Bi-Endian Designs

C

This appendix discusses ways in which platforms can meet the requirement of operating in both of the PowerPC endian modes, as required by the CHRP architecture. Current PowerPC processors assume that storage is Big-Endian when the processor is in Big-Endian mode or PowerPC Little-Endian when the processor is in Little-Endian mode. Therefore, when the processor is in Little-Endian mode, a translation of Little-Endian data must be made somewhere in the platform to PowerPC Little-Endian before the data reaches the PowerPC processor. Later sections discuss the design of graphics adaptors which support both Big-Endian and Little-Endian accesses and the effect on platform designs if the processor architecture changes.

C.1 Little-Endian Address and Data Translation

In order for PowerPC processors running in Little-Endian mode to correctly access an object in Little-Endian-organized storage, the object must have its bytes show up in the correct processor byte lanes, and the Big-Endian address must be changed to refer to the correct location (after the object has had its byte lanes reordered). This translation puts the data into PowerPC Little-Endian format and has three parts.

The first part of the translation achieves address conversion and is done by the PowerPC processor. It is summarized below (a more detailed discussion can be found in Book I, Appendix D of *The PowerPC Architecture* [1]).

The PowerPC architecture defines two status bits that determine whether a PowerPC processor uses Big-Endian or Little-Endian modes. The endian mode of the kernel and the endianess of the current operating mode are recorded in

two status bits. When Little-Endian mode is enabled, the effective address (EA) is modified in the PowerPC processor as shown in Table 64 on page 266 before it is used to reference memory.

Table 64. Address Modification for Little-Endian Mode

Data Access	EA Modifier
Byte	XOR with 0b111
Halfword	XOR with 0b110
Word	XOR with 0b100
Doubleword	(no change)

This address modification results in correct Little-Endian addresses being presented to memory for aligned accesses (as will become clearer in the example below). The PowerPC architecture allows unaligned accesses in Little-Endian mode to interrupt, so incorrect memory references will not be generated if the processor does not support them. One subtlety here is that string operations and load and store multiple are considered unaligned accesses, and thus may interrupt in Little-Endian mode also. There is also a transformation which can modify the EA to support unaligned accesses in Little-Endian mode, but it is beyond the scope of this discussion.

The second part of the translation ensures that the bytes of the object addressed show up on the correct byte lanes of the processor. In Little-Endian mode, this translation requires that the bytes of a doubleword be reversed between I/O storage and the processor. This byte reversal may happen either as the data is read from or written to a Little-Endian I/O device and put into System Memory, or it may occur as the data in Little-Endian order in System Memory is brought into the processor. The required byte alignment as a function of endian mode and access is summarized in Table 65 on page 267. The table assumes that the doubleword value 0x1011121314151617 is stored at address 0 (actually any doubleword-aligned address.)

Thus, a processor in Big-Endian mode that accesses the halfword at address 4 expects to see the value 0x1415. The most significant byte of the halfword, 0x14, appears in byte lane 4 and the least significant byte, 0x15, in byte lane 5. The table shows that applying the address mapping of Table 64 on page 266 to the address and reversing the bytes between storage and the processor will result in referencing the quantity 0x1415 in byte lanes 3 and 2 at halfword address 2 in a Little-Endian storage platform, as required.

Personal Use Copy - Not for Reproduction

Table 65. Bytes Accessed Versus Endian Mode

Big-Endian	Byte Address								Little-Endian
	0	1	2	3	4	5	6	7	
	7	6	5	4	3	2	1	0	
Byte at addr 0	10								Byte at addr 7
Byte at addr 1		11							Byte at addr 6
Byte at addr 2			12						Byte at addr 5
Byte at addr 3				13					Byte at addr 4
Byte at addr 4					14				Byte at addr 3
Byte at addr 5						15			Byte at addr 2
Byte at addr 6							16		Byte at addr 1
Byte at addr 7								17	Byte at addr 0
Halfword at 0	10	11							Halfword at 6
Halfword at 2			12	13					Halfword at 4
Halfword at 4					14	15			Halfword at 2
Halfword at 6							16	17	Halfword at 0
Word at addr 0	10	11	12	13					Word at addr 4
Word at addr 4					14	15	16	17	Word at addr 0
Doubleword at 0	10	11	12	13	14	15	16	17	Doubleword at 0
Instr at addr 0	i00	i01	i02	i03					Instr at addr 4
Instr at addr 4					i10	i11	i12	i13	Instr at addr 0

The third part of the translation requires that, in Little-Endian mode, addresses be generated correctly when addressing data in Little-Endian storage or when addressing I/O device addresses. For instance, since the effective address generated by the Little-Endian program as modified by the processor is used to access I/O, the programmer writing a device driver would have to pre-compensate the effective address used to access an adapter so that, after the

Personal Use Copy - Not for Reproduction

modification by the processor, the real address used to access the adapter is the real address of the target storage/register on the adapter. This translation must be done in every device driver and makes writing device drivers very error-prone. A better solution is to solve the address modification problem one time in hardware. Logic must be added to the I/O interface such that, when Little-Endian mode is selected, the added logic undoes the modification to the three low-order bits of the address. Then the unmodified (remodified) address used to access the I/O adapter is the same address as generated by the program. This platform design is required, since a programmer writing a device driver is able to use the control register addresses as specified in the adapter hardware reference manual.

In summary, whenever the machine is running in an endian mode different from the native mode of the processor (that is, the PowerPC processor expects Big-Endian storage order, but the platform is running in a Little-Endian mode) the address must be remapped and the byte lanes reversed somewhere on the way to or from the I/O subsystems.

C.2 Conforming Bi-Endian Designs

Several designs for implementing a Bi-Endian architecture are possible. Two approaches are described in the following material. The first approach is one in which both the memory and I/O subsystems are Bi-Endian, and the second approach is one in which memory is Big-Endian and the I/O subsystems are Bi-Endian.

C.2.1 Processor and I/O Mode Control

The mode of the platform will be changed by firmware, which has to perform the required functions to place the processor and I/O subsystem in the mode desired by the operating system. These designs are based on the assumption that the processor and I/O are in the same endian mode. No attempt has been made to design a platform where Little-Endian data is read and translated for a Big-Endian-mode processor running Big-Endian applications. The processor comes up in Big-Endian mode at power on or after a hardware reset. The synchronization requirements for changing the processor from one endian mode to the other are processor implementation-dependent and are specified in the Book IV, *Processor Implementation Features* document for the processor implementation.

Since the processor does not provide an external signal indicating the endian mode selected, the platform must provide a mechanism to allow firmware to control the endian mode of other subsystems. During configuration,

Personal Use Copy - Not for Reproduction

firmware will use this mechanism to select the endian mode to be used by storing the appropriate control value to the address of the control mechanism. The platform design may place the control mechanism address in the System Control Area or in Peripheral Memory space, whichever is more convenient. Firmware must be able to address and alter the mode control mechanism in both endian modes.

C.2.2 Approach #1—Bi-Endian Memory and Bi-Endian I/O Design

The Bi-Endian Memory and Bi-Endian I/O design for a Bi-Endian platform is shown in Figure 16 on page 271. I/O devices are categorized as "Transportable I/O," which consists of devices such as disks, tapes, and networks, and "Presentation I/O," which consists of devices such as graphics, audio, and video adaptors. For this design, memory, the processor, and Transportable I/O interfaces must support both endian modes. Data may exist on the Transportable devices in either Big-Endian or Little-Endian format; it is brought in, and sent out to the outside world, in either form. Presentation I/O are by design either Big-Endian or Little-Endian (or with extra hardware they may be Bi-Endian).

The processor accesses both storage and I/O through a controlled byte reversal multiplexor and controlled address modification function (called Xpose and Mod in Figure 16 on page 271). The address modification algorithm is shown in Table 64 on page 266. The byte reversal multiplexor reverses the position of each byte in a doubleword as shown in Table 67 on page 270. Similarly, when the transfer between the processor and I/O is on a 4-byte I/O bus, the byte reversal is performed as shown in Table 68 on page 270.

In this design, the memory and I/O are connected as if they were Big-Endian. In this case, byte 0 of storage or I/O is considered the most significant byte and passes through the controlled byte reversal multiplexor to byte 0 of the processor (MSB of the processor). The rules for applying the byte reversal multiplexor and the address modification are as shown in Table 66 on page 269.

