFO is disabled, THRESH is zero in the above formula. The last term of the formula, (16xTICP) is an inherent delay due to the microcode overhead required by the FDC. This delay is also data rate dependent.

The programmable FIFO threshold (THRESH) is useful in adjusting the floppy controller to the speed of the system. In other words, a slow system with a sluggish DMA transfer service request (DRQ for DMA mode or IRQ for interrupt mode). Conversely, a fast system with quick response to a data transfer service request uses a low value of THRESH.

Table 71 on page 40 is a table of TDRP and TICP values.

TDRP - Data Rate Period

TICP - Internal Clock Period

TCP - Clock Period

Table 71. Floppy FIFO Table

MFM Data Rate	TDRP	TICP	Example TICP Values
1 Mbps	1000	3 X TCP	125 ns
500 Kbps	2000	3 X TCP	125 ns
300 Kbps	3333	5 X TCP	208 ns
250 Kbps	4000	6 X TCP	250 ns

3.4.8 Digital Input Register (DIR)

Table 72. Digital Input Register (DIR) 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[2]	0x00	Read Only	See Table 73 on page 40	See Table 73 on page 40

Table 73. Diskette Drive Controller Status Register

Bit	Function	Value when OF passes to SW	Value on Software Reset
7	**Disk In Place:** When this bit is a 1 it reflects that the disk is in place.	x	x
6-3	**Reserved.** Must be set to 0b0000.	x	x
2,1	**Data Rate Select:** These bits indicate the status of the DRATE1-0 bits programmed throught the DSR or CCR	x	x
0	**(-High Density):** This bit is a 0 when the 1 Mb/s or 500 Kbps data rate is chosen, and a 1 when the 300 Kbps or 250 Kbps data rate is chosen.	x	x

3.4.9 Configuration Control Register (CCR)

Table 74. Configuration Control Register (CCR) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Value when OF passes to SW	Software Reset Value
1 byte	Direct	reg[2]	0x00	Write Only	See Table 75 on page 41	See Table 75 on page 41

This write-only register sets the transfer rate.

Table 75. Data Rate Control Register

Bit	Function	Value when OF passes to SW	Value on Software Reset
7-2	**Reserved:** Must be 0	x	x
1	**Data Rate Select:** Bits 1-0 select the data rate, as shown in the Table 76 on page 41	1	1
0	**Data Rate Select:** Bits 1-0 select the data rate, as shown in the Table 76 on page 41	0	0

Table 76. Data Rate Selection

Bits 1 0	Data Rate
00	500,000 bits per second
01	300,000 bits per second
10	250,000 bits per second
11	1,000,000 bits per second*

3.4.10 Autoeject Register (AEJ)

Table 77. Autoeject Register (AEJ) 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[3]	0x00	Read/Write	0x00	0x00

This table is provided for systems that optionally support floppy disk auto-eject.

The motor must be stopped before doing an eject. To cause an auto-eject, software must write to the least significant bit of this register with a 1, followed by a pause provided by software of at least 10 microseconds, followed by a write

of a 0, which will result in the auto-eject. All other bits in this register are reserved, therefore software must use a read-modify-write.

3.5 Floppy Drive Controller Programming Considerations

Each command is initiated by a multibyte transfer from the system microprocessor; the result can be a multibyte transfer back to the system microprocessor. Most transfers consist of three phases:

- *Command Phase:* The system microprocessor writes a series of command bytes to the controller directing it to perform a specific operation.
- *Execution Phase:* The controller performs the specified operation.
- *Result Phase:* After the operation is complete, status information is available to the system microprocessor through a sequence of read commands.

The Seek, Relative Seek, and Recalibrate commands have no result phase. After issuing these commands, the Sense Interrupt Status command must be issued for proper termination and verification of the head position (Present Cylinder Number parameter or PCN).

3.5.1 Controller Commands

Requirements:

3–13. For the device described herein, the commands described in Section 3.5.1.1, “Configure Command,” on page 43 through Section 3.5.1.22, “Write Deleted Data Command,” on page 55 must be implemented as specified.

The following are supported commands for diskette drive controller:

- Configure
- Dumpreg
- Format Track
- Lock
- Perpendicular Mode
- Read Data
- Read Deleted Data
- Read ID
- Read Track
- Recalibrate
- Relative Seek
- Scan Equal
- Scan High or Equal

- Scan Low or Equal
- Seek
- Sense Drive Status
- Sense Interrupt Status
- Specify
- Verify
- Version
- Write Data
- Write Deleted Data.
- Invalid Command Status

3.5.1.1 Configure Command

3.5.1.1.1 Command Phase

Table 78. Configure Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	1	0	0	1	1
Byte 1	0	0	0	0	0	0	0	0
Byte 2	0	EIS	EFF	PD	Thresh-old (THR)			
Byte 3	Precompensation Track Number (PTN)							

3.5.1.1.2 Execution Phase

Internal registers are read.

3.5.1.1.3 Result Phase

This command has no result phase.

3.5.1.2 Dumpreg Command

3.5.1.2.1 Command Phase

Table 79. Dumpreg Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	1	1	0

3.5.1.2.2 Execution Phase

Internal registers are written.

3.5.1.2.3 Result Phase

Table 80. Dumpreg Result Phase

	7	6	5	4	3	2	1	0
Byte 0	Present Track Number - Drive 0							
Byte 1	Present Track Number - Drive 1							
Byte 2	Present Track Number - Drive 2							
Byte 3	Present Track Number - Drive 3							
Byte 4	Stepping Rate Time Head Unload Time							
Byte 5	Head Load Time							DMA
Byte 6	Number of Sectors per Track/End of Track							
Byte 7	X	0	Reserved			X	GAP	WGATE
Byte 8	O	EIS	EFF	PD	Threshold			
Byte 9	Precompensation Track Number							

3.5.1.3 Format Track Command

3.5.1.3.1 Command Phase

Table 81. Format Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	MFM	0	0	1	1	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Number of Data Bytes in Sector							
Byte 3	Sectors per Track							
Byte 4	Gap Length							
Byte 5	Fill Byte							

3.5.1.3.2 Execution Phase

The system transfers four ID bytes (track, head, sector, bytes/sector) per sector to the floppy controller via DMA or Non-DMA modes. The entire track is formatted. The data block in the data field of each sector is filled with the data pattern byte.

3.5.1.3.3 Result Phase

Table 82. Format Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Reserved
Byte 4	Reserved
Byte 5	Reserved

Table 82. Format Result Phase ***(Continued)***

Byte 6	Reserved

3.5.1.4 Lock Command

3.5.1.4.1 Command Phase

Table 83. Lock Command Phase

	7	6	5	4	3	2	1	0
Byte 0	Lock	0	0	1	0	1	0	0

3.5.1.4.2 Execution Phase

An internal Register is written.

3.5.1.4.3 Result Phase

Table 84. Lock Result Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	Lock	0	0	0	0

3.5.1.5 Perpendicular Mode Command

3.5.1.5.1 Command Phase

Table 85. Perpendicular Mode Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	1	0	0	1	0
Byte 1	OW	0	D3	D2	D1	D0	GAP	WGATE

3.5.1.5.2 Execution Phase

Internal registers are written.

3.5.1.5.3 Result Phase

There is no result phase for the Perpendicular Mode command.

3.5.1.6 Read Data Command

3.5.1.6.1 Command Phase

Table 86. Read Data Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT0	MFM	SK	0	0	1	1	0
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Data Length							

3.5.1.6.2 Execution Phase

Data read from disk drive is transferred to system via DMA or non-DMA modes.

3.5.1.6.3 Result Phase

Table 87. Read Data Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.7 Read Deleted Data Command

3.5.1.7.1 Command Phase

Table 88. Read Deleted Data Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	SK	0	1	1	0	0
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							

Table 88. Read Deleted Data Command Phase ***(Continued)***

	7	6	5	4	3	2	1	0
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Data Length							

3.5.1.7.2 Execution Phase

Data read from disk drive is transferred to system via DMA or non-DMA modes.

3.5.1.7.3 Result Phase

Table 89. Read Deleted Data Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.8 Read ID Command

3.5.1.8.1 Command Phase

Table 90. Read ID Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	MFM	0	0	1	0	1	0
Byte 1	0	0	0	0	0	HD	US	US

3.5.1.8.2 Execution Phase

The controller reads the first ID field header bytes it can find and reports these bytes to the system in the result bytes.

3.5.1.8.3 Result Phase

Table 91. Read ID Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address

Table 91. Read ID Result Phase ***(Continued)***

Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.9 Read Track Command

3.5.1.9.1 Command Phase

Table 92. Read Track Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	MFM	0	0	0	0	1	0
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Data Length							

3.5.1.9.2 Execution Phase

Data read from the disk drive is transferred to the system via DMA or non-DMA modes.

3.5.1.9.3 Result Phase

Table 93. Read Track Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.10 Recalibrate Command

3.5.1.10.1 Command Phase

Table 94. Recalibrate Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	0	1	1	1

Table 94. Recalibrate Command Phase ***(Continued)***

	7	6	5	4	3	2	1	0
Byte 1	0	0	0	0	0	0	US	US

3.5.1.10.2 Execution Phase

The disk drive head is stepped out to Track 0.

3.5.1.10.3 Result Phase

This command has no result phase.

3.5.1.11 Relative Seek Command

3.5.1.11.1 Command Phase

Table 95. Relative Seek Command Phase

	7	6	5	4	3	2	1	0
Byte 0	1	DIR	0	0	1	1	1	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Relative Cylinder Number							

3.5.1.11.2 Execution Phase

The disk drive head is stepped in or out a programmable number of tracks.

3.5.1.11.3 Result Phase

This command has no result phase.

3.5.1.12 Scan Equal Command

3.5.1.12.1 Command Phase

Table 96. Scan Equal Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	SK	1	0	0	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Scan Test							

3.5.1.12.2 Execution Phase

Data transferred from the system to the controller is compared to data read from the disk.

3.5.1.12.3 Result Phase

Table 97. Scan Equal Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.13 Scan High or Equal Command

3.5.1.13.1 Command Phase

Table 98. Scan High or Equal Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	SK	1	1	1	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Scan Test							

3.5.1.13.2 Execution Phase

Data transferred from the systems to the controller is compared to data read from the disk.

3.5.1.13.3 Result Phase

Table 99. Scan High or Equal Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number

Table 99. Scan High or Equal Result Phase ***(Continued)***

Byte 6	Number of Data Bytes in Sector

3.5.1.14 Scan Low or Equal Command

3.5.1.14.1 Command Phase

Table 100. Scan Low or Equal Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	SK	1	1	0	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Scan Test							

3.5.1.14.2 Execution Phase

Data transferred from the system to the controller is compared to data read from the disk.

3.5.1.14.3 Result Phase

Table 101. Scan Low or Equal Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.15 Seek Command

3.5.1.15.1 Command Phase

Table 102. Seek Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	1	1	1
Byte 1	0	0	0	0	0	HD	US	US

Table 102. Seek Command Phase *(Continued)*

	7	6	5	4	3	2	1	0
Byte 2	New Track Number after Seek							

3.5.1.15.2 .Execution Phase

The disk drive head is stepped in or out to a programmable track.

3.5.1.15.3 Result Phase

This command has no result phase.

3.5.1.16 Sense Drive Status Command

3.5.1.16.1 Command Phase

Table 103. Sense Drive Status Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	0	1	0	0
Byte 1	0	0	0	0	0	HD	US	US

3.5.1.16.2 Result Phase

Table 104. Sense Drive Status Result Phase

Byte 0	Status Register 3

3.5.1.17 Sense Interrupt Status

3.5.1.17.1 Command Phase

Table 105. Sense Interrupt Status Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	1	0	0	0

3.5.1.17.2 Execution Phase

The status of the interrupt is reported.

3.5.1.17.3 Result Phase

Table 106. Sense Interrupt Status Result Phase

Byte 0	Status Register 0
Byte 1	Present Track Number

3.5.1.18 Specify Command

3.5.1.18.1 Command Phase

Table 107. Specify Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	0	0	0	1	1
Byte 1	Stepping Rate Time				Head Unload Time			
Byte 2	Head Load Time							DMA

3.5.1.18.2 Execution Phase

Internal registers are written.

3.5.1.18.3 Result Phase

This command has no result phase.

3.5.1.19 Verify Command

3.5.1.19.1 Command Phase

Table 108. Verify Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	SK	1	0	1	1	0
Byte 1	EC	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Address							
Byte 5	Sector Size							
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Byte Transfer Control							

3.5.1.19.2 Execution Phase

Data is read from the disk but not transferred to the system.

3.5.1.19.3 Result Phase

Table 109. Verify Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Address
Byte 6	Sector Size

3.5.1.20 Version Command

3.5.1.20.1 Command Phase

Table 110. Version Command Phase

	7	6	5	4	3	2	1	0
Byte 0	0	0	0	1	0	0	0	0

3.5.1.20.2 Result Phase

Table 111. Version Result Phase

	7	6	5	4	3	2	1	0
Byte 0	1	0	0	X	0	0	0	0

X can be a 1 or a 0.

3.5.1.21 Write Data Command

3.5.1.21.1 Command Phase

Table 112. Write Data Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	0	0	0	1	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Address							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							

Table 112. Write Data Command Phase ***(Continued)***

	7	6	5	4	3	2	1	0
Byte 6	End-of-Track							
Byte 7	Gap Length							
Byte 8	Data Length							

3.5.1.21.2 Execution Phase

Data is transferred from the system to the controller via DMA or non-DMA modes and written to the disk.

3.5.1.21.3 Result Phase

Table 113. Write Data Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.22 Write Deleted Data Command

3.5.1.22.1 Command Phase

Table 114. Write Deleted Command Phase

	7	6	5	4	3	2	1	0
Byte 0	MT	MFM	0	0	1	0	0	1
Byte 1	0	0	0	0	0	HD	US	US
Byte 2	Track Number							
Byte 3	Head Address							
Byte 4	Sector Number							
Byte 5	Number of Data Bytes in Sector							
Byte 6	End of Track							
Byte 7	Gap Length							
Byte 8	Data Length							

3.5.1.22.2 Execution Phase

Data is transferred from the system to the controller via DMA or non-DMA modes and written to the disk.

3.5.1.22.3 Result Phase

Table 115. Write Deleted Result Phase

Byte 0	Status Register 0
Byte 1	Status Register 1
Byte 2	Status Register 2
Byte 3	Track Number
Byte 4	Head Address
Byte 5	Sector Number
Byte 6	Number of Data Bytes in Sector

3.5.1.23 Invalid Command Status

3.5.1.23.1 Result Phase. The following status byte is returned to the system microprocessor when an invalid command has been received.

Table 116. Invalid Command Status Register

Byte 0	Status Register 0

Bits 6 and 7 in status register 0 are used to indicate command status. When an invalid command is processed, this information is returned to the system microprocessor in the invalid command status byte.

3.5.2 Command Status Registers Provided During Result Phase

This section provides definitions for status registers 0 through 3. These registers are read through the Data Register (FIFO) during the result phase of a floppy disk operation.

Requirements:

3–14. For the device described herein, the Command Status Registers described in Section 3.5.2.1, “Status Register 0 (ST0),” on page 56 through Section 3.5.2.4, “Status Register 3 (ST3),” on page 58 must be implemented as specified.

3.5.2.1 Status Register 0 (ST0)

Table 117. Status Register 0

Bit	Function	Value when OF passes to SW	Value on Software Reset
7, 6	**Interrupt Code:**These bits indicate the command interrupt status as given in Table 118 on page 57.	0	0
5	**Seek End:** This bit is set to 1 when the diskette drive completes the Seek or Recalibrate command, or a read or write operation with an implied Seek command.	0	0

Table 117. Status Register 0 ***(Continued)***

Bit	Function	Value when OF passes to SW	Value on Software Reset
4	**Equipment Check:** This bit is set to 1 if the '-track 0' signal fails to occur after the Recalibrate command is issued or Relative Seek command to step outward beyond track 0	0	0
3	**Reserved:** This bit is always set to 0	0	0
2	**Head Select:** This bit indicates the state of the '-head select' signal after the command was performed. When set to 1, head 1 was selected: when set to 0. head O was selected.	0	0
1, 0	**Drive Select:** These bits indicate the drive that was selected upon command completion	0	0

Table 118. Encodings for the Interrupt Code field (bits 7,6 of ST0)

Bits 7 6	Function
0 0	Normal Termination of Command
0 1	Abnormal Termination of Command
10	Invalid Command Issued
11	Internal drive ready status changed state during the drive polling mode. Only occurs after a hardware or software reset.

Table 119. Drive Select Bits

Bits 1 0	Function
0 0	Drive 0
0 1	Drive 1
1 0	Drive 2
1 1	Drive 3

3.5.2.2 Status Register 1 (ST1)

Table 120. Status Register 1

Bit	Function
7	**End-of-Track**: This bit is set to 1 when the controller tries to gain access to a sector beyond the final sector of a track
6	**Reserved:** This bit is always set to 0.
5	**Cyclic Redundancy Check (CRC) Error:** This bit is set to 1 when a CRC error is detected in the ID or data field.
4	**Overrun/Underrun Error:** This bit is set to 1 if the system does not service the diskette drive controller within an adequate period of time during data transfers.
3	**Reserved:** This bit is always set to 0.

Table 120. Status Register 1 ***(Continued)***

Bit	Function
2	**No Data:** This bit is set to 1 when • The controller cannot find the sector specified in the ID register during the execution of a Read Data, Read Deleted Data, or Read ID or Read Track command. • The controller cannot read the ID field without an error during the execution of a Read ID command • The starting sector cannot be found during the execution of a Read Track command
1	**Not Writable:** This bit is set to 1 when the '-write-protect' signal is active during a Write Data, Write Deleted Data, or Format Track command.
0	**Missing Address Mark:** This bit is set to 1 if the controller cannot detect an address mark. When this occurs, bit 0 of Status Register 2 indicates whether the missing address mark is an ID-address mark or a data-address mark.

3.5.2.3 Status Register 2 (ST2)

Table 121. Status Register 2

Bit	Function
7	**Reserved:** This bit is always set to 0.
6	**Control Mark:** This bit is set to 1 when the controller encounters a sector that has a deleted data-address mark during a Read Data or a Read Deleted Data encounters a data address mark.
5	**CRC Error in Data Field:** This bit is set to 1 if the controller detects an error in the data.
4	**Wrong Track:** This bit is set to 1 when the track number on the media is different from the track number issued by the command. When this occurs, bit 2 of Status Register 1 is also set to 1.
3	**Scan Equal Hit:** This bit is set to 1 during the Scan command when the conditions for Equal are satisfied.
2	**Scan Not Satisfied:** This bit is set to 1 during the Scan Command when the scan conditions are not satisfied.
1	**Bad Track:** This bit is set to 1 when the track number on the media is hex FF and the track number value stored in the IDregister is not hex FF. When this occurs, bit 2 of Status Register 1 is also set to 1.
0	**Missing Address Mark in Data Field:** This bit is set to 1 when the controller cannot find a data-address mark. This bit is set to 0 when an ID-address mark cannot be found. Bit 0 in Status Register 0 is also set if either address mark cannot be found.

3.5.2.4 Status Register 3 (ST3)

Table 122. Status Register 3

Bit	Function
7	**Reserved:** This bit is always set to 0
6	**Write Protect:** This bit indicates the status of the '-write-protect' signal from the diskette drive. When this bit is set to 1, the'-write-protect' signal is active.
5	**Reserved:** This bit is always set to 1.
4	**Track 0:** This bit indicates the status of the '-track O' signal from the diskette drive. When this bit is set to 1, the '-track 0 signal is active.
3	**Reserved:** This bit is always set to 1.

Table 122. Status Register 3 ***(Continued)***

Bit	Function
2	**Head Address:** This bit indicates the status of the '-head 1 select' signal from the diskette drive. When this bit is set to 1, the '-head 1 select' signal is active.
1, 0	**Drive Select:** These bits indicate the current selected drive.

3.6 Media Sense

Requirements:

3–15. The disk drive must be configured in a manner such that the Disk in Place signal, on the drive, is functionally a state indicator of the presence of media. The signal must assert an active level when a disk is present in the drive and an inactive level when there is no disk in the drive.

3.7 Floppy Drive Signal Connector Pin Assignment

Table 123. Signal Connector Pin Assignment

PIN	Signal Description	PIN	Signal Description
1	AUTO EJECT *	2	N. C.
3	Key	4	HD OUT
5	Ground	6	DISK IN PLACE *
7	Ground	8	INDEX
9	Ground	10	DRIVE SELECT 0
11	Ground	12	DRIVE SELECT 1
13	Ground	14	N. C.
15	Ground	16	MOTOR ON
17	Ground	18	DIRECTION
19	Ground	20	STEP
21	Ground	22	WRITE DATA
23	Ground	24	WRITE GATE
25	Ground	26	TRACK 00
27	Ground	28	WRITE PROTECT
29	Ground	30	READ DATA
31	Ground	32	HEAD SELECT
33	Ground	34	DISK CHANGE

Hardware Implementation Note: Signal pins AUTOEJECT, signal pin 1, DISK IN PLACE, signal pin 6, and HD OUT, signal pin 4, must be gated by Drive Select, if the drive is used in platforms that support multiple drives.

3.8 References

1. *National Semiconductor PC87308VUL SuperI/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 IEEE1284 Parallel Port* [24].

2. *National Semiconductor PC87332VLJ (3.3V/5V) and PC87332VLJ-5 (5V) (SuperI/O III Premium Green) with Floppy Disk Controller, Dual UARTs, IEEE 1284 Parallel Port, and IDE Interface* [26].

Legacy Interrupt Controller

4

4.1 Overview and General Requirements

Table 124 on page 61 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements of the Legacy Interrupt Controller on CHRP platforms.

Table 124. Minimum Platform Requirements for the Legacy Interrupt Controller

Subsystem	Specification	Portable	Personal	Server	Description
Interrupt Controller	8259 tree	R	R	R	8259 required for ISA compatibility.

The ISA Legacy Interrupt Controller is actually a set of two programmable interrupt controllers (PICs) referred to as PIC1 (master) and PIC2 (slave) in this book. Their interrupt lines may be set as edge or level sensitive. There are a number of possible modes for determining the order in which the priority of interrupts is resolved. The two controllers are cascaded in a master/slave relationship as shown in Figure 3.

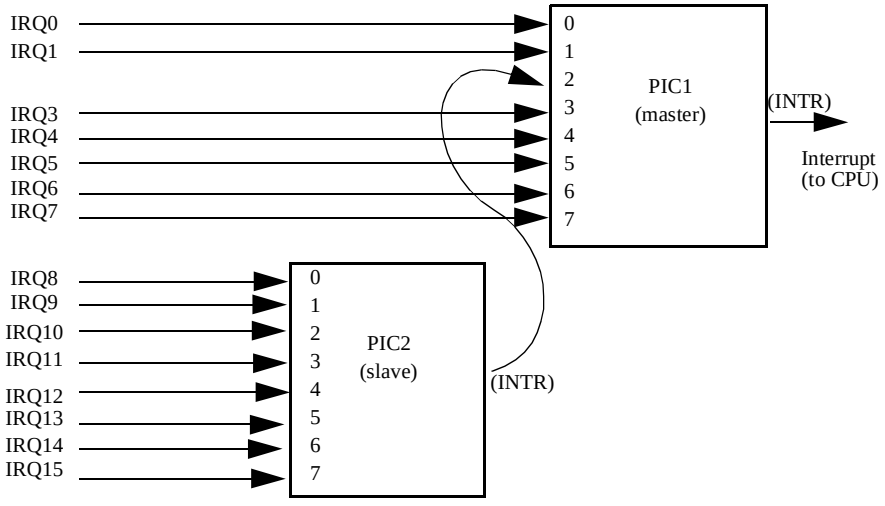


Figure 3. PIC1 (master) and PIC2 (slave) Interconnection

Requirements:

4–1. The Legacy Interrupt Controller must be represented by the Open Firmware properties “name=interrupt-controller, device_type=interrupt-controller, compatible=chrp,iic”.

4–2. The Legacy Interrupt Controller registers specified herein must be implemented precisely as defined, including the following characteristics and bit definitions:

- **a.** Table 129, “Initialization Command Word 1 Register (ICW1) Characteristics,” on page 68
- **b.** Table 130, “Initialization Command Word 1 Register (ICW1),” on page 68
- **c.** Table 131, “Initialization Command Word 2 Register (ICW2) Characteristics,” on page 68
- **d.** Table 132, “Initialization Command Word 2 Register (ICW2),” on page 68
- **e.** Table 133, “Initialization Command Word 3 Register (ICW3) Characteristics,” on page 69.
- **f.** Table 134, “Initialization Command Word 3 Register (ICW3),” on page 69
- **g.** Table 135, “Initialization Command Word 4 Register (ICW4) Characteristics,” on page 69
- **h.** Table 136, “Initialization Command Word 4 Register (ICW4),” on page 70
- **i.** Table 137, “Operation Command Word 1 Register (OCW1) Characteristics,” on page 70
- **j.** Table 138, “Operation Command Word 1 Register (OCW1),” on page 70
- **k.** Table 140, “Operation Command Word 2 Register (OCW2) Characteristics,” on page 71
- **l.** Table 141, “Operation Command Word 2 Register (OCW2),” on page 71
- **m.** Table 144, “Operation Command Word 3Register (OCW3) Characteristics,” on page 72
- **n.** Table 145, “Operation Command Word 3 Register (OCW3),” on page 73
- **o.** Table 149, “Interrupt Request Register (IRR) Characteristics,” on page 74
- **p.** Table 143, “Interrupt Level Select Encodings,” on page 72
- **q.** Table 151, “In-Service Register (ISR) Characteristics,” on page 75
- **r.** Table 143, “Interrupt Level Select Encodings,” on page 72
- **s.** Table 153, “Edge/Level Interrupt Control Register 1 (ELI1) Characteristics,” on page 76
- **t.** Table 154, “Edge/Level Interrupt Control 1 Register (ELI1),” on page 76
- **u.** Table 155, “Edge/Level Interrupt Control 2 Register (ELI2) Characteristics,” on page 76
- **v.** Table 156, “Edge/Level Interrupt Control 2 Register (ELI2),” on page 76

4–3. This device must be addressed purely with ISA I/O addresses.

Hardware and Software Implementation Note: The first 256 bytes of I/O space is not aliased (that is, all of the bits of the address are decoded).

4.2 Open Firmware Requirements

Requirements:

4–4. The registers shown in Table 125 on page 63 must all be present in the device defined in this chapter. The Open Firmware “reg” property information given in Table 125 on page 63 must be provided in the Open Firmware device tree for this device.

4–5. This device must be represented as an ISA device and a child of the ISA bridge node in the Open Firmware device tree.

4–6. This device must be represented in the OF device tree as specified in the *CHRP ISA Interrupt Controller Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware.*

4–7. All the reg property fields specified in table Table 125 on page 63 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.

4–8. The following events must occur during the OF initialization sequence:

a. The level sense circuit must be set to the Edge-sensitive mode (following the initialization procedure, interrupts are generated by a low-to-high transition high level on the interrupt request input.)

b. IRQ7 must be assigned priority 7.

c. PIC2 (slave) mode address must be set to 7.

d. The Special Mask mode must be cleared and the controller must be set to read the Interrupt Request register.

e. All Legacy Interrupt Controller registers must be initialized to the values indicated in the “Value in Register When OF Passes Control to OS” column of the characteristics table corresponding to the specific register.

4–9. PIC1 (master) interrupt controller must be initialized before PIC2 (slave) interrupt controller. Failure to do so will cause unexpected results.

A synopsis of the registers for the ISA Interrupt Controller is given in Table 125. The positions in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 125 on page 63. The format is described in the referenced Open Firmware bindings.

Table 125. Legacy Interrupt Controller Registers

Offset	Register	Open Firmware Reg Property
0	ICW1,OCW2,OCW3, IRR, ISR (PIC1 (master))	reg[1], size=2
1	ICW2,ICW3,ICW4,OCW1 (PIC1 (master))	
0	ICW1,OCW2,OCW3, IRR, ISR (PIC2 (slave))	reg[2], size=2
1	ICW2,ICW3,ICW4,OCW1 (PIC2 (slave))	

Table 125. Legacy Interrupt Controller Registers ***(Continued)***

Offset	Register	Open Firmware Reg Property
0	Edge Level Control register ELI1 (PIC1 (master))	reg[3], size=2
1	Edge Level Control register ELI2 (PIC2 (slave))	

4.3 Modes of Operation

The interrupt controller can be programmed to operate in a variety of modes through the Initialization Command Word registers (ICW) and the Operation Command Word registers (OCW).

A typical interrupt request occurs in the following manner:

1. One or more 'interrupt request' lines are set active, causing the corresponding bits in the Interrupt Request register (IRR) to be set to 1.
2. The interrupt controller evaluates the requests and sends an interrupt to the system microprocessor, if appropriate.
3. The system microprocessor receives the interrupt and enters the interrupt handler. The interrupt handler responds by reading 1 byte from the Interrupt Acknowledge address (at an address as specified by the *8259-interrupt-acknowledge* property of the OF device tree). This generates an 'interrupt acknowledge' cycle to the interrupt controller. (Refer to the *8259-interrupt-acknowledge* property in the CHRP System Binding for Open Firmware for information on how to perform the interrupt acknowledge.)
4. The controller prioritizes the unmasked bits in the Interrupt Request register and strobes the bit with the highest priority into the corresponding bit of the In-Service register (ISR). No data is sent to the system microprocessor.

   Hardware and Software Implementation Note: If an interrupt request is not present (for example, the duration of the request was too short), the interrupt controller issues an interrupt 7.

5. The interrupt controller responds by releasing the interrupt vector (as defined in Table 126 on page 65) on the data bus, where it is read by the system microprocessor.
6. The highest priority in-service bit remains set to 1 until the proper End of Interrupt (EOI) command is issued by the interrupt subroutine. If the source of the interrupt request is PIC2 (slave) interrupt controller, the EOI command must be issued twice, once for PIC1 (master) and once for PIC2 (slave). When the in-service bit is set to 1, all other interrupts with the same or lower priority are inhibited; interrupts with a higher priority cause an interrupt, but the interrupt is acknowledged only if the system microprocessor interrupt input has been re-enabled by software.

The EOI command has two forms, specific and non-specific. The controller responds to a Non-specific EOI command by resetting the highest in-service bit of those set. In a mode that uses a fully-nested interrupt structure, the highest in-service bit set is the level that was just acknowledged and serviced. In a mode that can use other than the fully-nested interrupt structure, a Specific EOI command is required to define which in-service bit to reset.

Software Implementation Note: An in-service bit masked by an Interrupt Mask register (see OCW1 for information about the Interrupt Mask register) bit cannot be reset by a Non-specific EOI command when in the Special Mask mode. See-Section 4.4.2.2, “Operation Command Word 2 Register (OCW2),” on page 71 for more information.

Requirements:

4–10. The Interrupt Controller must respond to a Non-specific EOI command by resetting the highest in-service bit of those set.

4–11. The Interrupt Controller must respond to a Specific EOI command by resetting the in-service bit that corresponds to the one specified by the command, except that an in-service bit masked by an Interrupt Mask register bit cannot be reset by a Non-specific EOI command when in the Special Mask mode.

4–12. The Interrupt Vector which is returned by the Interrupt Controller on an Interrupt Acknowledge cycle must be as defined in Table 126 on page 65.

Table 126. Interrupt Vector

Bit	Description
7-3	**Interrupt Vector Base Address:** This field defines the base address of the interrupt vector table which gives the base addresses for the interrupt routines associated with each IRQ on the controller.
2-0	**Interrupt:** Interrupt level that is currently requiring service and has been deemed to have the highest priority by this interrupt controller. IRQ0 is represented by 0b000, IRQ1 by 0b001, and so on.

4.3.1 Fully-Nested Mode

In the Fully-nested mode, interrupts are prioritized from 0 (highest priority) to 7 (lowest priority). This mode is automatically entered after initialization unless another mode has been programmed.

Software Implementation Note: The priorities can be changed by rotating the priorities through OCW2.

4.3.2 Special Fully-Nested Mode

The Special Fully-nested mode is used when the priority in PIC2 (slave) interrupt controller must be preserved. This mode is similar to the normal Nested mode with the following exceptions:

- When PIC2’s (slave’s) interrupt request is in service, PIC2 (slave) can still generate additional interrupt requests of a higher priority that are recognized by PIC1 (master), and initiate interrupts to the system microprocessor.
- Upon completion of the interrupt service routine, software must send a Non-specific EOI (see 4.4.2.2, “Operation Command Word 2 Register (OCW2),” on page 71) command to PIC2 (slave) and read PIC2 (slave)'s In-Service register to ensure that the interrupt just serviced was the only one generated by PIC2 (slave). If the register is not empty, additional interrupts are pending, and an EOI command must not be sent to PIC1 (master). If the register is empty, a Non-specific EOI command can be sent to PIC1 (master).

The Special Fully-nested mode is selected through ICW4.

4.3.3 Automatic Rotation Mode

The Automatic Rotation mode accommodates multiple devices having the same interrupt priority. After a device is serviced, it is assigned the lowest priority and must wait until all other devices requesting an interrupt are serviced once before the first device is serviced again.

The following example shows the status and priorities of the In-Service register bits before and after bit 4 of the Interrupt Request register is serviced by a Rotation on Non-specific EOI command.

Table 127. Automatic Rotation Mode

In-Service Register Bits	Status Before Service	Priority Before Rotate	Status After Service	Priority After Rotate
7	0	7(Lowest)	0	2
6	1(Pending)	6	1(Pending)	1
5	0	5	0	0
4	1(Pending)	4	0(Serviced)	7
3	0	3	0	6
2	0	2	0	5
1	0	1	0	4
0	0	0(Highest)	0	3

The Automatic Rotation mode is selected by issuing a Rotation on Non-specific EOI command through OCW2.

4.3.4 Specific Rotation Mode

The Specific Rotation mode allows the application programs to change the priority levels by assigning the lowest priority to a specific interrupt level. Once the lowest level priority is selected, all other priority levels change. The following example compares the normal Nested mode to the Specific Rotation mode with bit 5 of the Interrupt Request register set to the lowest priority.

Table 128. Specific Rotation Mode When IRQ5 Has the Lowest Priority

Interrupt Request Register Bits	Nested Mode Priority Level	Specific Rotation Mode Priority Level
7	7 (Lowest)	1
6	6	0 (Highest)
5	5	7 (Lowest)
4	4	6
3	3	5

Table 128. Specific Rotation Mode When IRQ5 Has the Lowest Priority ***(Continued)***

Interrupt Request Register Bits	Nested Mode Priority Level	Specific Rotation Mode Priority Level
2	2	4
1	1	3
0	0 (Highest)	2

The Specific Rotation mode is selected by issuing a Rotate on Specific EOI command or a Set Priority command through OCW2.

4.3.5 Special Mask Mode

The Special Mask mode allows application programs to selectively enable and disable any interrupt or combination of interrupts at any time during its execution. The Special Mask mode is selected through OCW3 register. Once the controller is in the Special Mask mode, setting a bit in OCW1 sets a corresponding bit in the Interrupt Mask register (see OCW1 for information about the Interrupt Mask register). Each bit set in the Interrupt Mask register masks the corresponding interrupt channel. Interrupt channels above and below a masked channel are not affected.

4.3.6 Poll Mode

Devices are serviced by software issuing a Poll command through OCW3. The first 'read' pulse following a Poll command is interpreted by the controller as an 'interrupt acknowledge' pulse; the controller sets the appropriate in-service bit and reads the priority level. The byte placed on the data bus during a 'read' pulse is shown in Table 148 on page 73.

4.4 Programming the Interrupt Controller

Before the system can be used, the interrupt controller must be programmed with four sequential writes to the four ICW registers. When PIC1 (master) is written at address reg[1]+0x00 or PIC2 (slave) at address reg[2]+0x00) with data bit 4 set to 1, the command is recognized as a write to ICW1. ICW1 is the first of four writes to the ICW registers that are required to program the interrupt controller. Once the interrupt controller is initialized by the ICW registers, the controller can be further programmed by the OCW registers to operate in other modes.

