Common Hardware Reference Platform Architecture Version 1.0 Change Notice

October 2, 1996

Apple/IBM/Motorola - CHRP Board

This paper presents a chronological list of all changes approved for the published version of the *PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture* (CHRP). These changes are approved by the CHRP Board. These changes will be incorporated into the next version of the CHRP document at the next printing.

Editorial Comments June 6, 1996

The following editorial comments have been received through June 6, 1996.

- Page xx add a paragraph before the paragraph starting “Sample implementations ...” to describe the use of this material by component manufacturers. “This document is one of those needed by vendors building chips and adaptors for use in these platforms. Component manufacturers must produce components which conform to this architecture.”
- Change the second paragraph on page xxiii to read “Big-Endian numbering of bytes is used in this document, unless indicated otherwise. Numbering of bits starts at zero for the most significant bit and continues to the least significant bit, unless indicated otherwise.”
- Change the third paragraph on page xxiii to indicate the table number. “Typographical ... in Table i on page xxiii.”
- Change the table number on page xxiii to be “Table i.”
- Change the last sentence in the paragraph and change the list under 2.5.1.4 on page 18. This paragraph should now read as follows:

“The *description* column ... Two documents are referenced extensively using an abbreviation in this description column. The abbreviations are as follows:

[20] refers to the *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference*. Use this document for additional information on the annotated specifications.

[23] refers to the *Macintosh Technology in the Common Hardware Reference Platform*. Use this document for specific physical, timing, and electrical requirements of the annotated specifications.”

- Remove the two instances of the phrase “(see note 1)” in the Specifications column for the Pointing Device row in Table 2 on page 19.
- Add the phrase “**For 64-bit addressing option in HBs:**” to the start of requirement 3-5 on page 29 and 227.

- Remove a redundant sentence in the last paragraph on page 37. This paragraph should read “Addresses in peripheral I/O Space from BIO+64 KB to BIO + (8MB - 1) are reserved for hardware use. HBs may treat ...”
- At the bottom of page 37 add a note which should read as follows:

“**Software Implementation Note:** When in the discontiguous I/O mode software should not put out an operation to Peripheral I/O Space which crosses a 32-byte boundary. Results from such operations are undefined.”

- Change the table reference for requirement 3-18 on page 39 and 230 to “Table 7 on page 44.”
- Add a new bullet under “Software Implementation Notes” on page 48. The bullet should read as follows: “The amount of address space relocated by the hardware, as specified by the *exception-relocation-size* property, may be larger than what is required by the software environment. In that case, the software environment may reclaim as System Memory some portion of the region which is relocated by the hardware. Memory reclaimed in this manner must not be addressed via the addresses which were relocated, but rather by their actual System Memory addresses.”
- The second sentence in the second full paragraph on page 62 needs the word “be” inserted. The sentence should read “It is also important to be able to ...”
- Requirement 6-2 on page 86 and 240 was an incomplete sentence and should read “The Interrupt Acknowledge register implemented in the interrupt controller must be read to acknowledge an interrupt.”
- Requirement 6-4 on page 86 and 240 should have the phrase “Memory Coherency not required” added and some other changes to be consistent with the PowerPC Architecture naming. The requirement should read “All interrupt control registers must be accessed via Caching Inhibited, Memory Coherence not required, and Guarded Storage mapping.”
- In the second Software Implementation Note on page 86 remove one of the duplicate words “Section” in the last sentence. The sentence should read “ ... see Section 12.2 ...”
- In the last Hardware Implementation Note on page 86 replace the word “are” with “is”. The sentence should read “The number ... supported is ...”
- Requirement 7-10 on page 94 and 241 should be modified to include all the floating point registers, “FPR0-FPR31.”

- Table 12 on page 100 should have “freeze_time_base” changed to “freeze-time-base” and “thaw_time_base” changed to “thaw-time-base” to be consistent with nomenclature in this section.
- Table 29 on page 120 has a missing word for Token 6. The comments column should read “... and must be documented in OS documentation if used.”
- The last sentence on requirement 8-1 on page 141 and 252 should be clarified. This requirement should read as follows:

“Platforms must implement ... is sufficient for a system with a single operating system installed.

a. Platforms must provide an additional 1 KB for each installed operating system beyond the first.”

- The Multi-boot approach documented in section 8.4.6 “Multi-Boot” on page 147 is under revision by the creators and an alternate approach will be published in a future update of this list.
- Add the phrase “**For the Multi-Boot option:**” at the start of requirement 8-23 on page 148 and 254.
- The last paragraph in Section 9.1.1 on page 152 needs some clarification. The paragraph should read “There are some ... between bridges under a single PHB. These uses of LOCK# are permitted.”
- Clarify requirement 9-5 on page 153 and 255 by replacing the first “share”. The requirement should read “PCI devices that do not reside in the Peripheral Memory Space ... interrupt source.”
- Change “in” to “is” in the last sentence of the first paragraph in section 9.2 on page 155. The sentence should read “... programming model is given ...”
- Add a clarifying phrase to requirement 9-15 on page 155 and 257. The requirement should read “When ... retain the programming model as specified in the *PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference*.”
- Table 51 at the bottom of page 163 needs to be corrected. Wording changes are shown in the fifth column, first, third and fourth rows of the following table fragment:

I/O	Memory	DMA - I/O to memory	Data parity on PCI bus	PHB signals PERR#	May be recoverable; see note following table
			Data parity on system bus	Machine check	
		DMA - memory to I/O	Data parity on PCI Bus	Device signals PERR#	May be recoverable; see note following table
			Data parity on system bus	Machine check or PHB signals target abort	
			Memory parity or uncorrectable ECC	Machine check	

- Change “state” to “level” in the fifth sentence in the last paragraph on page 193. The sentence should read “Because Domain 1 and Domain 2 are siblings, the power level ...”
- Remove the words “current and future” in the fourth sentence, first paragraph of section 11.2 on page 197. The sentence should read “Then Open Firmware properties ...”
- Change “Power-on-mask” to “Power_on_mask” in requirement 11-3 on page 201 and 259. The requirement should read “... and the Power_on_mask of the *power-off* ...”
- The last row in the first column of Table 63 on page 224 should read “Windows NT PowerPC Edition.”
- The Trademark list on page 296 should include the following:

“OS/2 International Business Machines Corporation”

- Page 298 item number 20 should read “*PowerPC Microprocessor Common Hardware Reference Platform: I/O Device Reference*, available at various electronic sites.”
- The third paragraph under “Sources for Documents” on page 299 should be reworded to make the URL clearer. It should read “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 lists the information at this directory.”

- Add Morgan Kaufmann's home page in the fifth paragraph under "Sources for Documents" on page 299. The paragraph should read "The documents ... may be ordered from Morgan Kaufmann at 1-800-745-7323 or through their home page at http://www.mkp.com. The PowerPC..."
- The sixth paragraph under "Sources for Documents" on page 299 should be reworded to correct the method of obtaining the document. The paragraph should read "The *Open PIC* ... is available in Acrobat readable format at http://www.amd.com/html/products/pcd/openpic/19725c.pdf. Communications about Open PIC may be sent to openpic@amd.com."
- The last paragraph on page 299 should be reworded to make the URL clearer. It should read "The PowerPC ... are available via anonymous FTP at ftp://playground.sun.com/pub/p1275/bindings/postscript."
- On page 300 under the title "Obtaining Additional Information" change the contact phone number in Asia to "In Asia (81)-775-87-4745 in Japanese."
- Change the last paragraph on page 300 to read "The current version and updates to *PowerPC Microprocessor Common Hardware Reference Platform: 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. The README file lists the information in this directory."
- The first paragraph on page 301 should be reworded to make the URL clearer. It should read "Several white papers ... anonymous FTP to ftp://ftp.austin.ibm.com/pub/technology/spec. The README file lists the files to retrieve."
- The last paragraph on page 301 should be reworded to give the URL. The paragraph should read "VESA documents ... at http://www.vesa.org."

Change Method of 64-bit RTAS Instantiate

This Architecture Change Request (ACR00003) was approved by the CHRP Board on June 6, 1996.

Issue

CHRP currently requires that instantiate-rtas be called in the mode (32 bit or 64 bit) in which the client program wishes to call RTAS. This requires that Open

Firmware be capable of being called in both of these modes since instantiate-rtas is an Open Firmware method.

This approach added the expense of developing and testing Open Firmware in both 32 bit mode and 64 bit mode and the effort of testing FCODE developed for 32 bit systems in a system executing in 64 bit mode.

Changes

- Reword requirement 7-6 on page 93 and 240 to be:

  "RTAS must be called with the SF MSR bit set to match the mode used to instantiate RTAS (0 for instantiate-rtas or 1 for instantiate-rtas-64) and the LE bit set to the same value that was in effect at the time that RTAS was instantiated."

- Add the words "or instantiate-rtas-64" following "instantiate-rtas" in the following locations:

  — the paragraph under 7.2.5 on page 97.

  — Requirement 7-25 on page 97 and page 242.

  — Table 11 on page 98.

  — Requirement 7-26 on page 98 and page 242.

  — Requirement 7-36 on page 102 and page 243.

- Reword requirement 7-34 on page 102 and 243 to be:

  "If the system is a 32 bit system, or if RTAS was instantiated by instantiate-rtas, then all cells in the RTAS argument buffer must be 32-bit sign extended values that are aligned to 4 byte boundaries."

- Reword requirement 7-35 on page 102 and page 243 to be:

  "If the system is a 64 bit system and if RTAS was instantiated by instantiate-rtas-64, then all cells in the RTAS argument buffer must be 64-bit sign extended values that are aligned to 8 byte boundaries."

Require both SCC and 16550

ACR00004 was approved by the CHRP Board on June 6, 1996.

Issue

Even though Mac OS supports both SCC and 16550, there may not be drivers for all devices on either. Also some applications may not support both.

Changes

- Remove the asterisk after the “R” in the Personal column for SCC and 16550 on the Serial Port row in table 2 on page 20.
- Add in the Description column the sentence “It is acceptable to provide the PC style interface or the Apple style interface by means of a plug-in card to achieve certification.”

Power_on_mask Definition

ACR00005 was approved by the CHRP Board on June 6, 1996.

Issue

The Power-off RTAS function is intended for platforms which may or may not be power management capable. In support of platforms which are not there needs to be a means for the platform to inform the OS which power on triggers it supports. A power management capable platform uses the power-management-events node of the device tree to inform the OS which of the architecturally defined power management event types it supports. The device tree of a non power management capable platform would not contain this node.

There needs to be an independent definition of power on triggers and a corresponding mechanism used to inform the OS as to support for a given platform.