Table 66: Rules for Byte Reversal and Address Modification

Mode	Function
Big-Endian	No byte reversal of data and no address translation
Little-Endian	Byte reversal of data and address translation

Personal Use Copy - Not for Reproduction

Table 67. Endian Mode Data Byte Reversal

I/O	Storage Byte	
Byte	BE Mode	LE Mode
0	0	7
1	1	6
2	2	5
3	3	4
4	4	3
5	5	2
6	6	1
7	7	0

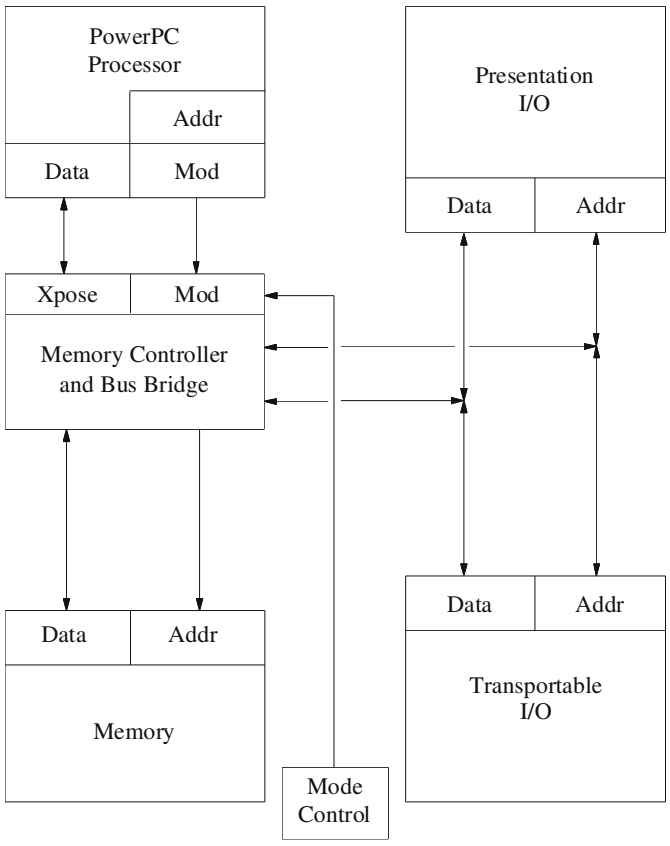
Table 68. Byte Reversal, Unequal Bus Widths

I/O		Storage Byte		Description
Word	Byte	BE Mode	LE Mode	
0	0	0	7	Word 0: Even addressed word
	1	1	6	
	2	2	5	
	3	3	4	
1	0	4	3	Word 1: Odd addressed word
	1	5	2	
	2	6	1	
	3	7	0	

From the viewpoint of the endianess of the data, I/O interfaces access only storage, not the processor or other I/O interfaces. I/O master transfers that move data from one device to another do not enter into the endianess translation.

The following three sections describe the interfaces that must be designed to perform this Bi-Endian support. For the three interfaces, both address processing and data handling are described.

Personal Use Copy - Not for Reproduction



Note: These are functional blocks, not chips.

Figure 16. Bi-Endian Platform with Bi-Endian Memory and I/O

C.2.2.1 Processor-to-Memory Interface

In this form of the design, memory is assumed to be in the same endian mode as the processor. Memory is accessed in two modes: as a result of cache-inhibited loads and stores, or as a result of cache reads or writes. Each of these operations will be described below.

Operations between cache and memory are always doublewords or larger, and as such the address in either endian mode is unaffected. Data is byte

Personal Use Copy - Not for Reproduction

reversed for Little-Endian mode and not byte reversed for Big-Endian mode. The result of these operations is that data brought into cache from Big-Endian memory is unchanged, and data brought into cache from memory in Little-Endian mode is byte reversed and stored as the PowerPC processor expects to see it (that is, most significant byte first). Writing data back to memory from cache in Little-Endian mode reverses the byte order and restores the data in Little-Endian order.

Cache-inhibited loads and stores have to adjust the data and addresses for the expected processor formats of data and addresses. For Little-Endian mode, the address generated by an instruction points to where the data exists in Little-Endian storage. The address is translated by the processor, and then translated again outside the processor which returns it to its original value. The data is byte reversed, putting it in the order expected by the processor logic on fetches or expected in storage for stores. For Big-Endian mode, the address is unchanged and the data is placed in memory in the order contained in the processor.

C.2.2.2 I/O-to-Memory Interface

There are no address or data transformations between memory and I/O. Data is placed in memory in the same order as it exists on the I/O devices independent of endian mode. The one exception might be Presentation I/O devices. Adaptors for these devices could be designed to accept data from Little-Endian and Big-Endian storage, or to always accept data only in one format. In this case, software would have to handle the data transformation. For example, a graphics adaptor that expected data in Little-Endian format would need software to byte reverse the data into Big-Endian mode before putting the data in memory.

C.2.2.3 Processor-to-I/O Interface

In Little-Endian mode, the address as modified by the processor must be modified again by the I/O interface such that the address used for the I/O access is the address computed by the storage instruction. Within the processor, the I/O addresses computed by a storage instruction are modified by the processor before the access is performed. Regardless of whether the access is to I/O space memory or a device control register, the address originally computed by the instruction is the address that must be used to access I/O space. The address modification algorithm shown in Table 64 on page 266 is used to remodify this I/O address. This function is shown in Figure 16 on page 271 in the box labeled “Mod,” which is controlled by the box labeled “Mode Control.”

Personal Use Copy - Not for Reproduction

The data transfer between the processor and I/O is managed in the same manner as the transfer between the processor and memory. Bytes are crossed between source and destination byte channels as indicated in Table 67 on page 270 and Table 68 on page 270. This byte transposition will occur once in Little-Endian mode to transform the processor-held data in Big-Endian format to Little-Endian format for I/O. In Big-Endian mode, the byte order is not transformed.

C.2.3 Approach #2—Bi-Endian I/O Design

The Bi-Endian I/O design for a Bi-Endian platform is shown in Figure 17 on page 274. For this design, storage is always Big-Endian, which means data is always stored with the most significant byte at the lowest address. Transportable I/O interfaces must support both endian modes. Data may exist on the Transportable devices in either Big-Endian or Little-Endian format; it is brought in, and sent out to the outside world, in either form. Presentation I/O are by design either Big-Endian or Little-Endian (or with extra hardware they may be Bi-Endian).

The processor accesses both storage and I/O. I/O master transfers that move data from one device to another do not enter into the endianess translation. From a platform viewpoint, I/O interfaces access only storage, not the processor or other I/O interfaces.

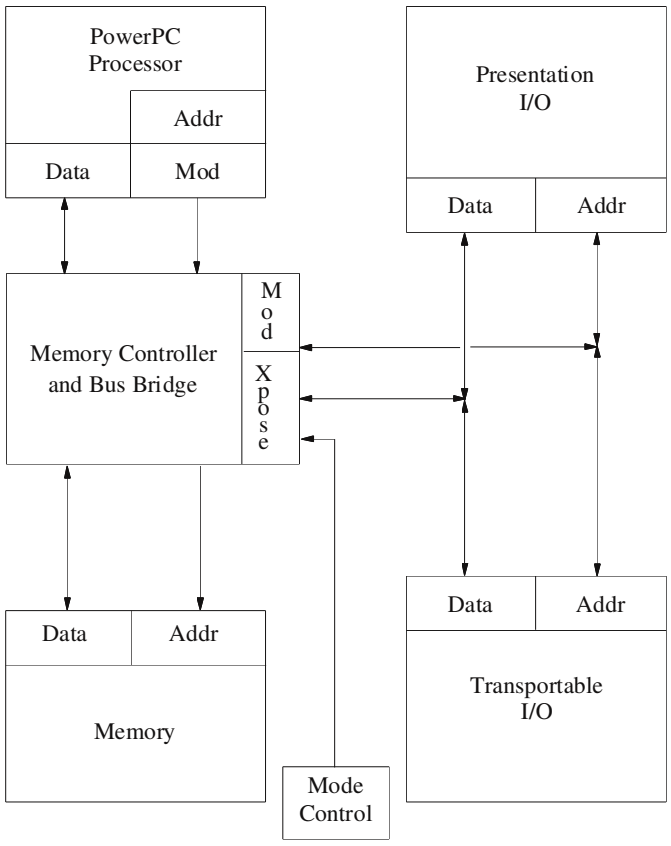
I/O accesses storage through a controlled byte reversal multiplexor and controlled address modification function (called Xpose and Mod in Figure 17 on page 274). This is further explained in Section C.2.3.2, “I/O-to-Memory Interface,” on page 274.

