

Architecture for Power Management on High Speed Serial Bus

Version 0.72

September 23, 1998

Sponsored by:

Not Yet Approved by:

Abstract: The purpose of this document is..

Keywords: 1394, power, power management

Revision History

Date	Revision	Comment
June 3, 1997	1.04	<ul style="list-style-type: none"> • Incorporated Informative section on the Power management model (4.1-4.5) • Clarified Power Manager operation and protocol (5.1.3; 5.2; 8.1) • Added support for Cable Power Sourcing capabilities and control (5.2.4, 5.2.5; 6.3.2) • Documented Wake-up based on ACK_TARDY protocol(8.2) • Added Notification events for Cable Power Sourcing capabilities change and Power state change
July 28, 1997	1.05	<ul style="list-style-type: none"> • Revised node power states to include Link-Off/PHY on (4.2; 5.2.2). • Deleted references to wakeup; covered in 1394a (4.5,8.2) • Deleted sections about STATE_SET and STATE_CLEAR registers(5.1); edited section about BusInfoBlock (6.1); covered in 1394a • Changed addressing of and clarified use of the POWER_CHANGE register (5.1.1; 5.2) • Fixed Cable Power Sourcing State register to be consistent with Power State register (added requested level field) (5.2.4) • Deleted the UNIT_NOTIFICATION_ADDRESS register (5.3, 5.3.2) • Revised Notification to refer only to power manager events (4.1, 4.5, 8.1)
October 24, 1997	0.7	<ul style="list-style-type: none"> • Changed name of Spec to 'Power State Management' and reset revision number to 0.70 • Re-wrote Model section for clarity • Deleted all node state definitions and controls • ...
December 29, 1997	0.7a	<ul style="list-style-type: none"> • Added first draft of power policy ownership sections
December 30, 1997	0.7b	<ul style="list-style-type: none"> • Revised power policy ownership sections • Made the concept of "manageable" nodes explicit • Reorganized sections so all register definitions are at the end of the document
January 2, 1998	0.7c	<ul style="list-style-type: none"> • Added Visio drawings
January 5, 1998	0.7d	<ul style="list-style-type: none"> • Rewrote section 4.4 <p>Moved Device Attachment scenario to "Power management procedures" section and edited to make consistent with section 4.4</p> <p>Revised Unit_Power_Level entry</p> <p>Eliminated Node_Power_Level entry</p>
January 19, 1998	0.71	<ul style="list-style-type: none"> • Revised configuration ROM entry definitions • Revised Unit_Power_Management register definitions • Got rid of all Word drawings so drawings print out okay. • Revised device attach sequence of steps
September 23, 1998	0.72	<ul style="list-style-type: none"> • Added Bus_ID to Power Change register

Table of Contents

1	OVERVIEW	1
1.1	Objectives of this specification	1
1.2	Applicability of this specification	1
2	REFERENCES.....	1
3	DEFINITIONS AND NOTATION.....	3
3.1	Definitions	3
3.1.1	Conformance	3
3.1.2	Glossary.....	3
3.1.3	Abbreviations	4
3.2	Notation	4
3.2.1	Numeric values	5
3.2.2	Bit, byte and quadlet ordering.....	5
3.2.3	Register specifications.....	6
4	POWER STATE MANAGEMENT MODEL (INFORMATIVE)	8
4.1	The Generic Device Power Management Model (informative).....	8
4.1.1	Power Policy and Policy Owners	8
4.1.2	Power Control	9
4.2	Unit power states (informative)	9
4.3	Link, PHY, and port power states (informative).....	10
4.4	Power Policy Owners and the Power States (informative).....	11
4.4.1	Unit Power Policy	11
4.4.2	Link Power Policy	11
4.4.3	PHY Power Policy	11
4.4.4	Port Power Policy.....	11
4.5	1394 Bus Power Manager (informative)	11
4.6	Batteries and battery groups (informative)	12
5	POWER MANAGER.....	12
5.1	Determination of the power manager	12
5.2	Power manager responsibilities	13
5.2.1	Managing power domains.....	13
5.2.2	Manageable and Non-Manageable Nodes.....	15
5.2.3	Notifications and the Power Manager	16
5.2.4	Power policy ownership arbitration.....	16
5.3	Example Scenarios	17
5.3.1	Single-initiator hard disk node scenario.....	17
5.3.2	Multiple PC nodes logged on to a multi-initiator hard disk node	17
5.3.3	PC node logged on to a multi-initiator hard disk node with exclusive use.....	18
5.3.4	Laptop PC switches for power provider to power consumer	18
6	POWER MANAGEMENT PROCEDURES	18
6.1	Device Attachment	18
6.1.1	Basic Power Management Support at Device Attach.....	18
6.1.2	Remote Wakeup Support	19
6.2	Power state change.....	19

6.3	Wake-up	20
7	POWER MANAGEMENT REGISTERS	20
7.1	Power manager registers	20
7.1.1	POWER_CHANGE register	22
7.1.2	Use table registers	22
7.2	Unit register	23
7.2.1	Unit_Pwr_State register	23
7.2.2	Unit_Pwr_Control register	24
7.2.3	BATTERY_STATE register	25
8	CONFIGURATION ROM	25
8.1	Bus information block	27
8.2	Root directory entries	28
8.2.1	Node_Power_Directory entry in the Root_Directory	28
8.2.2	Unit_Power_Directory entry in the Root_Directory	28
8.2.3	Power_Management_Registers entry in the Root_Directory	29
8.3	Node power directory entries	29
8.3.1	Node_Power_Management entry	30
8.3.2	Link_Power_State entry	30
8.3.3	PHY_Port_PwType entry	31
8.3.4	Power_Source_State entry	31
8.3.5	Node_Power_Characteristics entry	32
8.4	Unit power directory entries	32
8.4.1	Unit_Power_Management entry	32
8.4.2	Unit_Power_State entries (new version)	33
8.4.3	Battery_Group entry	33

1 Overview

IEEE Std 1394-1995, Standard for a High Performance Serial Bus, defines facilities in the cable environment for the distribution of power to connected devices. Although rudimentary facilities are also specified to permit Serial Bus devices to report their power requirements (*e.g.*, the *pwr_class* field in the self ID packets), the standard does not provide sufficient guidance to permit different vendors to implement power management protocols that guarantee interoperability of devices on the bus.