4.4.1 Initialization Command Word Registers

Requirements:

4–13. For the device described herein, all the registers defined in sections 4.4.1.1 through 4.4.1.4 must be implemented precisely as described herein.

4–14. During the initialization sequence ICW1 must be written before writing ICW2, ICW2 must be written before writing ICW3, and ICW3 must be written before writing ICW4.

4.4.1.1 Initialization Command Word 1 Register (ICW1)

Table 129. Initialization Command Word 1 Register (ICW1) 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	Sequenced	reg[1],reg[2]	0x00	Write Only	0x01	N/A

This is the first register written in a four register sequence. This byte must be set for each PIC. Each PIC has a different base address. The other three ICW registers are written consecutively at offset 0x1.

Table 130. Initialization Command Word 1 Register (ICW1)

Bit	Function
7 - 5	**Reserved:** Must be set to 0.
4	ICW/OCW Select: Must be set to 1 to select ICW1, ICW2, ICW3 and ICW4. 0 for OCW1, OCW2, OCW3.
3	**Edge/Level Bank Select (LTIM):** Must be set to 0
2-1	**Reserved:** Must be set to 0.
0	**ICW4 Write Required:** Must be set to 1.

4.4.1.2 Initialization Command Word 2 Register (ICW2)

Table 131. Initialization Command Word 2 Register (ICW2) 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	Sequenced	reg[1],reg[2]	0x01	Write Only	0x00 for PIC1 0x08 for PIC2	N/A

This byte defines the address returned as the Interrupt Vector. Bits 7 through 3 define the five high-order bits of the interrupt vector address as shown in Table 132. See also Table 126 on page 65 for a complete definition of the Interrupt Vector.

Table 132. Initialization Command Word 2 Register (ICW2)

Bit	Function
7-3	**Interrupt Vector Base Address:** This field defines the base address of the interrupt vector table which gives the base addresses for the interrupt routines associated with each IRQ on the controller. Must be set to 0b00000 for PIC1 (master) and to 0b00001 for PIC2 (slave)

Table 132. Initialization Command Word 2 Register (ICW2) ***(Continued)***

Bit	Function
2-0	**Reserved:** Must be set to 0b000.

4.4.1.3 Initialization Command Word 3 Register (ICW3)

Table 133. Initialization Command Word 3 Register (ICW3) 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	Sequenced	reg[1], reg[2]	0x01	Write Only	0x04 for PIC1 0x02 for PCI2	N/A

In PIC1 (master), this register has a value of 0x04 to tell PIC1 (master) that hardware interrupt IRQ2 indicates an interrupt is pending in the PIC2 (slave) controller. If IRQ2 wins the interrupt selection in PIC1 (master) then PIC1 (master) will assert the INT line to the processor. When the processor issues an acknowledge, PIC1 (master) will tell PIC2 (slave) to respond to the acknowledge cycle.

In PIC2 (slave), this register has a value of 0x02 to tell PIC2 (slave) that it is using hardware interrupt 2 to communicate with PIC1 (master). PIC2 (slave) compares this identification code which is broadcast from the PIC1 (master) controller; if they are equal, PIC2 (slave) releases the interrupt vector address on the data bus.

Table 134. Initialization Command Word 3 Register (ICW3)

Bit	Function
7-0	**Reserved:** Must be set to 0x04 for PIC1 and 0x02 for PIC2

4.4.1.4 Initialization Command Word 4 Register (ICW4)

Table 135. Initialization Command Word 4 Register (ICW4) 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	Sequenced	reg[1],reg[2]	0x01	Write Only	0x01	N/A

This byte is written to either PIC1 (master) or PIC2 (slave).

Table 136. Initialization Command Word 4 Register (ICW4)

Bit	Function
7-5	**Reserved:** Must be set to 0.
4	**Special Fully-Nested Mode (SFNM):** If 1, then the PIC is in Special Fully-nested mode.
3-2	**Reserved:** Must be set to 0b00.
1	**Automatic EOI:** Must be set to 0.
0	**80286/80386 Microprocessor Mode:** Must be set to 1.

4.4.2 Operation Command Word Registers

Requirements:

4–15. For the device described herein, all the registers defined in sections 4.4.2.1 through 4.4.4 must be implemented precisely as described herein.

4–16. Writing of OCW1 must be performed before the OCW2 and OCW3 registers are accessed, and following the ICW1-4 initialization sequence.

4.4.2.1 Operation Command Word1 Register (OCW1)

Table 137. Operation Command Word 1 Register (OCW1) 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	Sequenced	reg[1], reg[2]	0x01	Read/Write	0xFF	N/A

This byte controls the individual bits in the Interrupt Mask register. Reading this register will return the value in the Interrupt Mask register.

Table 138. Operation Command Word 1 Register (OCW1)

Bit	Function
7-0	**Interrupt Mask:** When a 1 is written to any bit in this register, the corresponding IRQ line is masked. See Table 139 on page 71 for the bit-to-interrupt level correspondence.

.

Table 139. OCW1 Bit Definitions

Bit	PIC1 (master)	PIC2 (slave)
7	Interrupt Level 7	Interrupt Level 15
6	Interrupt Level 6	Interrupt Level 14
5	Interrupt Level 5	Interrupt Level 13
4	Interrupt Level 4	Interrupt Level 12
3	Interrupt Level 3	Interrupt Level 11
2	Interrupt Level 2	Interrupt Level 10
1	Interrupt Level 1	Interrupt Level 9
0	Interrupt Level 0	Interrupt Level 8

Software Implementation Note: Masking of interrupt level 2 on PCI1 (master) will effectively mask off all the interrupts which come into PIC2 (slave).

4.4.2.2 Operation Command Word 2 Register (OCW2)

Table 140. Operation Command Word 2 Register (OCW2) 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],reg[2]	0x00	Write Only	0x00	N/A

This register controls the interrupt priority and EOI command.

Table 141. Operation Command Word 2 Register (OCW2)

Bit	Function
7-5	**Rotate and EOI Codes (R,SL, EOI):** These bits define the Rotate mode, EOI mode, or a combination of the two, as shown in Table 142
4-3	**OCW2 Select:** Must be 0b00 to select OCW2
2- 0	**Interrupt Level Select:** When bit 6, SL, is set to 1, the encoding of this field as shown in Table 143 determines which interrupt channel the command acts on

Table 142. Rotate and EOI Code

Bit 7	Bit 6	Bit 5	Function
0	0	0	Reserved
0	0	1	Non-specific EOI Command
0	1	0	No Operation
0	1	1	Specific EOI Command. Interrupt channel number determined by setting of bits 2-0 (see Table 143)
1	0	0	Reserved
1	0	1	Rotate on Non-specific EOI Command
1	1	0	Set Priority Command. Interrupt channel number determined by setting of bits 2-0 (see Table 143)
1	1	1	Rotate on Specific EOI Command. Interrupt channel number determined by setting of bits 2-0 (see Table 143)

Table 143. Interrupt Level Select Encodings

Bit 2	Bit 1	Bit 0	PIC1 (master) Function	PIC2 (slave) Function
0	0	0	Interrupt Level 0	Interrupt Level 8
0	0	1	Interrupt Level 1	Interrupt Level 9
0	1	0	Interrupt Level 2	Interrupt Level 10
0	1	1	Interrupt Level 3	Interrupt Level 11
1	0	0	Interrupt Level 4	Interrupt Level 12
1	0	1	Interrupt Level 5	Interrupt Level 13
1	1	0	Interrupt Level 6	Interrupt Level 14
1	1	1	Interrupt Level 7	Interrupt Level 15

4.4.2.3 Operation Command Word 3 Register (OCW3)

Table 144. Operation Command Word 3Register (OCW3) 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], reg[2]	0x00	Read/Write	0x00	N/A

This byte is written to either PIC1 (master) or PIC2 (slave).

Table 145. Operation Command Word 3 Register (OCW3)

Bit	Function
7	**Reserved:** Must be set to 0
6	**Special Mask Mode (SMM):** See Table 146 on page 73
5	**Enable Special Mask Mode (ESMM):** See Table 146 on page 73
4-3	**OCW3 Select:** Must be set to 0b01
2	**Poll Command:** When set to a 0, the Poll Command mode is disabled, and when set to a 1, the next read to one of the Interrupt Controller registers will be treated as an Interrupt Acknowledge cycle, with the data being returned as shown in Table 148 on page 73
1,0	**Register Read Command:** After programming these bits with the appropriate values as shown in Table 147 on page 73, the selected register (IRR or ISR) can be read repeatedly without having to write OCW3 each time

Table 146. Interpretation of SMM and ESMM Bits

Bit 6 (SMM)	Bit 5 (ESMM)	Function
0	0	No Action
0	1	Normal Mask Mode
1	0	No Action
1	1	Special Mask Mode

Table 147. Register Read Command Decoding

Bits 1-0	Function
0x00	No Action
0x01	No Action
0x02	**Read Interrupt Request Register (IRR):** See Table 149 on page 74
0x03	**Read In-Service Register (ISR):** See Table 151 on page 75

Table 148. Interrupt Controller Response to a Read Pulse In Poll Mode

Bit	Function
7	**Interrupt Present:** This bit is set to 1 if an interrupt is present.

Table 148. Interrupt Controller Response to a Read Pulse In Poll Mode *(Continued)*

Bit	Function
6-3	**Undefined:** These bits are not used and may be set to either 0 or 1.
2-0	**Highest Priority Level:** These bits contain the binary code of the highest priority level requesting service

4.4.3 Interrupt Request Register (IRR)

The IRR indicates interrupts which are pending and which have not yet been acknowledged.

Table 149. Interrupt Request Register (IRR) 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	Indirect (OCW3)	reg[1], reg[2]	0x00	Read Only	Undefined	N/A

A 1 in any bit position indicates that there is an interrupt pending, a 0 indicates that there is no interrupt pending.

Table 150. Interrupt Request Register (IRR)

Bit	PIC1 (master)	PIC2 (slave)
7	Interrupt Level 7	Interrupt Level 15
6	Interrupt Level 6	Interrupt Level 14
5	Interrupt Level 5	Interrupt Level 13
4	Interrupt Level 4	Interrupt Level 12
3	Interrupt Level 3	Interrupt Level 11
2	Interrupt Level 2	Interrupt Level 10
1	Interrupt Level 1	Interrupt Level 9
0	Interrupt Level 0	Interrupt Level 8

4.4.4 In-Service Register (ISR)

Requirements:

4–17. When an in-service bit is set to 1, the Interrupt Controller must inhibit all other interrupts with the same or lower priority.

4–18. When an in-service bit is set to 1, the Interrupt Controller must allow interrupts with a higher priority to cause an interrupt.

The ISR indicates interrupts which have been acknowledged but have not had an EOI issued for them.

Table 151. In-Service Register (ISR) 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	Indirect (OCW3)	reg[1], reg[2]	0x00	Read Only	Undefined	N/A

If the bit is a 1, then there is an interrupt that has been acknowledged but not yet EOI'ed.

Table 152. In-Service Register (ISR)

Bit	PIC1 (master)	PIC2 (slave)
7	Interrupt Level 7	Interrupt Level 15
6	Interrupt Level 6	Interrupt Level 14
5	Interrupt Level 5	Interrupt Level 13
4	Interrupt Level 4	Interrupt Level 12
3	Interrupt Level 3	Interrupt Level 11
2	Interrupt Level 2	Interrupt Level 10
1	Interrupt Level 1	Interrupt Level 9
0	Interrupt Level 0	Interrupt Level 8

4.4.5 Edge/Level Interrupt Control Registers (ELI)

These registers determine whether an interrupt is edge sensitive or level sensitive. After reset all registers are set to edge sensitive (positive edge; that is, low to high transition).

Requirements:

4–19. For the device described herein, all the registers defined in sections 4.4.5.1 through 4.4.5.2 must be implemented precisely as described herein.

4.4.5.1 Edge/Level Interrupt Control Register for PIC1 (master) (ELI1)

Table 153. Edge/Level Interrupt Control Register 1 (ELI1) 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[3]	0x00	Read/Write	Undefined	N/A

Table 154. Edge/Level Interrupt Control 1 Register (ELI1)

Bit	Function
7	I**NT7:** When this bit is 0, INT7 is edge sensitive, when this bit is a 1, INT7 is level-sensitive
6	**INT6:** When this bit is 0, INT6 is edge sensitive, when this bit is a 1, INT6 is level-sensitive
5	**INT5:** When this bit is 0, INT5 is edge sensitive, when this bit is a 1, INT5 is level-sensitive
4	**INT4:** When this bit is 0, INT4 is edge sensitive. when this bit is a 1, INT4 is level-sensitive
3	**INT3:** When this bit is 0, INT3 is edge sensitive, when this bit is a 1, INT3 is level-sensitive
2-0	**Reserved:** Always 0 (INT2, INT1, and INT0 are edge sensitive)

4.4.5.2 Edge/Level Interrupt Control Register for PIC2 (slave) (ELI2)

Table 155. Edge/Level Interrupt Control 2 Register (ELI2) 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[3]	0x01	Read/Write	Undefined	N/A

Table 156. Edge/Level Interrupt Control 2 Register (ELI2)

Bit	Function
7	**INT15:** When this bit is 0, INT15 is edge sensitive,.when this bit is a 1, INT15 is level-sensitive
6	**INT14:** When this bit is 0, INT14 is edge sensitive when this bit is a 1, INT14 is level-sensitive
5	**Reserved:** Always 0 (INT13 is edge sensitive)
4	**INT12:** When this bit is 0, INT12 is edge sensitive, when this bit is a 1, INT12 is level-sensitive
3	**INT11:** When this bit is 0, INT11 is edge sensitive, when this bit is a 1, INT11 is level-sensitive

Table 156. Edge/Level Interrupt Control 2 Register (ELI2) ***(Continued)***

Bit	Function
2	**INT10:** When this bit is 0, INT10 is edge sensitive, when this bit is a 1, INT10 is level-sensitive
1	**INT9:** When this bit is 0, INT9 is edge sensitive, when this bit is a 1, INT9 is level-sensitive
0	**Reserved:** Always 0 (INT8 is edge sensitive)

4.5 References

1. *Intel 82378ZB System I/O (SIO), Intel Corporation* [20].
2. *SL82C565 System I/O Controller with PCI Arbiter*, Symphony Laboratories Publication Number 2565; Version A.2. [21]

Parallel Port Controller

5

Printers have been attached to PCs, workstations, and servers for years with a parallel port which was developed several decades ago. Over time, this port has been enhanced to increase the data rates and add additional capabilities such as bi-directional capabilities, addressing, data transfer by DMA, and so on. This chapter describes the requirements for the Parallel Port Controller for platforms based on the Common Hardware Reference Platform architecture.

The following is a general description of the various modes in which the Parallel Port Controller can operate:

- *Standard Parallel Port (SPP) mode* is a combination of what the IEEE 1284 Specification [25] calls Compatibility mode and Nibble mode. This is one of the oldest modes and therefore the most common. Data transfer in this mode is mainly in the forward (host to device) direction, however four control lines can be used for transfer of data from the device to the host in the Nibble mode. All IEEE 1284 compatible devices are required to support the Nibble mode. In this mode there is no FIFO or DMA and all of the data transfer is done a byte at a time (or a nibble at a time in the Nibble mode) by the software, with software also directly controlling the control signals to the printer.
- *SPP Extended mode* adds to the SPP mode the capability of transferring data in the reverse direction (device to host) on the eight data lines. The IEEE 1284 Specification calls this the Byte mode.
- *Enhanced Parallel Port (EPP) mode* provides an interface from the host to the device which is similar to a typical microprocessor bus, with an address being provided followed by one or more data cycles. Although there are two modes detailed in some specifications (versions 1.7 and 1.9), only the more recent version, 1.9, is required by this architecture. In this mode, the address and data strobe signals to the device are controlled by the Parallel Port Controller hardware. This mode is tied to the microprocessor bus timing, with the bus being held off for the duration of the read or write to the bus (there is no First-In-First-Out (FIFO) buffer to speed-match the incoming data to the outgoing data).
- *Extended Capabilities Port (ECP) mode* has channel-like characteristics where there are 128 channels that can be addressed between the host and the device. All signalling is controlled by the Parallel Port Controller hardware and there is a FIFO to match the speed of the system to the speed of the peripheral device, so this is one of the faster modes of operation. In addition, filling and emptying of the FIFO by DMA is supported in this mode.
- *Parallel Port FIFO mode* is like the SPP mode except there is hardware control of the interface signals, there is a speed matching FIFO, and DMA is supported for FIFO fill. Note that the direction for this mode is always the forward direction.
- *Test FIFO mode* is a mode for testing of ECP device driver code and for diagnostic purposes. Data written to the FIFO can be read back. Data goes out on the data lines to the attached peripheral device, but the device does not see the data because the control signals are not activated.

The capabilities of these modes are summarized in Table 157 on page 80.

Table 157. Summary of the Parallel Port Controller Modes

Mode	Reverse Data Path	Control of Interface Signals by Hardware or Software?	FIFO Support	DMA Support	Device Address Control
SPP	4 bits via interface control signals	Software	No	No	None
SPP Extended	8 bit via data lines	Software	No	No	None
EPP	8 bit via data lines	Hardware	No	No	8 bit address
ECP	8 bit via data lines	Hardware	Yes	Yes	7 bit channel address
Parallel Port FIFO	Forward direction only (reverse support via SPP mode)	Hardware	Yes	Yes	None
Test FIFO	Reverse direction gives data written to FIFO in forward direction	N/A	Yes	Yes	N/A

5.1 General Requirements

Table 158 on page 80 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements of the Parallel Port Controller on CHRP platforms.

Table 158. Minimum Platform Requirements for the Parallel Port

Subsystem	Specification	Portable	Personal	Server	Description
Parallel Port	P1284 +ECP Mode	O	R	O	[20]

Requirements:

5–1. The Parallel Port Controller must be represented by the Open Firmware properties “name=parallel, device_type=parallel, compatible=chrp,ecp”. The compatible property may also specify the compatible property for the National Semiconductor 87308 in addition to “chrp,ecp”.

5–2. The Parallel Port Controller must protect against damage to the Parallel Port Controller electronics in the case where the device attached to the Parallel Port has its power on and the Parallel Port Controller has its power off.

5–3. The device described herein must support the following modes as defined by the IEEE 1284 Specification: Compatibility mode, Nibble mode (Compatibility mode along with Nibble mode are called Standard Parallel Port, SPP, in this specification), Byte mode (called SPP Extended mode in this specification), Enhanced Parallel Port (EPP) mode (version 1.9), and Extended Capabilities Port (ECP) mode.

5–4. The device described herein must also support the following modes as described herein: Parallel Port FIFO mode and Test FIFO mode.

5–5. This device must be addressed purely with ISA I/O addresses which can be aliased or non-aliased.

5–6. DMA transfers fro the Parallel Port Controller must be 8-bit DMA slave and must support at least the ISA Compatible Timing mode.

5–7. Interrupts generated by the Parallel Port Controller must be high true edge sensitive.

5–8. The Parallel Port Controller registers specified herein must be implemented precisely as defined, including the following characteristics and bit definitions:

- **a.** Table 160, “Data Register (DTR) Characteristics,” on page 84
- **b.** Table 161, “Data Register (DTR),” on page 84
- **c.** Table 162, “Status Register (STR or DSR) Characteristics,” on page 85
- **d.** Table 163, “Status Register (STR or DSR),” on page 85
- **e.** Table 164, “Control Register (CTR or DCR) Characteristics,” on page 85
- **f.** Table 165, “Control Register (CTR or DCR),” on page 86
- **g.** Table 166, “Extended Control Register (ECR) Characteristics,” on page 87
- **h.** Table 167, “Extended Control Register (ECR),” on page 87
- **i.** Table 168, “EPP Address Register (ADDR) Characteristics,” on page 88
- **j.** Table 169, “EPP Data Port 0 Register (DTR) Characteristics,” on page 88
- **k.** Table 170, “EPP Data Port 1 Register (DATA1) Characteristics,” on page 88
- **l.** Table 171, “EPP Data Port 2 Register (DATA2) Characteristics,” on page 89
- **m.** Table 172, “EPP Data Port 3 Register (DATA1) Characteristics,” on page 89
- **n.** Table 173, “ECP Address FIFO (AFIFO) Characteristics,” on page 90
- **o.** Table 174, “Address FIFO Register (AFIFO),” on page 90
- **p.** Table 175, “ECP Data FIFO Register (DFIFO) Characteristics,” on page 90
- **q.** Table 176, “ECP Data FIFO Register (DFIFO),” on page 90
- **r.** Table 177, “Parallel Port FIFO Register (CFIFO) Characteristics,” on page 91
- **s.** Table 178, “Parallel Port FIFO Register (CFIFO),” on page 91
- **t.** Table 179, “Test FIFO Register (TFIFO) Characteristics,” on page 91
- **u.** Table 180, “Test FIFO Register, TFIFO,” on page 91

Hardware Implementation Note: IEEE Std 1284-1994 specification recommends the use of the IEEE 1284-C connectors in all new designs (host side as well as peripheral side).

Software Implementation Note: Note that the operating system is expected to extract the DMA channel number from the node’s *dma* property. Requirement 5–6 means that the operating system need not extract the DMA transfer size, DMA count width, or master versus slave capability from the node’s *dma* property. Firmware must set these fields to the values that match the above requirement. Operating systems are recommended to extract the DMA timing from the node’s *dma* property, though they are free to default to the ISA Compatibility Timing mode.

Software Implementation Note: Note that the operating system is expected to extract the interrupt number from the node's *interrupts* property. Requirement 5–7 means that the operating system need not extract the interrupt type from the node's *interrupts* property. Firmware must set this field to the value that matches the above requirement.

5.2 Open Firmware Requirements

Requirements:

5–9. The Open Firmware "reg" property information given in Table 159 on page 83 must be provided in the Open Firmware device tree for this device.

5–10. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

5–11. This device must be represented in the OF device tree as specified in the *CHRP ISA Parallel Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*. [23]

5–12. All the reg property fields specified in Table 159 on page 83 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.

5–13. The following events must occur during the OF initialization sequence:

a. The Parallel Port Controller must be enabled and activated.

b. If there is I/O Range checking of the Parallel Port Controller it must be disabled.

c. The parallel Port Controller address space must be set up, enabled.

d. The interrupt level and type (edge versus level) must be setup, enabled, and reported in the OF device tree.

e. Eight-bit DMA for the Parallel Port Controller must be: set up, enabled, and reported in the OF device tree.

f. Demand DMA must be enabled

g. Both ECP and EPP 1.9 modes must be enabled.

h. The Parallel Port Clock must be enabled (that is, clock must be operational).

i. The Parallel Port Controller must be set up to disable its outputs when the Parallel Port is disabled.

j. All Parallel Port Controller registers must be initialized to the values indicated in the "Value in Register When OF Passes Control to OS" column of the characteristics table corresponding to the specific register.

The Parallel Port Controller reg properties are listed below in Table 159 on page 83. The positions in the reg property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 159 on page 83. The format is described in the referenced Open Firmware bindings.

Table 159. Parallel Port Controller Registers and Offsets

Name	Offset	Type	Mode	Reg Properties for Open Firmware
DTR	0x00	Read/Write	SPP, SPP Extended, EPP	reg[1], size=8
DATAR	0x00	Read Only	ECP	
AFIFO	0x00	Write Only	ECP	
STR	0x01	Read Only	SPP, SPP Extended, EPP Parallel Port FIFO, Test FIFO	
DSR			ECP	
CTR	0x02	Read/Write	SPP, SPP Extended, EPP Parallel Port FIFO, Test FIFO	
DCR			ECP	
ADDR	0x03	Read/Write	EPP	
DATA0	0x04	Read/Write	EPP	
DATA1	0x05	Read/Write	EPP	
DATA2	0x06	Read/Write	EPP	
DATA3	0x07	Read/Write	EPP	
CFIFO	0x00	Write Only	Parallel Port FIFO	reg[2], size=6
DFIFO	0x00	Read/Write	ECP	
TFIFO	0x00	Read/Write	Test FIFO	
Reserved	0x01			
ECR	0x02	Read/Write	All	
Reserved	0x03			
Reserved	0x04			
Reserved	0x05			

Hardware Implementation Note: The Parallel Port Controller cannot be located at the legacy LPT1 addresses because of the size of 8 bytes would cause it to overlap with other allocated I/O addresses.

5.3 Parallel Port Register Definitions

For the device described herein, all the registers are defined in sections 5.3.1.1 through 5.3.1.4. The purpose and operation of some of these registers changes depending on the mode in which the parallel controller is operating at the time.

5.3.1 Registers Used in Multiple Modes

The first of the registers in this section is defined for all modes except ECP. The rest of the registers in this section are defined for all modes.

5.3.1.1 DATA Register (DTR or DATAR)

This register is accessed in SPP, SPP Extended, and EPP 1.9 mode as a Read/Write register, and in ECP mode as a Read Only register. In all modes except ECP, this register is abbreviated DTR. In ECP mode this register is abbreviated DATAR. This register is for writing data to or reading data from the parallel interface.

Table 160. Data Register (DTR) 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 - Mode Selected	reg[1]	0x00	Read/Write	0x00	N/A

This register is defined as shown in Table 161.

Table 161. Data Register (DTR)

Bit	Description
7-0	**Data:** The action performed on a read or write depends on the mode in which the Parallel Port Controller is operating. In all modes, when the direction is in the forward direction (bit 5 of the Extended Control Register (ECR) is a 0), on a read the data returned is either the data in the Data register or the data on the input lines to the chip (signal lines PD7-0), depending on the implementation. In SPP mode on a write, the data in this register is driven directly to the output signal pins PD7-0. In SPP Extended mode: • On a write, data written to this register is latched and is gated to the output depending on the direction bit in the Control register (CTR). If the direction bit (bit 5) is a 0, then data transfer is from the Parallel Port to the device (forward direction) and therefore the data written is latched and gated to signal lines PD7-0. If the direction bit (bit 5) is a 1, then data transfer is from the device to the Parallel Port (reverse direction) and therefore the data written is latched into the Data register but not gated out onto the signal lines. • On a read with the direction bit a 1, then data transfer is from the device to the Parallel Port (reverse direction) and the data read is the data on the input lines to the chip (signal lines PD7-0). In EPP 1.9 mode: same as SPP mode except that in EPP mode, the busy bit of the STR must be checked to make sure it is a 1 (not busy) before writing to the DTR, in order to avoid possible corruption of an EPP cycle that is in progress. In ECP mode: For reads, same as SPP Extended mode. For writes, undefined.

Software Implementation Note: Damage to the drivers in the Parallel Port Controller interface or the attached device can occur if the controller and the attached device are driving data onto the data lines at the same time.

5.3.1.2 Status Register (STR or DSR)

This register returns the interrupt-pending status of the interface, and the real-time status of the connector pins. In all modes except ECP, this register is abbreviated STR. In ECP mode this register is abbreviated DSR.

Table 162. Status Register (STR or DSR) 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]	0x01	Read/Write	Undefined	N/A

The Status register is defined as shown in Table 163.

Table 163. Status Register (STR or DSR)

Bit	Description
7	**$\overline{\text{Busy}}$:** This bit represents the inverse of the state of the busy signal. When this bit is 0, the printer is busy and cannot accept data. In the Nibble mode, this is bit 3 and then bit 7 of the data. This bit is Read Only.
6	**$\overline{\text{Acknowledge}}$ ($\overline{\text{ACK}}$):** This bit represents the current state of the device acknowledge ($\overline{\text{ACK}}$) signal. When this bit is pulsed from a 1 to a 0 to a 1, the device has received a character and is ready to accept another. This bit is Read Only.
5	**Paper End (PE):** This bit represents the current state of the device paper end (PE) signal. When this bit is 1, the printer detected the end of the paper. In the Nibble mode, this is bit 2 and then bit 6 of the data. This bit is Read Only.
4	**Select (SLCT):** This bit represents the current state of the select (SLCT) signal. When this bit is 1, the printer has been selected. In the Nibble mode, this is bit 1 and then bit 5 of the data. This bit is Read Only.
3	**$\overline{\text{Error}}$:** This bit represents the current state of the device $\overline{\text{ERR}}$ signal. When this bit is 0, the printer has encountered an error condition. In the Nibble mode, this is bit 0 and then bit 4 of the data. This bit is Read Only.
2-1	**Reserved:** On a read, the returned data is undefined.
0	**Time-Out:** EPP Time-out bit. • When not in EPP mode: On a read, the returned data is undefined. • In EPP mode: A 0 indicates that no timeout has occurred and a 1 indicates that a time-out has occurred on an EPP cycle (minimum 10 microseconds). This bit is cleared to 0 after the STR is either read (some implementations) or written (other implementations). If cleared on a read of the bit, hardware must ignore writes to the bit. Software must write this bit to a 0 to guarantee that it gets reset.

5.3.1.3 Control Register (CTR or DCR)

In all modes except ECP, this register is abbreviated CTR. In ECP mode this register is abbreviated DCR. This register provides access to all the outbound control lines to the printer. All bits except bit 5 are Read/Write.

Table 164. Control Register (CTR or DCR) 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]	0x02	Read/Write	0xC0	N/A

The CTR (DCR) is defined as shown in Table 165.

Table 165. Control Register (CTR or DCR)

Bit	Description
7-6	**Reserved:** These bits are reserved and must be set to 0b11. On a read, the returned data is undefined.
5	**Direction:** When this bit is a 0, the Parallel Port Controller is in the forward direction (data transfer in the Parallel Port Controller to device direction). When this bit is a 1, the Parallel Port Controller is in the backward direction (data transfer in the device to Parallel Port Controller direction) and the Parallel Port Controller must not be driving the data lines. How this bit affects operations depends on which mode the Parallel Port Controller is operating. See also Table 161 on page 84. The Parallel Port Controller must be in the SPP Extended mode when this bit is changed from a 0 to a 1. • In SPP mode: This bit is Write Only. Must always be written to 0. Read as 0. • In SPP Extended mode: This bit is a Read/Write bit. • In EPP modes: Same as SPP except that this bit is a Read/Write bit and when there is an on-going EPP cycle, $\overline{RD}$ and $\overline{WR}$ signals on the bus determine the direction. This bit must not be changed from a 0 to a 1 in this state. • In ECP mode: This bit is a Read/Write bit. This bit must not be changed from a 0 to a 1 in this state. • In Parallel Port FIFO mode: This bit is Write Only. Must always be written to 0. • In Test FIFO mode: This bit is a Read/Write bit. This bit must not be changed from a 0 to a 1 in this state.
4	**Interrupt Enable:** This bit enables the $\overline{ACK}$ interrupt event (1=enable, 0=mask). An interrupt occurs when this bit is a 1 and either the $\overline{acknowledge}$ signal changes from active to inactive (non-DMA mode only) or when the EPP timeout bit (bit 0 of the Status register) gets set to 1 in the EPP mode. In all modes except the ECP mode, this bit floats (high-impedance state) the IRQ pin when this bit is 0.
3	**SLIN:** This bit controls the select in signal. When this bit is 1, the printer is selected. This bit must be set to a 1 before switching to the ECP mode. • In all modes except ECP mode: If the current state of the actual signal from the device is to be obtained by reading this register, this bit should be set to its inactive state (0). • In ECP mode: On a read, this bit will always reflect the value in the Parallel Port Controller's DCR register. • In EPP modes: When this bit is a 0, the $\overline{SLIN}$ signal is controlled by ECP hardware.
2	**$\overline{INIT}$:** This bit controls the initialize signal. When this bit is 0, the printer will be initialized. this bit must be set to a 1 before switching to the ECP mode. • In all modes except ECP mode: If the current state of the actual signal from the device is to be obtained by reading this register, this bit should be set to its inactive state (1). • In ECP mode: On a read, this bit will always reflect the value in the Parallel Port Controller's DCR register.
1	**AFD:** This bit controls the automatic feed XT signal. When this bit is 1, the $\overline{AFD}$ signal is driven low and the printer automatically spaces the paper up one line for every carriage return. • In all modes except ECP mode: If the current state of the actual signal from the device is to be obtained by reading this register, this bit should be set to its inactive state (0). • In ECP mode: On a read, this bit will always reflect the value in the Parallel Port Controller's DCR register. • In ECP and EPP modes: When this bit is a 0, the $\overline{AFD}$ signal is controlled by ECP hardware.
0	**STB:** This bit controls the $\overline{strobe}$ ($\overline{STB}$) signal to the printer. When this bit is 1 the $\overline{STB}$ line is activated, causing data is clocked into the attached device's data register. • In all modes except ECP mode: If the current state of the actual signal from the device is to be obtained by reading this register, this bit should be set to its inactive state (0). • In ECP mode: On a read, this bit will always reflect the value in the Parallel Port Controller's DCR register. • In Parallel Port FIFO mode and ECP mode: When this bit is 0, $\overline{STB}$ is controlled by Parallel Port Controller hardware.

5.3.1.4 Extended Control Register (ECR)

Table 166. Extended Control Register (ECR) 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[2]	0x02	Read/Write	0x15	N/A

The ECR register is defined as shown in Table 167.

Table 167. Extended Control Register (ECR)

Bit	Description
7-5	**Parallel Port Mode Bits:** These bits determine the mode in which the Parallel Port is operating. • 000 - SPP (Compatibility and Nibble modes together): In this mode, write cycles are controlled by the software. In this mode, the direction bit in the DCR is not used and PD7-0 are driven. The FIFO is reset and is empty. • 001 - SPP Extended (Byte mode): Read and Write are under software control, the FIFO is reset, and the direction bit is active. • 010 - Parallel Port FIFO: Write cycles are controlled by hardware with STB being generated by the hardware. The direction bit of the DCR is assumed to be 0 (this mode is defined for the forward direction only) and PD7-0 are driven. • 011 - ECP: The FIFO direction is controlled by the direction bit in the DCR. $\overline{STB}$ and $\overline{AFD}$ are controlled by hardware. • 100 - EPP (1.7 and 1.9). • 101 - Reserved. • 110 - Test FIFO: The FIFO is accessible via the TFIFO register. The hardware does not automatically fill and empty the FIFO. • 111 - Reserved.
4	**Interrupt Mask Bit:** When this bit is 0, an interrupt is generated on $\overline{ERR}$ assertion from the parallel interface (the high to low edge of $\overline{ERR}$). An interrupt is also generated if $\overline{ERR}$ is asserted when this bit is changed from 1 to 0. When this bit is a 1, no interrupt will be generated when $\overline{ERR}$ is activated.
3	**DMA Enable Bit:** When this bit is a 0, the DMA request signal (DMA0, 1, 2 or 3) that serves the selected DMA request signal is put into tri-state (high-impedance) state. When this bit is a 1, the DMA is enabled and the DMA starts when bit 2 of this register (ECP Service) is 0.
2	**Service Bit:** When the ECP clock is frozen, this bit is read as 0 regardless of its actual value (even though this bit may be modified by software when the ECP clock is frozen). When one of the following interrupt events occurs, an interrupt will be generated and this bit will be set to a 1 by the hardware: • Bit 3 of the ECR is a 1 and the terminal count is reached during DMA • Bit 3 of the ECR is 0 and bit 5 of the DCR (CTR) is 0 and there are eight or more bytes free in the FIFO • Bit 3 of the ECR is a 0 and bit 5 of the DCR (CTR) is a 1 and there are eight or more bytes to be read from the FIFO. When this bit is a 1, no further DMA requests or interrupt requests are generated. Writing a a 1 to this bit does not cause an interrupt.
1	**FIFO Full:** This bit is Read Only. A write to the ECR will not change its value. This bit continuously reflects the FIFO state. When the ECP clock is frozen this bit is read as 1, regardless of the actual FIFO state. When this bit is a 0, the FIFO has at least one free byte. When this bit is a 1, the FIFO is full.
0	**FIFO Empty:** This is a Read Only bit that continuously reflects the FIFO state. When this bit is a 0, the FIFO has at least one byte of data. When this bit is a 1, the FIFO is empty. If the ECP clock is frozen, this bit is a 1 regardless of the actual FIFO state.