The following three sections describe the interfaces that must be designed to perform this Bi-Endian support. For the three interfaces, both address processing and data handling are described.

C.2.3.1 Processor-to-Memory Interface

Address translations are performed by the processor. Cache block transfers between the processor and storage are always a doubleword or larger, and the address is not affected by the endian mode. With Big-Endian storage, Little-Endian-mode loads and stores of aligned scalars with caching enabled work correctly after the address translation. When Little-Endian mode is enabled, loads and stores with caching inhibited use the address as modified by the processor. These instructions work the same and have the same constraints as for loads and stores out of cache.

Personal Use Copy - Not for Reproduction



Note: These are functional blocks, not chips.

Figure 17. Bi-Endian Platform with Bi-Endian I/O

For the processor-to- or from-memory interface, the data handling is independent of endian mode. The memory stores multi-byte scalars in Big-Endian order, which has the most significant byte at the first byte of the address.

C.2.3.2 I/O-to-Memory Interface

In Little-Endian mode, storage addresses generated by I/O devices are modified using the address modification described in Table 64 on page 266 prior to

Personal Use Copy - Not for Reproduction

performing the access. This address modification is shown in the block labeled "Mod" on the right side of the Memory Controller and Bus Bridge, shown in Figure 17 on page 274. This address modification is performed for both Transportable I/O and Presentation I/O adaptors. This address modification is required to adjust for the format of the data in memory that has been byte reversed to place the MSB first.

Data transferred to and from I/O devices that must switch endian modes must have data that is stored in Little-Endian format converted to Big-Endian format. I/O transfers may be done at any implementation-determined width, but this byte reversal of data in Little-Endian mode is done by treating the data as a string of bytes that must be reversed within a doubleword (see Table 67 on page 270). For unaligned transfers or transfers of less than a doubleword, bytes must be crossed from the source byte channel to the destination byte channel as shown in this table. This design places a byte-reversing multiplexor in the path from I/O to memory. This byte reversal multiplexor is shown as the "Xpose" box in Figure 17 on page 274.

When the interface between main storage and I/O requires a conversion from a two-word bus to a one-word bus, the byte reversal should be done in accordance with Table 68 on page 270, which assumes a two-word bus to main storage and a one-word bus to I/O.

An unaligned access (for example, read or write of System Memory) that crosses a doubleword boundary must be performed as multiple accesses on processors which do not support unaligned accesses. The address modification algorithm described above does not work for unaligned accesses. One approach to handling unaligned accesses is to perform the access as multiple aligned accesses using byte, halfword, and word operations for which the main storage address is modified as described above.

The specific design of the Presentation I/O device adaptors will determine whether a second byte-reversing multiplexor is present on the device, and will influence how device drivers must interface with this device. For instance, an audio adaptor that has byte reversal logic in it may be placed on a bus, while a graphics adaptor that does not have byte reversal logic may be on the same or a different bus. Depending upon the mode of the processor and memory, neither device may need byte reversal or both may need it. For example, a graphics adaptor with a Little-Endian design (that is, registers and data are expected in Little-Endian order) could be addressed by a Big-Endian processor via software performing the byte reversal as the Big-Endian data was sent to the graphics adaptor. Alternatively, the graphics adaptor could be designed to perform the byte reversal. In this case, the adaptor would pass data straight through when the processor is in Little-Endian mode and do a byte reversal when the processor is in Big-Endian mode.

Personal Use Copy - Not for Reproduction

C.2.3.3 Processor-to-I/O Interface

In Little-Endian mode, the address as modified by the processor must be modified again by the I/O interface such that the address used for the I/O access is the address computed by the storage instruction. Within the processor, the I/O addresses computed by a storage instruction are modified by the processor before the access is performed. Regardless of whether the access is to I/O space memory or a device control register, the address originally computed by the instruction is the address that must be used to access I/O space. The address modification algorithm shown in Table 64 on page 266 is used to remodify this I/O address. This function is shown in Figure 17 on page 274 in the boxes labeled "Mod."

The data transfer between the processor and I/O is managed in the same manner as the transfer between I/O and memory except that the processor is the master (it provides address and control) rather than an I/O mechanism. Bytes are crossed between source and destination byte channels as indicated in Table 67 on page 270 and Table 68 on page 270. This byte reversal is performed on all I/O transfers. The processor performs unaligned accesses as multiple accesses to aligned doublewords and may transfer an odd number of bytes within an aligned doubleword.

C.3 Software Support for Bi-Endian Operation

The endian mode-switching logic is a function provided by firmware before control is passed to the operating system. The specific set of instructions is processor- and platform-implementation-dependent.

As pointed out above, data for Presentation I/O and multi-byte control data (registers) for any I/O device may have to be byte reversed by software. Services to perform these operations should be provided by the support software. Typically, this would be language syntax and compiler support for a load and store with byte reversal of 2-, 4- and 8-byte scalars. If the compilers for all languages do not support these forms of loads and stores, then the operating system should supply services that perform the byte reversal.

C.4 Bi-Modal Devices

For performance reasons, applications in many operating environments write directly to graphics adaptors. Graphics adaptors for CHRP implementations will provide both Big-Endian and Little-Endian data transfer methods. (See Ta-

Personal Use Copy - Not for Reproduction

ble 2 on page 19) It is recommended that graphics subsystems implementing this support use the design shown in Figure 18 on page 278, which applies to both the frame buffer and the graphics subsystem's register/command space. This should provide good performance when using the hardware acceleration features of the graphics subsystem. In this design, the graphics subsystem is accessed through address "apertures" which perform the endian switch of data as it passes through the aperture, as necessary.

The register/command apertures provide two ways (Big- and Little-Endian) to access the control data. They are defined address ranges within the address map of the adaptor. While two dedicated apertures are desired, one aperture may be used if it can programmably accommodate Big- or Little-Endian. The aperture labelled "BE" would always provide byte swapping in a two-aperture implementation. The aperture labelled "LE/BE" would provide no byte swapping in a two-aperture implementation, and provide byte swapping in a single-aperture implementation based on one of the following:

1. The size of the facility being accessed would automatically determine how swapping is to occur for a particular access (that is, half- or full-word, or no swapping).
2. Software would programmably set the swapping function to either half-word, full-word, or none as appropriate.

The frame buffer apertures provide two ways to access the pixel data. The frame buffer apertures are defined address ranges within the address map of the adaptor. Each frame buffer aperture can be independently controlled to provide one of the following modes, under software control:

1. No swapping.
2. Byte swapping within each halfword, for example, $ABCD \Rightarrow BADC$. This becomes $ABCDEFGH \Rightarrow BADCFEHG$ for a platform with a 64-bit data interface, such as the PCI bus with optional 64-bit bus extension.
3. Byte swapping across the entire word, for example, $ABCD \Rightarrow DCBA$. This becomes $ABCDEFGH \Rightarrow DCBAHGFE$ for a platform with a 64-bit data interface, such as the PCI bus with optional 64-bit bus extension.
4. Byte swapping across a doubleword, for example, $ABCDEFGH \Rightarrow HGFEDCBA$ for 64-bit pixels on a 64-bit extended data bus.

Hardware Implementation Note: Determination of the correct swapping mode for a given access may be performed automatically in the hardware. This approach is not recommended if there is a possibility

Personal Use Copy - Not for Reproduction

that the pixel depth in the frame buffer can be interpreted differently by multiple devices, or is not guaranteed to be “known” or implied by the state of the graphics subsystem hardware.