Changes

- Page 201 Section 11.2.1.3 and page 259 change requirement 11-3 to read:

“**For the Power Management option:** The Resume_mask of the suspend RTAS call and the Wakeup_mask of the hibernate RTAS call must be defined by the 64 bit quantity generated by ... specified in Table 62 on page199.”

- After requirement 11-3 add a new requirement called 11-3b:

"The Power_on_mask of the power-off RTAS call must be defined as specified in Table <62b>."

■ At the bottom of page 62 add a new table:

Table <62b>. Defined Power on Triggers

Bit	Semantics
0	Power switch
1	External ring indicate (RS 232 RI pin)
2	Internal ring indicate
3	LAN wakeup event
4	Real time clock wakeup event
5	Service processor wakeup event
6-63	Reserved

Author Note: The contents of table 62b were changed by ACR00024. Please refer to this change.

■ On page 127 Section 7.3.7.5 and page 248 change requirement 7-102 to refer to the new requirement and table:
"**If Software controlled** ... implementations, must be a bit mask of power on triggers, refer to 11-3b. If a ..."

No Audio Programming Model

ACR00007 was approved by the CHRP Board on June 6, 1996.

Issue

The CHRP I/O Device Reference specification does not contain a programming model for the audio subsystem.

Change

- Remove the line “Audio Controller” from requirement 9-14 on page 155 and page 257.

Clarify RTAS private data area usage

This Document Error Report (DER00002) was approved by the CHRP Board July 8, 96.

Issue

The RTAS sections on resource allocation and use (7.2.4) and RTAS instantiation (7.2.5) refer to the RTAS private data area in different ways, sometimes causing confusion. Also, handling the case when rtas-size is zero is unclear.

Change

- Page 96: Add to the first paragraph which follows “7.2.4 Resource Allocation and Use.” the following sentence. This memory is subsequently called the “RTAS private data area.”
- Page 97: Modify the paragraph which follows “7.2.5 Instantiating RTAS” as shown in the paragraph below.

“RTAS is instantiated by an explicit client interface service call into Open Firmware. The Open Firmware Device Tree contains a property (rtas-size, under the rtas node) which defines how much real memory RTAS requires for its private data area. The operating system allocates rtas-size bytes of real memory, and then invokes the instantiate-rtas method of the RTAS node, passing the real address of the private data area (or zero, if rtas-size is zero) as the rtas-base-address input argument. Firmware binds RTAS to that address, binds the addresses of devices that RTAS uses, performs any RTAS initialization, and returns the address of the rtas-call function that is appropriate for the current machine state.”

- Page 98: Modify the “Value” column of the first row of Table 11 as shown below:

Table 11. *Instantiate-rtas* Argument Call Buffer

Parameter Type	Name	Values
In	*rtas-base-address*	Real address of RTAS private data area or zero, if rtas-size is zero

■ On page 100 modify the heading row and the "Value" column of the first row of Table 13 as shown below:

Table 13. Open Firmware Device Tree Properties

Property name	Value
rtas-size	integer size of RTAS private data area in bytes or zero if allocation is not required

RTAS Call to Update Flash

ACR00001 and ACR00019 were approved by the CHRP Board on July 26, 1996.

Issue

Customers have asked for the ability to perform system administration operations remotely. Flash update, viewed as a system administration function, fits in this category.

Requiring the customer to use diskettes for updating flash code can be very cumbersome, especially in large installations and in configurations involving parallel systems.

A single way to update flash makes it possible for all OS's to have the capability to update the flash on any vendor's platform with one implementation.

Change

■ Add to table 12 on page 99 two lines following the system-reboot line as shown below:

Table 12. RTAS Tokens for Functions

RTAS Property/Function	Required	Notes
update-flash-and-reboot		See update-flash section
update-flash		

■ At the bottom of page 135 add the Section 7.3.9.2 as defined below.

7.3.9.2 Update-flash

The *update-flash* and *update-flash-and-reboot* functions are described in this section. They are essentially the same, except that the *update-flash-and -reboot* will both update the flash and call for a reboot. It does not return to the operating system if successful.

Requirements:

7-130+1. The RTAS *update-flash* and *update-flash-and-reboot* calls must be implemented using the argument call buffer defined by Table 40+1 on page ???.

Table 40+1 *update-flash* and *update-flash-and-reboot* Argument Call Buffer

Parameter Type	Name	Value
In	Token	Token for *update-flash* or for *update-flash-and-reboot*
	Number Inputs	1
	Number Outputs	1
	Block_list	A real pointer to a block list
Out	Status	0 Success (*update-flash-and-reboot* does not return on success) -1 Hardware cannot be updated -2 Image unacceptable to update program -3 programming failed when partially complete, and the flash is now corrupted -- reboot may fail

The -1 return is to cover the case where some condition prevents RTAS from being able to program the flash at this time. For example, the flash programming power supply is disconnected, a low-level security check (e.g., a switch or jumper) fails, or a test programming probe fails for an unknown reason.

The -2 return is to cover the case where embedded vendor/platform specific information in the image failed to conform to the required format or content for this platform, such as the firmware revision number or a CRC or some other check which was intended to ensure the integrity of the image.

The -3 return is to cover the case where the update failed before the image was fully updated. In this case, the OS has the responsibility for reporting the failure.

7-130+2. The block_list must start with a list length in bytes and must have pairs of real address and length of memory blocks, as shown in Table 40+2 on page ???.

7-130+3. The block list must be a sequence of cells as defined in requirements 7-34 and 7-35.