5.3.2 Registers Used Only in EPP Mode

The registers in this section are only defined in EPP modes.

5.3.2.1 EPP Address Register (ADDR)

Table 168. EPP Address Register (ADDR) 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 - Mode Selected	reg[1]	0x03	Read/Write	Undefined	N/A

Requirements:

5–14. Writing a Parallel Port device address to this register must initiate an EPP device/register selection operation.

5.3.2.2 EPP Data Port 0 Register (DATA0)

Table 169. EPP Data Port 0 Register (DTR) 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 - Mode Selected	reg[1]	0x04	Read/Write	Undefined	N/A

Requirements:

5–15. Accesses to this register must initiate device read or write operations to bits 7-0 of the 32 bit word to be transferred.

5.3.2.3 EPP Data Port 1 Register (DATA1)

Table 170. EPP Data Port 1 Register (DATA1) 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 - Mode Selected	reg[1]	0x05	Read/Write	Undefined	N/A

Requirements:

5–16. Accesses to this register must initiate device read or write operations to bits 15-8 of the 32 bit word to be transferred.

5.3.2.4 EPP Data Port 2 Register (DATA2)

Table 171. EPP Data Port 2 Register (DATA2) 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 - Mode Selected	reg[1]	0x06	Read/Write	Undefined	N/A

Requirements:

5–17. Accesses to this register must initiate device read or write operations to bits 23-16 of the 32 bit word to be transferred.

5.3.2.5 EPP Data Port 3 Register (DATA3)

Table 172. EPP Data Port 3 Register (DATA1) 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 - Mode Selected	reg[1]	0x07	Read/Write	Undefined	N/A

Requirements:

5–18. Accesses to this register must initiate device read or write operations to bits 31-24 of the 32 bit word to be transferred.

5.3.3 Registers Used Only in ECP Mode

The registers in this section are only defined in the ECP mode.

Software Implementation Note: When the OF compatible property includes the compatible name of the National Semiconductor 87308 or when the OF property *dma-308* is present, then the capability to do demand DMA gets disabled on exit from the ECP mode. In this case, to re-enable this capability, software should do the following:

1. Make sure that the Parallel Port Controller is quiesced
2. Write 0x05 to reg[1]+0x03
3. Read a byte from reg[1]+0x04
4. Set bit 6 of the byte just read to a 1
5. Write this byte to reg[1]+0x04

5.3.3.1 ECP Address FIFO Register (AFIFO)

In the ECP mode, when the Parallel Port's state is in the forward direction (bit 5 of the DCR is 0), a byte written into this register is pushed into the FIFO and tagged as a command. Writes to this register when the state of the Parallel Port is in the backward direction have no effect and the data is ignored.

Table 173. ECP Address FIFO (AFIFO) 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 - Mode Selected	reg[1]	0x00	Write Only	Undefined	N/A

The AFIFO is defined as shown in Table 174.

Table 174. Address FIFO Register (AFIFO)

Bit	Description
7-0	**Command:** These bits represent data that will be tagged as a command in the AFIFO register

5.3.3.2 ECP Data FIFO Register (DFIFO)

DFIFO is accessible in ECP mode. In the forward direction (bit 5 of the DCR is 0) a byte written to this register is pushed into the FIFO and tagged as data. In ECP mode, the DMA automatically writes to this register. Other write operations must address this register specifically. Reading this register has no effect and the data read is undefined.

In the backward direction (bit 5 of the DCR is 1) the ECP automatically issues ECP read cycles to fill the FIFO. Reading this register pops a byte from the FIFO. Writing this register has no effect and the data written is ignored.

Table 175. ECP Data FIFO Register (DFIFO) 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 - Mode Selected	reg[2]	0x00	Read/Write	Undefined	N/A

The DFIFO register is defined as shown in Table 176.

Table 176. ECP Data FIFO Register (DFIFO)

Bit	Description
7-0	**Data:** These bits represent the data that will be tagged as data when pushed onto or popped from the FIFO.

5.3.4 Registers Used Only in the Parallel Port FIFO Mode

CFIFO is accessible in Parallel Port FIFO mode. A byte written to CFIFO is pushed into the FIFO and tagged as data. In Parallel Port FIFO mode the DMA automatically writes to this register. Other write operations must address this register specifically. This mode is defined for the forward direction only, therefore this register is Write Only.

Table 177. Parallel Port FIFO Register (CFIFO) 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 - Mode Selected	reg[2]	0x00	Write Only	Undefined	N/A

The CFIFO register is defined as shown in Table 178.

Table 178. Parallel Port FIFO Register (CFIFO)

Bit	Description
7-0	**Data:** These bits represent the data that will be tagged as data when pushed onto the FIFO.

5.3.5 Registers Used Only in the Test FIFO Mode

TFIFO is accessible in Test FIFO mode. A byte written into this register is pushed into the FIFO. A byte read from this register is popped from the FIFO. The ECP does not issue a ECP cycle to transfer the data to or from the peripheral device. The TFIFO is readable and writable in both directions. In the forward direction (bit 5 of the DCR (CTR) is 0) PD0-7 are driven, but the data is undefined. The FIFO does not stall when overwritten or under-run (access is ignored). Bytes are always read from the top of the FIFO, regardless of the direction bit (bit 5 of the CTR or DCR).

Table 179. Test FIFO Register (TFIFO) 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 - Mode Selected	reg[2]	0x00	Read/Write	Undefined	N/A

The TFIFO register is defined as shown in Table 180.

Table 180. Test FIFO Register, TFIFO

Bit	Description
7-0	**Data:** These bits represent the data that will be tagged as data when pushed onto or popped from the FIFO.

5.4 References

1. *National Semiconductor PC87308VUL SuperI/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 IEEE1284 Parallel Port* [24]
2. *CHRP ISA Parallel Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware.* [23]
3. *IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers*, also called *IEEE Std 1284-1994 Specification* [25]

UART Controller

6

6.1 CHRP Requirements

Table 181 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements for a serial port on CHRP platforms.

Table 181. 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:

6–1. The UART serial device must be NS16550 compatible, as specified in this chapter.

6–2. The UART serial device must be addressed only with ISA I/O addresses.

6–3. The UART serial device must generate only low-to-high edge sensitive interrupts.

Software Implementation Note: Requirements 6–2 and 6–3 relieve the operating system from having to extract address type or interrupt type from the Open Firmware device tree.

6.2 UART Controller Open Firmware Requirements

Requirements:

6–4. The Open Firmware device tree node for the UART serial device described in this chapter must follow the specifications given in the *CHRP ISA Serial Port Device Binding* [28].

6–5. The Open Firmware "`reg`" property information given in Table 182, "UART Registers," on page 94 must be provided in the Open Firmware device tree for the UART serial device.

6–6. The UART serial device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

6.3 UART Registers

Requirements:

6–7. All the "**reg**" property fields specified in Table 182, "UART Registers," on page 94 must comply with the format given in the ISA/EISA/ISA-PnP binding to IEEE Std 1275-1994, as modified by the CHRP binding to IEEE Std 1275-1994.

6–8. The UART serial device must precisely implement the registers defined in Section 6.3, "UART Registers," on page 94.

6–9. The selection of the TDR/IER versus the DLL/DLH for the UART serial device must be exclusively controlled by the DLAB bit (bit 7) in the Line Control Register.

The positions in the "**reg**" property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 182 on page 94. The format is described in the referenced Open Firmware bindings.

Table 182. UART Registers

Address Offsets	Registers		Reg Properties
	Read	Write	
0x0	Receive Data Register (RDR)	Transmitter Data Register (TDR) OR[1] Divisor Latch Low Byte (DLL)	reg[1],size=8
1	Interrupt Enable Register (IER)	Interrupt Enable Register (IER) OR[1] Divisor Latch High Byte (DLH)	
2	Interrupt Identification Register (IIR)	FIFO Control Register (FCR)	
3	Line Control Register (LCR)	Line Control Register (LCR)	
4	Modem Control Register (MCR)	Modem Control Register (MCR)	
5	Line Status Register (LSR)	Writing the LSR is not recommended	
6	Modem Status Reg (MSR)	Modem Status Reg (MSR)	
7	Scratch Register (SCR)	Scratch Register (SCR)	

[1]These address offsets have dual register definitions based on the state of the DLAB bit in the Line Control Register (LCR). DLAB=0 references the TDR and IER; DLAB=1 references DLL and DLH

6.3.1 Transmitter Data Register (TDR)

Table 183. Transmitter Data Register (TDR) 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)=0	irba[0]	0x00	Write Only	Undefined	N/A

Table 184. Transmitter Data Register Bit Definition

Bit	Description
7-0	**Data Bits 7-0:** The Transmitter Data Register contains the character to be sent. Bit 0 is the least-significant bit and is the first bit sent serially.

6.3.2 Receiver Data Register (RDR)

Table 185. Receiver Data Register (RDR) 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)=0	irba[0]	0x00	Read Only	Undefined	N/A

Table 186. Receiver Data Register Bit Definition

Bit	Description
7-0	**Data Bits 7-0:** The Receiver Data Register contains the received character. Bit 0 is the least-significant bit and is the first bit received serially.

6.3.3 Interrupt Enable Register (IER)

The Interrupt Enable Register allows the four types of controller interrupts to separately activate the chip interrupt output signal. The interrupt system can be completely disabled by setting bits 0 through 3 of the Interrupt Enable register to 0. Similarly, by setting the appropriate bits of this register to 1, selected interrupts can be enabled. Disabling prevents the controller from generating the external interrupt to the system. All other system functions operate normally, including the setting of the Line Status and Modem Status registers.

Table 187. Interrupt Enable Register (IER) 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)=0	irba[0]	0x01	Read/Write	0x00	N/A

Table 188. Interrupt Enable Register Bit Definition

Bit	Description
7-4	**Reserved** = 0
3	**Enable Modem Status Interrupt:** When set to 1, this bit enables the Modem Status Interrupt.
2	**Enable Receiver Line Status Interrupt:** When set to 1, this bit enables the Receiver Line Status Interrupt.
1	**Enable Transmitter Holding Register Empty Interrupt:** When set to 1, this bit enables the Transmitter Holding Register Empty Interrupt.
0	**Enable Received Data Available Interrupt (Character and FIFO Mode) and Time-Out Interrupts (FIFO Mode Only):** When set to 1, this bit enables the Received Data Available Interrupt. In the FIFO mode, this bit also enables the time-out interrupts.

6.3.4 FIFO Control Register (FCR)

The FIFO Control Register is a write-only register at the same location as the read-only Interrupt Identification Register. The FIFO Control Register enables the FIFO registers, clears the FIFO registers, and sets the Receiver FIFO register trigger level.

Software Implementation Note: The Transmitter and Receiver FIFO registers are not accessible serial controller registers.

Table 189. FIFO Control Register (FCR) 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]	0x02	Write Only	Undefined	N/A

Table 190. FIFO Control Register Bit Definition

Bit	Description
7, 6	**Receiver FIFO Register Trigger:** These bits select the trigger level for the receiver-register interrupt, as shown in Table 191. When the number of bytes in the RCVR FIFO reaches the selected interrupt trigger level, a Receiver Data Available Interrupt is activated.
5,3	**Reserved** = 0

Table 190. FIFO Control Register Bit Definition ***(Continued)***

Bit	Description
2	**Transmitter FIFO Register Reset:** When this bit is set to 1, all bytes in the Transmitter FIFO register are cleared and its counter logic is reset to 0. The Transmitter Shift register is not cleared. This bit is self-clearing.
1	**Receiver FIFO Register Reset:** When this bit is set to 1, all bytes in the Receiver FIFO register are cleared and its counter logic is reset to 0. The Transmitter Shift register is not cleared. This bit is self-clearing.
0	**FIFO Mode Enable:** When this bit is set to 1, the FIFO mode is enabled. When this bit is changed, the transmit and receive (XMIT and RCV) FlFOs are cleared. When writing to any other FIFO Control register bits, this bit must be a 1.

Table 191. Receiver Trigger Level Bit Decodes

Bits 7,6	Receiver Trigger Level
0 0	1 Byte
0 1	4 Bytes
1 0	8 Bytes
1 1	14 Bytes

6.3.5 Interrupt Identification Register (IIR)

To minimize programming overhead during character mode transfers, the controller prioritizes interrupts into four levels:

- Priority 1 - Receiver-line-status
- Priority 2 - Received-data-available
- Priority 2 - Time-out (FIFO mode)
- Priority 3 - Transmitter-holding-register-empty
- Priority 4 - Modem status.

Information about a pending interrupt is stored in the Interrupt Identification Register (IIR). When the IIR is read, only the highest priority pending interrupt is indicated. After this interrupt condition is reset, other lower priority interrupts can be identified by re-reading the IIR.

Table 192. Interrupt Identity Register (IIR) 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]	0x02	Read Only	0x01	N/A

Table 193. Interrupt Identification Register Bit Definition

Bit	Description
7, 6	**FIFO Registers Enabled:** These two bits are set to 1 when FCR0=1 (FIFO mode enabled)
5, 4	**Reserved** = 0
3-0	**Interrupt ID (2-0), Interrupt Not Pending:** The interpretation of these bits is given in Table 194

Table 194. Interrupt Control Functions

Bits 3210	Priority	Type	Cause	Interrupt Reset Control
0001	n/a	none	no interrupt pending	n/a
0110	Highest	Receiver Line Status	Overrun, Parity or Framing error or Break Interrupt	Read the Line Status Register
0100	Second	Received data available	Data in the Receiver Buffer or the FIFO trigger level has been reached or a receive FIFO time-out occurs.	Read the Receiver Data Register or FIFO Register drops below the FIFO trigger level.
1100[1]	Second	Character Time-Out Indication	No characters have been removed from or put into the receiver FIFO register during the last four character times, and at least 1 character is in it at this time.	Read the Receiver Data Register.
0010	Third	Transmitter holding register empty	Transmitter holding register is empty	Read the Interrupt Identification Register or write to Transmitter Data Register
0000	Fourth	Modem status	Change in signal status from modem	Read the Modem Status Register.

[1] FIFO Mode only.

6.3.6 Line Control Register (LCR)

Asynchronous communications are programmed through the Line Control Register.

Table 195. Line Control Register (LCR) 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]	0x03	Read/Write	Undefined	N/A

Table 196. Line Control Register Bit Definition

Bit	Description
7	**Divisor Latch Access Bit (DLAB):** This bit is set to 1 to gain access to the divisor latches of the baud-rate generator. It is set to 0 to gain access to the Receiver Buffer, Transmitter Holding, or Interrupt Enable registers.
6	**Break Control:** When this bit is a 1, a break condition is transmitted to the receiving UART. A break condition is when the serial output is forced to the spacing state and remains there regardless of other transmitter activity. When this bit is set a 0, the break condition is not active.
5,4	**Parity Selection:** Bits 5 and 4 specify the parity selection as defined in Table 197. These bits have no effect if bit 3 is set to 0
3	**Parity Enable:** When bit 3 is set to 1, a parity bit is calculated based on the parity selection value in bits 5 and 4. This parity bit is generated on transmit and checked on receive. The parity bit appears between the last data bit and the first stop bit of the serial data. When bit 3 is set to 0, no parity bit is transmitted or checked.
2	**Number of Stop Bits:** The value of this bit, along with the value of bits 1 and 0, specify the number of stop bits (stop bit duration specified in terms of bit times). Table 198 gives the stop bit duration based on word length selection.
1,0	**Word Length Select, Bit 1,0:** These bits specify the number of data bits in each serial character that is sent or received as represented in Table 198

Table 197. Parity Selection Bit Description

Bits 5,4	Description
00	**Odd Parity Selected**
01	**Even Parity Selected**
10	**Mark Parity Selected:** A parity bit of logical 1 is always transmitted and checked.
11	**Space Parity Selected:** A parity bit of logical 0 is always transmitted and checked.

Table 198. Stop Bit Duration and Word Length

Bits 210	Stop Bit Duration (in bit times)	Word Length
000	1	5 Bits
001	1	6 Bits
010	1	7 Bits
011	1	8 Bits
100	1.5	5 Bits
101	2	6 Bits
110	2	7 Bits
111	2	8 Bits

6.3.7 Modem Control Register (MCR)

This 8-bit register controls the data exchange with the modem, data set, or peripheral device emulating a modem.

Table 199. Modem Control Register (MCR) 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]	0x04	Read/Write	0x00	N/A

Table 200. Modem Control Register Bit Description

Bit	Description
7-5	**Reserved** = 0
4	**Loop Mode:** This bit provides a loopback feature for diagnostic testing of the serial port. When bit 4 is set to 1: • Transmitter-serial-output is set to the marking state. • Receiver-serial-input is disconnected. • Output of the Transmitter Shift register is "looped back" to the Receiver Shift register input. • The modem control inputs (CTS, DSR, DCD, AND Rl) are disconnected. • The modem control outputs (DTR, RTS, OUT 1, and OUT 2) are internally connected to the (DSR, CTS, RI, and DCD) modem status inputs, respectively. • The modem control output pins are forced inactive.
3	**Out 2 (IRQ Output Control):** This bit is set = 1 to enable interrupts.
2	**Out 1:** This bit is not used in a normal mode. In loop mode, its status is reported to bit 6 (Rl) of the Modem Status register.
1	**Request-to-Send:** This bit controls the '-request to send' signal (-RTS) modem control output. When this bit is set to 1, -RTS is active. When this bit is set to 0, -RTS is inactive.
0	**Data-Terminal-Ready:** This bit controls the '-data terminal ready' signal (-DTR) modem control output. When this bit is set to1, -DTR is active. When this bit is set to 0, -DTR is inactive.

6.3.8 Line Status Register (LSR)

This 8-bit read-only register provides the system microprocessor with ongoing information about the data transfer.

Software Implementation Note: Writing to this register can produce unpredictable results.

Table 201. Line Status Register (LSR) 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]	0x05	Read Only	0xC0	N/A

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 (-Rl) 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 (-Rl) 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.

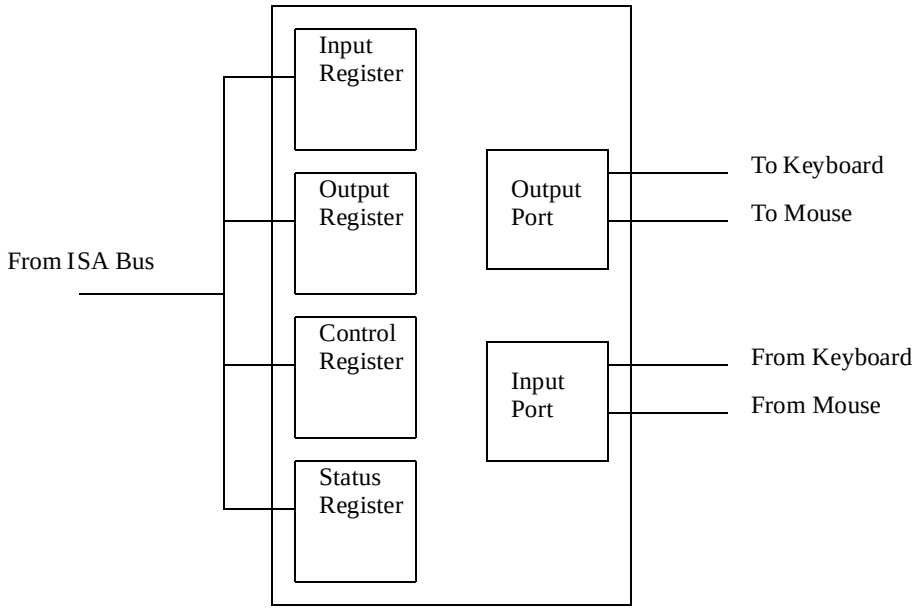


Figure 4. Block Diagram of Keyboard/Mouse Controller Register

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.

8 Audio

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

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 SCSI IDE PC Card	R O O O	R O O O	R O O O	"Medialess" systems are not covered by this architecture.
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		Reserved by 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.

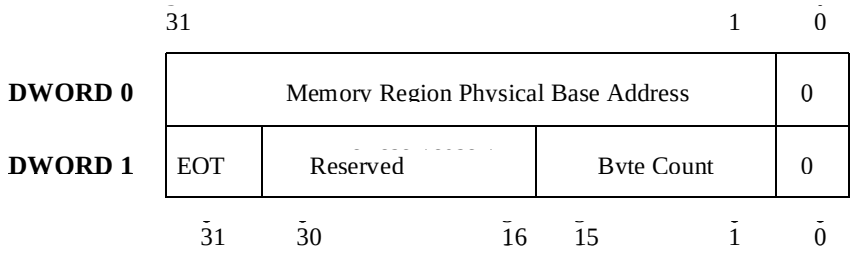


Figure 5. Physical Region Descriptors for Bus master IDE

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.

Table 288. Sequence Register

Bit	Function
7	**DMA**: The DMA bit indicates that the transfer is to be done using DMA. If this bit is enabled, FIFO accesses are disabled.
6	**TMode :** When set, the TMode bit puts the controller into Target mode for executing the command. The TMode bit is cleared for Initiator mode.
5	**Atn Bit:** For the Selection and Information Transfer phases, the Atn bit indicates that the Atn signal shall also be asserted.
4	**ActNeg:** If the ActNeg bit is set, the SCSI controller adopts Active Negation mode. In this mode it actively negates the Req and Ack signals by pulling them high instead of relying on line terminators. Only the Req and Ack signals are affected. If both ActNeg and TMode are set, the controller pulls Req actively both low and high; if TMode is clear, then Ack is actively negated.
3-0	**Seq:** The four Seq bits, seq3..seq0, contain the encoded value of the commands defined in Table 289 on page 151.

Table 289. Seq bit encoding

Command	Value	Normal completion	Exception	Error
Arb	0x1	Arbitration won (2.4µs)	ArbLost, SCSIReset, Reselect	
Sel	0x2	Destination selected	SelTO, SCSIReset, UnExpDisc	
Cmd	0x3	Transfer count exhausted	PhaseMM, SCSIReset, UnExp-Disc	
Status	0x4	Transfer count exhausted	PhaseMM, SCSIReset, UnExp-Disc	ParErr
DataOut	0x5	Transfer count exhausted	PhaseMM, SCSIReset, UnExp-Disc	
DataIn	0x6	Transfer count exhausted	PhaseMM, SCSIReset, UnExp-Disc	ParErr
MsgOut	0x7	Transfer count exhausted	PhaseMM, SCSIReset	
MsgIn	0x8	Transfer count exhausted	PhaseMM, SCSIReset	ParErr
BusFree	0x9	Bsy signal cleared	PhaseMM, SCSIReset	
EnParChk	0xA	3 clock cycles after write	No interrupts	
DisParChk	0xB	3 clock cycles after write	No interrupts	
EnReSel	0xC	3 clock cycles after write	No interrupts	
DisReSel	0xD	3 clock cycles after write	No interrupts	
RstMESH	0xE	3 clock cycles after write	No interrupts	
FlushFIFO	0xF	3 clock cycles after write	No interrupts	

These commands are described in Table 290 on page 151. SCSI output signals are desrcibed in *ANSI X3.131-199x Rev.10L,1992 (SCSI-2): Information Technology - Small Computer System Interface* [13].

Table 290. MESH SCSI Commands

Arbitrate Command (‘1’X)

Table 290. MESH SCSI Commands ***(Continued)***

Software can use the value in the SourceId register to perform SCSI bus arbitration. The Arbitrate command makes sure that the Bsy and Sel signals have been false for 1200 ns before asserting its encoded SCSI ID along with the Bsy signal. After asserting the Bsy signal, the controller shall wait for 2400 ns and then compare the IDs on the bus. If SourceID is the highest, the controller asserts the Sel signal and wins arbitration. If some other device has a higher ID or has asserted the Sel signal, then the controller has lost arbitration and shall release its Bsy signal and arbitration ID. At this point the ArbLost bit shall be set in the Exception register and an interrupt shall be generated. It is recommended that software load the DestinationID register with the initiator ID before commencing arbitration.
Selection Command
In response to the Selection command, the controller shall assert the ID indicated in the DestID register (along with SourceID) and then deassert the Bsy signal. The controller shall wait up to the time indicated in the SelTimeOut register (nominally 250 ms) for the Bsy signal to be asserted by the destination device. The Selection command assumes that the immediately prior command was the Arbitrate command and that the arbitration was won. If the Atn bit is set the Atn signal shall be asserted prior to releasing the Bsy signal indicating Selection phase. If the TMode bit is set, the IO bit shall be set and the reselection phase shall be entered
Information Transfer Commands
The Cmd, Status, DataIn, DataOut, MsgIn, and MsgOut commands execute until either the Transfer Count is exhausted, a phase mismatch occurs, or an error occurs. In the Initiator mode of any information transfer command, the Ack signal is left asserted until a subsequent command implies that it must be negated. This behavior is helpful in transitioning to the MessageOut phase or generally responding to messaging or data parity errors. An Information Transfer command can be overwritten at any time. If the TMode bit is set, the MESH controller shall set the Msg, CD, and IO signals to appropriate states and begin the Req/Ack action. In target mode, the Atn bit in the Sequence register is used as a phase comparison bit At the end of any Async data tranfer, when the transfer counter counts down to zero, the Ack signal shall remain low. This is for a variety of reasons—the host may choose to force a disconnect or there may have been a parity error. The Atn signal may need to be asserted prior to the trailing edge of Ack to go into MessageOut phase and to keep the Target from going into Synchronous data transfer before the Sync parameters are set up in the MESH controller. In general, Ack is asserted after the command completes. The Ack signal shall be negated upon issuance of the next command. Synchronous data transfer mode is enabled only during DataIn or DataOut phases. The MESH controller shall stay in synchronous mode between synchronous DataIn or DataOut commands. The Ack signal shall not hang in synchronous mode. For further information about synchronous data transfers, see Section 12.5.14.1, “Synchronous Data Transfers,” on page 157.
BusFree Command
The pseudo-phase BusFree command is used to indicate that the bus is expected to transition to the Bus Free phase, which is defined as the Bsy and Sel signals both false. In Initiator mode this implies that the Ack signal be negated first and then Bsy inspected to detect when it goes false. If the MESH controller detects Req asserted before Bsy goes false, then the PhaseMM bit shall be set. If the TMode bit is set, the MESH controller shall release the Bsy signal from the bus.
RstMESH Command
The RstMESH command is used to reset the MESH chip through software. Interrupts are masked by this command, so no interrupts are generated. The CmdDone signal shall be active until cleared through the Interrupt register. Note: This command does not cause a SCSI bus reset; this can be done only by direct manipulation of the Status1 register.
EnParChk Command
The EnParChk command enables parity checking on data transfers. Parity is always checked on reselection phases and generated on data transfer phases. No interrupts are generated by this command.
DisParChk Command
The DisParChk command disables parity checking on data transfers. Parity is always checked on reselection phases and generated on data transfer phases. No interrupts are generated by this command.
EnReSel Command
If the TMode bit is false, the EnReSel command enables reselection as a host. It is active until a SCSI bus reset occurs, the DisResel command is issued, or the controller is reselected as a Host. No interrupts are generated from issuing this command, but the Reselected Exception bit is set if the MESH controller is reselected as a host. If the TMode bit is set when this command is issued, it enables the MESH controller to be selected as a Target.
DisReSel Command

Table 290. MESH SCSI Commands ***(Continued)***

The DisReSel command disables selection or reselection as a host or target. Like the EnReSel command, depending on the state of TMode, it disables either reselection as a Host or selection as a Target. No interrupts are generated from this command.

12.5.5 Bus Status Register 0 (Status0)

Table 291. Bus Status Register 0 (Status0) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x04	R/W*	0x00	0x00

* During “read”, the resulting data state corresponds to an OR of the register bit with the SCSI signal state.

The Bus Status registers represent the state of the SCSI bus when read, and can be used to assert signals when written. Normally, these registers are only read to obtain the state of the bus. These registers can be written only when the Exception register is clear. In order to clear a register bit and its corresponding signal, a zero must be written to the status register. The meanings of the bits in Bus Status0 and Bus Status1 are given in Table 292 on page 153.

Table 292. Bus Status Register

Bit	Status0 Function:
7	Req32
6	Ack32
5	Req
4	Ack
3	Atn
2	Msg
1	CD
0	IO
	Status1 Function:
7	Rst
6	Bsy
5	Sel
4-0	reserved

The Req32 and Ack32 signals are reserved for potential use in systems that support 32-bit-wide SCSI.

Writing these registers directly may cause illegal bus phases and lost or corrupted data.

12.5.6 Bus Status Register 1 (Status1)

Table 293. Bus Status Register 1(Status1) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x05	R/W	0x00	0x00

See “Bus Status Register 0 (Status0)” on page 153.

12.5.7 Bus FIFO Count Register (FifoCount)

Table 294. Bus FIFO Count Register (FifoCount) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x06	R/W	0x00	0x00

The lower 5 bits of the FIFO Count register (4:0) are continually updated to represent the number of bytes in the FIFO. This register is synchronized with the processor interface so that it does not have to be protected from bounce.

Except for the case of synchronous data in (see section 11.4.1), for a data transfer command to be considered done, both the XferCount register and the FIFOCount must be 0. For example, if 6 bytes of CDB (command description block) are put into the FIFO and the transfer count register is not 0, the MESH controller shall expect to get XferCount more bytes before finishing. In this case, the XferCount register shall be set to zero before issuing the Cmd sequence command.

12.5.8 Exception Register (Exception)

Table 295. Exception Register 0High (Exception) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x07	R/W	0x00	0x00

The Exception register contains information about conditions which are not usually errors, but can cause the sequence to stop for processor intervention. It is cleared by writing the register with ones in the bits to be cleared or by writing a one to the exception bit in the Interrrupt register.. The meanings of the bits in the Exception register are given in Table 296 on page 154.The bits in the Exception register are discussed in the next sections.

Table 296. Exception Register

Bit	Function
7-6	Reserved
5	**SelWAtn:** The SelWAtn bit is set if the MESH controller was selected as a Target and the Atn signal on the SCSI bus was asserted.
4	**Selected:** The Selected bit is set if the the controller is selected as a Target and the Atn signal was not asserted at the time of the selection. When this bit becomes true, the SCSI bus IDs of the target and host are in the last byte of the FIFO.
3	**Reselected:** The Reselected bit is set if the the controller is reselected as a Host. When this bit becomes true, the SCSI bus IDs of the target and host are in the last byte of the FIFO.
2	**ArbLost:** The ArbLost bit indicates that arbitration was lost when executing the Arb command. This exception may represent one or more lost arbitrations, depending on the hardware implementation.
1	**PhaseMM:** The Phase Mismatch (PhaseMM) bit shall be set when the pending phase command expects a certain phase but another phase is driven by the Target. When this bit is set, the software must interrogate the bus Status0 register to determine the current phase of the bus and take action appropriately.
0	**SelTO:** The SelTO bit indicates that the time indicated by the SelectionTO register was exhausted prior to the Destination device being selected.

12.5.9 Error Register (Error)

Table 297. Error Register (Error) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x08	R/W	0x00	0x00

The Error register contains bits that indicate true error conditions. This register is cleared by writing ones to the bits to be cleared or by writing a one to the Error bit in the Interrupt register. The meanings of the bits in the Exception register are given in Table 298 on page 155.

Table 298. Error Register

Bit	Function
7	Reserved
6	**UnExpDisc:** The UnExpDisc bit indicates that the target released the Bsy signal during the time interval between successful selection and the issuing of the BusFree command.
5	**SCSIRs:** The SCSIRst bit indicates that the Rst signal was (or is) asserted. After receiving this interrupt, the SCSI bus Rst signal (bit 7 of Bus Status1) must be polled until the Rst signal is deasserted.
4	**SeqErr:** The Sequence Error bit indicates that a command was issued to the MESH controller while an Exception or error interrupt was pending.
3-0	**ParityErr:** The ParityError0 bit is set if the calculated parity of the incoming data does not match the Parity0 bit on the SCSI bus. The ParityErr3..ParityError1 bits are reserved for future use with up to 32-bit SCSI implementations.

The bits in the Error register are discussed in the next sections.

12.5.10 Interrupt Mask Register

Table 299. Interrupt Mask Register (InterruptMask) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x09	R/W	0x00	0x00

The Interrupt Mask register is used to mask out interrupts from any or all of the interrupt sources. The value of the register is all zeros upon reset. A value of 0 in a particular location masks (disables) the corresponding interrupt source. A value of 1 enables interrupts from the source. The meanings of the bits in the Interrupt mask register are given in Table 300 on page 155. A set condition in the mask corresponds to “masking’ the interrupt.

Table 300. Interrupt Mask Register)

Bit	Function
7-3	Reserved
2	ErrMask
1	ExcMask
0	CDoneMask

12.5.11 Interrupt Register

Table 301. Interrupt Register (Interrupt) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0A	R/W	0x00	0x00

The Interrupt register combines the Error, Exception, and CmdDone status bits into one interrupt source. The meanings of the bits in the Interrupt register are given in Table 302 on page 156.

Table 302. Interrupt Register

Bit	Function
7-3	Reserved
2	**Error:** The Error bit represents an OR-combination of all the bits in the Error register. It can be cleared by writing ones to the appropriate bits in the Error register or by writing 1 to this bit in the Interrupt register.
1	**Exception:** The Exception bit represents an OR-combination of all the bits in the Exception register. It can be cleared by writing ones to the appropriate bits in the Exception register or by writing 1 to this bit in the Interrupt register.
0	**CmdDone:** The CmdDone bit indicates that the last command issued is done. This bit is automatically cleared by issuing another command to the Command register. It can also be cleared by writing 1 to this bit in the Interrupt register.After a reset action, the CmdDone signal shall be asserted. Because the Interrupt Mask register is cleared on reset, no interrupts shall be generated.

The bits in the Interrupt register are discussed in the next sections.

12.5.12 SourceID Register

Table 303. SourceID Register (SourceID) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0B	R/W	0x00	0x00

The SourceID register contains the SCSI bus ID that is used for arbitration. It is also the register that the MESH controller shall respond to during selection phase (in Target mode) or reselection phase (in Initiator mode).

12.5.13 DestinationID Register

Table 304. DestinationID Register (DestinationID) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0C	R/W	0x00	0x00

The DestinationID register contains the SCSI bus ID of the device to be selected or reselected.

12.5.14 SyncParms Register (SyncParms)

Table 305. SyncParms Register (SyncParms) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0D	R/W	0x00	0x00

The SyncParms register contains two fields, as shown in Table 306 on page 157.