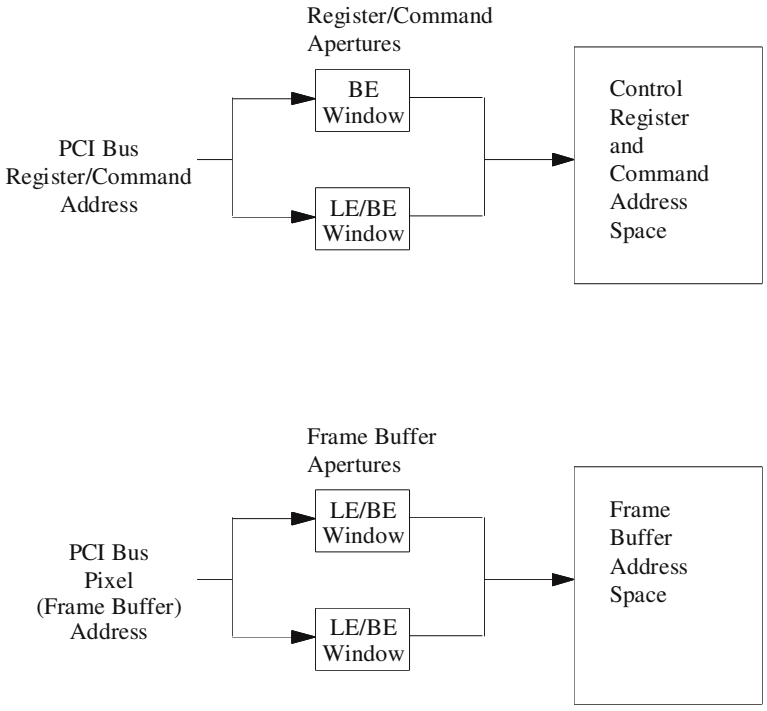


Figure 18. Bi-Endian Apertures for the Graphics Subsystem

With this capability, one aperture could access the frame buffer in Little-Endian mode, while the other could access it in one of the Big-Endian modes. Similarly, one aperture could be defined to swap for 16-bit pixels, while the other could be defined to swap for 24- or 32-bit pixels. Alternatively, several apertures may be defined to support the various pixel depths.

Platforms employing only one frame buffer aperture would provide the previously described swapping options under software control.

24-bit pixels are defined to occupy the least significant 3 bytes of a full word. The most significant byte may be used as an alpha byte, or may be

Personal Use Copy - Not for Reproduction

unused. 16-bit pixels are always halfword aligned, and 24- or 32-bit pixels are always full-word aligned.

Pixels between 32 and 64 bits are defined to occupy the least significant bytes in the 64-bit field, and are doubleword aligned.

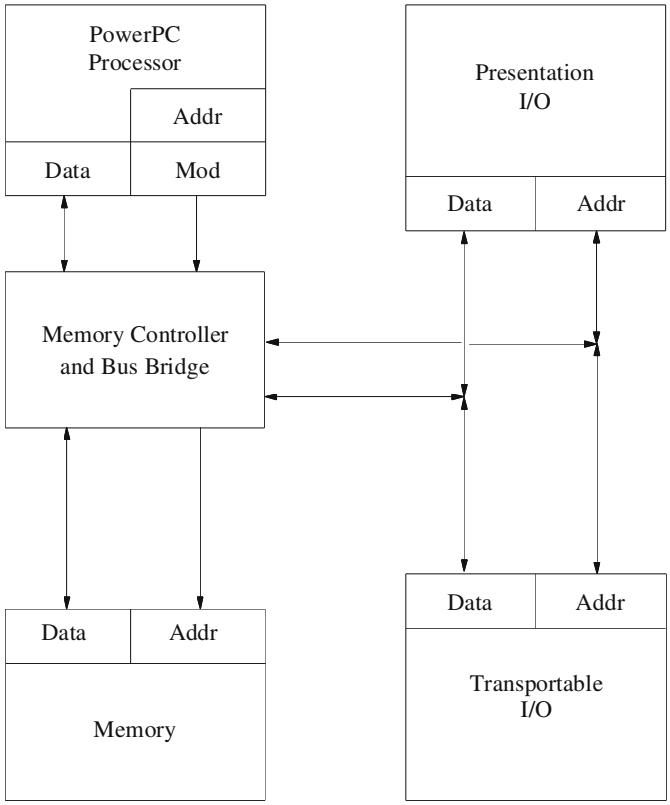
C.5 Future Directions in Bi-Endian Architecture

This section discusses possible future directions in the design of PowerPC microprocessors and the effect these changes would have on Bi-Endian platform design. The current implementations of Little-Endian mode in PowerPC microprocessors reduce internal processor complexity by moving some of the Bi-Endian support out of the processor.

Future implementations of the PowerPC architecture may support a true Little-Endian mode. In this true Little-Endian mode, data would go to the processor in Little-Endian format and be addressed from the processor with the Little-Endian address. In the interim, some PowerPC processor designs may implement support for Little-Endian unaligned operations.

Figure 19 on page 280 shows a design which could be used with future versions of the PowerPC processor implementing full Bi-Endian support. No external address modifications or byte reversal multiplexors are required. The previous designs may be migrated to this design by physically removing the byte reversal multiplexors and address modification logic components from the design, or by functionally removing them by changing the rules under which they are applied. For instance, Approach #1 (Figure 16 on page 271) would require the address modification and byte-transposing multiplexor to always be off. The address modification is no longer needed, because the processor would not modify the address since it deals with Little-Endian data in Little-Endian order. The byte reversal is not required because the processor accepts data in true Little-Endian order. The same applies to Approach #2.

Personal Use Copy - Not for Reproduction



Note: These are functional blocks, not chips.

Figure 19. Design with a Full Bi-Endian Processor

Personal Use Copy - Not for Reproduction

Architecture Migration Notes

D

This appendix provides information describing the migration from legacy systems. The PowerPC Reference Platform, Apple RISC architecture, and IBM RISC server systems were used in the development of the CHRP architecture. The objective was to reduce the porting effort of operating systems and applications coming from each of these environments. The information below describes the relationship of the Apple RISC architecture to the CHRP architecture and of the PowerPC Reference Platform to this architecture. The PowerPC Reference platform used IBM RISC client and server information.

Many components of this architecture are included for compatibility with the second generation Power Macintosh desktop products. Features of the second generation Power Macintosh and references to their use in this architecture are listed below:

- Based on the PowerPC microprocessor family. See Section 4.1, “Processor Architecture,” on page 49.
- Use of the PCI bus to support all I/O and system expansion. See 5.1 , “PCI Host Bridge (PHB) Architecture,” on page 69.
- Use of bus to bus bridges to support other buses such as NuBus™, SCSI, and IDE. See Section 5.2, “I/O Bus to I/O Bus Bridges,” on page 78.
- Use of Open Firmware for system start-up and to allow use of expansion cards from other architectures. See Section 2.2, “Firmware,” on page 13.
- Function of processor bus coherency. See Section 4.2.2, “Storage Ordering Models,” on page 57.

- Support for both Big-Endian and Little-Endian modes. See Section 2.3, “Bi-Endian Support,” on page 14.
- Provide Macintosh style I/O such as ADB, SCC, and LocalTalk. See Section 2.5, “Minimum System Requirements,” on page 16 and Chapter 9, “I/O Devices,” on page 151.

Some material used in this document was originally published in the *PowerPC Reference Platform Specification*, Version 1.1 [28]. For those currently developing products for the *PowerPC Reference Platform,* this evolution information will aid in understanding the architecture in this document. Table 69 on page 282 shows the sections in the *PowerPC Reference Platform Specification* and indicates their allocation in this document or the reason some material was not carried forward. Material carried from the *PowerPC Reference Platform Specification*, Version 1.1 has been modified and new material inserted.