7-130+4.A memory block included in the block_list must only include System Memory outside that reserved for firmware (both the RTAS data area and Open Firmware's memory defined by real-base and real-size).

7-130+5.A memory block included in the block_list must only include System Memory below 4 GB.

7-130+6.A memory block included in the block_list must be aligned on a 4 KB boundary.

7-130+7.A memory block included in the block_list must not cross a 256 MB boundary.

Table 40+2 Format of Block List

Length of block_list in bytes
Address of memory area 1
Length of memory area 1
...
Address of memory area n
Length of memory area n

7-130+8. The *update-flash-and-reboot* call must test the image to make sure it has the right format and is not damaged, update the flash from the block list and then perform a system reset and reboot, as for the system-reboot call.

7-130+9. The *update-flash* call must test the image to make sure it has the right format and is not damaged, update the flash from the block_list and then return to the OS.

Hardware and Software Implementation Note: Platform vendors should embed vendor-specific information in their flash images to identify the firmware unambiguously and to ensure that the firmware will operate correctly on the platform. Such information might include platform board model and revision numbers covered by the firmware, vendor name, and firmware revision number used for external display. This information should include a CRC or other check which ensures the

integrity of the data. The format of this information is left to the platform vendor.

Software Implementation Note: The execution time for these calls will be in the order of seconds, rather than “a few tens of microseconds” as noted on page 97.

Software Implementation Note: These RTAS flash update programs should display progress, completion, and error information via the display-character mechanism while the flash update is underway, if possible.

Software Implementation Note: The OS does not expect a return from the *update-flash-and-reboot* call other than for the case where the hardware can’t be accessed, the flash image is unacceptable to the RTAS flash update program, or the result of the update corrupted the flash.

Add requirement for PHB to do TCE table extent checking

ACR00013 was approved by the CHRP Board July 26,1996.

Issue

The set-64-bit-addressing OF method sets up the base and size of the TCE table, but currently no requirement exists for the hardware to do anything with the size parameter. The intent of specifying the size was to have the hardware check to see that a device doesn’t try to access PCI Memory Space which is not covered by the TCE table. There is even an existing error defined in Chapter 10 for this error (TCE extent error). This change request adds the missing requirement.

Change

■ On page 39, after requirement 3-18, add the following requirement:

3-18+1.For 64-bit addressing option in HBs: If the address that the PHB would use to access the TCE table (in order to get the TCE) would access outside of the TCE table, then the PHB must create a TCE extent error (see Chapter 10).

RTAS cache-control

ACR00015 was approved by the CHRP Board on July 26, 1996.

Issue

The requirement to take a phandle as input for the cache-control call is extra overhead for both the firmware and the OS. The correlation to the correct cache can be made by indicating the level of the cache with respect to the current processor. For SMP, this relies on the processor's PIR being preserved by the OS from boot to RTAS call, so firmware and RTAS can assume the same PIR contents.

Change

- Remove the supporting information from requirement 4-2 on page 50 and put it in a new paragraph under all the requirements. Add a reference to chapter 12 in this paragraph.

4-2. **For the Symmetric Multiprocessor option:** Multiprocessing platforms must use only processors which implement the processor identification register.

"See PowerPC 604 RISC Microprocessor User's Manual [6] for a definition of the processor identification register (PIR). See {reference to chapter 12} for additional information on the implementation, initialization, and use of the PIR."

- Change the "Cache_id" row of Table 41 on Page 136 as follows:

Table 41. cache-control Argument Call Buffer

Parameter Type	Name	Values
	Cache_id	level of the cache (1 = L1, 2 = L2, etc.)

- Add the following requirements and notes to Page 217.

12-9. For the Symmetric Multiprocessor option: SMP operating systems must not alter the PIR(s).

12-10. For the Symmetric Multiprocessor option: The platform must initialize each PIR to a value which is unique for each processor and corresponds to the Open PIC identity of that processor.

12-11. For the Symmetric Multiprocessor option: If firmware and RTAS use the PIR for processor identification, then they must use the same PIR value to refer to the same processor.

12-12. For the Symmetric Multiprocessor option: Firmware must not alter the PIR(s) once they are initialized.

Software Implementation Note: The operating system is permitted to read the PIR to determine the identity of the processor on which it is running. To minimize the differences between uniprocessor and multiprocessor run-time environments (since a uniprocessor might not have a PIR), the operating system may wish to copy this and other per-processor data into a data structure based off a pointer, perhaps kept in a SPRG.

Hardware and Software Implementation Note: Some processors which implement the 6xx bus use a portion of the PIR value to help identify bus transactions. These implementations typically provide a set of pins which are read when power is first applied to the processor to hardware-initialize the PIR value. On such processors, firmware and software should never attempt to write the PIR value, since the integrity of transactions on the bus would likely be lost as a result.

- A supporting change to the CHRP Open Firmware Binding, Version: 1.5 requires the deletion of section 6.3 on page 38.

NVRAM changes from System Binding

ACR00016 was approved by the CHRP Board on July 26,1996.

Issue

These changes are a result of work completed on the CHRP Open Firmware System Binding.

Change

- On page 11, Section 2.1.3.4 add to the second paragraph "See *PowerPC Microprocessor Common Hardware Reference Platform System Binding to IEEE Std 1275-1994 Standard for Boot (Initialization, Configuration) Firmware* for information relating to the multiboot process."

- Add the following note at the end of Section 8.2 on page 142.