Table 306. SyncParms

Bit	Function
7-4	**Sync Offset:** The SyncOffset field contains the depth to which the FIFO is capable of accepting synchronous data without delay. In the the case of MESH, this value may be from 1 to 15. If it is zero (as upon reset) then all tranfers shall be done asynchronously. See the ANSI *X3T9.2 SCSI-3 SCSI Parallel Interface (SPI)* specification for the complete definition of Synchronous Data transfer mode and the corresponding Synchronous Data Transfer Request (SDTR) and related negotiations that must be done before Synchronous Data mode is enabled. When the Synchronous Offset value in the SyncParms register is nonzero, the value in the SyncPeriod field is used to specify the period of the Ack signal output. When the Synchronous Offset value is 0, Asynchronous transfer mode is specified. In addition, the value in the SyncPeriod field specifies the minimum number of clock cycles between the negation of Ack and the time when the Req signal should be evaluated. Because of the dual-ranked synchronizers in hardware, the actual time may be greater by as much as 1 clock cycle. For example, if the SCSI bus is known to settle in 80 ns, a value of 4 in the SyncPeriod field shall guarantee that Req or Ack is not evaluated in the state machine for at least 80 ns.
3-0	**Sync Period:** The SyncPeriod field contains the period of the Req or Ack pulse when the MESH controller is in Synchronous Data transfer mode. The value must be adjusted to match the SyncPeriod time agreement in the SDTR messages. Technically, the ANSI SCSI specification allows resolution down to 4 ns, but the driving software must negotiate a period that is acceptable to both the controller and the Target device. A value of 0 indicates 10 MB per second and 3 indicates 5 MB per second. The equation for determining the period of the synchronous Ack signal is (4*clk + 2*clk*SyncPer) where *clk* represents one MESH Clk period and SyncPer is the value in the lower nybble of the SyncParms register. Fast synchronous operation is a special case, where 0 means 100 ns. For Asynchronous transfer modes, the SyncPeriod field must be set to a value of 2 or greater. This field is used to mask the trailing edge of the Req signal so that the internal logic does not incorrectly respond to SCSI bus noise created by cable reflections and other impedance mismatch side-effects. The SyncPeriod value represents the number of clocks delay after the trailing edge of Req before evaluating the Req signal again. The minimum value in the SyncPeriod field is 2 for asynchronous transfers. Anything less results in shorter setup times for data relative to Ack on writes. Achieving the fastest Asynchronous transfer rate with the MESH controller requires a SyncPeriod value of 2

12.5.14.1 Synchronous Data Transfers

Observe these cautions when performing synchronous data transfers:

- The MESH controller is not able to handle unsolicited synchronous data. Before the issuance of a command, the SyncParms register must be set to match the expected transfer mode.
- The count for the Command Descriptor Block may not be arbitrarily set to a value higher than the number of bytes sent. The CmdDone bit will get set with a corresponding Exception indicating a phase mismatch (the Target went from CommandPhase to Data phase unexpectedly). If the Target then begins to transfer data in synchronous mode, the MESH controller will not be able to handle the Req signals or any corresponding incoming data.
- Target mode does not support synchronous data transfers, regardless of the SyncParms value.

12.5.15 MESH ID Register (MeshID)

Table 307. Mesh ID Register (MeshID) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0E	R/W	0x00	0x00

The MESH ID register is used to indicate the version of the MESH controller and also to differentiate MESH from other controllers. This register is read-only and contains two sufields. Bits 7-5 are reserved for definition of vendor ID and should be ignored by software. Bits4-0 correspond to funtionality version ID and are used by software to determine the features and bug-fix level.

12.5.16 Selection TimeOut Register (SelTO)

Table 308. Selection TimeOut Register (SelTO) Characteristics

Register Size	Register Addressing	Associated Reg Property	Offset Address	Read/Wrt Characteristics	Hardware Reset Value	Software Reset Value
1 byte	Direct	reg[1]	0x0F	R/W	0x00	0x00

The Selection TimeOut register is used for selection and reselection timeouts with its 8-bit value representing the amount of time before the selection timeout is reached. Each count represents 10 ms when the controller is running at 50 MHz.

12.6 DBDMA Registers

This section briefly mentions the DBDMA registers for the MESH SCSI controller which are shown in Section 12.3, “Mesh SCSI Open Firmware Properties,” on page 148 and documented in Chapter 15, “Descriptor-Based DMA,” on page 171. The registers are the portion of the MESH SCSI Controller used for controlling bus master data transfers to/from system memory and are named:

1. DBDMA Transmit Channel Control
2. DBDMA Transmit Channel Status
3. DBDMA Interrupt Select
4. DBDMA Branch Select
5. DBDMA Wait Select

Requirements:

12–6. For the device described herein, all the registers and their associated function defined in Appendix A must be implemented precisely as described.

12.7 References

1. *ANSI X3.131-199x Rev.10L,1992 (SCSI-2): Information Technology - Small Computer System Interface* [13]

2. *Macintosh Technology in the Common Hardware Reference Platform* [18] (Use this document for specific physical, timing, and electrical requirements)
3. *Mac IO* [14] CHRP Open Firmware Bindings

Graphics

13

13.1 CHRP Requirements

Table 309 on page 161 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements of a graphics subsystem on CHRP platforms.

Table 309. Summary of Minimum Platform Requirements (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
Graphics		R	R	O	Servers do not require graphics during normal operation and need not support a graphics subsystem.
	1024x768	O	R	O	Portables may provide screen resolution in accordance with current state-of-the-art LCD technology.
	Bi-Endian	R	R	O	
	640x480x8 LFB	R	R	O	
	VGA	O	O	O	Some operating environments will prefer platforms that provide hardware support of VGA for performance considerations.

The CHRP Specification requires that graphics subsystems provide both Little Endian and Big Endian interfaces to frame buffers. *Portable* and *Personal* CHRP platforms must minimally provide a 640x480x8bit linear frame buffer (LFB) for use by the operating system (OS) until such time that the OS installs the drivers necessary to fully operate the video subsystem. It is recommended that systems provide Video Graphics Array (VGA) capability; certain emulation and DOS environments require hardware VGA support for performance reasons.

13.2 Bi-Endian Interface

CHRP platforms support both Big Endian (BE) and Little Endian (LE) operating systems and applications. Requiring software to meet a single interface regardless of its own endianess is impractical due to the pervasiveness of the changes that would be required, not only to the OS, but perhaps even applications. There are also environments that may require concurrent BE/LE access to the graphics controller and its frame buffer. An example of this might be a system with a BE operating system with some LE frame grabber hardware on the bus. The graphics controller must respond correctly to both of these interfaces in a performance sensitive manner.

Requirements:

13–1. Graphics controllers must provide both a Big Endian (BE) and a Little Endian (LE) interface to the frame buffer.

Hardware and Software Implementation Note: An endian-neutral aperture may be provided for the control interface, however, if there are control registers greater than a single byte in size, separate BE and LE interfaces are recommended. The capability to simultaneously access the BE/LE apertures (dual apertures) is recommended, although a single aperture under mode control is acceptable.

13.3 Linear Frame Buffer

A Linear Frame Buffer (LFB) is a linear range of System Memory or Peripheral Memory Space designated to correspond to the two dimensional arrangement of pixels on the display device, where successive memory bytes are assumed to correspond to successive pixels, possibly with undisplayed bytes at the end of each scan line.

The **`fb8`** generic *frame buffer support package* implements the display device low level interfaces for frame buffers with 8 bits per pixel. **`fb8`** provides routines such as blink/erase/reset/invert screen, insert/delete characters/lines, toggle cursor, and draw-logo (see *IEEE Std 1275-1994, IEEE Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices* [9]).

The *8-bit Graphics Extension* adds simple graphics methods to the frame buffer's drivers, including pixel addressable reading and writing of rectangular sections of the frame buffer. Methods are also provided for color-table programming (see *Open Firmware Recommended Practice: 8-bit Graphics Extension* [34]).

The *16-color Text Extension* adds new FCodes that can be used by the **`fb8`** support package for displaying color text. The display support package includes the property named "**`iso6429-1983-colors`**" to indicate the presence of these extensions. The ISO 6429-1983 standard forms the basis of the 16-color model used by this extension. See *Open Firmware Recommended Practice: 16-color Text Extension* [35].

Requirements:

13–2. CHRP platforms must minimally support a 640x480 linear frame buffer with the Open Firmware characteristics and interfaces described in the *CHRP Linear Frame Buffer Device Binding*.

13.4 Video Graphics Array (VGA)

Requirements:

13–3. VGA compatibility is optional; if VGA compatibility is provided, it must adhere to the programming model given in Appendix A, "VGA Programming Model," on page 195.

13–4. If VGA compatibility is provided, it must conform with the characteristics and interfaces described in the *CHRP VGA Display Device Binding*.

13–5. If VGA compatibility is provided, the display node of the Open Firmware device tree must indicate a compatible property of "`pnpPNP,900`". Conversely, if the compatibility property of "`pnpPNP,900`" is indicated in the display node of the Open Firmware device tree, the programming model described in Appendix A, "VGA Programming Model," on page 195 must be present.

13.5 References

1. *IEEE Std 1275-1994, IEEE Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices* [9]

2. *Open Firmware Recommended Practice: 8-bit Graphics Extension* [34]

3. *Open Firmware Recommended Practice: 16-color Text Extension* [35]

4. *CHRP Linear Frame Buffer Display Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [36]

5. *CHRP VGA Display Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [37]

Versatile Interface Adapter (VIA) 14

14.1 CHRP Requirements

Table 310 on page 165 is an excerpt from *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*, showing the minimum requirements for VIA on CHRP platforms.

Table 310. Summary of Minimum Platform Requirements (excerpt)

Subsystem	Specification	Portable	Personal	Server	Description
VIA		R	R	M	

14.2 VIA Open Firmware Requirements

Requirements:

14–1. The Open Firmware device tree node for the device described in this chapter must follow the specifications given in the *Mac I/O* [14] Open Firmware device binding.

14–2. The Open Firmware "`reg`" property information given in Table 311 on page 167 must be provided in the Open Firmware device tree for the VIA device.

14.3 Introduction

14.3.1 Legacy Application

Macintosh computers have used Versatile Interface Adapter (VIA) integrated circuits to provide a variety of control and interface functions. The VIA contains internal control and data registers that can be accessed by software to configure and determine the state of the VIA and the signals that the VIA monitors.

Hardware and Software Implementation Note: VIA hardware will be removed in later versions of this platform specification. It is included here and in initial platforms to support legacy Mac OS code.

Requirements:

14–3. New capabilities in operating systems must not depend in any way on the continued existence of the facilities described in the VIA chapter.

14.3.2 Internal Timings

VIA functions synchronize to an internally generated clock of period = 1.27655 µs; most normal operations will require at least this amount of time to complete. MacOS has used this fact to insert delays of known duration into the execution of timing-critical code. This practice *must not be proliferated* since the VIA function will be removed from this platform definition in the future.

Requirements:

14–4. New software must not utilize the timing characteristics associated with operations on the VIA hardware as described in Section 14.3.2, “Internal Timings,” on page 166 of the VIA chapter.

14.4 VIA Registers

Requirements:

14–5. All the “**`reg`**” property fields specified in Table 311 on page 167, “VIA Registers”, must comply with the format given in the PCI binding to IEEE Std 1275-1994, as modified by the CHRP binding to IEEE Std 1275-1994.

The positions in the “**`reg`**” property (denoted by reg[1], reg[2], etc.) and the value of the size fields are stated in Table 311 on page 167. The format is described in the referenced Open Firmware bindings.

Table 311. VIA Registers

Address Offsets	Registers	Reg Properties
0x0000	Data Register B	reg[1],size=7680
0x0200	Data Register A	
0x0400	Data Direction Register B	
0x0600	Data Direction Register A	
0x0800	Timer T1 counter (low-order byte)	
0x0A00	Timer T1 counter (high-order byte)	
0x0C00	Timer T1 latch (low-order byte)	
0x0E00	Timer T1 latch (high-order byte)	
0x1000	Timer T2 counter (low-order byte)	
0x1200	Timer T2 counter (high-order byte)	
0x1400	Shift Register	
0x1600	Auxiliary Control Register	
0x1800	Peripheral Control Register	
0x1A00	Interrupt Flag Register	
0x1C00	Interrupt Enable Register	
0x1E00	Data Register A	

14.4.1 Data Register B

Table 312. Data Register B 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	base_adr[0]	0x0000	Read/Write	Undefined	N/A

There are two data registers in each VIA, called *Data register A* and *Data register B*, each with its own data direction register. The Data Register bit definitions are system dependent. They are 8-bit bidirectional buses used for transfer of data, control, and status between the VIA and a peripheral device. Table 313 shows the bit assignments for Data Register B.

Table 313. Data Register B Bit Definition

Bit	I/O	Description
PB0-PB7		**Unused:** If programmed as inputs, these bits will read 0.

14.4.2 Data Register A

Table 314. Data Register A 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	base_adr[0]	0x0200	Read/Write	Undefined	N/A

There are two data registers in each VIA, called *Data register A* and *Data register B*, each with its own data direction register. The Data Register bit definitions are system dependent. They are 8-bit bidirectional buses used for transfer of data, control, and status between the VIA and a peripheral device. Table 315 shows the bit assignments for Data Register A.

Table 315. Data Register A Bit Definition

Bit	I/O	Description
PA0-PA2		**Unused:** If programmed as inputs, these bits will read 0.
PA3	Output	**Sync Modem:** Modem clock select: 1: Select the external serial clock to drive the SCC's /RTxCA pin 0: Select the C3.6864MHz clock to drive the SCC cell **Note:** This is but one implementation of SCC clock selection (platform dependent).
PA4		**En WaitReqA:** Lets the WaitReq_L signal from port A of the SCC appear on the PA7 input pin.
PA5		**Unused:** If programmed as inputs, these bits will read 0.
PA6	Output	**En WaitReqB:** Lets the WaitReq_L signal from port B of the SCC appear on the PA7 input pin.
PA7	Input	**SCC WREQ:** Reflects the state of the Wait/Request pins from the SCC.

14.4.3 Data Direction Register B

Table 316. Data Register B 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	base_adr[0]	0x0400	Read/Write	Undefined	N/A

Table 317. Data Register B Bit Definition

Bit	Description
0-7	A bit set as 1 in a data direction register causes the corresponding bit of the data register to be used for output, whereas a 0 causes it to be used for input. The Macintosh Operating System sets up the data direction registers at system start-up or reset.

14.4.4 Data Direction Register A

Table 318. Data Register A 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	base_adr[0]	0x0600	Read/Write	Undefined	N/A

Table 319. Data Register A Bit Definition

Bit	Description
0-7	A bit set as 1 in a data direction register causes the corresponding bit of the data register to be used for output, whereas a 0 causes it to be used for input. The Macintosh Operating System sets up the data direction registers at system start-up or reset.

14.4.5 Event Timers

The VIA event timers are unused in CHRP compliant platforms.

14.4.6 Shift Register

The Shift Register is unused in CHRP compliant platforms.

14.4.7 Auxiliary Control Register

The Auxiliary Control register controls various parameters pertaining to the VIA timers and the VIA Shift register, and is unused in CHRP compliant platforms.

14.4.8 Peripheral Control Register

The VIA Peripheral Control register allows software to set some very low-level parameters for certain VIA signals. ***Do not write the Peripheral Control register.*** The result of a read operation is undefined.

Requirements:

14–6. Software must not write to the Peripheral Control Register in the VIA hardware.

14.4.9 Interrupt Flag Register

The Interrupt Flag Register (IFR) is used to indicate the presence of interrupt conditions generated internally by VIA functions or from external interrupt conditions on the control input pins.

14.4.10 Interrupt Enable Register

The Interrupt Enable Register (IER) is used to enable/disable the interrupts indicated in the Interrupt Flag Register. There is bit in the IER corresponding to each bit in the IFR.

Descriptor-Based DMA 15

15.1 Overview

This chapter defines the hardware and software operation of descriptor-based direct memory access (DBDMA) technology for I/O data transfers.

The DBDMA architecture uses *command descriptors*, which are structured as linked arrays. For the sake of simplicity, the scalability of this model has been limited; for example, there are no mechanisms for sharing DMA devices among processors or for efficiently reporting completion status from large numbers of DMA devices. Another (more complex) model may be needed in platforms containing large numbers of processors or I/O devices. For more information about command descriptors, see Section 15.7, “Commands,” on page 182.

There is a more extensive DBDMA architecture which is documented in the *Macintosh Technology in the Common Hardware Reference Platform* [18]. Only those DBDMA features used by the devices described in this document are documented here.

15.2 DBDMA Characteristics

The DBDMA architecture has the following characteristics:

- **Fixed-size commands.** All command entries are 16 bytes long.
- **Fewest processor interrupts.** No more than one interrupt is needed per I/O operation.
- **Initiator-resident data structures.** All command and status list components are expected to be located in initiator-resident system memory
- **Simple command structure.** Descriptors are organized in a simple linear array. Linking can be accomplished through an optional branch action, either associated with a data transfer command or through a NOP command that specifies a branch.
- **Logically distinct channels.** To avoid dynamic usage conflicts, the following distinct DMA channels are provided for controlling transfers of independent data streams:
  - **Duplex channels.** Different DMA channels are provided for logically distinct data-transfer paths, such as full duplex transmit and receive.
  - **Event channels.** A special DMA channel can be provided for reporting asynchronous events from any controller channel.
- **Conditional waits.** DMA commands can be conditionally suspended, based on the DMA controller’s internal Channel Status bits. Since external access to certain of these state bits is provided, they can be used to temporarily suspend DMA activity until a flow-control conflict is resolved.

- **Embedded branches.** To minimize command latency in channel programs that require branching, the INPUT, OUTPUT, and NOP commands have a conditional branch capability.

15.3 Conventions

The discussion of DBDMA in this specification uses the conventions described in this section.

- The term *descriptor* refers to the DMA command list elements, which are often simply buffer descriptors.
- The term *DMA* indicates that data transfers are performed by a relatively simple state machine (called the *target*) that processes commands generated by a relatively sophisticated processor (called the *initiator*).
- Since the devices defined herein that use DBDMA are PCI devices, only little endian addressing for DBDMA is supported. The little-endian variant of the DBDMA architecture is defined by the ordering of bytes in any quadlet register within the DMA unit or the command/status memory locations it accesses. For such quadlets, the data byte with the smallest address is the least significant byte.
- The formats of storage locations, field values, and DMA-command constants are defined in this appendix. To minimize confusion when these are referenced, a consistent nomenclature is used throughout. This nomenclature is defined in Table 320 on page 172.

Table 320. Name notation examples

Name	Description
CommandPtr	Register or element of a memory-resident data structure **Explanation:** In names of registers and memory locations, the first letter is capitalized. For multiple-component names, like CommandPtr, the first letters of additional run-together words are also capitalized.
commandDependent	Name of a field within a register or memory-resident data structure **Explanation:** In names of fields, the letters within the first word are not capitalized. For multiple-component field names, like commandDependent, the first letters of additional run-together words are also capitalized.
Command.field	Name of a field within a register or memory-resident data structure
CMD_VALUE	Name of a defined constant value **Explanation:** In names of defined constants, all letters within the name are capitalized. For multiple-component field names, like OUTPUT_MORE, underscore characters separate the name components.

Requirements:

15–1. Every DBDMA reserved field must be 0 when written and must be ignored when read. This must be true whether the reserved field is accessed by the DBDMA hardware or by device driver software.

15–2. Some address pointers are constrained to be multiples of 4, 8, or 16 bytes. For example, the NewCommandPtr value is always 16-byte aligned. The least-significant bits of these aligned addresses must be reserved, to ensure deterministic behavior when unaligned address values are used.

15.4 Controller Registers

15.4.1 Register Organization

Controllers using the DBDMA architecture described herein have two distinct sets of registers:

- Channel Registers: These are the DBDMA registers that are described in this appendix.
- Device Registers: These are registers which are device specific and are not described in this appendix. Refer to the specific devices which use DBDMA for definitions of these registers.

Table 321 shows the offset for each of the channel registers. The offset value appears literally in the Command address field when the register is accessed indirectly (Command.key = KEY_REGS). When accessed directly by the host, or by a channel command (Command.key = KEY_SYSTEM) the register's address is the sum of the system-defined channel base address and the register's offset value.

Table 321. Channel registers

Offset	Register	Required /optional	
0x00	ChannelControl	Required	ChannelControl is used to control the activity of the channel.
0x04	ChannelStatus	Required	ChannelStatus is used to observe the activity of the channel.
0x08	Reserved		
0x0C	CommandPtrLo	Required	CommandPtrLo is used to indicate where the channel's command list is located in memory.
0x10	InterruptSelect	Optional: See chapter on applicable device	
0x14	BranchSelect	Optional: See chapter on applicable device	
0x18	WaitSelect	Optional: See chapter on applicable device	

The initial values, read values, and write effects for the channel registers are shown in Table 322 on page 173.

Table 322. DBDMA register summary

Register Name	Initial Value	Read Value	Write Effect
ChannelControl	n.a.	0x00000000	update selected bits
ChannelStatus	0x00000000	ChannelStatus	ignored
CommandPtrLo	undefined	CommandPtrLo	store if idle
InterruptSelect	undefined	InterruptSelect	store
BranchSelect	undefined	BranchSelect	store
WaitSelect	undefined	WaitSelect	store

Detailed descriptions of the full register set are contained in the next sections.

15.4.2 ChannelControl Register

Table 323. ChannelControl 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 bytes	Direct	See chapter on device using DBDMA channels	0x00	Read/Write	Undefined	N/A

The format of the ChannelControl register is shown in Table 324. The ChannelControl register serves as the writing port for bits that are presented in the ChannelStatus register..

Table 324. Channel Control Register

Bit	Function
31 -16	Channel Control Mask: The ChannelControl.mask field selects which of the lower 16 bits are to be modified by a write action. Bits in the lower half of the ChannelControl register are written only if the corresponding bits in ChannelControl.mask are set.
15-0	Channel Control Data: The value in the ChannelControl.data field is conditionally written to the ChannelStatus register, based on the ChannelControl.mask field. Note that certain bits of the ChannelStatus register are not writeable or should be written only to a set state or to a cleared state. See Section 15.4.3 for full details.

15.4.3 ChannelStatus Register

Table 325. ChannelStatus 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 bytes	Direct	See chapter on device using DBDMA channels	0x04	Read/Write	Undefined	N/A

The read-only ChannelStatus register provides access to the channel’s internal status bits. The ChannelControl register bits are also visible through the ChannelStatus register. A write to the ChannelStatus register shall be ignored. The format of the ChannelStatus register is shown in Table 326 on page 174.

Table 326. Channel Status Register

Bit	Function
31 -16	Reserved

Table 326. Channel Status Register ***(Continued)***

Bit	Function
15	ChannelStatus.run: ChannelStatus.run is set to 1 by software to start execution by the channel. This should be done only after the CommandPtr registers are initialized; otherwise, the operation of the channel is undefined. Software can clear this bit to 0 to abort the operation of the channel. When channel operation is aborted data transfers are terminated, status is returned, and an interrupt is generated (if requested in the Command.i field of the command descriptor). Data that is stored temporarily in channel buffers may be lost.
14	ChannelStatus.pause: ChannelStatus.pause is set to 1 by software to suspend command processing. Hardware responds by suspending data transfers and command execution and then clearing the ChannelStatus.active bit. Software must reset ChannelStatus.pause to 0 to allow command processing to resume.
13	ChannelStatus.flush: ChannelStatus.flush can be set to 1 by software, to force a channel that is executing an INPUT_MORE or INPUT_LAST command to update memory with any data that has been received from the device but has not yet been written to memory. When the memory update is complete, hardware updates the xferStatus and resCount fields in the current memory resident channel command, then clears the flush bit. Thus, a partial status update is characterized by 1 in the flush bit of the xferStatus field and a final status update is characterized by 0.
12	ChannelStatus.wake: ChannelStatus.wake is set to 1 by software to cause a channel that has gone idle to wake up, refetch the command pointed to by CommandPtr, and continue processing commands. The channel becomes idle after executing a STOP command. The STOP command does not increment the CommandPtr register.ChannelStatus.wake shall be reset to 0 by hardware immediately after each command fetch.
11	ChannelStatus.dead: ChannelStatus.dead is set to 1 by hardware when the channel halts execution due to a catastrophic event such as a bus or device error. The current command is terminated and hardware attempts to write status back to memory. Further commands are not executed. If a hardwired interrupt signal is implemented, the controller shall generate an unconditional interrupt. When hardware sets the ChannelStatus.dead bit, the ChannelStatus.active bit is simultaneously deasserted. Hardware shall reset ChannelStatus.dead to 0 when the ChannelStatus.run bit is cleared by software.
10	ChannelStatus.active: ChannelStatus.active is set to 1 by hardware in response to software setting the ChannelStatus.run bit to 1. ChannelStatus.active is reset to 0 by hardware in response to software clearing the ChannelStatus.run bit or setting the ChannelStatus.pause bit. It is also reset to 0 after a STOP command is executed or when hardware sets the ChannelStatus.dead bit to 1.
9	Reserved
8	ChannelStatus.bt: ChannelStatus.bt is set by hardware at the completion of the NOP, INPUT, and OUTPUT commands to indicate whether a branch has been taken. Branching is governed by the Command.b field in the command descriptor, the ChannelStatus.s7..s0 bits, and the values in the mask and value fields of the BranchSelect register. The presence of this bit in the Command.xferStatus field allows software to follow the actual channel program flow with minimal overhead.
7-0	ChannelStatus.s7-s0: Each DMA channel has up to eight general purpose state bits (ChannelControl.s7 .. ChannelControl.s0) that can be written through the ChannelControl register and read through the ChannelStatus register. In some channel implementations, it may be desirable to provide an additional method for setting and clearing these bits directly through hardwired connections—for example, when errors occur or to indicate the completion of a logical record. These bits can be tested at the completion of each command to determine if certain actions should be taken. These actions may include generation of a hardwired interrupt signal, suspension of further command processing, and branching to a new location for further command fetches.

The run and pause bits are control bits that are set and cleared by software. The flush and wake bits are command bits that are set by software and are cleared by hardware when a given action has been performed. The dead, active, and Bt bits are hardware status bits. The bits s7..s0 can be used for general purpose status and control. Their meaning is channel-specific, and they can be controlled by either hardware or software.

15.4.4 CommandPtrLo Register

Table 327. CommandPtrLo 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 bytes	Direct	See chapter on device using DBDMA channels	0x0C	Read/Write	Undefined	N/A

The and CommandPtrLo register specifies the address of the next command entry to be fetched. Since all channel commands are 16-byte aligned, the four least-significant bits of the CommandPtrLo register must always be written with zeros. If they are written with a nonzero value, the operation of the channel is indeterminate and the value returned when these bits are read is undefined. Table 328 on page 176shows the format of the CommandPtr register.

Table 328. CommandPtrLo

Bit	Function
31 -0	Address Aligned on a 16 byte boundary

The CommandPtrLo registers can be read at any time, but writes to the CommandPtrLo register are ignored unless the ChannelStatus.run and ChannelStatus.active bits are both 0.

15.4.5 InterruptSelect Register

Table 329. Interrupt Select 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 bytes	Direct	See chapter on device using DBDMA channels	0x10	Read/Write	Undefined	N/A

The InterruptSelect register is used to generate the interrupt condition bit that is tested at the completion of each command to determine if a hardwired interrupt signal should be asserted. Its format is shown in Table 330 on page 176.

Table 330. Interrupt Select Register

Bit	Function
31 -24	Reserved. Undefined on reads.
23-16	Mask
15-8	Reserved. Undefined on reads.

Table 330. Interrupt Select Register ***(Continued)***

Bit	Function
7-0	Value

The 8-bit mask and value fields affect the generation of interrupts when command descriptors are processed.

An interrupt condition signal, which can be tested by each channel command, is generated according to the following equation:

```
c = (ChannelStatus.s7..s0 & InterruptSelect.mask)
      == (InterruptSelect.value & InterruptSelect.mask)
```

15.4.6 BranchSelect Register

Table 331. BranchSelect 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 bytes	Direct	See chapter on device using DBDMA channels	0x14	Read/Write	Undefined	N/A

The BranchSelect register is used to generate the branch condition bit that is tested at the completion of each command to determine if a branch should be performed. The BranchSelect register has the same format as the InterruptSelect register.

The branch condition signal is generated according to the following equation:

```
c = (ChannelStatus.s7..s0 & BranchSelect.mask)
      == (BranchSelect.value & BranchSelect.mask)
```

15.4.7 WaitSelect Register

Table 332. WaitSelect 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 bytes	Direct	See chapter on device using DBDMA channels	0x18	Read/Write	Undefined	N/A

The WaitSelect register is used to generate the wait condition bit that is tested at the completion of each command to determine if command execution should be suspended. The WaitSelect register has the same format as the InterruptSelect register.

The wait condition signal is generated according to the following equation:

```
c = (ChannelStatus.s7..s0 & WaitSelect.mask)
      == (WaitSelect.value & WaitSelect.mask)
```

15.5 Summary of DBDMA Operations

DBDMA operations are expected to involve an initiator, which is the component that generates commands. In a typical operation, the initiator generates a new sequence of commands and appends them to a preexisting command list in memory. A wakeup signal is then sent to the target, to initiate the target's processing of the recently appended commands. This process is illustrated in the Figure 6 on page 178.

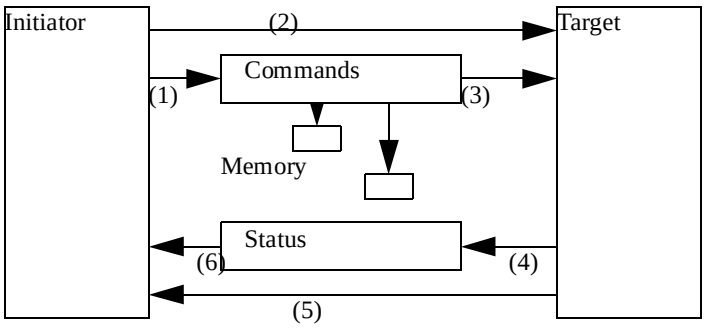


Figure 6. DBDMA Operation

A target is a component that connects to one or more I/O devices such as a disk controller or network interface. When the target is activated, it reads and executes the new commands. As command processing continues, the target returns status information to prespecified locations and sends a signal to the initiator, indicating that another descriptor has been processed.

There are no provisions for having multiple initiators concurrently append additional command entries; multiple initiators are expected to serialize their command list updates through other shared-memory semaphores that are beyond the scope of this specification. The unsharable nature of the command list implies that another architecture may be appropriate when the DMA controller is shared by loosely-coupled multiprocessors.

Costs are reduced by eliminating the needs to support status-queue structures. Status information is simply returned to prespecified locations within command descriptors, which are checked by the processor after an interrupt is received. The mandatory polling of multiple status locations implies that another architecture may be appropriate when large numbers of DMA controllers are shared by one processor.

15.5.1 Multiplexed Channels

The DBDMA architecture assumes that each device is connected to its own data-transfer channel. I/O driver software is thus freed from the responsibility of dynamically assigning DMA channels to I/O devices. Two data-transfer

channels would typically be assigned to a full-duplex device, such as Ethernet; one is sufficient for a half-duplex device such as a disk controller.

In most cases, the transfer of control information (such as a seek address) is performed before the data transfer is initiated. Similarly, status information (such as short-transfer counts) is returned after the data transfer finishes. In both cases, the control, data transfers, and control transfers are performed sequentially, so the same DMA channel resources can be shared.

Other forms of status information (such as a disk pack removal) are unexpected and cannot be associated with a data transfer channel. A special event channel may be used to report such asynchronous events.

Since the event channel is rarely used, it may be shared by all devices attached to a DBDMA controller. Hence the controller may contain one shared event channel and one or more device-connected data transfer channels, as shown in Figure 7 on page 179.

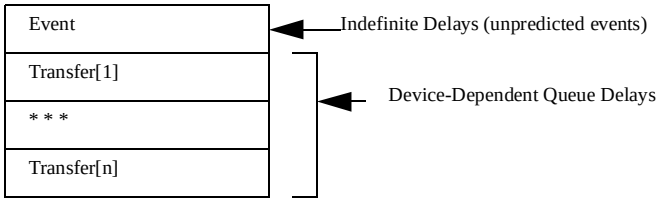


Figure 7. Multiplexed channel types

The commands in the event list are expected to be Read commands that (on demand) return unexpected event status. In this case, the returned list-index value *n* is used by I/O driver software to identify the affected transfer list, transfer[n]. Only one or two Read commands are expected to be queued, since a queued event command is not preallocated to any list but can be used to return event status from any of the affiliated lists. Note that the commands may remain in the event list for an indefinite period of time, depending on the arrival rate of unexpected events.

The commands in the transfer[1] through transfer[n] lists are expected to be used individually (to support half-duplex traffic) or in pairs (to support full-duplex traffic). The commands in these lists are typically transient, in that they are constantly consumed as I/O operations are performed. However, some I/O operations (such as terminal read actions or flow-controlled write actions) may have an indefinite lifetime.

The event channel is optional; it need not be implemented when the interrupt mechanism and device status registers are sufficient to report device-dependent asynchronous events.

15.5.2 Command-List Structure

The command list space is allocated by the initiator, which fills a set of command-list entries before passing ownership of them to the target. The target fetches command entries and performs the command-entry-specified data-transfer operation, processing command entries until a STOP entry is reached. The structure of a typical command list is shown in Figure 8.

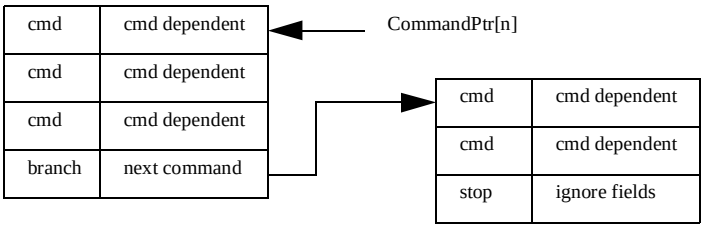


Figure 8. Command list structure

The command list may be structured in several different ways: to loop on itself (a circular queue), to link individual data-transfer operations (linked lists), or to link groups of data-transfer operations (a hybrid approach, as shown in Figure 8).

Note that branches may be embedded in most commands. Therefore, the branch shown in the example could be collapsed into the previous command or it could be encoded as a NOP command that specified a branch. The detailed structure of command lists is dependent on device driver software conventions and is not specified by the DBDMA architecture.

In the idle state, the target has a CommandPtr[n] value for each of the *n* multiplexed channels that are supported. When activated on channel *n*, a command entry is fetched from the physical address specified by CommandPtr[n]. Command fetching and execution normally continue until a STOP command is executed.

The autonomous fetching and execution of multiple commands provides a mechanism to execute different data-transfer operations (command chaining), provide several memory addresses for a single data-transfer operation (scattered memory addressing), or provide several target addresses for a single data-transfer operation (scattered device addressing).

During the execution of a command entry, the CommandPtr[n] value is either incremented by 16, replaced by the command-entry's branchAddress value, or left unchanged (for the STOP command value).

Additional commands may be dynamically appended to an existing command descriptor list while the previously initiated DMA operations are in progress. Command-list appending is performed by creating an additional command list and overwriting the STOP command at the end of the old command list with a NOP or a branch to the first command of the new list.

The WAIT field can be used to provide flow control within a channel. A channel can be programmed to wait until signaled by the processor or by another device.

15.6 Design Model

This section presents the DBDMA design model, which illustrates how the DBDMA components are structured

15.6.1 Controller Components

A DBDMA controller is expected to connect multiple devices indirectly to a system bus that supports read and write transactions to system memory. Processor interrupts are either generated through special signal lines (for motherboard designs) or by sending wakeup events (a 32-bit write to a prespecified register address).

The DBDMA controller may have shared states, as well as channel states that are replicated for each of several connected devices, as shown in Figure 9 on page 181.

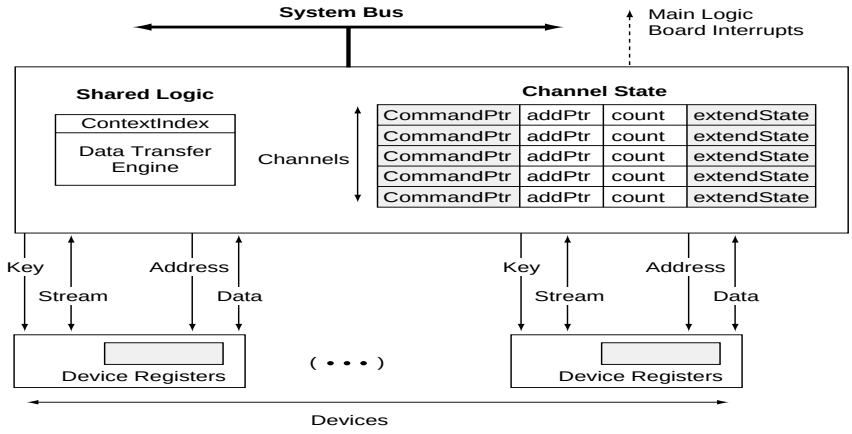


Figure 9. DBDMA controller model

Hardware is expected to use some form of ContextIndex state to identify which of the DMA contexts is currently active, but this state is not directly or indirectly accessible to software. The data-transfer engine provides the facilities for performing the requested data transfers.