Table 69. PowerPC Reference Platform Specification Evolution

PowerPC Reference Platform Section	Presentation in this Document
Chapter 1, “Introduction”	Chapter 1, “Introduction,” on page 1
Chapter 2, “Hardware Configuration”	Section 2.5, “Minimum System Requirements,” on page 16
Chapter 3, “Architecture through Section 3.1 System Topology”	Chapter 1, “Introduction,” on page 1
Section 3.2, “System Memory”	Section 4.2.2.1, “Memory Coherence,” on page 57
Section 3.3, “I/O Memory”	Section 4.2.2.1, “Memory Coherence,” on page 57
Section 3.4, “System I/O,” and Section 3.5 Self Modifying Code”	Section 4.2.2.1, “Memory Coherence,” on page 57
Section 3.6, “Resource Locking”	Section 9.1.1, “Resource Locking,” on page 152
Section 3.7. “Bus Errors”	Chapter 10, “Error and Event Notification,” on page 157
Section 3.8, “Memory Map”	Section 4.2.1, “System Memory,” on page 56
Section 3.8.1, “Example Memory Maps”	Chapter 3, “System Address Map,” on page 23
Section 3.9, “Memory Ordering”	Section 4.2.2.1, “Memory Coherence,” on page 57
Section 3.10, “Configuration and Diagnostics”	Not covered in the Architecture
Section 3.11, “Power Management”	Chapter 11, “Power Management,” on page 185
Section 3.12, “Bi-Endian Support”	Section 2.3, “Bi-Endian Support,” on page 14

Personal Use Copy - Not for Reproduction

Table 69. PowerPC Reference Platform Specification Evolution ***(Continued)***

PowerPC Reference Platform Section	Presentation in this Document
Section 3.13, “Multiprocessor Considerations”	Chapter 12, “The Symmetric Multiprocessor Option,” on page 215
Section 3.14, “Alignment Considerations”	Section 4.1.4.1, “Unaligned Little-Endian Scalar Operations,” on page 53 and Section 4.1.4.2, “Little-Endian Multiple Scalar Operations,” on page 54
Section 3.15, “Support for Loads and Stores to System I/O Bus,” and Section 3.16, “Cache-Inhibited Loads and Stores to System Memory”	Section 4.1.4.3, “Direct-Store Segment Support,” on page 54
Section 3.17, “PowerPC Architecture Features Not Recommended”	Section 4.1.4, “PowerPC Architecture Features Deserving Comment,” on page 53
Chapter 4, “Machine Abstraction”	Chapter 7, “Run-Time Abstraction Services,” on page 91
Chapter 5, “Boot Process,” through Section 5.4, “Transferring Control”	The legacy process is defined in the current version of the *PowerPC Reference Platform Specification* [28]
Section 5.5, “NVRAM”	NVRAM for the legacy process is described in the *PowerPC Reference Platform Specification* [28]. NVRAM for the Open Firmware process is described in Chapter 8, “Non-Volatile Memory,” on page 141
Section 5.6, “Residual Data”	Residual data for the legacy process is described in the *PowerPC Reference Platform Specification* [28]
Section 5.7, “Open Firmware Extension”	Refer to the *PowerPC Microprocessor Common Hardware Reference Platform System binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [10]
Chapter 6, “Reference Implementation”	Hardware system vendors may publish separate books with implementation examples for compliant platforms
Appendix A, “Implementation Examples”	Separate books prepared by hardware system vendors
Appendix A.7, “Proposed Diagnostic Strategy”	Not covered in the Architecture
Appendix B, “Bi-Endian Design Guidance”	Appendix C, “Bi-Endian Designs,” on page 265
Appendix C, “Additional Compliant Subsystems and Devices”	Separate books prepared by hardware system vendor
Appendix D, “Windows NT”	Obtain information from the operating systems
Appendix E, “AIX”	Obtain information from the operating systems
Appendix F, “Workplace OS”	Obtain information from the operating systems
Appendix G, “Solaris”	Obtain information from the operating systems

Personal Use Copy - Not for Reproduction

Table 69. PowerPC Reference Platform Specification Evolution ***(Continued)***

PowerPC Reference Platform Section	Presentation in this Document
Appendix H, “Taligent”	Obtain information from Taligent™
Appendix I, “PowerPC Supplement to IEEE 1275”	See *PowerPC Microprocessor Common Hardware Reference Platform System binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* [10]
Appendix J, “Plug and Play Extensions”	See the *PowerPC Reference Platform Specification* [28]
Appendix K, “Dump of Residual Data”	Residual data is described in the *PowerPC Reference Platform Specification* [28]
Obtaining Additional Information	Section , “Obtaining Additional Information,” on page 300
Bibliography	Section , “Bibliography,” on page 297
Acronyms and Abbreviations	Section , “Glossary,” on page 285
Glossary	Section , “Glossary,” on page 285
Trademark Information	Section , “Trademark Information,” on page 295
Index	Section , “Index,” on page 303

Personal Use Copy - Not for Reproduction

Glossary

This glossary contains an alphabetical list of terms, phrases, and abbreviations used in this document.

Term	Definition
AC	Alternating current
AD	Address data line
ADB	Apple Desktop Bus
addr	Address
Architecture	The hardware/software interface definition or software module to software module interface definition.
ASCII	American National Standards Code for Information Interchange
ASR	Address Space Register
BAT	Block address translation
BE	Big-Endian or Branch Trace Enable bit in the MSR
BIM	Bottom of initial memory
BIO	Bottom of Peripheral Input/Output Space
BIOS	Basic input/output system
Boundedly undefined	Describes some addresses and registers which when referenced provide one of a small set of predefined results.

BPM	Bottom of Peripheral Memory
BSCA	Bottom of System Control Area
BSM	Bottom of System Memory
CD-ROM	Compact disk read-only memory
CHRP	Common Hardware Reference Platform
CIS	Client interface service
COM	Communication
CPU	Central processing unit
CR	Condition Register
CTR	Count Register
DABR	Data Address Breakpoint Register
DAC	Dual address cycle
DAR	Data Address Register
DBAT	Data block address translation
DBDMA	Descriptor based direct memory access
DDC 1	Display data channel for unidirectional information from displays.
DEC	Decrementer
DIMM	Dual in-line memory module
DOS	Disk operating system
DR	Data relocate bit in MSR
DSISR	Data Storage Interrupt Status Register
DMA	Direct memory access
EA	Effective address
EAR	External Access Register
ECC	Error checking and correction
ECP	Extended capability port
EE	External interrupt enable bit in the MSR
EISA	Extended industry standard architecture

Personal Use Copy - Not for Reproduction

EPA	Environmental protection agency
EPOW	Environment and power warning
ERR	Exception Relocation Register
ESCD	Extended system configuration data
FCode	A computer programming language defined by the Open Firmware standard which is semantically similar to the Forth programming language, but is encoded as a sequence of binary byte codes representing a defined set of Forth words.
FE0	Floating-Point exception mode 0 bit in the MSR
FE1	Floating-Point exception mode 1bit in the MSR
FP	Floating-Point available bit in the MSR
FPR	Floating-Point register
FPSCR	Floating-Point Status And Control Register
FSM	Finite state machine
GB	Gigabytes - as used in this document it is 2 raised to the power of 30
HB	Host Bridge
Hz	Hertz
IBAT	Instruction block address translation
ID	Identification
IDE	Integrated device electronics
IDU	Interrupt delivery unit
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input/output
I/O bus master	Any entity other than a processor, cache, memory controller, or host bridge which supplies both address and data in write transactions or supplies the address and is the sink for the data in read transactions.
ILE	Interrupt Little-Endian bit in MSR
Instr	Instruction
IP	Interrupt prefix bit in MSR

Personal Use Copy - Not for Reproduction

IPI	Interprocessor interrupt
IPL	Initial program load
IR	Instruction relocate bit in MSR register or infrared
IrDA	Infrared Data association which sets standards for infrared support including protocols for data interchange.
ISA	Industry standard architecture (typically refers to the PC local bus)
ISO	International Standards Organization
ISR	Interrupt source register
ISU	Interrupt source unit
JEIDA	Japan Electronic Industry Development Association
KB	Kilobytes - as used in this document it is 2 raised to the power of 10
KHz	Kilo Hertz
LAN	Local area network
LCD	Liquid crystal display
LE	Little-Endian bit in MSR or Little-Endian
LED	Light emitting diode
LFB	Linear frame buffer
LR	Link Register
L1	Primary cache
L2	Secondary cache
MB	Megabytes - as used in this document it is 2 raised to the power of 20
ME	Machine check enable
MESH	Macintosh enhanced SCSI hardware
MFM	Modified frequency modulation
MHz	Mega Hertz
MOD	Address modification bit in the MSR
MOU	Memorandum of understanding

Personal Use Copy - Not for Reproduction

MP Multiprocessor

MSB Most significant byte

MSR Machine State Register

N/A Not applicable

Nibble Refers to the first or last four bits in an 8 bit byte

NVRAM Nonvolatile random access memory

OF Open Firmware

OS Operating system

OUI Organizationally unique identifier

PC Personal computer

PC Card A memory or I/O card compatible with the *PC Card Standard* [17]. 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)

Peripheral I/O Space
The range of real addresses which are assigned to the I/O Space of a Host Bridge (HB) and which are sufficient to contain all of the *Load* and *Store* address space requirements of all the devices in the I/O Space of the I/O bus that is generated by the HB. A keyboard controller is an example of a device which may require Peripheral I/O Space addresses. This portion of the system address space was referred to as “System I/O” in the *PowerPC Reference Platform Specification.*

Peripheral Memory Space
The range of real addresses which are assigned to the Memory Space of a Host Bridge (HB) and which are sufficient to contain all of the *Load* and *Store* address space requirements of the devices in the Memory Space of the I/O bus that is generated by the HB. The frame buffer of a graphics adapter is an example of a device which may require Peripheral Memory Space addresses. This portion of the system address space was re-

Personal Use Copy - Not for Reproduction

ferred to as “I/O Memory” in the *PowerPC Reference Platform Specification* [28].