"**Software Implementation Note:** Although the data areas of CHRP NVRAM partitions are not required to have error checking, it is strongly recommended that the system software implement robust data structures and error checking. Loss of NVRAM structures due to data corruption can be catastrophic, potentially leading to OS reinstallation and/or complete system initialization."

- On Page 143, Table 47, in the "Description" column for the "checksum" row add a clarifying statement to the last sentence. This sentence should read "The checksum algorithm ... value. A valid header cannot have a checksum of zero."

- On Page 143, Table 47, in the "Description" column for the "length" row the sentence "This allows ... =1MB." should be replaced with "A length of zero is invalid."

- On Page 143, Table 47, in the "Description" column for the "name" row the first paragraph should be replaced with the paragraph below:

"The name field is a 12 byte string (or a null-terminated string of less than 12 bytes) used to identify a particular partition within a signature group. In order to reduce the likelihood of a naming conflict, each platform-specific or OS-specific partition name should be prefixed with a company name as specified under the description of the "name" string in the *IEEE Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices,* that is, a company name string in one of the three forms described in the reference, followed by a comma (","). If the company name string is null, the name will be interpreted as 'other'."

- Change Table 48. NVRAM Signatures on page 144 as follows:

Table 48. NVRAM Signatures

Signature	Signature Type	Ownership Class	# Required	Description
0x02	Support Processor	Any support processor	0 to n	Reserved for support processor use.
0x50	Open Firmware	Firmware	1	Open Firmware Configuration Variables.
0x51	Firmware	Firmware	0 to n	General firmware usage
0x52	Hardware	Firmware	1 to n	These partitions are used to store administrative machine data such as serial numbers, ec levels, etc. This is often referred to as Vital Product Data (VPD).
0x70	System	Global	1	The System partition is used to store environment variables that must be accessed by firmware and operating systems.
0x71	Configuration	Global	0 to n	Configuration partitions are used to store configuration data generated by the system firmware or the operating system that is required at boot time to properly configure the system.
0x72	Error Log	Global	0 to n	This partition is used to store the error logs built as a result of various RTAS calls.
0x7E	Vendor-defined	Global	0 to n	"name" prefix required, see Table 47 on 143.
0x7F	Free Space	Global	0 to n	This signature is used to mark free space in the NVRAM array. The **name** field of all signature 0x7F partitions must be set to 0x7...77.
0xA0	OS	Any OS	0 to n	General operating system usage.

Note: Any signature not defined above is reserved.

- On page 145, Section 8.4.3, System (0x70), replace the first paragraph with "System partitions are used for storing information (typically, configuration variables) accessible to both Open Firmware and operating systems. Refer to the *PowerPC Microprocessor Common Hardware Reference Platform (CHRP) System Binding to: IEEE Std 1275-1994* for the definition of the contents of the System partition named *common*.
- On page 145, Section 8.4.3, System (0x70) delete the Software Implementation Note.
- Delete Section 8.4.6, "Multi-Boot (0x73)" on page 147.

Indicate Required Indicators

ACR00017 was approved by the CHRP Board July 26, 1996.

Issue

Table 12 on page 99 indicates for set-indicator that “Some specific indicators are required and some are optional”. Section 7.3.6.2, “Set-indicator”, does not indicate which indicators are required.

Change

- Add a new requirement worded as follows:

7-88+. Of the indicator types defined by Table 29 on page 120, RTAS must implement at least Tone Frequency and Tone Volume.

- In Table 29 on page 120 for the “Tone Volume” row and “Examples/Comments” column add the following explanation sentence. “RTAS should select the closest implemented volume for values between zero (off) and 100 (full on).

RTAS PCI config read/write clarification

ACR00022 was approved by the CHRP Board on July 26,1996.

Issue

CHRP Specification Sections 7.3.5.1 and 7.3.5.2 do not say anything about the endianess of the data in configuration space and whether it should be returned “as-is” or “byte-reversed” depending on the endianess of the processor. Requirement 5-12 would infer that the byte-reversed instructions should be used if the processor is in Big-Endian mode (since the data in PCI configuration space is always Little-Endian), but the requirement is not clear for RTAS since RTAS is not considered software.

Change

- On page 115 add the following requirement and software note:

7-72+. RTAS must follow the rules of Table 9 on page 76 when accessing PCI configuration space.

Software implementation Note: Since PCI Configuration space is defined to be Little-Endian, RTAS will access this area using the byte-reversed forms of the Load and Store instructions when operating in Big-Endian mode (LE=0). In this fashion, the values passed are defined to be Little-Endian when LE=1 and Big-Endian when LE=0.

RTAS Hibernate

DER00004 was approved by the CHRP Board on July 26, 1996.

Issue

AIX already has all the logic to write out the hibernation image to disk. We see little reason to change that. But there may be reasons that are platform specific that would lead us to call hibernate to power off the platform for hibernation. The platform may want to output indicators that we do not know of, the platform may have different triggers for resume than for normal power on, etc. So we are thinking of using hibernate with a null block list (first word, length in bytes, of the block list is 0).

Change

Add a note at the end of section 7.3.8.1.2 on page 134

Software Implementation Note: The OS is not required to specify a non-zero amount of data. RTAS should be prepared to deal with a pointer to a block list describing no non-zero-sized memory areas. That is to say, the block list would contain a length in bytes, and one or more sets of area descriptions each containing a length of zero. In this case RTAS would put the system in the hibernate state without recording any data.

Load and Store Ordering Rules

ACR00006 was approved by the CHRP Board on September 23, 1996.