The DBDMA architecture provides a well-defined interface to attached I/O devices. An I/O device may be connected to a data-byte stream, where the controller handles the movement of unaddressed data bytes into or out of system memory. The controller provides a 3-bit key value. Four of the Command.key encodings are used to identify which byte-stream is being used. The standard data stream is typically STREAM0 and an optional control/status stream is typically STREAM1.

As an alternative option, a DBDMA controller may support the direct movement of data between device registers and memory. In this case, the controller provides a sequence of device register addresses while the data is being transferred. An additional Command.key encoding is used to identify transfers directed to device registers.

Each channel provides a few memory-mapped registers. These registers include the CommandPtr register (used to initialize the sequencer’s pointer to the command list) and the ChannelStatus and ChannelControl registers. Writes to ChannelControl registers can reactivate idle command-list processing. Optional BranchSelect, InterruptSelect, and WaitSelect registers can be used to specify the manner in which the conditional execution of the branch, interrupt, and wait actions is selected.

15.6.2 System Bus Errors

Unrecoverable errors on system memory accesses includes parity and addressing errors. A busy-retry error is treated as a recoverable bus error, and the bus action is retried until it finishes successfully or the DBDMA controller’s command processing is terminated.

An unrecoverable error sets the dead bit in the ChannelStatus register of the currently executing DMA channel. Software must explicitly reset the ChannelStatus.run bit to 0 and later set it back to 1 to return to an operational state.

15.6.3 Device Status

The device's control and status registers are mapped into system memory space, so the processor can access them directly using read and write transactions. For some applications, the processor is required to update these registers at the end of every data transfer, so there is no need for the channel program to return device status explicitly.

However, some devices have the intelligence to process multiple data transfer requests autonomously. For these applications, processor performance would be reduced unless the channel could autonomously return device status before initiating the next data transfer. The DBDMA architecture provides two mechanisms for autonomously returning device status:

- **Status stream.** An additional data-transfer command can be used to transfer a status-byte stream into memory. A unique Command.key value is used to distinguish the status stream from the normal data-transfer stream.
- **Status registers.** An additional data-transfer command can be used to transfer a range of device register addresses into memory. Another unique Command.key value is used to distinguish this memory-mapped transfer from the normal data-transfer stream.

Simple device status information may also be returned in the Command.xferStatus field described in Section Bit.

15.6.4 Conditional Actions

The INPUT, OUTPUT and NOP commands provide Command.b and Command.branchAddress fields that specify the condition and address for optional branches in command sequencing. A similar mechanism is used to specify conditional interrupt generation and conditional suspension of command processing.

15.7 Commands

All the commands listed herein are mandatory and must be implemented as specified.

15.7.1 Command Formats

The general format of a DBDMA command descriptor contains 16 bits of opcode fields (including a 4-bit cmd field), a 16-bit reqCount field, a 32-bit address parameter, and a 32-bit cmdDep parameter, as shown in Table 324 on page 174. The xferStatus and resCount fields are used by the channel to report status after a command finishes. Although the location and size of the cmdDep field is standardized for all commands, its interpretation is dependent on the Command.cmd field value.

Table 333. Command Entry Formatr

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27	Reserved
	26-24	key
	23-22	Reserved
	21-20	i
	19-18	b
	17-16	w
	15-0	ReqCount
Word 2	31-0	Address
Word 1	31-0	CmdDep
Word 0	31-16	XferStatus
	15-0	ResCount

Bit 31-28, Word 3 (Command.cmd): The Command.cmd field specifies which type of data transfer is performed, as defined in Table 334 and described in the following sections.

Table 334. Command.cmd field values

Value	Name	Description
0	OUTPUT_MORE	Transfer more memory to stream
1	OUTPUT_LAST	Transfer last memory to stream
2	INPUT_MORE	Transfer more stream to memory
3	INPUT_LAST	Transfer last stream to memory
4	STORE_QUAD	Store immediate 4-byte value
5	LOAD_QUAD	Load immediate 4-byte value
6	NOP	No data transfer
7	STOP	Suspend command processing
8–15		Reserved

Bit 26-24, Word 3 (Command.key): The Command.key field is used to select which of the device access ports is used, as specified in Table 335 on page 183.

Table 335. Effects of Command.key field

Value	Name	Description	Usage
0	KEY_STREAM0	Default device stream	Data
1	KEY_STREAM1	Device-dependent stream	Control and status
2	KEY_STREAM2	Device-dependent stream	
3	KEY_STREAM3	Device-dependent stream	
4		Reserved	
5	KEY_REGS	Channel-state register space	
6	KEY_SYSTEM	System memory-mapped space	

Table 335. Effects of Command.key field ***(Continued)***

Value	Name	Description	Usage
7	KEY_DEVICE	Device memory-mapped space	

With INPUT and OUTPUT commands, the key field may specify alternate device streams (e.g. data and status). With LOAD_QUAD and STORE_QUAD commands, the key field specifies to which address space the immediate data is to be transferred. Table 336 on page 184 summarizes the operation of the key field in combination with each of the data transfer commands. A key field value of KEYSTREAM0 through KEYSTREAM3 combined with a LOAD_QUAD or STORE_QUAD command is illegal and generates undefined results.

Table 336. Data transfer operations

Command	Key	Source	Destination
OUTPUT	KEY_STREAM0..3	SysMem(address)	Stream 0..3
	KEY_SYSTEM	SysMem(address)	SysMem(Data2Ptr)
	KEY_DEVICE	SysMem(address)	Device(Data2Ptr)
	KEY_REGS	SysMem(address)	ChannelRegs(Data2Ptr)
INPUT	KEY_STREAM0..3	Stream 0..3	SysMem(address)
	KEY_SYSTEM	SysMem(Data2Ptr)	SysMem(address)
	KEY_DEVICE	Device(Data2Ptr)	SysMem(address)
	KEY_REGS	ChannelRegs(Data2Ptr)	SysMem(address)
STORE_QUAD	KEY_STREAM0..3	Illegal; see note below	See note below
	KEY_SYSTEM	data32	SysMem(address)
	KEY_DEVICE	data32	Device(address)
	KEY_REGS	data32	ChannelRegs(address)
LOAD_QUAD	KEY_STREAM0..3	Illegal; see note below	See note below
	KEY_SYSTEM	SysMem(address)	data32
	KEY_DEVICE	Device(address)	data32
	KEY_REGS	ChannelRegs(address)	data32

Bit 21-20, Word 3 (Command.i): Upon completion of each command, a hardwired interrupt may optionally be generated. Interrupt generation is controlled by the 2-bit Command.i field in conjunction with an internal interrupt condition bit that is generated by the channel. The channel generates the interrupt condition based on the current values of the ChannelStatus.s7..s0 bits along with mask and data values found in the InterruptSelect register of each channel. The suggested algorithm for generating the interrupt condition is given in Table 337 on page 184.

Table 337. Algorithm for Generating Interrupt

```
c = (ChannelStatus.s7..s0 & InterruptSelect.mask)
== (InterruptSelect.value & InterruptSelect.mask)
```

Bit In many implementations there will be no need to allow all of the general purpose status bits to generate interrupts. In these cases, the generation of the interrupt condition may be as simple as tying it to a single status bit or tying it to 0. The equation above should be used only in those cases where it is desired to allow interrupts to be generated based on the values of multiple status bits. Table 338 on page 185 indi-

cates how the interrupt condition bit is used in conjunction with the Command.i field to determine whether or not to generate an interrupt.

Table 338. Effects of Command.i field

Value	Effect
0	Never interrupt
1	Interrupt if the Interrupt Condition bit is set
2	Interrupt if the Interrupt Condition bit is cleared
3	Always interrupt

In implementations that have no hard-wired interrupt signal (such as a DBDMA controller used with certain I/O expansion buses), the Command.i field value is ignored.

Bit 19-18, Word 3 (Command.b): Upon completion of each data transfer command, the CommandPtr is updated to point to either the next command or the target of a branch. Branching is controlled by the Command.b field in conjunction with an internal branch condition bit that is generated by the channel. The channel generates the branch condition based on the current values of the ChannelStatus.s7..s0 bits along with a mask and a data value found in the BranchSelect register of each channel. The algorithm for generating the branch condition is given in Table 339 on page 185.

Table 339. Algoritm for Generating Branch Condition

```
c = (ChannelStatus.s7..s0 & BranchSelect.mask)
== (BranchSelect.value & BranchSelect.mask)
```

Table 340 on page 185 indicates how the branch condition bit is used in conjunction with the Command.b field to determine whether or not to take the branch.

Table 340. Effects of Command.b field

Value	Effect
0	Never branch
1	Branch if the Branch Condition bit is set
2	Branch if the Branch Condition bit is cleared
3	Always branch

17-16, Word 3, (Command.w): Upon completion of each command, channel operation may optionally be suspended. This action is controlled by the Command.w field in conjunction with an internal wait condition bit that is generated by the channel interface. The channel interface generates the wait condition based on the current values of the ChannelStatus.s7..s0 bits along with mask and data values found in the WaitSelect register of each channel. The suggested algorithm for generating the wait condition is given in Table 341 on page 185.

Table 341. Algorithm for Wait Condition

```
c = (ChannelStatus.s7..s0 &
WaitSelect.mask)
      == (WaitSelect.value &
WaitSelect.mask)
```

Table 340 on page 185 indicates how the wait condition bit is used in conjunction with the Command.w field to determine whether or not to suspend command execution. If a wait action is invoked, it occurs before the normal command completion sequence (branch determination, status update, interrupt generation, or next command fetch).

Table 342. Effects of Command.w field

Value	Effect
0	Never wait
1	Branch if the Wait Condition bit is set
2	Branch if the Wait Condition bit is cleared
3	Always wait

Bit 15-0, Word 3 (Command.reqCount): The Command.reqCount field is used to specify the number of bytes to be transferred by data transfer commands.

Bit 31-0, Word 2 (Command.address): The Command.address field is used to specify the primary 32-bit address used in data transfers. With INPUT and OUTPUT commands, it is always a system memory address and points to a data buffer. With LOAD_QUAD and STORE_QUAD commands, it is the address to or from which the immediate data is to be moved. In the latter case, it may point to system memory space, device memory space, or channel registers, as controlled by the Command.key field.

Bit 31-0, Word 1 (Command.cmdDep): The meaning and format of the 32-bit Command.cmdDep field is dependent on the Command.cmd values. With STORE_QUAD and LOAD_QUAD commands it is the data value to be written or read. With input and output commands it specifies a 32-bit conditional branch target. See the descriptions of individual commands in the next sections.

Bit 31-16, Word 0 (Command.xferStatus): Upon completion of a command, the current content of the channel's ChannelStatus register is written to the Command.xferStatus field. (The format of the ChannelStatus register is defined in Section 15.4.3, "ChannelStatus Register," on page 174.) When Command.xferStatus is written to memory, the bit corresponding to ChannelStatus.active is always set to 1. This serves as an indication that the corresponding command has been executed, and both the xferStatus and resCount fields have been updated. To make use of this feature, host software should clear the active bit in the Command.xferStatus field to 0 when the command descriptor is initialized.The ChannelControl.flush bit provides a mechanism for software to force buffered data to be written to memory and for status to be updated before the completion of the command. Partial status reports are characterized by the flush bit being set to 1 in the xferStatus field. Some devices may wish to provide intermediate status autonomously, identifying how many data bytes have been transferred, before completing the transfers specified by the command descriptor. This may be accomplished by providing the device with the ability to set the flush bit in the ChannelControl register. Intermediate status reports may be useful in cases such as terminal input, where the host needs to respond quickly to input data but would like to avoid interrupts on every byte. The intermediate status capability is optional. When it is implemented, the conditions under which partial status is returned are device-dependent; for example, status could be returned periodically, when signaled through device-dependent state bits, when a minimum number of data bytes are available, or in response to a combination of these conditions.

Bit 15-0, Word 0 (Command.resCount): Upon completion of a command, the 16-bit residual byte count value is written to the Command.resCount field. This value is normally 0, but may be more when the device prematurely terminates the data transfer.

Note: The Command.resCount and Command.xferStatus fields shall be updated in an indivisible operation.

15.7.2 INPUT and OUTPUT Commands

The INPUT_MORE, INPUT_LAST, OUTPUT_MORE, and OUTPUT_LAST commands transfer data between system memory and the device stream. The OUTPUT_MORE and OUTPUT_LAST commands transfer data from memory (which typically goes into the attached device). The INPUT_MORE and INPUT_LAST commands transfer data (which typically comes from the attached device) into memory. Data chaining is accomplished through the use of the MORE and LAST versions of these commands. The MORE commands indicate that the current buffer is not expected to complete a logical record (such as a network packet). The LAST commands indicate that the buffer is expected to complete a logical record. The reqCount field specifies the number of bytes to be transferred, the address field specifies a starting system memory address, and the branchAddress field specifies the address from which to fetch the next command if the branch test is successful. The Command.key field selects which of the device access ports is used. These fields are shown in Table 343 on page 187.

Table 343. Input and Output Command Formatr

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27	Reserved
	26-24	key
	23-22	Reserved
	21-20	i
	19-18	b
	17-16	w
	15-0	ReqCount
Word 2	31-0	Address
Word 1	31-0	Branch Address
Word 0	31-16	XferStatus
	15-0	ResCount

Note: An OUTPUT or INPUT command that performs a conventional transfer between system memory and an unaddressed device stream should specify a Command.key value of KEY_STREAM0. Additional stream identifiers allow access to other streams that may be provided by the device, such as control and status information. Command.key values of KEY_SYSTEM and KEY_DEVICE are used in conjunction with an (optional) Data2Ptr register to perform two-address transfers between system memory and the specified space.

15.7.3 STORE_QUAD Command

The STORE_QUAD command stores a 32-bit immediate value into memory. The 32-bit data32 field specifies the data value, and the address field (in combination with the key field) specifies the destination address, as shown in Table 344 on page 188.

Table 344. Store_Quad Command Format

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27	Reserved
	26-24	key
	23-22	Reserved
	21-20	i
	19-18	Reserved
	17-16	w
	15-3	Reserved
	2-0	ReqCount
Word 2	31-0	Address
Word 1	31-0	Data32
Word 0	31-16	XferStatus
	15-0	Reserved

The field normally used for specifying whether a branch is to be taken (Command.b) is reserved in this command. Software shall write it with a 0 value. The operation of the channel is undefined when a nonzero value is written to this field.

The only valid values for the reqCount field are 1, 2 and 4. Only aligned transfers are supported. For example, if the count is 4, the low-order two bits of the address are assumed to be 0 and are ignored. Likewise, if the count is 2, the low-order bit of the address is assumed to be 0. When the count is less than 4 bytes, the least significant bytes of data32 are transferred.

STORE_QUAD maps illegal reqCount values into legal ones as shown in Table 345 on page 188. In Table 345 on page 188, *x* represents a bit that is ignored.

Table 345. STORE_QUAD reqCount Values

reqCount	Effective value
0b00*x*	1
0b01*x*	2
0b1*xx*	4

During the execution of a STORE_QUAD command, the Command.key value specifies the destination address for the immediate data. Only key values of KEY_REGS, KEY_SYSTEM, and KEY_DEVICE are allowed. The operation of the channel is undefined when other key values are used.

15.7.4 LOAD_QUAD Command

The LOAD_QUAD command shall load a 32-bit immediate value from memory. The 32-bit data32 field is the destination and the address field (in combination with the key field) specifies the source address, as shown in Table 346 on page 189.

Table 346. Load_Quad Command Format

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27	Reserved
	26-24	key
	23-22	Reserved
	21-20	i
	19-18	Reserved
	17-16	w
	15-3	Reserved
	2-0	ReqCount
Word 2	31-0	Address
Word 1	31-0	Data32
Word 0	31-16	XferStatus
	15-0	Reserved

The field normally used for specifying whether a branch is to be taken (Command.b) is reserved in this command. Software shall write it with a 0 value. The operation of the channel is undefined when a nonzero value is written to this field.

LOAD_QUAD handles the size and alignment of data transfers in the same way that the STORE_QUAD command does.

During the execution of a LOAD_QUAD command, the Command.key value specifies the source address for the immediate data. Only key values of KEY_REGS, KEY_SYSTEM, and KEY_DEVICE are allowed. The operation of the channel is undefined when other key values are used.

15.7.5 NOP Command

The NOP command performs no data transfers. However, it can use standard mechanisms to specify that interrupt, branch, or wait actions be performed. The NOP command is shown in Table 347 on page 189.

Table 347. NOP Command Format

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27-22	Reserved
	21-20	i
	19-18	b
	17-16	w
	15-0	Reserved
Word 2	31-0	Address
Word 1	31-0	Data32
Word 0	31-16	XferStatus
	15-0	Reserved

15.7.6 STOP Command

The STOP command deactivates channel processing. It is placed at the end of a command list to indicate that there are currently no further commands to be processed. The only effect of the STOP command is that the channel goes idle and clears the ChannelStatus.active bit. The format of the STOP command is shown in Table 348 on page 190.

Table 348. Stop Command Format

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -28	cmd
	27-0	Reserved
Word 2	31-0	Reserved
Word 1	31-0	Reserved
Word 0	31-0	Reserved

To add additional commands to an active channel program, the following sequence should be performed:

1. Append the new commands to the existing channel program in memory.
2. Overwrite the STOP command with a NOP command (or the first new channel command).
3. Set the wake bit in the ChannelControl register to 1. This lets the channel process the new commands whether or not it had already completed the old commands and gone idle.

15.8 Asynchronous Event Packet Formats

The DBDMA architecture defines a special channel to allow interfaces to report low frequency asynchronous events. This shared resource alleviates the need for an additional channel to be allocated to each interface. The reporting of modem signal changes (such as DCD or CTS) is one example of a situation in which this facility might be used.

this feature is optional. In the presence of appropriate interrupt mechanisms and device status registers, it may not be needed.

The asynchronous event channel's command list is composed of a series of INPUT_LAST commands with a count of 4 bytes. When an asynchronous event occurs, the current INPUT_LAST command is executed by writing a 4-byte EventPacket to the address indicated by Command.address. This packet consists of a 16-bit eventSource field and a 16-bit eventType field, as shown in Table 349 on page 190.

Table 349. Asynchronous Event Packet Format

Word	Bit	Function
Word 3 (Little Endian Byte Ordering)	31 -0	
Word 2	31-0	
Word 1	31-0	
Word 0	31-0	

The eventSource field identifies the source of the asynchronous event. It should contain a value that identifies the channel affiliated with the event. The eventType field identifies the nature of the event that was generated by the source. The assignment of values within this field is implementation-specific.

15.9 Hardwired Interrupts

Requirements:

15–3. The PCI devices described herein are allowed one interrupt each to the system. DBDMA interrupts must be shared with the other interrupt requirements of the device as described in the device chapters.

15.10 Examples

This section presents examples of implementations of the DBDMA architecture.

15.10.1 Command Queuing

This section describes how the DBDMA architecture might be used to support a circular queue structure for commands —in this example, for Ethernet packet transmission.

Table 350 illustrates a sample channel program. In this example, three commands are required for each packet: one to transmit the header, one to transmit the data, and a final one to receive the transmit status information. An interrupt is generated when the last command of the sequence finishes.

Table 350. Queued transmission commands

No.	Command	key	i	b	w	reqCount	Address	cmdDep
0	OUTPUT_MORE	STREAM0	NEVER	NEVER	NEVER	100	&Hdr1	NA
1	OUTPUT_LAST	STREAM0	NEVER	NEVER	NEVER	1000	&Buf1	NA
2	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status1	NA
3	OUTPUT_MORE	STREAM0	NEVER	NEVER	NEVER	100	&Hdr2	NA
4	OUTPUT_LAST	STREAM0	NEVER	NEVER	NEVER	1000	&Buf2	NA
5	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status2	NA
6	STOP->NOP	NA	NA	NA	NA	NA	NA	NA
7	OUTPUT_MORE	STREAM0	NEVER	NEVER	NEVER	100	&Hdr2	NA
8	OUTPUT_LAST	STREAM0	NEVER	NEVER	NEVER	1000	&Buf2	NA
9	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status2	NA
10	STOP	NA	NA	NA	NA	NA	NA	NA
11	-	-	-	-	-	-	-	-
12	-	-	-	-	-	-	-	-
13	NOP (Branch)	NA	NEVER	ALWAYS	NEVER	NA	NA	&Cmd0

In the example, commands 0 through 5 represent a request to transmit two packets. Initially these commands are followed by a STOP command because there is no further work to be done. When a new request for a packet transmission is received from a client, it can be added to the queue without waiting for completion of the first two packets. This is accomplished by appending the new commands to the list, changing the STOP to a NOP, and setting the wake bit in the ChannelControl register to 1.

Adding commands to the queue is illustrated by commands 6 through 10 in the example. Note that there are several alternatives to the insertion of the NOP command. For example, the STOP command could be overwritten with the new OUTPUT_MORE command after the remaining commands and the new STOP command are written. Software could also replace the stop with a NOP that branches to a new command group, terminated with a STOP command.

Command groups may continue to be added in this manner until the allocated descriptor blocks are exhausted. At that point, a branch back to the first descriptor can be executed, and new command groups can overwrite those which have been executed and released by hardware. (Hardware releases descriptors by writing to the Command.xferStatus field.) The branch can either take the form of a static, standalone command, as illustrated here, or can be embedded in the last data transfer command.

15.10.2 Ethernet Reception

Note: An Ethernet implementation using DBDMA is not documented herein. This example is given to demonstrate DBDMA operation.

This section illustrates one possible way in which the DBDMA architecture can be used to support Ethernet packet reception and mini-buffer allocation. In this example, memory utilization is optimized by allocating buffers that are substantially smaller than the maximum packet size. Thus, in the typical situation where the majority of packets received are quite small, less physical memory must be allocated in order to buffer a given number of packets.

As shown in Table 351, the command list is structured as a number of channel command pairs. After each input buffer has been filled, the channel checks for packet completion. If the packet is complete, a command is executed that fetches the receive status from the device's status stream and stores it in system memory. An interrupt is generated when this command finishes. Note that the first command in the command list is a STORE_QUAD. This initializes the BranchSelect register to the value required for the following conditional branches by specifying that the end of frame bit be tested.

Table 351. Ethernet receive channel commands

No.	Command	key	i	b	w	reqCount	Address	cmdDep
0	STORE_QUAD	SYSTEM	NEVER	NEVER	NEVER	4	&BrSelReg	00400040 <EOF=1>
1	INPUT_MORE	STREAM0	NEVER	!Cond	NEVER	256	&Buffer0	&Cmd3
2	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status0	NA
3	INPUT_MORE	STREAM0	NEVER	!Cond	NEVER	256	&Buffer1	&Cmd5
4	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status1	NA
5	INPUT_MORE	STREAM0	NEVER	!Cond	NEVER	256	&Buffer2	&Cmd7
6	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status2	NA
7	INPUT_MORE	STREAM0	NEVER	!Cond	NEVER	256	&Buffer3	&Cmd2
8	INPUT_LAST	STREAM1	ALWAYS	NEVER	NEVER	4	&Status3	NA
9	STOP	NA	NA	NA	NA	NA	NA	NA
10	NOP (BRANCH)	NA	NEVER	ALWAYS	NEVER	NA	NA	&Cmd1

Only four input buffers are used in this example; in practice many more receive buffers would typically be allocated. The command list is structured as a circular queue of buffers. A STOP command must always be inserted after the last buffer/status command pair. This delineates the buffers that are owned by the controller from the returned buffers that are owned by the host, and prevents the controller from overwriting any filled buffers that have not been released by the host.

As buffers are filled and returned by the controller, new receive buffers are added to the queue in their place, and the position of the STOP command is advanced. This progression of the command list structure is shown in Table 352. Note that this example assumes that the first two received packets each fit within a single mini-buffer and that the third packet requires two mini-buffers.

Table 352. Ethernet receive command sequence

No.	Initial command list	After Buffer 0 is filled	After Buffer 1 is filled	After Buffers 2 and 3 are filled
0	STORE_QUAD	STORE_QUAD	STORE_QUAD	STORE_QUAD
1 2	Buffer0 Status0	STOP NOP	Buffer5 Status5	Buffer5 Status5
3 4	Buffer 1 Status 1	Buffer 1 Status 1	STOP NOP	Buffer 6 Status 6
5 6	Buffer 2 Status 2	Buffer 2 Status 2	Buffer 2 Status 2	Buffer 7 Status 7
7 8	Buffer 3 Status 3	Buffer 3 Status 3	Buffer 3 Status 3	STOP NOP
9 10	STOP NOP (w. Branch)	Buffer 4 Status 4 (w. Branch)	Buffer 4 Status 4 (w. Branch)	Buffer 4 Status 4 (w. Branch)

15.10.3 Full Handshake Flow Control

The DBDMA architecture also allows large device-to-device transfers to be performed autonomously, using small memory-resident buffers for intermediate data storage.

As an example, consider the live capture of the most-recent video images on a disk device, as illustrated in Figure 10. In the most general case, the transient data transfer rates of the two devices may be arbitrarily different—the disk would have a higher transient data transfer rate, but the data transfer rate would drop to 0 during seek operations. For illustration purposes, assume that a small video sample is being saved in a circular buffer on disk.

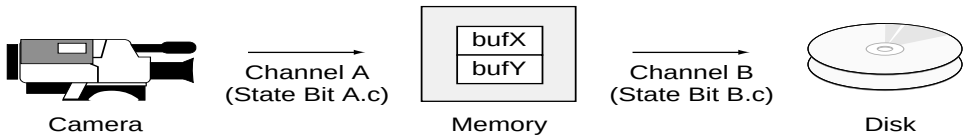


Figure 10. Flow-controlled device model

The DBDMA channels contain state bits that can influence the conditional execution of WAIT actions. Thus, channel programs can use this capability to spin-wait on externally generated events. For the data camera, flow control involves waiting for buffer-use bits to be set by the disk when a data buffer has been read from memory. For the disk, flow control involves waiting for buffer-use bits to be set by the data camera when a new data buffer has been written into memory.

The channelA program uses a STORE_QUAD command to set a channelB state bit (B.s0) after each data buffer has been written; channelA waits for its own state bit (A.s0) to be set by channelB after each data buffer has been consumed. The structure of these two channel programs is illustrated in Figure 11. In the diagram, dotted lines are used to illustrate how the STORE_QUAD command entries in one program are coupled to the WAIT actions in the other.

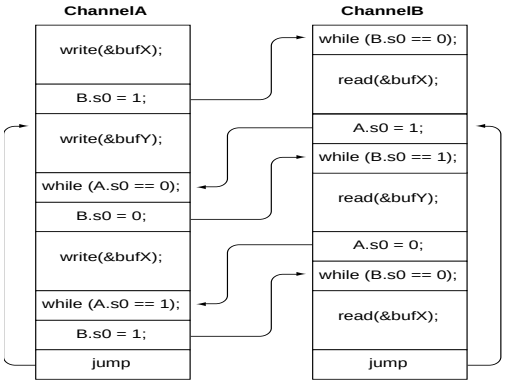
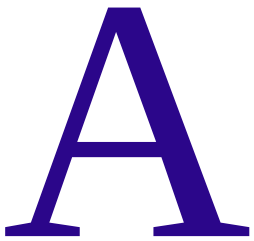


Figure 11. Synchronizing flow-controlled channel programs

VGA Programming Model



A.1 Introduction

IBM introduced the Video Graphics Array (VGA) display standard in 1987 to provide a higher resolution color graphics capability to its PS/2 line of personal computers. VGA BIOS was provided to control the VGA hardware, but programming directly to the hardware has become the more common approach for performance reasons. Since its introduction, VGA has been “cloned” by many manufacturers, claiming register-level compatibility. If a software developer is programming the hardware directly, 100 percent compatibility is imperative; unfortunately, this is not always the case.

The VGA function includes a video buffer, a video digital-to-analog converter (DAC), a CRT controller, a sequencer unit, a graphics controller, and an attribute controller.

Video memory consists of at least 256KB; its use and mapping depend on the mode selected. It is configured as four 64KB memory maps:

The *video DAC* contains the color palette that is used to convert the video data into the video signal that is sent to the display. Three analog signals (red, green, and blue) are output from the DAC.

The *CRT controller* generates horizontal and vertical synchronization signal timings, addressing for the regenerative buffer, cursor and underline timings, and refresh addressing for the video memory.

The *sequencer* generates basic memory timings for the video memory and the character clock for controlling regenerative buffer fetches. It allows the system to access memory during active display intervals by periodically inserting dedicated system microprocessor memory cycles between the display memory cycles. Map mask registers in the sequencer are available to protect entire memory maps from being changed.

The *graphics controller* is the interface between the video memory and the attribute controller during active display times, and between video memory and the system microprocessor during memory accesses.

During active display times, memory data is latched and sent to the attribute controller. In graphics modes, the memory data is converted from parallel to serial bit-plane data before being sent; in alphanumeric modes, the parallel attribute data is sent.

During system accesses of video memory, the graphics controller can perform logical operations on the memory data before it reaches video memory or the system data bus. These logical operations are composed of four logical write modes and two logical read modes. The logical operators allow enhanced operations, such as a color compare in the read mode, individual bit masking during write modes, internal 32-bit writes in a single memory cycle, and writing to the display buffer on non-byte boundaries.

The *attribute controller* takes in data from video memory through the graphics controller and formats it for display. Attribute data in alphanumeric mode and serialized bit-plane data in graphics mode are converted to an 8-bit color value. Each color value is selected from an internal color palette of 64 possible colors (except in 256-color mode). The color value is used as a pointer into the video DAC where it is converted to the analog signals that drive the display.

A.2 VGA Modes

VGA provides alphanumeric modes for text processing and graphics modes for graphics processing. The following table describes the alphanumeric (A/N) and all points addressable (APA) graphics modes available in standard VGA monitors. Each color is selected from 256K possibilities, and gray shades from 64 possibilities. The variations within these modes are selected by setting the number of scan lines. The scan line count is set before the mode set is executed. In the 200-scan-line modes, the data for each scan line is scanned twice. This double scanning allows the 200-scan-line image to be displayed in 400 scan lines.

Table 353. VGA Video Modes

Mode (Hex)	Type	Colors	Alpha Format	Buffer Start	Box Size	Max. Pages	Freq	Resolution (hxv Pixels)	Double Scan	Border Support
0,1	A/N	16	40x25	B8000	8x8	8	70Hz	320x200	Yes	No
0*,1*	A/N	16	40x25	B8000	8x14	8	70Hz	320x350	No	No
0+,1+	A/N	16	40x25	B8000	9x16	8	70Hz	360x400	No	No
2,3	A/N	16	80x25	B8000	8x8	8	70Hz	640x200	Yes	Yes
2-,3-	A/N	16	80x25	B8000	8x14	8	70Hz	640x350	No	Yes
2+,3+	A/N	16	80x25	B8000	9x16	8	70Hz	720x400	No	Yes
4,5	APA	4	40x25	B8000	8x8	1	70Hz	320x200	Yes	No
6	APA	2	80x25	B8000	8x8	1	70Hz	640x200	Yes	Yes
7	A/N	-	80x25	B0000	9x14	8	70Hz	720x350	No	Yes
7+	A/N	-	80x25	B0000	9x16	8	70Hz	720x400	No	Yes
D	APA	16	40x25	A0000	8x8	8	70Hz	320x200	Yes	No
E	APA	16	80x25	A0000	8x8	4	70Hz	640x200	Yes	Yes
F	APA	-	80x25	A0000	8x14	2	70Hz	640x350	No	Yes
10	APA	16	80x25	A0000	8x14	2	70Hz	640x350	No	Yes
11	APA	2	80x30	A0000	8x16	1	60Hz	640x480	No	Yes
12	APA	16	80x30	A0000	8x16	1	60Hz	640x480	No	Yes
13	APA	256	40x25	A0000	8x8	1	70Hz	320x200	Yes	Yes
Note: * or + indicate enhanced modes										

Mode 3+ is the default mode with a color display attached and mode 7 is the default mode with a monochrome display attached. The graphics controller senses whether the display is monochrome or color and initializes itself to the appropriate mode.

A.2.1 Alphanumeric Modes

The alphanumeric modes are modes 0 through 3 and 7. The mode chart lists the variations of these modes. The data format for alphanumeric modes is the same as the data format on the IBM Color/Graphics Monitor Adapter, the IBM Monochrome Display Adapter. and the IBM Enhanced Graphics Adapter.

The video subsystem is initialized according to the selected mode and the color values are loaded into the video DAC. These color values can be changed to give a different color set to select from. Bit 3 of the attribute byte may be redefined by the Character Map Select register to act as a switch between character sets, giving the Programmer access to 512 characters at one time.

When an alphanumeric mode is selected, the character font patterns are transferred from the ROM to map 2. The system stores the character data in map 0, and the attribute data in map 1. In the alphanumeric modes, the programmer views maps 0 and 1 as a single buffer. The CRT controller generates sequential addresses, and fetches one character code byte and one attribute byte at a time. The character code and row scan count are combined to make up the address into map 2, which contains the character font. The appropriate dot patterns are then sent to the attribute controller, where color is assigned according to the attribute data.

Every display character position in the alphanumeric mode is defined by two bytes in the display buffer. Both the color/graphics and the monochrome emulation modes use the following 2-byte character/attribute format.

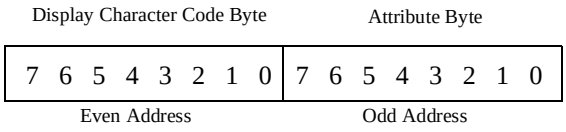


Figure 12. Character/Attribute Format

The functions of the attribute byte are defined in the following table. Bit 7 can be redefined in the Attribute Mode Control register to give 16 possible background colors; its default is to control character blinking. Bit 3 can be redefined in the Character Map Select register to select between two character fonts; its default is to control foreground color selection

.

Table 354. Attribute Byte Definitions

Bit	Color	Function
7	B/I	Blinking or Background Intensity
6	R	Background Color
5	G	Background Color
4	B	Background Color
3	I/CS	Foreground Intensity or Character Font Select
2	R	Foreground Color
1	G	Foreground Color
0	B	Foreground Color

The following are the color values loaded for the 16-color modes.

Table 355. Firmware Color Initialization

IRGB	Color
0000	Black
0001	Blue
0010	Green
0011	Cyan
0100	Red
0101	Magenta
0110	Brown
0111	White
1000	Gray
1001	Light Blue
1010	Light Green
1011	Light Cyan
1100	Light Red
1101	Light Magenta
1110	Yellow
1111	White (High Intensity)

Both 40-column and 80-column alphanumeric modes are supported. The features of the 40-column alphanumeric modes (all variations of modes 0 and 1) are:

- 25 rows of 40 characters
- 2,000 bytes of video memory per page
- One character byte and one attribute byte per character.

The features of the 80-column alphanumeric modes (all variations of modes 2, 3, and 7) are:

- 25 rows of 80 characters
- 4,000 bytes of video memory per page
- One character byte and one attribute byte per character.

A.2.2 Graphics Modes

This section describes the graphics modes supported. The colors in this section are generated when the BIOS is used to set the mode. BIOS initializes the video subsystem and the DAC palette to generate these colors. If the DAC palette is changed, different colors are generated.

A.2.2.1 320 x 200 Four Color Graphics (Modes 4 and 5)

Addressing, mapping, and data format are the same as the 320 x 200 PEL mode of the IBM Color/Graphics Monitor Adapter. The display buffer is configured at 0xB8000. Bit image data is stored in memory maps 0 and 1. The two bit planes (C0 and C1) are each formed from bits from both memory maps.

Features of this mode are:

- A maximum of 200 rows of 320 pels
- Double-scanned to display as 400 rows
- Memory-mapped graphics
- Four colors for each PEL
- Four pels per byte
- 16,000 bytes of read/write memory.