Peripheral Space
: Refers to the physical address space which may be accessed by a processor, but which is controlled by a host bridge. At least one peripheral space must be present and it is referred to by the suffix 0. A host bridge will typically provide access to at least a memory space and possibly to an I/O space.

PHB
: PCI Host Bridge

PIC
: Programmable interrupt controller

PIR
: Processor Identification Register

Platform
: Refers to the hardware plus firmware portion of a system composed of hardware, firmware, and operating system.

PM
: Power management

PnP
: Plug and play

POST
: Power-on self test

PR
: Privileged bit in the MSR

Processor revision number
: A 16-bit number that distinguishes between various releases of a particular processor version, for example different engineering change levels.

PVN
: Processor version number. Uniquely determines the particular processor and PowerPC architecture version.

PVR
: Refers to the Processor Version Register. A register in each processor that identifies its type. The contents of the PVR include the processor version number and processor revision number.

RAM
: Random access memory

Real address
: A real address results from doing address translation on an effective address when address translation is enabled. If address translation is not enabled, the real address is the same as the effective address. An attempt to fetch from, load from, or store to a real address that is not physically present in the machine may result in a Machine Check interrupt.

Personal Use Copy - Not for Reproduction

Reserved — The term “reserved” is used within this document to refer to bits in registers or areas in the address space which should not be referenced by software except as described in this document.

Reserved for firmware use — Refers to a given location or bit which may not be used by software, but are used by firmware.

Reserved for future use — Refers to areas of address space or bits in registers which may be used by future versions of the architecture.

RFU — Reserved for future use

RI — Recoverable interrupt bit in the MSR

RISC — Reduced instruction set computing

ROM — Read only memory

RPN — Real page number

RTAS — Run-time abstraction services

RTC — Real time clock

SCC — Serial communications controller

SCSI — Small computer system interface

SDR — Storage Description Register

SE — Single-step trace enabled bit in the MSR

SF — Processor 32-bit or 64-bit processor mode bit in the MSR

Shrink-wrap OS — A single version of an operating system that runs on all compliant platforms.

Shrink-wrap Application — A single version of an application program that runs on all compliant platforms with the applicable operating system.

SIMM — Single in-line memory module

SMP — Symmetric multiprocessor

Snarf — An industry colloquialism for cache-to-cache transfer. A typical scenario is as follows: (1) cache miss from cache A, (2) line found modified in cache B, (3) cache B performs castout of

Personal Use Copy - Not for Reproduction

modified line, and (4) cache A allocates the modified line as it is being written back to memory.

Snoop — The act of interrogating a cache for the presence of a line, usually in response to another party on a shared bus attempting to allocate that line.

SPRG — Special Purpose Registers for General use

SR — System Registers

SRR — Save/Restore Register

System — Refers to the collection of hardware, system firmware, and operating system software which comprise a computer model.

System address space — The total range of addressability as established by the processor implementation.

System Control Area — Refers to a range of addresses which contains the system ROM(s) and an unarchitected, reserved, platform-dependent area used by firmware and Run-Time Abstraction services for control of the platform. The ROM areas are defined by the OF properties in the *openprom* and *os-rom* nodes of the OF device tree.

System firmware — Refers to the collection of all firmware on a system including Open Firmware, RTAS and any legacy firmware.

System Memory — Refers to those areas of memory which form a coherency domain with respect to the PowerPC processor or processors that execute application software on a system.

System software — Refers to the combination of operating system software, device driver software, and any hardware abstraction software, but excludes the application software.

TB — Time base

TBD — To be determined

TBEN — Time base enable signal on a PowerPC processor

TCE — Translation Control Entry

Personal Use Copy - Not for Reproduction

TEA	Transaction error acknowledge signal on a PowerPC processor
TEMR	Top of Emulated Memory Register
TIO	Top of Peripheral Input/Output Space
Third party DMA	The process by which an entity independent of the data source and sink generates addresses for transfers from the source to the sink; or a noun referring to the independent entity; or a noun referring to the transfer done using this technique.
TLB	Translation lookaside buffer
TODC	Time of day clock
TPM	Top of Peripheral Memory
TSM	Top of System Memory
tty	Teletypewriter or ASCII character driven terminal device
UCT	Universal coordinated time
URL	Universal resource locator
VESA	Video Electronics Standards Association
VGA	Video graphics array
VIA	Versatile interface adapter
VPD	Vital product data
XER	Fixed-Point Exception Register
Xpose	Transpose data during LE mode transfer
x86	Intel processors such as 80386, 80486,...

Personal Use Copy - Not for Reproduction

Blank page when cut - 294

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., International Business Machines Corporation, and Motorola, Inc.
Ethernet	Xerox
IBM	International Business Machines Corporation
Intel	Intel Corporation
LocalTalk	Apple Computer, Inc.
Macintosh	Apple Computer, Inc.
Mac OS	Apple Computer, Inc.
Micro Channel	International Business Machines Corporation
Motorola	Motorola, Inc.
NetWare	Novell, Inc.

NuBus	Texas Instruments
Power Macintosh	Apple Computer, Inc.
PowerPC	International Business Machines Corporation
PS/2	International Business Machines Corporation
Solaris	SunSoft
Taligent	Taligent, Inc.
Windows	Microsoft Corporation
Windows NT	Microsoft Corporation

Personal Use Copy - Not for Reproduction

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. *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
2. *Computer Architecture: A Quantitative Approach,* Second Edition Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-329-8
3. *How To Create Endian-Neutral Software for Portability*, IBM TR54.837 (also published in *Dr. Dobb's Journal* for October and November 1994)
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. *Open PIC Multiprocessor Interrupt Controller Register Interface Specification,* Revision 1.2

9. IEEE 1275, *IEEE Standard for Boot (Initialization Configuration) Firmware, Core Requirements and Practices,* IEEE part number DS02683, ISBN 1-55937-426-8

10. *PowerPC Microprocessor Common Hardware Reference Platform System binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

11. *PowerPC processor binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware*

12. *PCI Bus Binding to IEEE 1275 Standard for Boot (Initialization, Configuration) Firmware*

13. IEEE 996, *A Standard for an Extended Personal Computer Back Plane Bus*

14. *PCI Local Bus Specification*, Revision 2.1

15. *PCI System Design Guide*, Revision 1.0, September 1993

16. *PCI to PCI Bridge Architecture Specification*

17. *PC Card Standard*, February 1995

18. *PowerPC 60x Microprocessor Interface Specification*, by IBM and has limited availability

19. *PowerPC 6xx Bus Definition*, by IBM and has limited availability

20. *PowerPC Microprocessor Common Hardware Reference Platform: Hardware Technical Reference,* available from IBM

21. *Plug and Play ISA Specification,* Version 1.0a, Microsoft Corporation

22. *Extended System Configuration Data Specification*, Version 1.01, available with other Plug and Play information

23. *Macintosh Technology in the Common Hardware Reference Platform*, Morgan Kaufmann Publishers, Inc., San Francisco, CA, ISBN 1-55860-393-X

24. *Inside Macintosh*, Addison-Wesley Publishing Company, Reading, MA.