Issue

Requirement 5-6 only solves a small part of the problem described in the PCI Local Bus Specification version 2.1, on pages 116-117. Since it does not solve the whole problem, and since non-CHRP systems deal with this PCI anomaly, this change deletes 5-6, which is difficult to implement in the PHB.

Change

- On page 72, delete requirement 5-6.
- On page 72, replace the entire last paragraph which reads: "With the exception of the one case ... operations from the processors)." with the following note:

Software Implementation Note: The PCI architecture delayed transaction operations creates a potential data inconsistency problem when two entities are accessing the same address when that address is on the opposite side of a PCI to PCI bridge which supports delayed transactions. In particular, if a PCI device requests a read of an address on the other side of one of these PCI to PCI bridges and that read gets delayed, and then a second device writes to that address through the same bridge and then that second device reads the exact same address with the same length as the first device, the second device will get the data from the first device's read (instead of the data that it just wrote). The two devices doing the reading and writing can be the same device; for example, it might be the PHB and the *Loads* and *Stores* to the same address may be coming from the same processor or different processors. This potential delayed transaction data inconsistency scenario is documented in the PCI Local Bus specification.

- On page 74, replace the last paragraph of section 5.1.2.3 which reads "Except for scenario 1, software must create the appropriate protocols to assure that the problem does not occur. Scenario 1 is prevented from happening by requirement 5-6." with the following:

  Software should create the appropriate protocols to avoid these scenarios.

Identification of Vendor-unique Extended

Error Log Formats

ACR00014 was approved by the CHRP Board on September 23, 1996.

Issue

The CHRP extended error log provides a means of defining vendor-specific log formats, indicated by a format indicator of 12, 13, 14, or 15 in byte 2, bits 4:7 of the general extended error log format (see page 176 in CHRP). However, if two companies both decide to define format 12 for their own vendor-specific purposes, there is no way to tell the formats apart. The same problem occurs for any vendor-specific log data that is appended after the standard 40-byte log. Since this log data may get forwarded across the network or to a service provider, proper identification of the log data is critical.

This ACR requires that for vendor-specific error log formats, the first four bytes of that format must contain a unique identifier for the company that has defined the format. This is being raised as an ACR so that all vendors who use vendor-specific formats will implement it consistently, preventing misinterpretation of log content.

Change

- In table 54 on page 176, change the line in the Description column for Byte 2, Bits 4:7 that says “(12-15) Vendor specific” to say “(12-15) Vendor specific, see Table 54B on page 22”
- Add a Table 54B, as follows:

Table 54B. Vendor-specific error log format, bytes 12-39

Byte	Bit(s)	Description
12-15	Company identifier of the company that has defined the format for this vendor specific log type.	
15-39	Vendor-specific log data	

- On page 175, add a requirement and description.

10–18. When extended product-unique error log data is appended after the first 40 bytes of the Extended Error Log Format, the first 4 bytes of this ex-

tended log data (bytes 40-43) must contain a company identifier for the company that defined the format of this extended log data.

Requirement 10–18 and Table 54B on page 22 require that four bytes of the vendor-specific format contain a unique identifier for the company that has defined the format. The description of the “name” string in the “IEEE 1275 Standard for Boot (Initialization Configuration) Firmware: Core Requirements and Practices” provides alternatives for defining this identifier. Examples of these unique identifiers include stock ticker labels and Organizationally Unique Identifiers (OUIs). Since the different options in IEEE 1275 provide different guarantees of uniqueness and different identifier lengths, the company should use its best judgement in selecting a unique identifier that fits the four character field. The length of this field is limited to 4 bytes to conserve available log data space. As an example, if Allied Information Monitoring (a fictional name for the purposes of this example) were to create a vendor-specific log format 12, then bytes 12-15 of such a log may contain “AIM<null>”.

This identifier is intended to apply to the company that defines the specific format, and may be used by other companies that wish to be compatible with that format. For example, if another company wanted to take advantage of existing support in one of the operating systems by using an AIM-specific error log format for logs generated on their own platform, their log would have to contain an identifier of “AIM<null>”.

- Update Figure 12 to indicate bytes 40-43 as the location for the company identifier.

RTAS set-time-for-power-on

ACR00018 was approved by the CHRP Board on September 23, 1996.

Issue

The current specification places no restrictions on the time allowed for this call. Some hardware supports about a 30 day horizon.

Change

- On page 109, correct the table reference in requirement 7-55 to “Table 21 on page 109.”
- Page 109. Add the following requirements prior to requirement 7-56:

7–56+1. Hardware must support power on times of up to four weeks into the future, at a minimum.

7–56+2.RTAS must schedule the time for power on as close as it can approach to the desired time.

■ Page 109. Add a note following these new requirements:

Software Implementation Note: Hardware limitations on the duration of the power-on timer may result in power-on sooner than requested by software. If present in the RTAS node, the Open Firmware property "power-on-max-latency" gives in days the maximum power-on duration capability of the hardware. If the property is not present, software should expect the default of a maximum of 28 days. A "day" is defined as 24 hour increments from the current time.

Get-sensor-state return of state and value

ACR00021 was approved by the CHRP Board on September 23, 1996.