The video memory is organized into two banks of 8,000 bytes each using the following format. Address 0xB8000 contains the PEL information for the upper left corner of the display area.

The following figure shows the format for each byte.

Table 356. PEL Format, Modes 4 and 5

Bit	Function
7	C1- First Display PEL
6	C0 - First Display PEL
5	C1- Second Display PEL
4	C0 - Second Display PEL
3	C1 - Third Display PEL
2	C0 - Third Display PEL
1	C1- Fourth Display PEL
0	C0 - Fourth Display PEL

A.2.2.2 640 x 200 Two Color Graphics (Mode 6)

Addressing, scan-line mapping, and data format are the same as the 640 x 200 PEL black and white mode of the IBM Color/Graphics Monitor Adapter. The display buffer is configured at 0xB8000. Bit image data is stored in memory map 0 and comprises a single bit plane (C0). Features of this mode are:

- A maximum of 200 rows of 640 pels
- Double scanned to display as 400 rows
- Same addressing and scan-line mapping as 320 x 200 graphics
- Two colors for each PEL
- Eight pels per byte
- 16,000 bytes of read/write memory.

The following shows the format for each byte. The bit definition for each PEL is 0 equals black and 1 equals intensified white.

Table 357. PEL Format, mode 6

Bit	Function
7	First Display PEL
6	Second Display PEL
5	Third Display PEL
4	Forth Display PEL
3	Fifth Display PEL
2	Sixth Display PEL
1	Seventh Display PEL
0	Eighth Display PEL

A.2.2.3 640 x 350 Graphics (mode F)

This mode emulates the EGA graphics with the monochrome display and the following attributes: black, video, blinking video, and intensified video. A resolution of 640 x 350 uses 56,000 bytes of video memory to support the four attributes. This mode uses maps 0 and 2; map 0 is the video bit plane (C0), and map 2 is the intensity bit plane (C2). Both planes reside at address 0xA0000.

The two bits, one from each bit plane, define one PEL. The bit definitions are given in the following table.

Table 358. Bit Definitions C2,C0

C2 C0	PEL Color
0 0	Black
01	White
1 0	Blinking White
1 1	Intensified White

Memory is organized with successive bytes defining successive pels. The first eight pels displayed are defined by the byte at 0xA0000, the second eight pels at 0xA0001, and so on. The most significant bit in each byte defines the first PEL for that byte.

Since both bit planes reside at address 0xA0000, the user must select the plane to update through the Map Mask register of the sequencer.

A.2.2.4 640 x 480 Two Color Graphics (mode 11 hex)

This mode provides two color graphics with the same data format as mode 6.

The bit image data is stored in map 0 and comprises a single bit plane (C0). The video buffer starts at 0xA0000. The first byte contains the first eight pels; the second byte at 0xA0001 contains the second eight pels, and so on. The bit definition for each PEL is 0 equals black and 1 equals intensified white.

A.2.2.5 16-Color Graphics Modes (Modes 10 hex, D, E, and 12 hex)

These modes support 16 colors. For all modes, the bit image data is stored in all four memory maps. Each memory map contains the data for one bit plane. The bit planes are C0 through C3 and represent the following colors:

C0= Blue

C1 = Green

C2 = Red

C3 = Intensified

The four bits define each PEL on the screen by acting as an address (pointer) into the internal palette in the Type 2 video.

The display buffer resides at address 0xA0000. The Map Mask register selects any or all of the maps to be updated when the system writes to the display buffer.

A.2.2.6 256-Color Graphics Mode (mode 13 hex)

This mode provides graphics with the capability of displaying 256 colors at one time

The display buffer is sequential, starts at address 0xA0000, and is 64,000 bytes long. The first byte contains the color information for the upper-left PEL. The second byte contains the second PEL, and so on, for 64,000 pels (320 x 200). The bit image data is stored in all four memory maps and comprises four bit planes. The four bit planes are sampled twice to produce eight bit-plane values that address the video DAC.

In this mode, the internal palette of the video subsystem is loaded by BIOS and should not be changed. The first 16 locations in the external palette, which is in the video DAC, contain the colors compatible with the alphanumeric modes. The second 16 locations contain 16 evenly spaced gray shades. The next 216 locations contain values based on a hue-saturation-intensity model tuned to provide a usable, generic color set that covers a wide range of color values.

The following figure shows the color information that is compatible with the colors in other modes:

Table 359. Compatible Color Coding

PEL Bits 76543210	Color Output
00000000	Black
00000001	Blue
00000010	Green
00000011	Cyan
00000100	Red
00000101	Magenta
00000110	Brown
00000111	White
00001000	Gray
00001001	Light Blue
00001010	Light Green

Table 359. Compatible Color Coding ***(Continued)***

PEL Bits 76543210	Color Output
00001011	Light Cyan
00001100	Light Red
00001101	Light Magenta
00001110	Yellow
00001111	White (High Intensity)

Each color in the palette can be programmed to one of 256K different colors.

The features of this mode are:

• A maximum of 200 rows with 320 pels

• Double scanned to display as 400 rows

• Memory-mapped graphics

• 256 of 256K colors for each PEL

• One byte per PEL

• 64,000 bytes of video memory.

A.3 Registers

There are six groups of registers in the video subsystem. All video registers are readable except the system data latches and the attribute address flip-flop. The following table lists the register groups, their l/O addresses with the type of access (read or write), and page reference numbers.

The question mark in the address can be a hex B or D depending on the setting of the I/O address bit in the Miscellaneous Output register, described in “General Registers” on page 203.

Software Implementation Note: All registers in the video subsystem are read/write. The value of reserved bits in these registers must be preserved. Read the register first and change only the bits required.

Table 360. VGA Subsystem Register Overview:

Registers	R/W	Port Address
General Registers		
Sequencer Registers		
Address Register	R/W	0x03C4
Data Registers	R/W	0x03C5
CRT Controller Registers		
Address Register	R/W	0x03?4
Data Registers	R/W	0x03?5
Graphics Controller Registers		
Address Register	R/W	0x03CE
Data Registers	R/W	0x03CF

Table 360. VGA Subsystem Register Overview: ***(Continued)***

Registers	R/W	Port Address
Attribute Controller Registers		
Address Register	R/W	0x03C0
Data Registers	W	0x03C0
	R	0x03C1
Video DAC Palette Registers		
Write Address	R/W	0x03C8
Read Address	W	0x03C7
Data	R/W	0x03C9
PEL Mask	R/W	0x03C6

A.3.1 General Registers

Table 361. General Registers:

Register	Read Address	Write Address
Miscellaneous Output Register	0x03CC	0x03C2
Input Status Register 0	0x03C2	-
Input Status Register 1	0x03?A	-
Feature Control Register	0x03CA	0x03?A
Video Subsystem Enable Register	0x03C3	0x03C3

A.3.1.1 Miscellaneous Output Register

Read Address: 0x03CC

Write Address: 0x03C2

7	6	5	4	3	2	1	0
VSP	HSP	—	—	CS		ERAM	IOS

Miscellaneous Output Register

— : Set to 0, undefined on Read
VSP : Vertical Sync Polarity
HSP : Horizontal Sync Polarity
PB : Page Bit for Odd/Even
CS : Clock Select
ERAM : Enable RAM
IOS : I/O Address Select

The register fields are defined as follows:

VSP Determines the polarity of the vertical sync pulse and can be used (with HSP) to control the vertical size of the display by utilizing the autosynchronization feature of VGA displays.
= 0 selects a positive vertical retrace sync pulse.

HSP Determines the polarity of the horizontal sync pulse.
= 0 selects a positive horizontal retrace sync pulse.
Bits 7 (VSP) and 6 (HSP) select the vertical size as follows:

Bits	
7 6	***Vertical Size***
0 0	- Reserved
0 1	- 400 lines
1 0	- 350 lines
1 1	- 480 lines

PB Selects the upper/lower 64K page of memory when the system is in an eve/odd mode (modes 0,1,2,3,7).
= 0 selects the low page
= 1 selects the high page

CS These two bits select the clock source as below: The external clock is driven through the auxiliary video extension. The input clock should be kept between 14.3 MHz and 28.4 MHz.

Bits	
3 2	***Function***
0 0	- Selects 25 MHz clock for 640/320 Horizontal pels
0 1	- Selects 28 MHz clock for 720/360 Horizontal pels
1 0	- Reserved
1 1	- Reserved

ERAM Controls system access to the display buffer.
= 0 disables address decode for the display buffer from the system
= 1 enables address decode for the display buffer from the system

IOS This bit selects the CRT controller addresses. When set to 0, this bit sets the CRT controller addresses to 0x03Bx and the address for the Input Status Register 1 to 0x03BA for compatibility with the monochrome adapter.
When set to 1, this bit sets CRT controller addresses to 0x03Dx and the Input Status Register 1 address to 0x03DA for compatibility with the color/graphics adapter.
The Write addresses to the Feature Control register are affected in the same manner.

A.3.1.2 Input Status Register 0

Read-only Address: 0x03C2

7	6	5	4	3	2	1	0
CI	—	—	SS	—	—	—	—

Input Status Register 0

— : Set to 0, undefined on Read
CI : CRT Interrupt
SS : Switch Sense

The register fields are defined as follows:

CI When set to 1, this bit indicates a vertical retrace interrupt is pending

SS Returns the status of the four sense switches as selected by the CS field of the Miscellaneous Output Register.

A.3.1.3 Input Status Register 1

Read-only Address: 0x03?A

7	6	5	4	3	2	1	0
—	—	—	—	VR	—	—	DE

Input Status Register

— : Set to 0, undefined on Read
VR : Vertical Retrace
DE : Display Enable

The register fields are defined as follows:

VR When set to 1, this bit indicates that the display is in a vertical retrace interval.This bit can be programmed, through the Vertical Retrace End register, to generate an interrupt at the start of the vertical retrace.

DE When set to 1, this bit indicates a horizontal or vertical retrace interval. This bit is the real-time status of the inverted 'display enable' signal. Programs have used this status bit to restrict screen updates to the inactive display intervals in order to reduce screen flicker. The video subsystem is designed to eliminate this software requirement; screen updates may be made at any time without screen degradation.

A.3.1.4 Feature Control Register

Read Address 0x03CA

Write Address: 0x03?A

All bits are reserved.

7	6	5	4	3	2	1	0
Feature Control							

A.3.1.5 Video Subsystem Enable Register

This register (0x03C3) is reserved.

7	6	5	4	3	2	1	0
Video Subsystem Enable							

A.3.2 Sequencer Registers

The Address register is at address **0x03C4** and the data registers are at address **0x03C5**. All registers within the sequencer are read/write.

Table 362. Sequencer Registers

Register	Index (hex)
Sequencer Address	—
Reset	00
Clocking Mode	01
Map Mask	02
Character Map Select	03
Memory Mode	04

A.3.2.1 Sequencer Address Register

The Address register is at address **0x03C4**. This register is loaded with an index value that points to the desired sequencer data register.

7	6	5	4	3	2	1	0
—	—	—	—	—	SA		

Sequencer Address Register

— : Set to 0, undefined on Read
SA : Sequencer Address

The register field is defined as follows:

SA These bits contain the index value that points to the data register to be accessed.

A.3.2.2 Reset Register

Address 0x03C4; Data 0x03C5; Index 0x00.

7	6	5	4	3	2	1	0
—	—	—	—	—	—	SR	ASR

Reset Register

— : Set to 0, undefined on Read
SR : Synchronous Reset
ASR : Asynchronous Reset

The register fields are defined as follows:

SR When set to 0, this bit commands the sequencer to synchronously clear and halt. Bits 1 and 0 must be 1 to allow the sequencer to operate. To prevent the loss of data, bit 1 must be set to 0 during the active display interval before changing the clock selection. The clock is changed through the Clocking Mode register or the Miscellaneous Output register.

ASR When set to 0, this bit commands the sequencer to asynchronously clear and halt. Resetting the sequencer with this bit can cause loss of video data.

A.3.2.3 Clocking Mode Register

Address 0x03C4; Data 0x03C5; Index 0x01.

7	6	5	4	3	2	1	0
—	—	SO	SH4	DC	SL	1	D89

Clocking Mode Register

— : Set to 0, undefined on Read
1 : Set to 1, undefined on Read
SO : Screen Off
SH4 : Shift 4
DC : Dot Clock
SL : Shift Load
D89 : 8/9 Dot Clocks

The register fields are defined as follows:

SO When set to 1, this bit turns off the display and assigns maximum memory bandwidth to the system. Although the display is blanked, the synchronization pulses are maintained. This bit can be used for rapid full-screen updates.

SH4 When the Shift 4 field and the Shift Load Field are set to 0, the video serializers are loaded every character clock. When the Shift 4 field is set to 1, the video serializers are loaded every forth character clock, which is useful when 32 bits are fetched per cycle and chained together in the shift registers.

DC When set to 0, this bit selects the normal dot clocks derived from the sequencer master clock input. When this bit is set to 1, the master clock will be divided by 2 to generate the dot clock. All other timings are affected because they are derived from the dot clock. The dot clock divided by 2 is used for 320 and 360 horizontal PEL modes.

SL When this bit and bit 4 are set to 0, the video serializers are loaded every character clock. When this bit is set to 1, the video serializers are loaded every other character clock, which is useful when 16 bits are fetched per cycle and chained together in the shift registers. The Type 2 video behaves as if this bit is set to 0; therefore, programs should set it to 0.

D89 When set to 0, this bit directs the sequencer to generate character clocks 9 dots wide; when set to 1, it directs the sequencer to generate character clocks 8 dots wide. The 9-dot mode is for alphanumeric modes 0+, 1+, 2+, 3+, 7, and 7 + only; the 9th dot equals the 8th dot for ASCII codes 0xC0 through 0xDF. All other modes must use 8 dots per character clock. See the line graphics character bit in the “Attribute Mode Control Register” on page 228

A.3.2.4 Map Mask Register

Address 0x03C4; Data 0x03C5; Index 0x02.

7	6	5	4	3	2	1	0
—	—	—	—	M3E	M2E	M1E	M0E

Map Mask Register, Index 0x02

— : Set to 0, undefined on Read
M3E : Map 3 Enable
M2E : Map 2 Enable
M1E : Map 1 Enable
M0E : Map 0 Enable
D89 : 8/9 Dot Clocks

The register fields are defined as follows:

M*E — When set to 1, these bits enable system access to the corresponding map. If all maps are enabled, the system can write its 8-bit value to all four maps in a single memory cycle. This substantially reduces the system overhead during display updates in graphics modes.
Data scrolling operations can be enhanced by enabling all maps and writing the display buffer address with the data stored in the system data latches. This is a Read-Modify-Write operation.
When odd/even modes are selected, maps 0 and 1 and maps 2 and 3 should have the same map mask value.
When chain 4 mode is selected, all maps should be enabled.

A.3.2.5 Character Map Select Register

Address 0x03C4; Data 0x03C5; Index 0x03.

In alphanumeric modes, bit 3 of the attribute byte normally defines the foreground intensity. This bit can be redefined as a switch between character sets allowing 512 displayable characters. To enable this feature:

1. Set the extended memory bit in the Memory Mode register (0x04) to 1.

2. Select different values for character map A and character map B.

This function is supported by BIOS and is a function call within the character generator routines.

7	6	5	4	3	2	1	0
—	—	MAH	MBH	MAL		MBL	

Character Map Select Register, Index 0x03

— : Set to 0, undefined on Read
MAH : Character Map A Select (MSB)
MBH : Character Map B Select (MSB)
MAL : Character Map A Select (LSB)
MBL : Character Map B Select (LSB)

The register fields are defined as follows:

MAH This bit is the most-significant bit for selecting the location of character map A.

MBH This bit is the most-significant bit for selecting the location of character map B.

MAL These bits and bit 5 select the location of character map A. Map A is the area of map 2 containing the character font table used to generate characters when attribute bit 3 is set to 1. The selection is shown in Table 363 on page 209.

MBL These bits and bit 4 select the location of character map B. Map B is the area of map 2 containing the character font table used to generate characters when attribute bit 3 is set to 0. The selection is shown in Table 364 on page 209.

Bits 532	Map Selected	Table Location
000	0	1st 8KB of display memory plane 2
001	1	3rd 8KB
010	2	5th 8KB
011	3	7th 8KB
100	4	2nd 8KB
101	5	4th 8KB
110	6	6th 8KB
111	7	8th 8KB

Table 363. Character Map Select A

Bits 410	Map Selected	Table Location
000	0	1st 8KB of display memory plane 2
001	1	3rd 8KB
010	2	5th 8KB
011	3	7th 8KB
100	4	2nd 8KB
101	5	4th 8KB
110	6	6th 8KB
111	7	8th 8KB

Table 364. Character Map Select B

A.3.2.6 Memory Mode Register

Address 0x03C4; Data 0x03C5; Index 0x04.

7	6	5	4	3	2	1	0
—	—	—	—	CH4	OE	EM	—

Memory Mode Register, Index 0x04

— : Set to 0, undefined on Read
CH4 : Chain 4
OE : Odd/Even
EM : Extended Memory

The register fields are defined as follows:

CH4 This bit controls the map selected during system read operations. When set to 0, this bit enables system addresses to sequentially access data within a bit map by using the Map Mask register. When set to 1, this bit causes the two low-order bits to select the map accessed as shown below.

Address Bits	
A0	***A1***	***Map Selected***
0	0	0
0	1	1
1	0	2
1	1	3

OE When this bit is set to 0, even system addresses access maps 0 and 2, while odd system addresses access maps 1 and 3. When this bit is set to 1, system addresses sequentially access data within a bit map, and the maps are accessed according to the value in the Map Mask register (index 0x02).

EM When set to 1, this bit enables the video memory from 64KB to 256KB. This bit must be set to 1 to enable the character map selection described for the previous register.

A.3.3 CRT Controller Registers

A data register is accessed by writing its index to the Address register at address 0x03D4 or 0x03B4, and then writing the data to the access port at address 0x03D5 or 0x03B5. The l/O address used depends on the setting of the l/O address select bit (bit 0) in the Miscellaneous Output register, which is described in “General Registers” on page 203. The following figure shows the variable part of the address as a question mark.

Software Implementation Note: When modifying a register, the setting of reserved bits must be preserved. Read the register first and change only the bits required.

Table 365. CRT Controller Registers

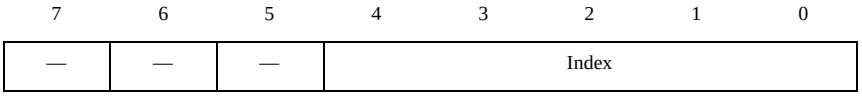
Register Name	Address	Index
Address	0x03?4	—
Horizontal Total	0x03?5	0x0
Horizontal Display-Enable End	0x03?5	0x01
Start Horizontal Blanking	0x03?5	0x02
End Horizontal Blanking	0x03?5	0x03
Start Horizontal Retrace Pulse	0x03?5	0x04
Index values not listed are reserved.		

Table 365. CRT Controller Registers ***(Continued)***

Register Name	Address	Index
End Horizontal Retrace	0x03?5	0x05
Vertical Total	0x03?5	0x06
Overflow	0x03?5	0x07
Preset Row Scan	0x03?5	0x08
Maximum Scan Line	0x03?5	0x09
Cursor Start	0x03?5	0x0A
Cursor End	0x03?5	0x0B
Start Address High	0x03?5	0x0C
Start Address Low	0x03?5	0x0D
Cursor Location High	0x03?5	0x0E
Cursor Location Low	0x03?5	0x0F
Vertical Retrace Start	0x03?5	0x10
Vertical Retrace End	0x03?5	0x11
Vertical Display-Enable End	0x03?5	0x12
Offset	0x03?5	0x13
Underline Location	0x03?5	0x14
Start Vertical Blanking	0x03?5	0x15
End Vertical Blanking	0x03?5	0x16
CRT Mode Control	0x03?5	0x17
Line Compare	0x03?5	0x18
Index values not listed are reserved.		

A.3.3.1 Address Register

This register is at address 0x03?4, and is loaded with an index value that points to the data registers within the CRT controller.



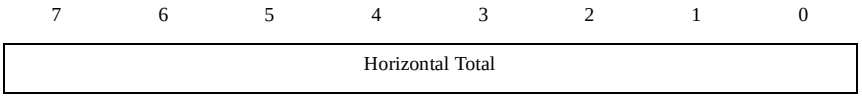
Address Register

— Set to 0, undefined on Read

Index : These bits are the index that points to the data register accessed through address 0x03D5 or 0x03B5.

A.3.3.2 Horizontal Total Register

Address 0x03?4; Data 0x03?5; Index 0x00

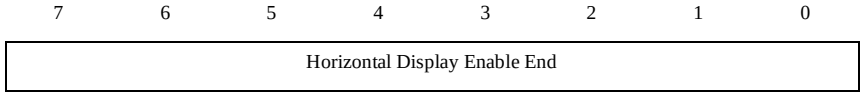


This register the total number of characters in the horizontal scan interval including the retrace time. The value directly controls the period of the 'horizontal retrace' signal. A horizontal character counter in the CRT controller counts the character clock inputs; comparators are used to compare the register value with the character's horizontal width to provide horizontal timings. All horizontal and vertical timings are based on this register.

The value contained in this register is the total number of characters minus 5.

A.3.3.3 Horizontal Display-Enable End Register

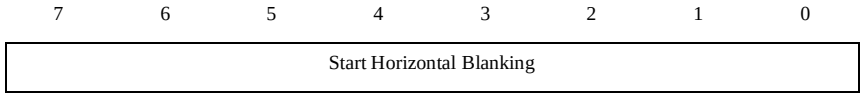
Address 0x03?4; Data 0x03?5; Index 0x01



The value in this register defines the length of the 'horizontal display-enable' signal, and determines the number of character positions per horizontal line. The value contained in this register is the total number of displayed characters minus 1.

A.3.3.4 Start Horizontal Blanking Register

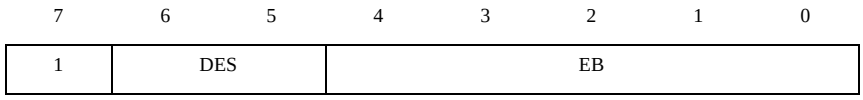
Address 0x03?4; Data 0x03?5; Index 0x02



The value in this register is the horizontal character count where the 'horizontal blanking' signal goes active.

A.3.3.5 End Horizontal Blanking Register

Address 0x03?4; Data 0x03?5; Index 0x03



End Horizontal Blanking Register, Index 0x03

1 : Set to 1, undefined on Read
DES : Display Enable Skew Control
EB : End Blanking

The register fields are defined as follows:

DES — These two bits determine the amount of skew of the 'display enable' signal. This skew control is needed to provide sufficient time for the CRT controller to read a character and attribute code from the video buffer, to gain access to the character generator, and go through the Horizontal PEL Panning register in the attribute controller. Each access requires the 'display enable' signal to be skewed one character clock so that the video output is synchronized with the horizontal and vertical retrace signals. The skew values are shown below.

DES Field

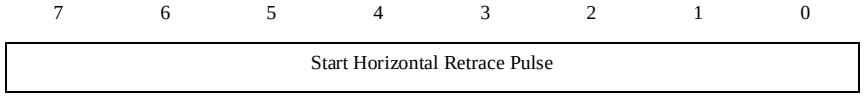
6 5	***Amount of Skew***
0 0	No character clock skew
0 1	One character clock skew
1 0	Two character clock skew
1 1	Three character clock skew

Note: Character skew is not adjustable on the Type 2 video and the bits are ignored; however, programs should set these bits for the appropriate skew to maintain compatibility.

EB — These bits are the five low-order bits of a 6-bit value that is compared with the value in the Start Horizontal Blanking register to determine when the 'horizontal blanking' signal will go inactive. The most-significant bit is bit 7 in the End Horizontal Retrace register (index 0x05).
To program these bits for a signal width of W, the following algorithm is used: the width W, in character clock units, is added to the value from the Start Horizontal Blanking register. The six low-order bits of the result are the 6-bit value programmed.

A.3.3.6 Start Horizontal Retrace Pulse Register

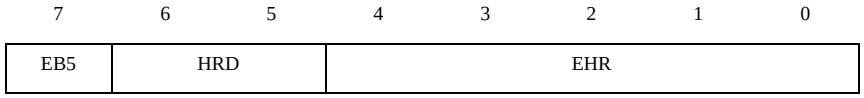
Address 0x03?4; Data 0x03?5; Index 0x04



These bits are used to center the screen horizontally by specifying the character position where the 'horizontal retrace' signal goes active.

A.3.3.7 End Horizontal Retrace Register

Address 0x03?4; Data 0x03?5; Index 0x05



End Horizontal Retrace Register, Index 0x05

EB5 : End Horizontal Blanking, Bit 5
HRD : Horizontal Retrace Delay
EHR : End Horizontal Retrace

The register fields are defined as follows:

EB5 — This bit is the most significant bit (bit 5) of the end horizontal blanking value in the End Horizontal Blanking register (index 0x03).

The register fields are defined as follows: ***(Continued)***

HRD — These bits control the skew of the 'horizontal retrace' signal. The value of these bits is the amount of skew provided (from 0 to 3 character clock units). For certain modes, the 'horizontal retrace' signal takes up the entire blanking interval. Some internal timings are generated by the falling edge of the 'horizontal retrace' signal. To ensure that the signals are latched properly, the 'retrace' signal is started before the end of the 'display enable' signal and then skewed several character clock times to provide the Proper screen centering.

EHR — These bits are compared with the Start Horizontal Retrace register to give a horizontal character count where the 'horizontal retrace' signal goes inactive.
To program these bits with a signal width of W, the following algorithm is used: the width W, in character clock units, is added to the value in the Start Retrace register. The five low-order bits of the result are the 5-bit value programmed.

A.3.3.8 Vertical Total Register

Address 0x03?4; Data 0x03?5; Index 0x06

7	6	5	4	3	2	1	0
Vertical Total							

These are the eight low-order bits of a 10-bit vertical total. The value for the vertical total is the number of horizontal raster scans on the display, including vertical retrace, minus 2. This value determines the period of the 'vertical retrace' signal.

Bits 8 and 9 are in the Overflow register (index 0x07).

A.3.3.9 Overflow Register

Address 0x03?4; Data 0x03?5; Index 0x07

7	6	5	4	3	2	1	0
VRS9	VDE9	VT9	LC8	VBS8	VRS8	VDE8	VT8

Overflow Register, Index 0x07

VRS9 : Vertical Retrace Start, Bit 9
VDE9 : Vertical Display Enable End, Bit 9
VT9 : Vertical Total, Bit 9
LC8 : Line Compare, Bit 8
VBS8 : Vertical Blanking Start, Bit 8
VRS8 : Vertical Retrace Start, Bit 8
VDE8 : Vertical Display Enable End, Bit 8
VT8 : Vertical Total, Bit 8

The register fields are defined as follows:

VRS9 — Bit 9 of the Vertical Retrace Start register (index 0x10).
VDE9 — Bit 9 of the Vertical Display Enable End register (index 0x12).
VT9 — Bit 9 of the Vertical Total register (index 0x06).
LC8 — Bit 8 of the Line Compare register (index 0x18).
VBS8 — Bit 8 of the Start Vertical Blanking register (index 0x15).
VRS8 — Bit 8 of the Vertical Retrace Start register (index 0x10).
VDE8 — Bit 8 of the Vertical Display Enable End register (index 0x12).

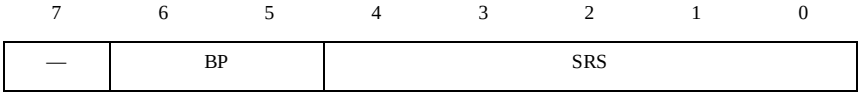
The register fields are defined as follows: ***(Continued)***

VRS9 Bit 9 of the Vertical Retrace Start register (index 0x10).

VT8 Bit 8 of the Vertical Total register (index 0x06).

A.3.3.10 Preset Row Scan Register

Address 0x03?4; Data 0x03?5; Index 0x08



Preset Row Scan Register, Index 0x08

— : Set to 0, Undefined on Read
BP : Byte Panning
SRS : Starting Row Scan Count

The register fields are defined as follows:

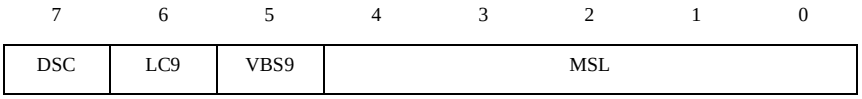
BP These two bits control byte panning in multiple shift modes. These bits are used in pel-panning operations. and should normally be set to 0.

SRS These bits specify the row scan count for the row starting after a vertical retrace. The row scan counter is incremented every horizontal retrace time until the maximum row scan occurs. When the maximum row scan is reached, the row scan counter is cleared (not preset).

Note: The CRT controller latches the start address at the start of the vertical retrace. These register values should be loaded during the active display time.

A.3.3.11 Maximum Scan Line Register

Address 0x03?4; Data 0x03?5; Index 0x09



Maximum Scan Line Register, Index 0x09

DSC : 200 to 400 Line Conversion
LC9 : Line Compare, Bit 9
VBS9 : Start Vertical Blanking, Bit 9
MSL : Maximum Scan Line

The register fields are defined as follows:

DSC When this bit is set to 1, 200-scan-line video data is converted to 400-scan-line output. To do this, the clock in the row scan counter is divided by 2, which allows the 200-line modes to be displayed as 400 lines on the display (this is called double scanning; each line is displayed twice). When this bit is set to 0, the clock to the row scan counter is equal to the horizontal scan rate.

LC9 Bit 9 of the Line Compare register (index 0x18).

VBS9 Bit 9 of the Start Vertical Blanking register (index 0x15).

The register fields are defined as follows: ***(Continued)***

DSC When this bit is set to 1, 200-scan-line video data is converted to 400-scan-line output. To do this, the clock in the row scan counter is divided by 2, which allows the 200-line modes to be displayed as 400 lines on the display (this is called double scanning; each line is displayed twice). When this bit is set to 0, the clock to the row scan counter is equal to the horizontal scan rate.

MSL These bits specify the number of scan lines per character row. The value of these bits is the maximum row scan number minus 1.

A.3.3.12 Cursor Start Register

Address 0x03?4; Data 0x03?5; Index 0x0A

7	6	5	4	3	2	1	0
—	—	CO	RSCB				

Cursor Start Register, Index 0x0A

— : Set to 0, Undefined on Read
CO : Cursor Off
RSCB : Row Scan Cursor Begins

The register fields are defined as follows:

CO When set to 1, this bit disables the cursor.

RSCB These bits specify the row within the character box where the cursor begins. The value of these bits is the first line of the cursor minus 1. When this value is greater than that in the Cursor End register, no cursor is displayed.

A.3.3.13 Cursor End Register

Address 0x03?4; Data 0x03?5; Index 0x0B

7	6	5	4	3	2	1	0
—	CSK		RSCE				

Cursor End Register, Index 0x0B

— : Set to 0, Undefined on Read
CSK : Cursor Skew Control
RSCE : Row Scan Cursor Ends

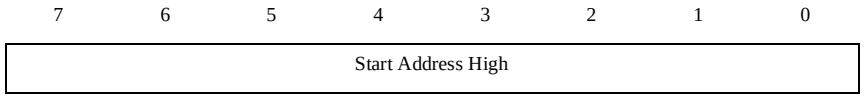
The register fields are defined as follows:

CSK These bits control the skew of the cursor. The skew value delays the cursor by the selected number of character clocks from 0 to 3. For example, a skew of 1 moves the cursor right one position on the screen.

RSCE These bits specify the row within the character box where the cursor ends. If this value is less that the cursor start value, no cursor is displayed.

A.3.3.14 Start Address High Register

Address 0x03?4; Data 0x03?5; Index 0x0C

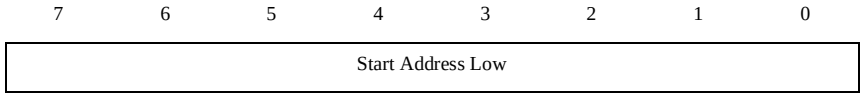


These are the eight high-order bits of a 16-bit value that specifies the starting address for the regenerative buffer. The start address points to the first address after the vertical retrace on each screen refresh.

Software Implementation Note: The CRT controller latches the start address at the start of the vertical retrace. These register values should be loaded during the active display time.

A.3.3.15 Start Address Low Register

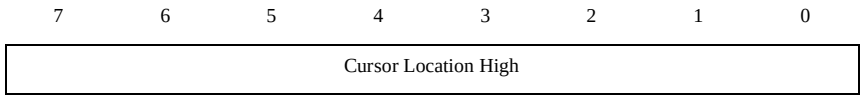
Address 0x03?4; Data 0x03?5; Index 0x0D



These are the eight low-order bits of the starting address for the regenerative buffer.

A.3.3.16 Cursor Location High Register

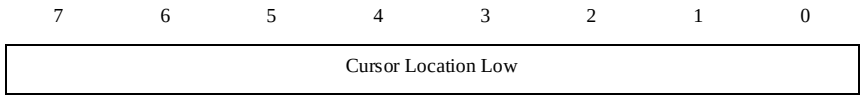
Address 0x03?4; Data 0x03?5; Index 0x0E



These are the eight high-order bits of the 16-bit cursor

A.3.3.17 Cursor Location Low Register

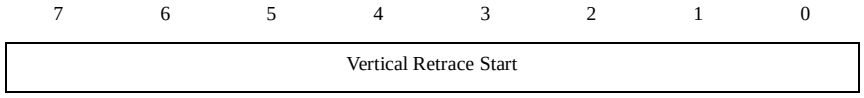
Address 0x03?4; Data 0x03?5; Index 0x0F



These are the eight low-order bits of the cursor location.

A.3.3.18 Vertical Retrace Start Register

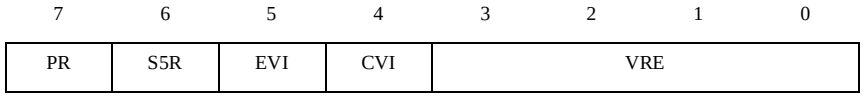
Address 0x03?4; Data 0x03?5; Index 0x10



These are the eight low-order bits of the 9-bit start position for the 'vertical retrace' pulse; it is programmed in horizontal scan lines. Bit 8 is in the Overflow register (index 0x07).

A.3.3.19 Vertical Retrace End Register

Address 0x03?4; Data 0x03?5; Index 0x11



Vertical Retrace End Register, Index 0x11

PR : Protect Registers 0-7
S5R : Select 5 Refresh Cycles
EVI : Enable Vertical Interrupt
CVI : Clear Vertical Interrupt
VRE : Vertical Retrace End

The register fields are defined as follows:

PR — When set to 1, this bit disables write access to the CRT controller registers at index 00 through 07. The line compare bit in the Overflow register (index 0x07) is not protected.

S5R — When set to 1, this bit generates five memory refresh cycles per horizontal line. When set to 0, this bit selects three refresh cycles. Selecting five refresh cycles allows use of the VGA chip with 15.75 kHz displays. This bit should be set to 0 for supported operations.

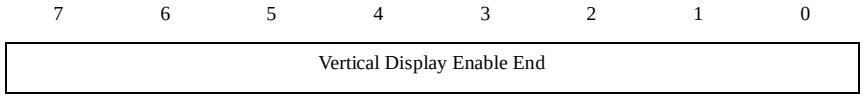
EVI — When set to 0, this bit enables a vertical retrace interrupt. The vertical retrace interrupt is IRQ2. This interrupt level can be shared; therefore, to determine whether the video generated the interrupt, check the CRT interrupt bit in Input Status Register 0.

CVI — When set to 0, this bit clears a vertical retrace interrupt. At the end of the active vertical display time, a flip-flop is set to indicate an interrupt. An interrupt handler resets this flip-flop by first setting this bit to 0, then resetting it to 1.