25. *Designing PCI Cards and Drivers for Power Macintosh Computers*, APDA, P.O. Box 319, Buffalo, NY 14207

26. *Technical Introduction to the Macintosh Family*, Second Edition, Addison-Wesley Publishing Company, Reading, MA, ISBN 0-201-62215-7

Personal Use Copy - Not for Reproduction

27. *RISC System/6000 PowerPC System Architecture*, Morgan Kaufmann Publishers, San Francisco, CA., ISBN 1-55860-344-1

28. *PowerPC Reference Platform Specification*, Version 1.1

29. *PCI Multimedia Design Guide*

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 and on CompuServe. The FTP server address is ftp.austin.ibm.com and the material is placed in the directory /pub/technology/spec. The document is maintained on a library of the PowerPC forum.

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. 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.

The *Open PIC Multiprocessor Interrupt Controller Register Interface Specification* is available from Advanced Micro Devices, Inc. or Cyrix, Inc. Send a request which includes your postal address to openpic@amd.com.

The *PowerPC Microprocessor Common Hardware Reference Platform System binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* and *PowerPC processor binding to: IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* are available via

Personal Use Copy - Not for Reproduction

anonymous FTP to playground.sun.com. These documents are located on the path /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 (33)-6713-5757 in French
- Within Europe (33)-6713-5756 in Italian
- Within Europe (49)-511-516-3444 in English
- Within Europe (49)-511-516-3555 in German
- In Asia (81)-755-87-4745 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.

Updates to *The PowerPC Microprocessor Common Hardware Reference Processor: A System Architecture* are available via anonymous FTP. The FTP server address is ftp.austin.ibm.com and the material is placed in the directory /pub/technology/spec. This document and other related documents are

Personal Use Copy - Not for Reproduction

described on the home page at www.austin.ibm.com at the directory /pow-rinfo.html.

Several white papers are available from IBM. These papers expand on the information in this specification. These papers are available via anonymous FTP to ftp.austin.ibm.com and in the directory /pub/technology/spec. The papers which were available at the time of publishing this document are as follows:

- *Bi-Endian Designs in PowerPC Reference Platform* by Shien-Tai Pan
- *PowerPC Endian Switch Code* by Gary Tsao
- *Plug and Play for PowerPC Reference Platform* by Gary Tsao
- *L2 Cache Design for PowerPC-Based Systems* by Allan Steel
- *The PowerPC™ Hardware Reference Platform,* by Steve MacKenzie, David Tjon, Allan Steel, and Steve Bunch

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 www.vesa.org.

Personal Use Copy - Not for Reproduction

Blank page when cut - 302

Index

Numerics

603
  power management 51
603 processor family 51
604 processor family 51
60x bus 53
620 processor family 51
64-bit address space 51
64-bit Addressing 14
64-bit execution mode 51
64-bit implementations 50, 51
6xx bus 53

A

ADB 16, 19
address map
  areas 23
  compatibility holes 24
  decode and translate table 32
  example: 32-bit with one HB 33
  example: 32-bit with two HBs 34
  example: 64-bit with one HB 35
  initial memory alias spaces 25
  io-hole 24
  legend of terms 23
  Peripheral I/O Space 24
  Peripheral Memory Space 23
  processor-hole 25
  routing and translation 28
  System Control Area 24
  System Memory 23
  Undefined 24
AIX 223
aliasing
  I-bit 58
  M-bit 58
  W-bit 58
alphanumeric input device 19
asymmetric multiprocessor compared with SMP 216
atomic stores 57
atomic update model 64
attached processor compared with SMP 216
audio 20

B

B2 security, uniprocessor 51
BE. *See* Big-Endian
Big-Endian
  multiple scalar operations 55
  PHB implications 74
Boot 8, 11
bridge
  ISA to PC Card 83
  PC Card 83
  PCI to ISA 82
  PCI to PC Card 83
  PCI to PCI 81
bridge architecture 52

C

cache
  disabled state 66
  enabled 66
  external 65
  flushing 65
  initialization 66
  in-line 66
  in-line, reservation support 66
  internal 65
  invalidate 65
  low power state 66
  memory 65
  RTAS cache-control 66
  state bits 66
cache control 136
cache flushing 65
cache line 58
cache memory 65
cache, external 19
Caching Inhibited (I bit) 58

Cardbus 83
CD-ROM 19
Check stop condition 160
check-exception 110
checksum, NVRAM 143
CIS 9
Client Interface Services in SMP booting 220
coherence
  memory 57
  symmetric multiprocessor 57
coherency
  TCE 40, 41
coherency domain 60
coherency granule 58
compatibility holes
  defined 24
  PCI Host Bridge option 70
Configuration 9
configuration, NVRAM partition 144, 145
consistency model 59
cpu
  OF properties 51

D

DAC. *See* Dual Address Cycle
Device tree 9
direct memory access 20
direct-store segments 54
discontiguous I/O
  defined 37
  figure showing 38
  initial state 37
display-character 104
DMA 20
DMA ordering 73
Dual Address Cycle
  defined 77
  option of the PHB 77

E

emulation-assists 51
endian modes 265
endianess 14
energy conservation 185
EPA Energy Star compliance 197
ERR Register 47
error log, NVRAM partition 144, 147
event-scan 110
exception handler 93
external cache 65
external control instructions 55

F

firmware 13
  NVRAM partition 144
firmware storage 19
floppy disk 19

G

graphics 20
Guarded Storage (G bit) 58

H

hard disk 19
hardware, NVRAM partition 145
HB. *See* Host Bridge
Hibernate 13, 128, 189
Host Bridge 61
  address map example 33, 34, 35
  option to accept 64-bit DMA addresses 40
  PCI. *See* PCI Host Bridge

I

I/O bus to I/O bus bridge
  general requirement 79
  PCI to ISA 82
  PCI to ISA, maximum number 82
  PCI to PCI 81
I/O interrupt group 87
I/O subsystem 60
I-bit aliasing 58
IDU 87
infrared 20
initial memory alias spaces
  defined 25
  peripheral-memory-alias 25
Initialization 9
instantiate-rtas 97, 102
internal cache 65
Inter-Processor Interrupts in booting 221
Interrupt Acknowledge Cycle 77
Interrupt Controller
  architecture 85
  Distributed implementation 86
  single-chip implementation 89
interrupt controller 20
  8259 support 77
Interrupt Delivery Unit 87
Interrupt Source Unit 87
io-hole

Personal Use Copy - Not for Reproduction

defined 24
DMA decode 32
IPI in booting 221
ISA 9, 145
device participation in DMA data transfers 79, 80
DMA controller requirement 82
ISA to PC Card bridge 83
PCI to ISA bridge 82
ISU 87

K

keyboard 16, 19

L

LE. *See* Little-Endian
legacy firmware 13
Little-Endian 265–268
address and data translation 265
PHB implications 74
PowerPC Little-Endian 265
load and store string operations 55

M

Mac OS 224
machine check 96
Machine check interrupt 160
master in booting SMP 219, 221, 222
M-bit aliasing 58
memory coherence 57
Memory Coherence Required (M bit) 58
memory controllers 64
full power mode 65
initialization 65
memory, cache 65
memory, non-volatile 19
memory, system 19
minimum requirements
alphanumeric input device 19
audio 20
cache, external 19
CD-ROM 19
DMA 20
firmware storage 19
floppy disk 19
graphics 20
hard disk 19
infrared 20
interrupt controller 20
memory, system 19
network 20
non-volatile memory 19
OS ROM 19
parallel port 20
pointing device 19
real time clock 20
ROM, OS 19
serial port 20
VIA timer 20
mouse 16, 19
MSR 93
multiboot 11
multi-boot, NVRAM partition 144, 147
multiple scalar operations
Big-Endian 55
Little-Endian 54
multiprocessor. *See* SMP

N

NetWare 224
network 20
Non-Volatile Memory 19, 141
NVRAM 19, 104, 141

O

OF method
change-address-map 36
instantiate-rtas 97
set-64-bit-addressing 39, 42, 70
set-discontiguous-io 37
set-initial-aliases 25
set-io-hole 25
set-processor-hole 25
OF property
32-64-bridge 51
603-power-management 51
603-translation 51
64-bit 51, 52
64-bit-addressing 39, 70
64-bit-dma 40, 70, 77
8259-interrupt-acknowledge 78
external-control 51
for RTAS 100
general-purpose 51
graphics 51
initial-memory-alias 25
Interrupt controller 89
interrupt-ranges 89
io-hole 25
memory 57
memory-controller 65