Issue

This ACR addresses three items:

1. Environmental sensors such as Thermal and Battery voltage may be monitored within a platform - for example by a support processor. If the value goes outside of some predefined limit an EPOW may be invoked. Software should be able to see what sensor was out of limits and which were normal. For instance, an EPOW of WARN_COOLING does not tell software which temperature sensor or fan was out of limits in the case of multiple sensors. In addition, when the sensor returns to within limits there is no indication that the "WARN_COOLING event is over. This change provides information to an operating system to allow it to determine these conditions.
2. The Units of the Battery Voltage sensor are in volts which provides little precision for this sensor. The units should be changed to millivolts to give more precision.
3. Table 30 refers to "Current sensor's state" and Table 31 has a column labeled "Defined Values" which is the returned value. Clarify this nomenclature.

Change

■ Add a requirement after 7-89:

7–89+1. If a platform tests sensor values against limits, then RTAS must return the result of these tests using the status output parameter."

Change the "Out" row of table 30 as follows:

Table 55. *get-sensor-state* Argument Call Buffer

Parameter Type	Name	Values
Out	Status	13: Sensor value >= critical high 12: Sensor value >= Warning high 11: Sensor value normal 10: Sensor value <= Warning low 9: Sensor value <= critical low 0: Success -1: Hardware Error -2: Hardware Busy, Try again later -3: No such sensor implemented
	State	Current value as defined in the |"Defined Values" column of Table 31 on page 121

■ In table 31 on page 122 change the "Defined Values" column of the "Battery Voltage" row to read "Voltage (in units of millivolts)".

■ Add implementation notes to explain the additional status codes.

Hardware Implementation Note: Some platforms may compare the value of environmental sensors (such as the Battery Voltage or Thermal Sensor) to some limits. When the value of the sensor meets or exceeds a limit the platform may take some action. RTAS makes the operating system aware of the relationship of the sensor values to the limit by using the status code to return this information.

Software and Hardware Implementation Note: The meaning of these limits are as follows:

■ - "Critical high" means the sensor value is greater than or equal to this limit. A platform may take some action and may initiate an EPOW (see section 10.2.2). Operating systems may take some action to correct this situation or to perform an orderly shutdown.

- - “Warning high” means the sensor value is greater than or equal to this limit, but less than the critical high limit. A platform may initiate a warning EPOW. Operating systems may take some action to bring this reading back into the normal range.
- - “Normal” means that RTAS is aware of the limits and the value is within these operating limits.
- - “Warning low” means the sensor value is less than or equal to this limit, but greater than the critical low limit. A platform may initiate a warning EPOW. Operating systems may take some action to bring this reading back into the normal range.
- - “Critical low” means the sensor value is less than or equal to this limit. A platform may take some action and may initiate an EPOW. Operating systems may take some action to correct this situation or to perform an orderly shutdown.

Where:

- A “critical” state is defined as a condition where the sensor value of the measured item indicates that it is outside the allowable operating parameters of the system, and that a failure is imminent unless some immediate action is taken.
- A “warning” state is defined as a condition where the sensor value of the measured item indicates that it is outside the expected operating parameters for normal operation, but has not yet reached a critical state. The variance is significant enough that either system software or an operator may want to take some action to bring the parameter back into the normal range.

Platform Implementation Note: The existence of this sensor state reporting capability should not be construed as a requirement to have any limits on sensors or to always have all four limits.

RTAS query-cpu-stopped-state

ACR00023 was approved by the CHRP Board on September 23, 1996.

Issue

The RTAS rule about allowing only one cpu to call RTAS at a time is somewhat problematical in the case of the stop-self call. There is no way to know when the cpu is stopped so another call can be made, since the call does not return.

The basic problem with the RTAS stop-self service is that it is a non-terminal RTAS call that does not return to the caller. This basically implies that an operating system can not acquire the RTAS lock prior to calling stop-self.

Change

■ Page 95: Change Requirement 7-13 as follows:

7–13. Except as noted in requirements 7-19 and 7–19+1, the operating system must ensure that RTAS calls are not re-entered and are not simultaneously called from different processors in a multi-processor system.

■ Page 96: Add requirement 7–19+1:

7–19+1. The stop-self service need only be serialized with calls to the stop-self, start-cpu, and set-power-level services. The operating system is free to call other RTAS services on other processors while a processor is being stopped.

■ Page 100: Add to the bottom of Table 12 query-cpu-stopped-state

Table 12. RTAS Tokens for Functions

RTAS property/function	Required?	Notes
query-cpu-stopped-state	Required in SMP Platforms	

■ Page 130: Change requirement 7-107 to read as follows:

7–107. In an SMP system, all other processors must be in the stopped state before invoking suspend. This is the responsibility of the operating system.

■ Page 140+: add Section 7.3.11.5 as follows:

7.3.11.5 query-cpu-stopped-state

The query-cpu-stopped-state primitive is used to query a different processor to determine its status with respect to the RTAS stopped state.

Requirements:

7-143. **For the Symmetric Multiprocessor option:** RTAS must implement a query-cpu-stopped-state call which returns the CPU_status of the processor specified by the CPU_id parameter. This call must be implemented using the argument call buffer defined by Table 46+1 on page 29.

Table 46+1. query-cpu-stopped-state Argument Call Buffer

Parameter Type	Name	Values
In	Token	Token for query-cpu-stopped-state
	Number Inputs	1
	Number Outputs	2
	CPU_id	Token identifying the processor to be queried, obtained from the reg value for the CPU in the Open Firmware device tree
Out	Status	0: Success -1: Hardware Error -2: Hardware Busy, Try again later
	CPU_status	0: The processor is in the RTAS stopped state 1: Stop-self is in progress 2: The processor is not in the RTAS stopped state

Firmware Implementation Note: RTAS serialization may be required between the stop-self and the query-cpu-stopped-state calls.