VRE — The Vertical Retrace Start register is compared with these four bits to determine where the 'vertical retrace' signal goes inactive. It is programmed in units of horizontal scan lines. To program these bits with a signal width of W, the following algorithm is used: the width W, in horizontal scan units, is added to the value in the Start Vertical Retrace register. The four low-order bits of the result are the 4-bit value programmed.

A.3.3.20 Vertical Display Enable End Register

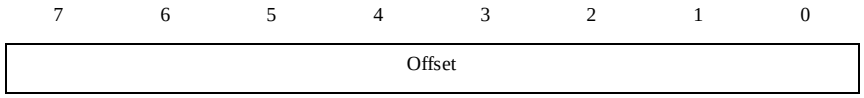
Address 0x03?4; Data 0x03?5; Index 0x12



These are the eight low-order bits of a 10-bit value that defines the vertical-display-enable end position. The two high-order bits are contained in the Overflow register (index 0x07). The 10-bit value is equal to the total number of scan lines minus 1.

A.3.3.21 Offset Register

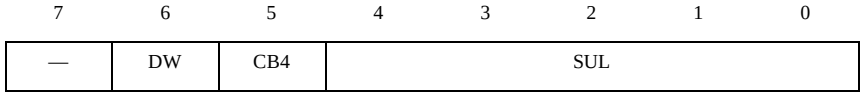
Address 0x03?4; Data 0x03?5; Index 0x013



These bits specify the logical line width of the screen. The starting memory address for the next character row is larger than the current character row by 2 or 4 times the value of these bits. Depending on the method of clocking the CRT controller, this address is either a word or doubleword address.

A.3.3.22 Underline Location Register

Address 0x03?4; Data 0x03?5; Index 0x14



Underline Location Register, Index 0x14

— : Set to 0, Undefined on Read
DW : Doubleword Mode
CB4 : Count by 4
SUL : Start Underline

The register fields are defined as follows:

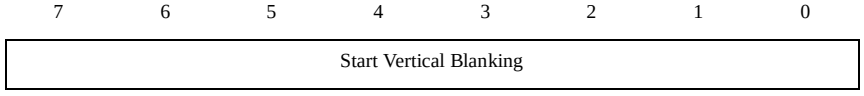
DW — When this bit is set to 1, memory addresses are doubleword addresses. See the description of the word/byte mode bit (bit 6) in the “CRT Mode Control Register” on page 220.

CB4 — When this bit is set to 1, the memory-address counter is clocked with the character clock divided by 4, which is used when doubleword addresses are used.

SUL — These bits specify the horizontal scan line of a character row on which an underline occurs. The value programmed is the scan line desired minus 1.

A.3.3.23 Start Vertical Blanking Register

Address 0x03?4; Data 0x03?5; Index 0x15



These are the eight low-order bits of a 10-bit value that specifies the starting location for the 'vertical blanking' signal. Bit 8 is in the Overflow register (index 0x07) and bit 9 is in the Maximum Scan Line register (index 0x09). The 10-bit value is the horizontal scan line count where the 'vertical blanking' signal becomes active minus 1.

A.3.3.24 End Vertical Blanking Register

Address 0x03?4; Data 0x03?5; Index 0x16

7	6	5	4	3	2	1	0
End Vertical Blanking							

This register specifies the horizontal scan count where the 'vertical blanking' signal becomes inactive. The register is Programmed in units of the horizontal scan line.

To program these bits with a 'vertical blanking' signal of width W, the following algorithm is used: the width W, in horizontal scan line units, is added to the value in the Start Vertical Blanking register minus 1. The eight low-order bits of the result are the 8-bit value programmed.

A.3.3.25 CRT Mode Control Register

Address 0x03?4; Data 0x03?5; Index 0x17

7	6	5	4	3	2	1	0
RST	WB	ADW	—	CB2	HRS	SRC	CMS0

CRT Mode Control Register, Index 0x17

— : Set to 0, Undefined on Read
RST : Hardware Reset
WB : Word/Byte Mode
ADW : Address Wrap
CB2 : Count by Two
HRS : Horizontal Retrace Select
SRC : Select Row Scan Counter
CMS0 : Compatibility Mode Support

The register fields are defined as follows:

RST — When set to 0, this bit disables the horizontal and vertical retrace signals and forces them to an inactive level. When set to 1, this bit enables the horizontal and vertical retrace signals. This bit does not reset any other registers or signal outputs.

WB — When this bit is set to 0, the word mode is selected. The word mode shifts the memory-address counter bits to the left by one bit; the most-significant bit of the counter appears on the least-significant bit of the memory address outputs.
The doubleword bit in the Underline Location register (0x14) also controls the addressing. When the doubleword bit is 0, the word/byte bit selects the mode. When the doubleword bit is set to 1, the addressing is shifted by two bits.
When set to 1, bit 6 selects the byte address mode.

ADW — This bit selects the memory-address bit, bit MA 13 or MA 15, that appears on the output pin MA 0, in the word address mode. If the VGA is not in the word address mode, bit 0 from the address counter appears on the output pin, MA 0.
When set to 1, this bit selects MA 15. In odd/even mode, this bit should be set to 1 because 256KB of video memory is installed on the system board. (Bit MA 13 is selected in applications where only 64KB is present. This function maintains compatibility with the IBM Color/Graphics Monitor Adapter.)

The register fields are defined as follows: ***(Continued)***

CB2 When this bit is set to 0, the address counter uses the character clock. When this bit is set to 1, the address counter uses the character clock input divided by 2. This bit is used to create either a byte or word refresh address for the display buffer.

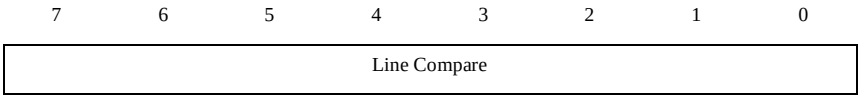
HRS This bit selects the clock that controls the vertical timing counter. The clocking is either the horizontal retrace clock or horizontal retrace clock divided by 2. When this bit is set to 1. the horizontal retrace clock is divided by 2.
Dividing the clock effectively doubles the vertical resolution of the CRT controller. The vertical counter has a maximum resolution of 1024 scan lines because the vertical total value is 10-bits wide. If the vertical counter is clocked with the horizontal retrace divided by 2, the vertical resolution is doubled to 2048 scan lines.

SRC This bit selects the source of bit 14 of the output multiplexer. When this bit is set to 0, bit 1 of the row scan counter is the source. When this bit is set to 1, the bit 14 of the address counter is the source.

CMS0 This bit selects the source of bit 13 of the output multiplexer. When this bit is set to 0, bit 0 of the row scan counter is the source, and when this bit is set to 1, bit 13 of the address counter is the source.
The CRT controller used on the IBM Color/Graphics Adapter was capable of using 128 horizontal scan-line addresses. For the VGA to obtain 640-by-200 graphics resolution, the CRT controller is programmed for 100 horizontal scan lines with two scan-line addresses per character row. Row scan address bit 0 becomes the most-significant address bit to the display buffer. Successive scan lines of the display image are displaced in 8KB of memory. This bit allows compatibility with the graphics modes of earlier adapters.

A.3.3.26 Line Compare Register

Address 0x03?4; Data 0x03?5; Index 0x18



This register contains the eight low-order bits of the 10-bit compare target. When the vertical counter reaches the target value, the internal start address of the line counter is cleared. This creates a split screen where the lower screen is immune to scrolling. Bit 8 is in the Overflow register (index 0x07), and bit 9 is in the Maximum Scan Line register (index 0x09).

A.3.4 Graphics Controller Registers

The Address register for the graphics controller is at address 0x03CE. The data registers are at address 0x03CF. All registers are read/write.

Table 366. Graphics Controller Registers

Register Name	Address	Index
Address	0x03CE	—
Set/Reset	0x03CF	0x0
Enable Set/Reset	0x03CF	0x01
Color Compare	0x03CF	0x02
Data Rotate	0x03CF	0x03
Read Map Select	0x03CF	0x04
Graphics Mode	0x03CF	0x05
Miscellaneous	0x03CF	0x06

Table 366. Graphics Controller Registers ***(Continued)***

Register Name	Address	Index
Color Don't Care	0x03CF	0x07
Bit Mask	0x03CF	0x08

A.3.4.1 Address Register

The Address register is at address **0x03CE**

7	6	5	4	3	2	1	0
—	—	—	—	Index			

This register is loaded with the index value that points to the desired data register within the graphics controller.

A.3.4.2 Set/Reset Register

Address 0x03CE; Data 0x03CF; Index 0x00

7	6	5	4	3	2	1	0
—	—	—	—	SR3	SR2	SR1	SR0

Set/Reset Register, Index 0x00

— : Set to 0, Undefined on Read
SR3 : Set/Reset Map 3
SR2 : Set/Reset Map 2
SR1 : Set/Reset Map 1
SR0 : Set/Reset Map 0

The register fields are defined as follows:

SR* When write mode 0 is selected, the system writes the value of each set/reset bit to its respective memory map. For each write operation, the set/reset bit, if enabled, is written to all eight bits within that map. Set/reset operation can be enabled on a map-by-map basis through the Enable Set/Reset register.

A.3.4.3 Enable Set/Reset Register

Address 0x03CE; Data 0x03CF; Index 0x01

7	6	5	4	3	2	1	0
—	—	—	—	ESR3	ESR2	ESR1	ESR0

Enable Set/Reset Register, Index 0x01

— : Set to 0, Undefined on Read
ESR3 : Enable Set/Reset Map 3
ESR2 : Enable Set/Reset Map 2
ESR1 : Enable Set/Reset Map 1

Enable Set/Reset Register, Index 0x01

ESR0 : Enable Set/Reset Map 0

The register fields are defined as follows:

ESR* These bits enable the set/reset function used when write mode 0 is selected in the Graphics Mode register (index 0x05). When the bit is set to 1, the respective memory map receives the value specified in the Set/Reset register. When Set/Reset is not enabled for a map, that map receives the value sent by the system.

A.3.4.4 Color Compare Register

Address 0x03CE; Data 0x03CF; Index 0x02

7	6	5	4	3	2	1	0
—	—	—	—	CC3	CC2	CC1	CC0

Color Compare Register, Index 0x02

— : Set to 0, Undefined on Read
CC3 : Color Compare Map 3
CC2 : Color Compare Map 2
CC1 : Color Compare Map 1
CC0 : Color Compare Map 0

The register fields are defined as follows:

CC* These bits are the 4-bit color value to be compared when the read mode bit in the Graphics Mode register is set to 1. When the system does a memory read, the data returned from the memory cycle will be a 1 in each bit position where the four maps equal the Color Compare register. If the read mode bit is 0, the data is returned without comparison.
All bits of the corresponding map's byte are compared with the color compare bit. Each of the eight bit positions in the selected byte are compared across the four maps, and a 1 is returned in each position where the bits of all four maps equal their respective color compare values.

A.3.4.5 Data Rotate Register

Address 0x03CE; Data 0x03CF; Index 0x03

7	6	5	4	3	2	1	0
—	—	—	FS		RC		

Data Rotate Register, Index 0x03

— : Set to 0, Undefined on Read
FS : Function Select
RC : Rotate Count

The register fields are defined as follows:

FS Data written to the video buffer can be operated on logically by data already in the system latches. The Function Select field determines whether and how this is done.
Data can be any of the choices selected by the write mode bits except system latches, which cannot be modified. If rotated data is selected also, the rotate is performed before the logical operation. The logical operations selected are shown in the following table.

FS Field

4	***3***	***Function***
0	0	Data Unmodified
0	1	Data ANDed with Latched Data
1	0	Data ORed with Latched Data
1	1	Data XORed with Latched Data

RC In write mode 0, these bits select the number of positions the system data is rotated to the right during a system Memory Write operation. To write data that is not rotated in mode 0, all bits are set to 0.

A.3.4.6 Read Map Select Register

Address 0x03CE; Data 0x03CF; Index 0x04

7	6	5	4	3	2	1	0
—	—	—	—	—	—	MS	

Read Map Select Register, Index 0x04

— : Set to 0, Undefined on Read
MS : Map Select

The register field is defined as follows:

MS These bits select the memory map for system read operations. This register has no effect on the color compare read mode. In odd/even modes, the value can be a binary 00 or 01 to select the chained maps 0, 1 and the value can be a binary 10 or 11 to select the chained maps 2, 3.

A.3.4.7 Graphics Mode Register

Address 0x03CE; Data 0x03CF; Index 0x05

7	6	5	4	3	2	1	0
—	C256	SR	OE	RM	—	WM	

Graphics Mode Register, Index 0x05

— : Set to 0, Undefined on Read
C256 : 256 - Color Mode
SR : Shift Register Mode
OE : Odd/Even
RM : Read Mode
WM : Write Mode

The register fields are defined as follows:

C256 When set to 0, this bit allows bit 5 to control the loading of the shift registers. When set to 1, this bit causes the shift registers to be loaded in a manner that supports the 256-color mode.

SR When set to 1, this bit directs the shift registers in the graphics controller to format the serial data stream with even-numbered bits from both maps on even-numbered maps, and odd-numbered bits from both maps on the odd-numbered maps. This bit is used for modes 4 and 5.

OE When set to 1, this bit selects the odd/even addressing mode used by the IBM Color/Graphics Monitor Adapter. Normally, the value here follows the value of Memory Mode register bit 2 in the sequencer.

RM When this bit is set to 1, the system reads the results of the comparison of the four memory maps and the Color Compare register.
When this bit is set to 0, the system reads data from the memory map selected by the Read Map Select register, or by the two low-order bits of the memory address (this selection depends on the chain-4 bit in the Memory Mode register of the sequencer).

WM The write mode selected and its operation are defined below. The logic operation specified by the function select bits is performed on system data for modes 0, 2, and 3.

WM Field	*Mode Description*
00	Each memory map is written with the system data rotated by the count in the Data Rotate register. If the set/reset function is enabled for a specific map, that map receives the 8-bit value contained in the Set/Reset register.
01	Each memory map is written with the contents of the system latches. These latches are loaded by a system Read operation.
10	Memory map n (0 through 3) is filled with eight bits of the value of data bit n.
11	Each memory map is written with the 8-bit value contained in the Set/Reset register for that map (the Enable Set/Reset register has no effect). Rotated system data is ANDed with the Bit Mask register to form an 8-bit value that performs the same function as the Bit Mask register in write modes 0 and 2 (see also “Bit Mask Register” on page 226).

A.3.4.8 Miscellaneous Register

Address 0x03CE; Data 0x03CF; Index 0x06

7	6	5	4	3	2	1	0
—	—	—	—	MM		OE	GM

Miscellaneous Register, Index 0x06

— : Set to 0, Undefined on Read
MM : Memory Map
OE : Odd/Even
GM : Graphics Mode

The register fields are defined as follows:

MM These bits control the mapping of the regenerative buffer into the system address space. The bit functions are defined below.

MM Field

3	***2***	***Addressing Assignment***
0	0	A0000 for 128KB
0	1	A0000 for 64KB
1	0	B0000 for 32 KB
1	1	B8000 for 32 KB

OE When set to 1, this bit directs the system address bit, A0, to be replaced by a higher-order bit. The odd map is then selected when A0 is 1, and the even map when A0 is 0.

GM This bit controls alphanumeric mode addressing. When set to 1, this bit selects graphics modes, which also disables the character generator latches.

A.3.4.9 Color Don't Care Register

Address 0x03CE; Data 0x03CF; Index 0x07

7	6	5	4	3	2	1	0
—	—	—	—	M3	M2	M1	M0

Color Don’t Care Register, Index 0x07

— : Set to 0, Undefined on Read
M3 : Compare Map 3
M2 : Compare Map 2
M1 : Compare Map 1
M0 : Compare Map 0

The register fields are defined as follows:

M* These bits select whether a map is going to participate in the color compare cycle. When the bit is set to 1, the bits in that map are compared.

A.3.4.10 Bit Mask Register

Address 0x03CE; Data 0x03CF; Index 0x08

7	6	5	4	3	2	1	0
Bit Mask							

When the bit is set to 1, the corresponding bit position in each map can be changed. When the bit set to 0, the bit position in the map is masked to prevent change, provided that the location being written was the last location read by the system microprocessor.

The bit mask applies to write modes 0 and 2. To preserve bits using the bit mask, data must be latched internally by reading the location. When data is written to preserve the bits, the most current data in the latches is written in those positions. The bit mask applies to all maps simultaneously.

A.3.5 Attribute Controller Registers

Each register for the attribute controller has two addresses. Address 0x03C0 is the write address and 0x03C1 is the read address. The individual data registers are selected by writing their index to the Address register (0x03C0).

Table 367. Attribute Controller Registers:

Register Name	Write Address	Read Address	Index
Address	0x03C0	0x03C0	—
Internal Palette	0x03C0	0x03C1	0x0-0x0F
Attribute Mode Control	0x03C0	0x03C1	0x10
Overscan Color	0x03C0	0x03C1	0x11
Color Plane Enable	0x03C0	0x03C1	0x12
Horizontal PEL Panning	0x03C0	0x03C1	0x13
Color Select	0x03C0	0x03C1	0x14

A.3.5.1 Address Register

This read/write register is at address **0x03C0**.

The attribute controller registers do not have an input bit to control selection of the address and data registers. An internal address flip-flop controls this selection. Reading Input Status Register 1 clears the flip-flop and selects the Address register.

After the Address register has been loaded with the index, the next Write operation to 03C0 loads the data register. The flip-flop toggles for each Write operation to address 0x03C0. It does not toggle for Read operations to 03C0 or 03C1.

7	6	5	4	3	2	1	0
—	—	IPAS	Index				

Address Register

— : Set to 0, Undefined on Read
IPAS : Internal Palette Address Source

The register fields are defined as follows:

IPAS — This bit is set to 0 to load color values to the registers in the internal palette. It is set to 1 for normal operation of the attribute controller.
Note: Do not access the internal palette while this bit is set to 1. While this bit is 1, the Type 1 video subsystem disables accesses to the palette; however, the Type 2 does not, and the actual color value addressed cannot be ensured.

Index — These bits contain the index to the data registers in the attribute controller.

A.3.5.2 Internal Palette Registers 0 through F

Write 0x03C0; Read 0x03C1; Index 0x00-0x0F

7	6	5	4	3	2	1	0
—	—	P5	P4	P3	P2	P1	P0

Internal Palette Registers

— : Set to 0, Undefined on Read
P5 to P0 : Palette Data

The register fields are defined as follows:

P5-P0 These 6-bit registers allow a dynamic mapping between the text attribute or graphic color input value and the display color on the CRT screen. When set to 1, this bit selects the appropriate color. The Internal Palette registers should be modified only during the vertical retrace interval to avoid problems with the displayed image. These internal palette values are sent off-chip to the video DAC, where they serve as addresses into the DAC registers.

Bit	***Color***
0	Blue
1	Green
2	Red
3	Secondary Blue
4	Secondary Green
5	Secondary Red

Software Implementation Note: These registers can be accessed only when bit 5 in the Address register is set to 0. When the bit is 1, writes are "don't care" and reads return undefined data.

A.3.5.3 Attribute Mode Control Register

Write 0x03C0; Read 0x03C1; Index 0x10.

7	6	5	4	3	2	1	0
PS	PW	PP	—	EB	ELG	ME	G

Attribute Mode Control Register

— : Set to 0, Undefined on Read
PS : P5, P4 Select
PW : PEL Width
PP : PEL Panning Compatibility
EB : Enable Blink/Select Background Intensity
ELG : Enable Line Graphics Character Code
ME : Mono Emulation
G : Graphics/Alphanumeric Mode

The register fields are defined as follows:

PS This bit selects the source for the P5 and P4 video bits that act as inputs to the video DAC. When this bit is set to 0, P5 and P4 are the outputs of the Internal Palette registers. When this bit is set to 1, P5 and P4 are bits 1 and 0 of the Color Select register.

PW When this bit is set to 1, the video data is sampled so that eight bits are available to select a color in the 256-color mode (0x13). This bit is set to 0 in all other modes.

PP When this bit is set to 1, a successful line-compare in the CRT controller forces the output of the PEL Panning register to 0 until a vertical synchronization occurs, at which time the output returns to its programmed value. This bit allows a selected portion of a screen to be panned.
When this bit is set to 0, line compare has no effect on the output of the PEL Panning register.

The register fields are defined as follows: ***(Continued)***

EB When this bit is set to 0, the most-significant bit of the attribute selects the background intensity (allows 16 colors for background). When set to 1, this bit enables blinking.

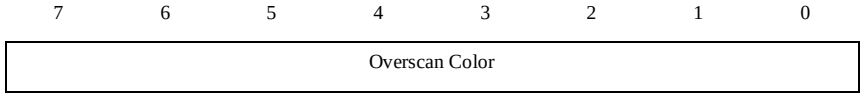
ELG When this bit is set to 0, the ninth dot will be the same as the background. When set to 1, this bit enables the special line-graphics character codes for the monochrome emulation mode. This emulation mode forces the ninth dot of a line graphic character to be identical to the eighth dot of the character. The line-graphics character codes for the monochrome emulation mode are 0xC0 through 0xDF. For character fonts that do not utilize these line-graphics character codes, bit 2 should be set to 0 to prevent unwanted video information from displaying on the CRT screen.
BIOS will set this bit, the correct dot clock, and other registers when the 9-dot alphanumeric mode is selected.

ME When this bit is set to 1, monochrome emulation mode is selected. When this bit is set to 0, color emulation mode is selected.

G When set to 1, this bit selects the graphics mode of operation.

A.3.5.4 Overscan Color Register

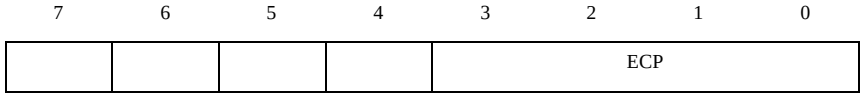
Write 0x03C0; Read 0x03C1; Index 0x11.



These bits select the border color used in the 80-column alphanumeric modes and in the graphics modes other than modes 4, 5, and D. (Selects a color from one of the DAC registers.)

A.3.5.5 Color Plane Enable Register

Write 0x03C0; Read 0x03C1; Index 0x12.



Color Plane Enable Register

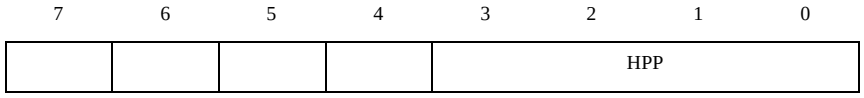
— : Set to 0, Undefined on Read
ECP : Enable Color Plane

The register field is defined as follows:

ECP Setting a bit to 1, enables the corresponding display-memory color plane.

A.3.5.6 Horizontal PEL Panning Register

Write 0x03C0; Read 0x03C1; Index 0x13.



Attribute Mode Control Register

— : Set to 0, Undefined on Read
HPP : Horizontal PEL Panning

The register field is defined as follows:

HPP These bits select the number of pels that the video data is shifted to the left. PEL panning is available in both alphanumeric and graphics modes. The following table shows the number of bits shifted for each mode.

Table 368. Image Shifting

Register Value	Number of Pels Shifted to the Left		
	Mode Hex 13	A/N Modes*	All Other Modes
0	0	1	0
1	—	2	1
2	1	3	2
3	—	4	3
4	2	5	4
5	—	6	5
6	3	7	6
7	—	8	7
8	—	0	—

* Only mode 7 and the A/N modes with 400 scan lines.

A.3.5.7 Color Select Register

Write 0x03C0; Read 0x03C1; Index 0x14.

7	6	5	4	3	2	1	0
—	—	—	—	SC7	SC6	SC5	SC4

Color Select Register

— : Set to 0, Undefined on Read
SC7 : S_color 7
SC6 : S_color 6
SC5 : S_color 5
SC4 : S_color 4

The register fields are defined as follows:

SC7, SC6 In modes other than mode 13 hex, these are the two most-significant bits of the 8-bit digital color value to the video DAC. In mode 13 hex, the 8-bit attribute is the digital color value to the video DAC. These bits are used to rapidly switch between sets of colors in the video DAC.

SC5, SC4 These bits can be used in place of the P4 and P5 bits from the Internal Palette registers to form the 8-bit digital color value to the video DAC. Selecting these bits is done in the Attribute Mode Control register (index 0x10). These bits are used to rapidly switch between colors sets within the video DAC.

A.3.6 Video Digital to Analog Converter

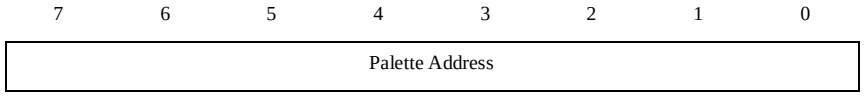
The video digital-to-analog converter (DAC) integrates the function of a color palette and three internal DACs for driving an analog display. The DAC has 256 registers containing 18 bits each to allow the display of up to 256 colors from a possible 256K colors.

Table 369. DAC Registers

Register Name	Read/ Write	Address
Palette Address (Write Mode)	R/W	0x03C8
Palette Address (Read Mode)	W	0x03C7
DAC State	R	0x03C7
Palette Data	R/W	0x03C9
Pel Mask	R	0x03C6

A.3.6.1 Palette Address Write Mode Register

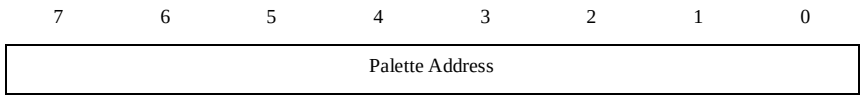
Read/Write Address 0x03C8



This register contains the 8-bit address used to access the 256 color registers during a write operation. Color data from the Palette Data Register is loaded into the palette registers in three separate output cycles per write operation. At the end of the third output to the Palette Data Register, the Palette Address Register will automatically increment.

A.3.6.2 Palette Address Read Mode Register

Write Address 0x03C7



This register contains the 8-bit address used to access the 256 color registers during a read operation. Color data from the palette registers is loaded into the Palette Data Register in three separate output cycles per write operation. At the end of the third output to the Palette Data Register, the Palette Address Register will automatically increment.

A.3.6.3 DAC State Register

Read Address 0x03C7

7	6	5	4	3	2	1	0
—	—	—	—	—	—	STA	

This is a read-only register that returns the last active operation in bits 1 and 0. If the last operation was a read operation, both bits are set to 1. If the last operation was a write, both bits are set to 0.

A.3.6.4 Palette Data Register

Read/Write Address 0x03C9

7	6	5	4	3	2	1	0
Palette Data							

The Palette Data Register is really 18 bits wide to correspond to the three 6-bit RGB representations in the palette registers. Since the system interface is 8 bits, three read/write operations are needed to access this register for each palette register.

A.3.6.5 Pel Mask Register

Read Address 0x03C6

7	6	5	4	3	2	1	0
Pel Mask							

This read only register is initialized by the mode set software and should not be further modified.

A.3.6.6 Device Operation

The palette address (P7 - P0) and the blanking input are sampled on the rising edge of the PEL clock. After three more PEL clock cycles, the video reflects the state of these inputs.

During normal operation the palette address is used as a pointer to one of the 256 data registers in the palette. The value in each data register is converted to an analog signal for each of the three outputs (red, green, blue). The blanking input is used to force the video output to 0 volts. The blanking operation is independent of the palette operation.

Each data register is 18 bits wide: 6 bits each for red, green, and blue. The data registers are accessible through the system interface.

A.3.6.7 Video DAC to System Interface

The Palette Address register holds an 8-bit value that is used to address a location within the video DAC. The Palette Address register responds to two addresses; the address depends on the type of palette access, Read or Write. Once the address is loaded, successive accesses to the data register automatically increment the address register.

For palette Write operations, the address for the Palette Address register is 0x03C8. A write cycle consists of writing three successive bytes to the Data register at address 0x03C9. The six least-significant bits of each byte are concatenated to form the 18-bit palette data. The order is red value first, then green, then blue.

For palette Read operations, the address for the Palette Address register is 0x03C7 (in the read mode, the Palette Address register is write only). A read cycle consists of reading three successive bytes from the Data register at address 0x03C9. The six least-significant bits of each byte contain the corresponding color value. The order is red value first, then green, then blue.

If the Palette Address register is written to during a Read or Write cycle, a new cycle is initialized and the unfinished cycle is terminated. The effects of writing to the Data register during a Read cycle or reading from the Data register during a Write cycle are undefined and can change the palette contents.

The DAC State register is a read-only register at address 0x03C7. Bits 1 and 0 return the last active operation to the DAC. If the last operation was a Read operation, both bits are set to 1. If the last operation was a Write, both bits are set to 0.

Reading the Read Palette Address register at 0x03C8 or the DAC State register at 0x03C7 does not interfere with read or write cycles.

A.3.6.8 Programming Considerations

As explained in “Video DAC to System Interface” on page 232, the effects of writing to the Data register during a read cycle or reading from the Data register during a write cycle are undefined and can change the palette contents. Therefore, the following sequence must be followed to ensure the integrity of the color palette during accesses to it:

1. 1. Disable interrupts.
2. 2. Write the address to PEL Address register.
3. 3. Write or read three bytes of data.
4. 4. Go to Step 2, repeat for the desired number of locations.
5. 5. Enable interrupts.

All accesses to the DAC registers are byte-wide l/O operations.

To prevent "snow" on the screen, an application reading data from or writing data to the DAC registers should ensure that the blank input to the DAC is asserted. This can be accomplished either by restricting data transfers to retrace intervals (use Input Status register 1 to determine when retrace is occurring) or by using the Screen Off bit located in the Clocking Mode register in the sequencer.

Do not write to the PEL Mask register (0x03C6). Palette information can be changed as a result. This register is correctly initialized to 0xFF during a mode set.

A.4 VGA Programming Considerations

The following are some programming considerations for the VGA:

1. The following rules must be followed to guarantee the critical timings necessary to ensure the proper operation of the CRT controller:
    a. The value in the Horizontal Total register must be at least 0x19.

b. The minimum positive pulse width of the 'horizontal synchronization' signal must be four character clock units.

c. The End Horizontal Retrace register must be programmed such that the 'horizontal synchronization' signal goes to 0 at least one character clock time before the 'horizontal display enable' signal goes active.

d. The End Vertical Blanking register must be set to a minimum of one horizontal scan line greater than the line-compare value.

2. When PEL panning compatibility is enabled in the Attribute Mode Control register, a successful line compare in the CRT controller forces the output of the Horizontal PEL Panning register to 0's until a vertical synchronization occurs. When the vertical synchronization occurs, the output returns to the programmed value. This allows the portion of the screen indicated by the Line Compare register to be operated on by the Horizontal PEL Panning register.

3. A write to the Character Map Select register becomes valid on the next whole character line. This will prevent deformed character images when changing character generators in the middle of a character scan line.

4. For mode 13 hex, the attribute controller is configured so that the 8-bit attribute in video memory becomes the 8-bit address (P0-P7) into the video DAC. The user should not modify the contents of the Internal Palette registers

5. The following is the sequence for accessing the attribute data registers:

   • 1. Disable interrupts.

   • 2. Reset the flip-flop for the Attribute Address register.

   • 3. Write the index.

   • 4. Access the data register.

   • 5. Enable interrupts.

6. • The Color Select register in the attribute controller section allows the programmer to rapidly switch color sets in the video DAC. Bit 7 of the Attribute Mode Control register controls the number of bits in the Color Select register used to address the color information in the video DAC (either two or four bits are used). By changing the value in the Color Select register, an application can switch color sets in graphics and alphanumeric modes (mode 13 hex does not use this feature).

   For multiple color sets, the user must load the color values.

7. An application that saves the video state must store the four bytes of information contained in the system microprocessor latches in the graphics controller subsection. These latches are loaded with 32 bits from video memory (8 bits per map) each time the system reads from video memory. The application needs to:

   a. Use write mode 1 to write the values in the latches to a location in video memory that is not part of the display buffer, such as the last location in the address range.

   b. Save the values of the latches by reading them back from video memory.

   If memory addressing is in the chain-4 or odd/even mode, reconfigure the memory as four sequential maps prior to performing the sequence above.

8. The Horizontal PEL Panning register allows programs to control the starting position of the display area on the screen. The display area can be shifted to the left up to eight PEL positions. In single-byte shift modes, to

pan to the PEL position above 8, the CRT controller start address is incremented and the PEL Panning register is reset to 0.

In multiple shift modes, the byte-panning bits (in the Reset Row Scan register) are used as extensions to the Horizontal PEL Panning register. This allows panning across the width of the video output. For example, in the 32-bit shift mode, the byte pan and pel-panning bits provide panning up to 31 bits. To pan from position 31 to 32, the CRT controller start address is incremented and the panning bits, both PEL and byte, are reset to 0.

Further panning can be accomplished by changing the start-address value in the CRT controller registers, Start Address High and Start Address Low. The sequence is:

1. Use the Horizontal PEL Panning register to shift the maximum number of bits to the left.
2. Increment the start address.
3. Set the Horizontal PEL Panning register so that no bits are shifted.

The screen will now be shifted one PEL to the left of the position it was in at the end of Step 1. Step 1 through Step 3 are repeated as often as necessary.

4. When using a split-screen application that scrolls a second screen on top of the first screen and operating in a mode with 200 scan lines, the Line Compare register (CRT Controller register 0x19) must contain an even value. This is a requirement of the double scanning logic in the CRT controller.
5. If the value in the Cursor Start register (CRT Controller register 0x0A) is greater than that in the Cursor End register (CRT Controller register 0x0B), the cursor is not displayed.
6. In 8-dot character modes, the underline attribute produces a solid line across adjacent characters. In 9-dot character modes, the underline across adjacent characters is dashed. In 9-dot modes with the line-graphics characters (C0 - DF character codes), the underline is solid.

A.4.1 Programming the Registers

Each of the video components has an address register and a number of data registers. The data registers have addresses common to all registers for that component. The individual registers are selected by a pointer (index) in its Address register. To write to a data register, the Address register is loaded with the index of the desired data register, then the data register is loaded by writing to the common l/O address.

The general registers do not share a common address; they each have their own I/O address.

See “Video DAC to System Interface” on page 232 for details on programming the video DAC.

For compatibility with the IBM Enhanced Graphics Adapter (EGA), the internal video subsystem palette is programmed the same as the EGA. Using BIOS to program the palette will produce a color compatible to that produced by the EGA. mode 13 hex (256 colors) is programmed so that the first 16 locations in the DAC produce compatible colors.

When BIOS is used to load the color palette for a color mode and a monochrome display is attached, the color palette is changed. The colors are summed to produce shades of gray that allow color applications to produce a readable screen.

Modifying the following bits must be done while the sequencer is held in a synchronous reset through its Reset register:

- Bits 3 and 0 of the Clocking Mode register.
- Bits 3 and 2 of the Miscellaneous Output register.

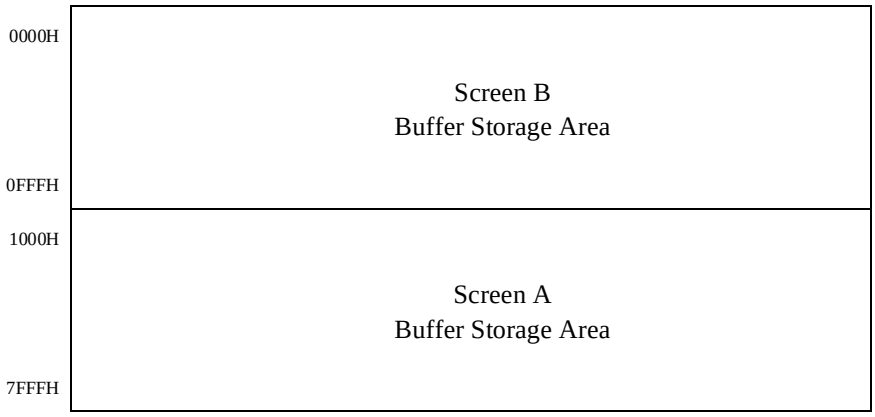
A.4.2 RAM Loadable Character Generator

The character generator is RAM loadable and can support characters up to 32 scan lines high. Three character fonts are stored in BIOS, and one is automatically loaded when an alphanumeric mode is selected. The Character Map Select register can be programmed to redefine the function of bit 3 of the attribute byte to be a character-font switch. This allows the user to select between any two character sets residing in map 2, and effectively gives the user access to 512 characters instead of 256. Character fonts can be loaded off-line, and up to eight fonts can be loaded at any one time. The character generator is in map 2 and must be protected using the map mask function.