Personal Use Copy - Not for Reproduction

performance-monitor 51
processor-hole 25
reg 89
tlbia 51
Off (system power state) 190
Open Firmware 1, 13
Open PIC 85
operating systems 223
option
data buffering in PHB 70
HB accepts 64-bit DMA addresses 40
instruction queuing in PHB 70
io-hole 24
PC Emulation 44
PCI Dual Address Cycle 70
PHB 64-bit addressing 70
PHB Dual Address Cycle support 77
processor-hole 25
optional instructions
external control 55
ordering instructions 60
ordering of DMA data 73
ordering of *Load and Store operations* 71
OS/2 224

P

pages reserved for implementation specific use 51
parallel port 20
partitions 144
PC Card 83
PC Card bridge 83
PC Emulation option
defined 44
Exception Relocation Register (ERR) 47
state of holes and aliases 47
Top of Emulated Memory Register (TEMR) 47
PCI
configuration space 114
Dual Address Cycle (DAC) 77
Host Bridge. *See* PCI Host Bridge
Interrupt Acknowledge Cycle 77
PCI master defined 78
PCI master participation in DMA 78
PCI to ISA bridge 82
PCI to PC Card bridge 83
PCI to PCI bridge 81
read-pci-config 115
write-pci-config 116
PCI Host Bridge
DMA ordering 73
64-bit addressing option 70
compatibility holes option 70
data buffering in 70
data coherency 71
defined 69
Dual Address Cycle option 70
endianess 74
general requirements 69
instruction queuing in 70
Interrupt Acknowledge Cycle 77
ISA bridge attachment requirement 82
LE bit and DMA operations 75
LE bit and Load/Store operations 75
PCI master
defined 78
participation in DMA data transfers 80
PCMCIA 83
Peripheral I/O Address Space
decoding and translation 32
Peripheral I/O Space
boundary alignment 30
changing size 36
defined 24
discontiguous I/O mode 37
non-overlap requirement 29
number required per HB 30
PCI to PCI bridge limitation 81
PHB requirement 77
size restrictions 30
space not addressable 37
Peripheral Memory Space
boundary alignment 30
changing size 36
decoding and translation 32
defined 23
non-overlap requirement 29
number required per HB 30
peripheral-memory-alias 25
size restrictions 30
system-memory-alias 25
peripheral-memory-alias
defined 25
Load and Store decode 32
personal 16
PHB. *See* PCI Host Bridge
platform-aware software 49
Plug and Play 146
pointing device 19
portable 16

Personal Use Copy - Not for Reproduction

POST 8
power management 185–213
  batteries 195
  concepts 185
  device power states 187
  EPA Energy Star compliance 197
  hardware requirements 197
  mechanism 186
  Open Firmware properties 201
  policy 123, 186
  power domain control point 192
  power domain dependency tree 192
  power domains 124, 125, 191
  power management events 195
  software requirements 210
  system power states 187
  system power transitory states 190
power management disabled 188
power management enabled 188
power management events 195
power-off 96, 127
Power-on-mask 127
PowerPC Little-Endian 265
PowerPC Little-Endian System Memory 65
Powerup 191
processor configuration in SMPs 220
processor identification register 50
processor interface
  32-bit 53
  64-bit 53
processor version number 51
processor-hole
  defined 25
  Load and Store decode 32
PS/2 16, 19
PVN 51

R

real mode 94
real time clock 20
real-time clock 106
reboot 13
recursion in boot process with SMPs 222
requirement
  16-bit PC Card support 83
  contiguous System Memory 29
  data buffering in PHB 71
  Exception Relocation Register (ERR) 47
  I/O bus to I/O bus bridges 79
  I/O DMA routing and translation 29
  instruction queuing in PHB 71
  ISA bridge attached to a PHB 82
  ISA DMA controller 82
  Load and Store routing and translation 28
  maximum number of ISA bridges 82
  ordering of DMA data 73
  PCI Interrupt Acknowledge Cycle 77
  peripheral-memory-alias 31
  PHB data coherency 71
  PHB Peripheral I/O Space 77
  system-memory-alias 31
  TCEs must be in System Memory 40
  Top of Emulated Memory Register(TEMR) 47
  Translation Control Entry translation 29
reservation protocol 64
Reset 9
restart-rtas 97
Resume 190
ROM 24
ROM, OS 19
RTAS 1, 12, 13
  argument buffer 101
  assume-power-management 123, 126
  cache-control 66, 136
  calling conventions 101
  cell size 102
  check-exception 96, 112
  display-character 118
  event-scan 110
  freeze-time-base 137
  get-power-level 125
  get-sensor-state 121
  get-time-of-day 107
  hibernate 131
  nvram-fetch 96, 105
  nvram-store 96, 105
  read-pci-config 115
  register usage 94
  relinquish-power-management 123, 126
  restart-rtas 97, 104
  set-indicator 96, 119
  set-power-level 124
  set-time-for-power-on 109
  set-time-of-day 108
  start-cpu 140
  stop-self 139
  suspend 129
  system-reboot 96, 135
  thaw-time-base 138
  tokens for functions 99
  write-pci-config 116
RTAS private data area 96

Personal Use Copy - Not for Reproduction

RTAS Status Word Values 103
RTAS stopped state 139, 140
rtas-call 92, 93, 97, 98, 102
rtas-display-device 104
RTC 20

S

SCC 20
self modifying code 60
sequential ordering 57
sequentially consistent 59
serial ordering 57
serial port 20
serialization order 58
server 16
service processor use in booting SMP 222
set-power-level 96
signature 142
single copy atomic accesses 57
single copy atomic stores 57
slbia 50
slbie 50
SMP 215–222
  asymmetric multiprocessor compared 216
  attached processor compared 216
  boot process 218–222
  client 215
  freeze-time-base 137
  Inter-Processor Interrupt use in booting 221
  interrupts 215
  master in booting 219, 221, 222
  operating system support 216
  processor configuration 220
  purpose 215
  quasi-equal timing 217, 218
  requirements 217
  server 215
  service processor in booting 222
  special register for boot 219
  start-cpu 140
  stop-self 139
  symmetry 216
  system organization 216
  thaw-time-base 138
  uniprocessor compared 216
  very special register in booting 220
snooping 59
Soft Reset 86
Solaris 224
special register for booting SMPs 219
Standby 188
start-cpu 139
Status Word Values 103
stop-self 140
storage
  Write-Through 64
storage ordering models 57
strong ordering 57
Suspend 12, 128, 189
symmetric multiprocessor. *See* SMP
system address map
  defined 23
  legend of terms 26
  valid hole and alias combinations 26
System Control Area
  address decoding 32
  defined 24
  non-overlap requirement 29
  PHB defined registers in 78
System Memory
  address decoding 32
  Big-Endian 65
  compatibility holes 24
  contiguous requirement 29
  defined 23
  first area 29
  Little-Endian 65
  non-overlap requirement 29
  reserved addresses 36
System Memory requirements 56
system-memory-alias
  defined 25
  DMA decoding 32

T

TBEN signal 137
TCE. *See* Translation Control Entry
TEM register 47
TEMR (Top of Emulated Memory Register) 47
tightly-coupled multiprocessor. *See* SMP
Time Base Register 137, 138
time of day clock 20
timebase 51
tlbie 50
tlbsync 50
TOD clock 20
Topology 2
Translation Control Entry
  data coherency related 41
  error on accessing 40
  figure showing 43
  no device modification allowed 41

Personal Use Copy - Not for Reproduction

no DMA allowed to TCE table 41
operation 41
requirement to be BE 39
requirement to be in System Memory 40
requirement to implement 39
True Little-Endian System Memory 65

U
uniprocessor compared with SMP 216
used-by-rtas 95, 104

V
very special register in booting SMPs 220
VIA Timer 20

W
Wakeup 191
WARP 224
W-bit aliasing 58
weakly consistent 59
Windows NT 224
Write Through (W bit) 58

Personal Use Copy - Not for Reproduction