Software Implementation Note: The operating system must perform a stop-self on the desired processor, then periodically call query-cpu-stopped-state on another processor until the desired processor is stopped before calling set-power-level to power off the desired processor.

Defined Power on Triggers

ACR00024 was approved by the CHRP Board on September 23, 1996.

Issue

ACR 00005 added table 62b for power on triggers, but defined different mask positions than table 62. It would be convenient for software, if the power on mask was a subset of table 62 and included every possible power on event.

Clarify that the power on events supported on a specific platform should be reported to the operating system via the property “power-on-triggers” of the Open Firmware device tree.

Change

- Remove the second sentence in requirement 7-102, “If a..., then hardware should....”. This sentence is not a requirement and should be in the clarification text below the requirement.
- Add a paragraph below requirement 7-102.

If a bit in the Power_on_mask is set to 1, then the hardware should enable the corresponding hardware power-on mechanism. The Open Firmware property “power-on-triggers” in the RTAS node is used to indicate which of these triggers are supported by the particular platform.

- Change table 62 to add the service processor event.

Table 62. Defined Power Management Event Types

Event	Type Field Value	Semantics
Service processor	113	A service processor initiated power management event
Reserved	114-159	Reserved for future use.

- Replace table 62b which was defined in ACR00005.

Table 62b. Defined Power On Triggers

Bit	Event	Semantics
0	Power Switch On	Type Field Value 96 In Table 62

Table 62b. Defined Power On Triggers

Bit	Event	Semantics
1	Reserved	Reserved for future use
2	Lid Open	Type Field Value 98 in Table 62
3-4	Reserved	Reserved for future use
5	Wake Button	Type Field Value 101 in Table 62
6-7	Reserved	Reserved for future use
8	Switch to Battery	Type Field Value 104 in Table 62
9	Switch to AC	Type Filed Value 105 in Table 62
10	Keyboard or mouse activity	Type Field Value 106 in Table 62
11	Reserved	Reserved for future use
12	Enclosure Closed	Type Field Value 108 in Table 62
13	Ring Indicate	Type Field Value 109 in Table 62
14	LAN Attention	Type Field Value 110 in Table 62
15	Time Alarm	Type Field Value 111 in Table 62
16	Configuration change	Type Field Value 112 in Table 62
17	Service Processor	Type Field Value 113 in Table 62
18-63	Reserved	Reserved for future use

Add System Bus

ACR00025 was approved by the CHRP Board on September 23, 1996.

Issue

Since devices are allowed on the system bus, the I/O Device Reset States table should be expanded to include the system bus.

Change

■ Add a new row on Page 10, Table 1 as follows:

Table 1>. I/O Device Reset States

Bus	Devices Left Open by Open Firmware	Other Devices
System	Configured per OF Device Tree Interrupts inactive DMA inactive No outstanding I/O operations	The device is in hardware reset state (see note) or inactive: Interrupts inactive DMA inactive

Note: May optionally be *configured*, but must be *inactive*.

Clarification of display-character

ACR00028 was approved by the CHRP Board on September 23, 1996.

Issue

CHRP does not specify the character set which is displayable with the RTAS display-character command.

Change

■ Change Section 7.3.6.1, 4th line to “This call is intended to display the alpha-numeric characters . . .”

■ Add requirement 7-86+1 after requirement 7-86:

7–86+1. The ASCII characters that must be displayed are generally those coded from 0x20 to 0x7E.

- a-z (0x61-0x7A)
- A-Z (0x41-0x5A)
- 0-9 (0x30-0x39)
- SPACE (0x20)

- !”#$%&’()*+,-./ (0x21-0x2F)
- :;<=>?@ (0x3A-0x40)
- [\]^_‘ (0x5B-0x60)
- {|}~ (0x7B-0x7E)

Software Implementation Note: Care should be taken in using the full character set for all systems as some characters may not be available or may display in a different fashion. For instance, the currency symbol, $ (0x24), may be modified to a national currency symbol. Other currently known differences occur for the reverse slant, \ (0x5C), and the tilde, ~ (0x7E).

RTAS Functions

Table 12 on page 24 shows the cumulative changes of all new and modified RTAS functions.

Table 12. RTAS Tokens for Functions

RTAS property/function	Required?	Notes
restart-rtas	Required	
nvram-fetch	Required	Execution time proportional to amount of data
nvram-store	Required	Execution time proportional to amount of data
get-time-of-day	Required	
set-time-of-day	Required	
set-time-for-power-on		
event-scan	Required	
check-exception	Required	
read-pci-config	Required	
write-pci-config	Required	
display-character		
set-indicator	Required	Some specific indicators are required, and some are optional

Table 12. RTAS Tokens for Functions ***(Continued)***

RTAS property/function	Required?	Notes
get-sensor-state		
set-power-level	Required in Power Managed Platforms	
get-power-level		
assume-power-management		
relinquish-power-management		
power-off		Provided for platforms with software controlled power off capability, with or without other Power Management capability
hibernate		
suspend		
system-reboot	Required	
update-flash-and-reboot		See update-flash section
update-flash		
cache-control	Required in Power Managed Platforms	
	Required in SMP Platforms	
freeze_time_base	Required in SMP Platforms	Turn off time base
thaw_time_base		Turn time base back on
stop-self		
start-cpu		
query-cpu-stopped-state	Required in SMP |Platforms	

This is the last page of the CHRP Version 1.0 Change Notice