A.4.3 Creating a Split Screen

The VGA hardware supports a split screen. The top portion of the screen is designated as screen A, and the bottom portion is designated as screen B.

The following figure shows the screen mapping for a system containing a 32KB alphanumeric storage buffer, such as the VGA. Information displayed on screen A is defined by the Start Address. High and Low registers of the CRT controller. Information displayed on screen B always begins at video address 0x0000.



The Line Compare register of the CRT controller performs the split screen function. The CRT controller has an internal horizontal scan line counter and logic that compares the counter value to the value in the Line Compare register and clears the memory address generator when a comparison occurs. The linear address generator then sequentially addresses the display buffer starting at location 0. Each subsequent row address is determined by the 16-bit addition of the start-of-line latch and the Offset register.

Screen B can be smoothly scrolled onto the display by updating the Line Compare register in synchronization with the 'vertical retrace' signal. Screen B information is not affected by scrolling operations that use the Start Address registers to scroll through the screen A information.

When pel-panning compatibility is enabled (Attribute Mode Control register), a successful line comparison forces the output of the Horizontal PEL Panning register to 0's until vertical synchronization occurs. This feature allows the information on screen B to remain unaffected by pel-panning operations on screen A.

Requirements Summary

B

This appendix lists all the requirements of this architecture. The requirements are collected under the chapter heading. This appendix may be used for a quick reference list of all requirements for building a standard platform. The requirements list follows:

Chapter 1 General Requirements and Information

1–1. The interfaces define in this document must not be accessed by RTAS or marked as *used-by-rtas* with two exception as follows:

a. Any RTAS service that does not return to the operating system may access any interface.

b. The interfaces in Chapter 13, “Graphics,” on page 161 may be accessed by the RTAS *display-character* function, if the graphics interface in the Open Firmware device tree node does not contain the *used-by-rtas* property, but is the device indicated by the *rtas-display-device* property.

1–2. If the implementation of the devices in this document have additional registers which are not part of the programming model given in this document, then these registers must not be required for correct operation of the device by the operating system and they must all be declared in the Open Firmware device tree to prevent other devices from being assigned to their addresses.

1–3. Any controller not contained on standard adaptor cards for their I/O bus must have the *builtin* property.

Chapter 2 ISA DMA Controller

2–1. An ISA DMA Controller must be compatible with the Intel 82378ZB System I/O (SIO), except as noted in the following requirements in this chapter.

2–2. This device (ISA DMA Controller) must be addressed only with ISA I/O addresses.

2–3. This device must be represented by the Open Firmware properties “name=dma-controller, device_type=dma-controller, compatible= chrp,dma”.

2–4. This device must have the Open Firmware properties defined in CHRP ISA DMA Controller Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware.

2–5. The registers shown in Table 2 on page 5, Register Map for ISA DMA Device, must all be present in the device defined in this chapter.

2–6. All the reg property fields specified in Table 2 on page 5 must comply with the format given in the ISA Bus Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware.

2–7. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

2–8. For ISA DMA, the DACK# signal must be active low.

2–9. For ISA DMA, the DREQ signal must be active high.

2–10. For ISA DMA, software must assume the terminal count (TC) is an output from the DMA controller. This means some ISA devices may not work if they depend on TC being an input to the DMA controller.

2–11. For ISA DMA, a hardware design must not produce an error if 0b0 is written to bit 6 of DCEM1 or DCEM2.

2–12. For operations involving ISA DMA slave devices, software must assume the hardware only allows addressing modes as follows:

a. Channels 0-3 are 8-bit count by bytes addressing mode. The DMA address, count, and page registers must be programmed accordingly.

b. Channels 5-7 are 16-bit count by word (2 bytes) addressing mode. The DMA address, count, and page registers must be programmed accordingly.

c. 16-bit count by bytes addressing mode is not supported.

2–13. For ISA DMA, a hardware design must not produce an error if 0b00 is written to bits [3:2] of DCEM1, or if 0b01 is written to bits [3:2] of DCEM2.

2–14. Software must assume the DMA Write All Mask Bits Registers (WAM1, WAM2) are write-only.

2–15. During Scatter/Gather operations, the High Page Registers (described in Section 2.4.13, “Current High Page Register 0 (CHIP0),” on page 13 through Section 2.4.14, “Current High Page Register 1-7 (CHIP1-CHIP7),” on page 14) and Low Page Registers (described in Section 2.4.9, “Current Low Page Register 0 (CLOP0),” on page 12 through Section 2.4.10, “Current Low Page Register 1-7 (CLOP1-CLOP7),” on page 13) must increment appropriately as part of the full 32 bit transfer address.

2–16. Scatter/Gather Descriptor: The Memory Buffer Starting Physical Address field in the SGD must be programmed as follows:

a. For 8-bit count by byte operations, this field is programmed with the byte address of the buffer to be transferred. The address may be even or odd.

b. For 16-bit count by word operations, this field is programmed with the even byte address of the buffer to be transferred. Thus, bit 0 in this field will be 0.

2–17. IRQ 13 must be reserved for the ISA DMA device.

2–18. The Scatter/Gather Descriptor Table Pointer Registers must be programmed with a single 32-bit write.

2–19. The ISA DMA Controller must not preclude an ISA bus master from using any of the seven available DMA channels, regardless of addressing mode.

2–20. The ISA DMA hardware must provide at least 24-bit addressing to system memory for ISA bus masters.

Chapter 3 Floppy Disk Controller

3–1. A Floppy Disk Controller that is compliant with the interfaces as defined in this section must be represented in the Open Firmware device tree for the auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=chrp,fdc. For the non-auto-eject version of the device, it is represented by the following properties; name=fdc, device_type=fdc, and compatible=pnpPNP,700.

3–2. The firmware must configure the Floppy Disk Controller in the PS/2™ mode, with enhanced auto-eject or without auto-eject.

3–3. This device must be addressed purely with ISA I/O addresses which can be aliased or non-aliased.

3–4. DMA transfers for the floppy disk controller must be 8-bit DMA slave and must support at least the ISA Compatible Timing mode.

3–5. Interrupts generated by the floppy disk controller must be high true edge sensitive.

3–6. The floppy disk controller must support 3.5 inch media with an unformatted capacity of 2 megabytes and a formatted capacity of 1.44 megabytes.

3–7. The floppy disk controller must support the 3.5 inch media with an unformatted capacity of 4 megabytes and a formatted capacity of 2.88 megabytes for purposes of floppy disk port tape backup units.

3–8. The three sets of registers shown in Table 52 on page 33 must be present in the Open Firmware device tree for the auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=chrp,fdc.

3–9. The first two sets of registers shown in Table 52 on page 33 must be present in the Open Firmware device tree for the non-auto-eject version of the device as identified by the following properties; name=fdc, device_type=fdc, and compatible=pnpPNP,700.

3–10. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

3–11. A floppy disk controller must be reported in the Open Firmware device tree as specified in the *CHRP ISA Floppy Device Binding* [16].

3–12. For the device described herein, all the registers defined in Section 3.4.1, “Status Register A (SRA),” on page 33 through Section 3.4.10, “Autoeject Register (AEJ),” on page 41 must be implemented as described herein.

3–13. For the device described herein, the commands described in Section 3.5.1.1, “Configure Command,” on page 43 through Section 3.5.1.22, “Write Deleted Data Command,” on page 55 must be implemented as specified.

3–14. For the device described herein, the Command Status Registers described in Section 3.5.2.1, “Status Register 0 (ST0),” on page 56 through Section 3.5.2.4, “Status Register 3 (ST3),” on

page 58 must be implemented as specified.

3–15. The disk drive must be configured in a manner such that the Disk in Place signal, on the drive, is functionally a state indicator of the presence of media. The signal must assert an active level when a disk is present in the drive and an inactive level when there is no disk in the drive.

Chapter 4 Legacy Interrupt Controller

4–1. The Legacy Interrupt Controller must be represented by the Open Firmware properties “name=interrupt-controller, device_type=interrupt-controller, compatible=chrp,iic”.

4–2. The Legacy Interrupt Controller registers specified herein must be implemented precisely as defined, including the following characteristics and bit definitions:

a. Table 129, “Initialization Command Word 1 Register (ICW1) Characteristics,” on page 68

b. Table 130, “Initialization Command Word 1 Register (ICW1),” on page 68

c. Table 131, “Initialization Command Word 2 Register (ICW2) Characteristics,” on page 68

d. Table 132, “Initialization Command Word 2 Register (ICW2),” on page 68

e. Table 133, “Initialization Command Word 3 Register (ICW3) Characteristics,” on page 69.

f. Table 134, “Initialization Command Word 3 Register (ICW3),” on page 69

g. Table 135, “Initialization Command Word 4 Register (ICW4) Characteristics,” on page 69

h. Table 136, “Initialization Command Word 4 Register (ICW4),” on page 70

i. Table 137, “Operation Command Word 1 Register (OCW1) Characteristics,” on page 70

j. Table 138, “Operation Command Word 1 Register (OCW1),” on page 70

k. Table 140, “Operation Command Word 2 Register (OCW2) Characteristics,” on page 71

l. Table 141, “Operation Command Word 2 Register (OCW2),” on page 71

m. Table 144, “Operation Command Word 3Register (OCW3) Characteristics,” on page 72

n. Table 145, “Operation Command Word 3 Register (OCW3),” on page 73

o. Table 149, “Interrupt Request Register (IRR) Characteristics,” on page 74

p. Table 143, “Interrupt Level Select Encodings,” on page 72

q. Table 151, “In-Service Register (ISR) Characteristics,” on page 75

r. Table 143, “Interrupt Level Select Encodings,” on page 72

s. Table 153, “Edge/Level Interrupt Control Register 1 (ELI1) Characteristics,” on page 76

t. Table 154, “Edge/Level Interrupt Control 1 Register (ELI1),” on page 76

u. Table 155, “Edge/Level Interrupt Control 2 Register (ELI2) Characteristics,” on page 76

v. Table 156, “Edge/Level Interrupt Control 2 Register (ELI2),” on page 76

4–3. This device must be addressed purely with ISA I/O addresses.

4–4. The registers shown in Table 125 on page 63 must all be present in the device defined in this chapter. The Open Firmware “reg” property information given in Table 125 on page 63 must be provided in the Open Firmware device tree for this device.

4–5. This device must be represented as an ISA device and a child of the ISA bridge node in the Open Firmware device tree.

4–6. This device must be represented in the OF device tree as specified in the *CHRP ISA Interrupt Controller Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware.*

4–7. All the reg property fields specified in table Table 125 on page 63 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.

4–8. The following events must occur during the OF initialization sequence:

a. The level sense circuit must be set to the Edge-sensitive mode (following the initialization procedure, interrupts are generated by a low-to-high transition high level on the interrupt request input.)

b. IRQ7 must be assigned priority 7.

c. PIC2 (slave) mode address must be set to 7.

d. The Special Mask mode must be cleared and the controller must be set to read the Interrupt Request register.

e. All Legacy Interrupt Controller registers must be initialized to the values indicated in the “Value in Register When OF Passes Control to OS” column of the characteristics table corresponding to the specific register.

4–9. PIC1 (master) interrupt controller must be initialized before PIC2 (slave) interrupt controller. Failure to do so will cause unexpected results.

4–10. The Interrupt Controller must respond to a Non-specific EOI command by resetting the highest in-service bit of those set.

4–11. The Interrupt Controller must respond to a Specific EOI command by resetting the in-service bit that corresponds to the one specified by the command, except that an in-service bit masked by an Interrupt Mask register bit cannot be reset by a Non-specific EOI command when in the Special Mask mode.

4–12. The Interrupt Vector which is returned by the Interrupt Controller on an Interrupt Acknowledge cycle must be as defined in Table 126 on page 65.

4–13. For the device described herein, all the registers defined in sections 4.4.1.1 through 4.4.1.4 must be implemented precisely as described herein.

4–14. During the initialization sequence ICW1 must be written before writing ICW2, ICW2 must be written before writing ICW3, and ICW3 must be written before writing ICW4.

4–15. For the device described herein, all the registers defined in sections 4.4.2.1 through 4.4.4 must be implemented precisely as described herein.

4–16. Writing of OCW1 must be performed before the OCW2 and OCW3 registers are accessed, and

following the ICW1-4 initialization sequence.

4–17. When an in-service bit is set to 1, the Interrupt Controller must inhibit all other interrupts with the same or lower priority.

4–18. When an in-service bit is set to 1, the Interrupt Controller must allow interrupts with a higher priority to cause an interrupt.

4–19. For the device described herein, all the registers defined in sections 4.4.5.1 through 4.4.5.2 must be implemented precisely as described herein.

Chapter 5 Parallel Port Controller

5–1. The Parallel Port Controller must be represented by the Open Firmware properties “name=parallel, device_type=parallel, compatible=chrp,ecp”. The compatible property may also specify the compatible property for the National Semiconductor 87308 in addition to “chrp,ecp”.

5–2. The Parallel Port Controller must protect against damage to the Parallel Port Controller electronics in the case where the device attached to the Parallel Port has its power on and the Parallel Port Controller has its power off.

5–3. The device described herein must support the following modes as defined by the IEEE 1284 Specification: Compatibility mode, Nibble mode (Compatibility mode along with Nibble mode are called Standard Parallel Port, SPP, in this specification), Byte mode (called SPP Extended mode in this specification), Enhanced Parallel Port (EPP) mode (version 1.9), and Extended Capabilities Port (ECP) mode.

5–4. The device described herein must also support the following modes as described herein: Parallel Port FIFO mode and Test FIFO mode.

5–5. This device must be addressed purely with ISA I/O addresses which can be aliased or non-aliased.

5–6. DMA transfers fro the Parallel Port Controller must be 8-bit DMA slave and must support at least the ISA Compatible Timing mode.

5–7. Interrupts generated by the Parallel Port Controller must be high true edge sensitive.

5–8. The Parallel Port Controller registers specified herein must be implemented precisely as defined, including the following characteristics and bit definitions:

a. Table 160, “Data Register (DTR) Characteristics,” on page 84

b. Table 161, “Data Register (DTR),” on page 84

c. Table 162, “Status Register (STR or DSR) Characteristics,” on page 85

d. Table 163, “Status Register (STR or DSR),” on page 85

e. Table 164, “Control Register (CTR or DCR) Characteristics,” on page 85

f. Table 165, “Control Register (CTR or DCR),” on page 86

g. Table 166, “Extended Control Register (ECR) Characteristics,” on page 87

h. Table 167, “Extended Control Register (ECR),” on page 87

i. Table 168, “EPP Address Register (ADDR) Characteristics,” on page 88

j. Table 169, “EPP Data Port 0 Register (DTR) Characteristics,” on page 88

k. Table 170, “EPP Data Port 1 Register (DATA1) Characteristics,” on page 88

l. Table 171, “EPP Data Port 2 Register (DATA2) Characteristics,” on page 89

m. Table 172, “EPP Data Port 3 Register (DATA1) Characteristics,” on page 89

n. Table 173, “ECP Address FIFO (AFIFO) Characteristics,” on page 90

o. Table 174, “Address FIFO Register (AFIFO),” on page 90

p. Table 175, “ECP Data FIFO Register (DFIFO) Characteristics,” on page 90

q. Table 176, “ECP Data FIFO Register (DFIFO),” on page 90

r. Table 177, “Parallel Port FIFO Register (CFIFO) Characteristics,” on page 91

s. Table 178, “Parallel Port FIFO Register (CFIFO),” on page 91

t. Table 179, “Test FIFO Register (TFIFO) Characteristics,” on page 91

u. Table 180, “Test FIFO Register, TFIFO,” on page 91

5–9. The Open Firmware “reg” property information given in Table 159 on page 83 must be provided in the Open Firmware device tree for this device.

5–10. This device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

5–11. This device must be represented in the OF device tree as specified in the *CHRP ISA Parallel Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*. [23]

5–12. All the reg property fields specified in Table 159 on page 83 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.

5–13. The following events must occur during the OF initialization sequence:

a. The Parallel Port Controller must be enabled and activated.

b. If there is I/O Range checking of the Parallel Port Controller it must be disabled.

c. The parallel Port Controller address space must be set up, enabled.

d. The interrupt level and type (edge versus level) must be setup, enabled, and reported in the OF device tree.

e. Eight-bit DMA for the Parallel Port Controller must be: set up, enabled, and reported in the OF device tree.

f. Demand DMA must be enabled

g. Both ECP and EPP 1.9 modes must be enabled.

h. The Parallel Port Clock must be enabled (that is, clock must be operational).

i. The Parallel Port Controller must be set up to disable its outputs when the Parallel Port is disabled.

j. All Parallel Port Controller registers must be initialized to the values indicated in the “Value in Register When OF Passes Control to OS” column of the characteristics table corresponding to the specific

register.

5–14. Writing a Parallel Port device address to this register must initiate an EPP device/register selection operation.

5–15. Accesses to this register must initiate device read or write operations to bits 7-0 of the 32 bit word to be transferred.

5–16. Accesses to this register must initiate device read or write operations to bits 15-8 of the 32 bit word to be transferred.

5–17. Accesses to this register must initiate device read or write operations to bits 23-16 of the 32 bit word to be transferred.

5–18. Accesses to this register must initiate device read or write operations to bits 31-24 of the 32 bit word to be transferred.

Chapter 6 UART Controller

6–1. The UART serial device must be NS16550 compatible, as specified in this chapter.

6–2. The UART serial device must be addressed only with ISA I/O addresses.

6–3. The UART serial device must generate only low-to-high edge sensitive interrupts.

6–4. The Open Firmware device tree node for the UART serial device described in this chapter must follow the specifications given in the CHRP ISA Serial Port Device Binding [28].

6–5. The Open Firmware “reg” property information given in Table 182, “UART Registers,” on page 94 must be provided in the Open Firmware device tree for the UART serial device.

6–6. The UART serial device must be represented as an ISA device and a child of an ISA bridge node in the Open Firmware device tree.

6–7. All the “reg” property fields specified in Table 182, “UART Registers,” on page 94 must comply with the format given in the ISA/EISA/ISA-PnP binding to IEEE Std 1275-1994, as modified by the CHRP binding to IEEE Std 1275-1994.

6–8. The UART serial device must precisely implement the registers defined in Section 6.3, “UART Registers,” on page 94.

6–9. The selection of the TDR/IER versus the DLL/DLH for the UART serial device must be exclusively controlled by the DLAB bit (bit 7) in the Line Control Register.

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.

Chapter 7 ISA Keyboard/Mouse Controller

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.

7–5. 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].

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.

Chapter 8 Audio

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.

Chapter 9 ESCC

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.

9–4. 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 .

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.

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.

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.

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.

9–21. 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.

Chapter 10 Apple Desktop Bus Controller

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

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.

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

10–8. 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.

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

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.

10–14. The ADB controller shall load the command that was sent into the CDRF, shall set the NRE error in the sta-

tus register, and shall set the DFB bit in the interrupt source register.

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.

10–17. 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.

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).

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.

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

Chapter 11 IDE Drive Controller

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.

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.

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.

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.

Chapter 12 SCSI

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.

12–2. 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.

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. 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.

12–6. For the device described herein, all the registers and their associated function defined in Appendix A must be implemented precisely as described.

Chapter 13 Graphics

13–1. Graphics controllers must provide both a Big Endian (BE) and a Little Endian (LE) interface to

the frame buffer.

13–2. CHRP platforms must minimally support a 640x480 linear frame buffer with the Open Firmware characteristics and interfaces described in the CHRP Linear Frame Buffer Device Binding.

13–3. VGA compatibility is optional; if VGA compatibility is provided, it must adhere to the programming model given in Appendix A, “VGA Programming Model,” on page 195.

13–4. If VGA compatibility is provided, it must conform with the characteristics and interfaces described in the CHRP VGA Display Device Binding.

13–5. If VGA compatibility is provided, the display node of the Open Firmware device tree must indicate a compatible property of “pnpPNP,900”. Conversely, if the compatibility property of “pnpPNP,900” is indicated in the display node of the Open Firmware device tree, the programming model described in Appendix A, “VGA Programming Model,” on page 195 must be present.

Chapter 14 Versatile Interface Adapter (VIA)

14–1. The Open Firmware device tree node for the device described in this chapter must follow the specifications given in the Mac I/O [14] Open Firmware device binding.

14–2. The Open Firmware “reg” property information given in Table 311 on page 167 must be provided in the Open Firmware device tree for the VIA device.

14–3. New capabilities in operating systems must not depend in any way on the continued existence of the facilities described in the VIA chapter.

14–4. New software must not utilize the timing characteristics associated with operations on the VIA hardware as described in Section 14.3.2, “Internal Timings,” on page 166 of the VIA chapter.

14–5. All the “reg” property fields specified in Table 311 on page 167, “VIA Registers”, must comply with the format given in the PCI binding to IEEE Std 1275-1994, as modified by the CHRP binding to IEEE Std 1275-1994.

14–6. Software must not write to the Peripheral Control Register in the VIA hardware.

Chapter 15 Descriptor-Based DMA

15–1. Every DBDMA reserved field must be 0 when written and must be ignored when read. This must be true whether the reserved field is accessed by the DBDMA hardware or by device driver software.

15–2. Some address pointers are constrained to be multiples of 4, 8, or 16 bytes. For example, the NewCommandPtr value is always 16-byte aligned. The least-significant bits of these aligned addresses must be reserved, to ensure deterministic behavior when unaligned address values are used.

15–3. The PCI devices described herein are allowed one interrupt each to the system. DBDMA interrupts must be shared with the other interrupt requirements of the device as described in the device chapters.

Glossary

This glossary contains an alphabetical list of terms, phrases, and abbreviations used in this document.

Term	Definition
µs	Microsecond or one millionth of a second
ACD	Active device register
ADB	Apple Desktop Bus
ADC	Analog-to-digital converter
ADDR	Address register
AEJ	Autoeject register
AFIFO	Address first in - first out register
A/N	Alphanumeric
ANSI	American National Standards Institute
APA	All points addressable
Architecture	The hardware/software interface definition or software module to software module interface definition.
ASCII	American National Standards Code for Information Interchange
BA	Base address register
BC	Base count register
BE	Big Endian
BIOS	Basic input/output system
BLOP	Base low page register
CA	Current address register
CBP	Clear byte pointer register
CC	Current count register
CCR	Configuration control register
CDRF	Command/data register file

CHIP	Current high page register
CHRP	Common Hardware Reference Platform
CLOP	Current low page register
CMD	Command register
CRT	Cathode ray tube which normally refers to a computer monitor
CTL	Control register
CTR	Control register
DAC	Digital-to-analog converter
DAT	Data register
DATA	EPP data port register
DATAR	Data register
DBDMA	Descriptor based direct memory access
DCM	DMA channel mode register or DMA clear mark register
DCEM	DMA extended mode register
DCOM	DMA command register
DCR	Control register
DET	Detect register
DFIFO	Data first in - first out register
DIR	Digital input register
DLH	Baud rate divisor high order register
DLL	Baud rate divisor low order register
DMA	Direct memory access
DMC	DMA master clear register
DOR	Digital Output Register
DR	DMA request register
DRS	Data rate select register
DS	DMA status register
DSR	Status register
DTC	Data type count register
DTR	Data register
ECP	Extended capability parallel port
ECR	Extended control register
EDR	Extended data register

EIR	Extended index register
ELI	Edge level interrupt control register
ENH	Enhanced register
EOL	End of line
EPP	Enhanced parallel port
ESR	Error status register
ESSC	Extended serial communications controller
FCR	First in - first out control register
FIFO	First in - first out or bus first in - first out register
FifoCount	FIFO count register
GB	Gigabytes - as used in this document it is 2 raised to the power of 30
Hz	Hertz
ICW	Initialization control word
ID	Identification
IDE	Integrated device electronics
IEEE	Institute of Electrical and Electronics Engineers
IER	Interrupt enable register
IIR	Interrupt identification register
I/O	Input/output
IN	Input register
IrDA	Infrared Data association which sets standards for infrared support including protocols for data interchange
IRQ	Interrupt request
IRR	Interrupt request register
ISA	Industry standard architecture (typically refers to the PC local bus)
ISE	Interrupt source enable register
ISO	International Organization of Standards
ISR	In-service register
IST	Interrupt status register
KB	Kilobytes - as used in this document it is 2 raised to the power of 10
KHz	Kilo Hertz
LCD	Liquid crystal display
LCR	Line control register
LE	Little Endian

LFB	Linear frame buffer
lsb	Least significant bit
LSR	Line status register
LTPC	LocalTalk protocol controller
MCR	Modem control register
MSR	Main status register or Modem status register
MB	Megabytes - as used in this document it is 2 raised to the power of 20
MESH	Macintosh enhanced SCSI hardware
MeshFeature	MESH features register
MeshID	MESH identification register
ms	Milliseconds or thousandths of a second
N/A	Not applicable
ns	Nanoseconds or billionths of a second
OCW	Operation control word
OF	Open Firmware
OS	Operating system
OUT	Output register
PC	Personal computer
PC Card	A memory or I/O card compatible with the *PC Card Standard* []. When cards are referred to as PC Cards, what is being addressed are those characteristics common to both 16-bit PC Cards and CardBus PC Cards.
PCI	Peripheral Component Interconnect
PCMCIA	Personal Computer Memory Card International Association (see PC Card)
PCR	Primary command register
PEL	Picture element
PIC	Programmable interrupt controller
Platform	Refers to the hardware plus firmware portion of a system composed of hardware, firmware, and operating system.
PPRD	Primary PRD table address register
PRD	Physical region descriptor
PSR	Primary status register
RAM	Random access memory
RC	Recovery count register
RDR	Receiver data register

Reserved	The term “reserved” is used within this document to refer to bits in registers or areas in address spaces which should not be referenced by software except as described in this document. Normally when writing other bits in a register with reserved bits, the register should be read, the non reserved bits set, and the register should be written.
RMSK	Receive mask register
ROM	Read only memory
SCC	Serial communications controller
SCDR	Special character detect register
SCR	Scratch control register or secondary command register
SCSI	Small computer system interface
SelTO	Selection time out register
SGC	Scatter/gather command register
SGD	Scatter gather descriptor
SGIS	Scatter/gather interrupt status register
SGPTR	Scatter/gather descriptor table pointer register
SGS	Scatter/gather status register
SP	Special character register
SPP	Standard parallel port
SPRD	Secondary PRD table address register
SR	Status register
SRQ	Service request
SSR	Secondary status register
ST	Status register
STA	Status register
STR	Status register
STRT	Start register
SyncParms	Synchronous parameters register
System	Refers to the collection of hardware, system firmware, and operating system software which comprise a computer model.
System firmware	Refers to the collection of all firmware on a system including Open Firmware, RTAS, and any legacy firmware.
System software	Refers to the combination of operating system software, device driver software, and any hardware abstraction software, but excludes the application software.
TC	Terminal count register

TDR	Table drive register or Transmitter data register
TMR	Timer register
UART	Universal asynchronous receiver/transmitter
VESA	Video Electronics Standards Association
VGA	Video graphics array
VIA	Versatile interface adapter
WAM	Write all mask bits register
WSM	Write single bit mask register
x86	Intel processors such as 80386, 80486,...
XferCount	Transfer count register

Trademark Information

The following terms, denoted by a registration symbol (®) or trademark symbol (™) on the first occurrence in this publication, are registered trademarks or trademarks of the companies as shown in the list below:

Trademark	Company
Apple Desktop Bus	Apple Computer, Inc.
AIX	International Business Machines Corporation
Apple	Apple Computer, Inc.
CHRP	Apple Computer, Inc.
Ethernet	Xerox
GeoPort	Apple Computer, Inc.
IBM	International Business Machines Corporation
LocalTalk	Apple Computer, Inc.
Macintosh	Apple Computer, Inc.
Mac	Apple Computer, Inc.
Motorola	Motorola, Inc.
PowerPC	International Business Machines Corporation
PS/2	International Business Machines Corporation
Solaris	SunSoft
Windows	Microsoft Corporation
Windows NT	Microsoft Corporation

Bibliography

This section lists documents which were referenced in this specification or which provide additional information. Following this list are two sections. The first section gives useful information for obtaining these documents. The second section lists contacts and sources for additional helpful information. The documents are listed below:

1. *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture,* Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-394-8
2. *The PowerPC Architecture: A Specification for a New Family of RISC Processors,* Second Edition, Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-316-6
3. *Computer Architecture: A Quantitative Approach,* Second Edition Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-329-8
4. *PowerPC 603 RISC Microprocessor User's Manual*, IBM order number MPR603UMU-01, Motorola order number MPC603UM/AD
5. *PowerPC 603 RISC Microprocessor Technical Summary,* Rev 3 IBM order number MPR603TSU-03, Motorola order number MPC603/D
6. *PowerPC 604 RISC Microprocessor User's Manual*, IBM order number MPR604UMU-01, Motorola order number MPC604UM/AD
7. *PowerPC 604 RISC Microprocessor Technical Summary,* Rev 1 IBM order number MPR604TSU-02, Motorola order number MPC604/D
8. *Technical Introduction to the Macintosh Family,* Second Edition, Addison-Wesley Publishing Company, Reading, MA, ISBN 0-201-62215-7
9. IEEE 1275, *IEEE Standard for Boot (Initialization Configuration) Firmware, Core Requirements and Practices,* IEEE part number DS02683, ISBN 1-55937-426-8
10. *PCI Local Bus Specification*, Revision 2.1
11. *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference*, available at various electronic sites
12. *PowerPC Reference Platform Specification*, Version 1.1
13. *ANSI X3.131-199x Rev.10L,1992 (SCSI-2): Information Technology - Small Computer System Interface*
14. *Mac I/O* binding for Open Firmware
15. *CHRP ISA Bus Binding to: IEEE Standard 1275-1994*
16. *CHRP ISA Floppy Device Bindingto: IEEE Standard 1275-1994*

17. *Inside Macintosh: Devices*

18. *Macintosh Technology in the Common Hardware Reference Platform*, Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-393-X

19. *ADB-The Untold Story* available at http://dev.info.apple.com/technotes/Archive/Hardware/hw_01.html

20. *Intel 82378ZB System I/O (SIO) Specification*

21. *SL82C565 System I/O Controller With PCI Arbiter*, from Winbond Systems Laboratory

22. *CHRP ISA DMA Controller Device Binding to IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

23. *CHRP ISA Parallel Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

24. *National Semiconductor PC87308VUL SuperI/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 IEEE1284 Parallel Port*

25. *IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers*, also called *IEEE Std 1284-1994 Specification*

26. *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*

27. *IEEE Std 1275-1994, IEEE Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices*, published by IEEE.

28. *CHRP ISA Serial Port Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

29. *ANSI ATA Specification Revision 3.2*

30. *ANSI ATAPI Specification (Current Revision?)*

31. *SFF IDE PCI Bus Master Specification??*

32. *PCI Sign IDE PCI Bus Master Specification??*

33. *Open Firmware Bindings for SCSI devices/bus????*

34. *Open Firmware Recommended Practice: 8-bit Graphics Extension*, published by the Open Firmware Working Group

35. *Open Firmware Recommended Practice: 16-color Text Extension*, published by the Open Firmware Working Group

36. *CHRP Linear Frame Buffer Display Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

37. *CHRP VGA Display Device Binding to: IEEE 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

38. *Personal Computer Standard V2.0 (SurePath BIOS)*

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

Sources for Documents

This section provides useful information for obtaining the documents listed above.

The manuals for the PowerPC microprocessors or the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* are available from the following sources:

- IBM at 1-800-PowerPC (1-800-769-3772) in the U.S.A
  - If the 1-800-PowerPC number can not be reached or if multilingual operators are required, use 1-708-296-6767
- IBM at (39)-39-600-4455 in Europe
- Motorola at 1-800-845-MOTO (6686)

The *PowerPC Reference Platform Specification* in PostScript format is available via anonymous FTP at ftp://ftp.austin.ibm.com/pub/technology/spec and on CompuServe in the library of the PowerPC forum. The FTP server contains a README file which list the information at this directory.

The *PowerPC Reference Platform Specification* and *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* are available in Europe electronically from a bulletin board service (BBS). The BBS may be reached at +39-39-600-5076 and supports up to 28.8 KBPS.

The documents published by Morgan Kaufmann Publishers may be purchased in bookstores and may be ordered from Morgan Kaufmann at 1-800-745-7323 or through their home page at http://www.mkp.com. The PowerPC versions may be available from IBM publications sources at 1-800-426-6477 in the US only. In other countries, order from your local source for IBM literature.

PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture is available in html format from http://www.austin.ibm.com/tech/chrp/chrp_book.htlm and in Acrobat viewable PDF format from http://chrp.apple.com.

The approved Open Firmware bindings including *Open Firmware Recommended Practice: 16-color Text Extension* and *Open Firmware Recommended Practice: 8-bit Graphics Extension* are available via anonymous FTP to ftp://playground.sun.com/pub/p1275/bindings/postscript/*.ps.

Call 1-212-642-4900 for orders or inquiries pertaining to ANSI and ISO standards.

To order copies of EIA standards, contact Global Engineering Documents at 1-800-854-7179.

Copies of IEEE standards may be obtained by calling 1-800-678-IEEE. For information about IEEE standards, call 908-562-3800.

To purchase PCI documents, call 1-800-433-5177 in the U.S.A. or 1-503-797-4207 outside the U.S.A.

Copies of PCMCIA standards including the *PC Card Standard* may be obtained by calling PCMIA at 1-408-720-0107 or Fax to 1-408-720-9416 or by calling JEIDA at +81-3-3433-1923 or Fax to +81-3-3433-6350.

Plug and Play information is available on CompuServe Plug and Play forum (GO PLUGPLAY).

Obtaining Additional Information

Several sources exist for obtaining additional information about the PowerPC microprocessor and the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture*.

Hardware system vendors may obtain information on IBM components or the IBM design kits which give further information on their reference implementation by contacting IBM at the following numbers:

- IBM at 1-800-PowerPC (1-800-769-3772) in the U.S.A
- Within Europe (49)-511-516-3444 in English
- Within Europe (49)-511-516-3555 in German
- In Asia (81)-775-87-4745 in Japanese

Updates to the *The PowerPC Architecture: A Specification for a New Family of RISC Processors* are available at http://www.austin.ibm.com/tech/ppc-chg.html.

The curernt version and updates to *The PowerPC Microprocessor Common Hardware Reference Processor: A System Architecture* and *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference* are available via anonymous FTP from ftp://ftp.austin.ibm.com/pub/technology/spec/chrp.

OEMs, IHVs, and ISVs may obtain information on Motorola products and Motorola system design kits by contacting their local Motorola sales office or 1-800-845-MOTO (6686).

Information on the full range of Motorola's semiconductor, software, design kits, and system products may be found at Motorola's PowerPC World Wide Web home page located at http://www.mot.com/PowerPC/.

Information on the full range of IBM products may be found at IBM's World Wide Web home page located at http://www.ibm.com/.

Information on the full range of Apple products may be found at Apple's World Wide Web home page located at http://www.apple.com/.

An electronic forum on CompuServe has been established for the discussion of PowerPC reference platform topics and for obtaining answers to questions on the PowerPC reference platform. Go to the "PowerPC" forum on CompuServe and join the "Reference Platform" topic.

Information on IrDA documents is available on the World Wide Web at ftp://hplose.hpl.hp.com.

VESA documents including the DDC 1 standard are available on the World Wide Web at http://www.vesa.org.

Index