1.1 Objectives of this specification

This specification extends the power management facilities of IEEE Std 1394-1995 to achieve the following goals:
Enable standard system support for power-managed Serial Bus devices in order to promote integration with desktop computers or other peer applications (such as set top boxes).
Enable power-efficient Serial Bus devices by providing a framework for reporting and controlling low-power modes.
Improve the user's perception of device availability through such features as soft power-off, rapid power-on, and the automatic wake-up of remote applications upon a signal from the device.

1.2 Applicability of this specification

This specification is about the manipulation of 1394-bus specific power control mechanisms by software entities residing on the bus for the purpose of conserving power while maximizing device availability for end-users. While any entity on the Serial Bus could provide the described control, the intention is that software running on an intelligent computing device such as a personal computer could best implement Serial Bus-wide power management. Such software offers the benefit of closer integration of various PC applications with units within nodes on the Serial Bus.

The software that provides power management control for units within all the manageable nodes on the Serial Bus is called the Power Manager. There is only one Power Manager at a time on the Serial Bus. Not all nodes on the Serial Bus are manageable by the Power Manager. The only nodes on the Serial Bus that may be power managed by the Power Manager are

Outside the PC or set top box chassis.
Not in a Device Bay subsystem.

For more information about the Power Manager, see section 4, "Power Management Model (Informative)."

Manageable devices may be bus-powered or self-powered. The principal difference is that self-powered devices may consume power at more than one voltage level.

2 References

The following standards contain provisions that, through reference in this text, constitute provisions of this specification. At the time of publication, the editions indicated were valid. All standards are subject to revision; users of this specification are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

IEEE Std 1394-1995, Standard for a High Performance Serial Bus

ISO/IEC 13213:1994, Control and Status Register (CSR) Architecture for Microcomputer Buses

IEEE P1394a, Draft Standard for a High Performance Serial Bus (Supplement)

3 Definitions and notation

3.1 Definitions

3.1.1 Conformance

Several keywords are used to differentiate levels of requirements and optionality, as follows:

expected: A keyword used to describe the behavior of the hardware or software in the design models assumed by this specification. Other hardware and software design models may also be implemented.

ignored: A keyword that describes bits, bytes, quadlets, octlets or fields whose values are not checked by the recipient.

may: A keyword that indicates flexibility of choice with no implied preference.

reserved: A keyword used to describe objects—bits, bytes, quadlets, octlets and fields—or the code values assigned to these objects in cases where either the object or the code value is set aside for future standardization. Usage and interpretation may be specified by future extensions to this or other standards. A reserved object shall be zeroed or, upon development of a future standard, set to a value specified by such a standard. The recipient of a reserved object shall not check its value. The recipient of a defined object shall check its value and reject reserved code values.

shall: A keyword that indicates a mandatory requirement. Designers are required to implement all such mandatory requirements to assure interoperability with other products conforming to this specification.

should: A keyword that denotes flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “is recommended.”

3.1.2 Glossary

The following terms are used in this specification:

byte: Eight bits of data.

doublet: Two bytes, or 16 bits, of data.

initial node space: The 256 terabytes of Serial Bus address space that is available to each node. Addresses within initial node space are 48 bits and are based at zero. The initial node space includes initial memory space, private space, initial register space and initial units space. See ISO/IEC 13213:1994 and IEEE Std 1394-1995 for more information on address spaces.

initial register space: A one kilobyte portion of initial node space with a base address of FFFF F000 0000₁₆. Core registers defined by ISO/IEC 13213:1994 are located within initial register space as are Serial Bus-dependent registers defined by IEEE Std 1394-1995.

initial units space: A portion of initial node space with a base address of FFFF F000 0400₁₆. This places initial units space adjacent to and above initial register space. The CSR's and other facilities defined by unit architectures are expected to lie within this space.

kilobyte: A quantity of data equal to 2^{10} bytes.

node ID: The 16-bit node identifier defined by IEEE Std 1394-1995 that is composed of a bus ID portion and a physical ID portion. The physical ID is uniquely assigned as a consequence of Serial Bus initialization.

octlet: Eight bytes, or 64 bits, of data.

power domain: A portion of the Serial Bus topology that receives power from a single power source. The fact that cable power shall not be propagated by nodes that supply power, along with other power class and topology information in the self ID packets, permits the power manager to parse a Serial Bus topology into disjoint power domains.

quadlet: Four bytes, or 32 bits, of data.

register: A term used to describe quadlet aligned addresses that may be read or written by Serial Bus transactions. In the context of this specification, the use of the term register does not imply a specific hardware implementation. For example, in the case of split transactions that permit sufficient time between the request and response subactions, the behavior of the register may be emulated by a processor within the module.

split transaction: A transaction that consists of separate request and response subactions. Subactions are considered separate if ownership of the bus is relinquished between the two. A transaction that is not split is called a unified transaction.

system memory: The portions of any node's memory resource that are directly addressable by a Serial Bus address and which accepts, at a minimum, quadlet read and write access. Computers are the most common example of nodes that make system memory addressable from Serial Bus, but any node, including those usually thought of as peripheral devices, may have system memory.

terabyte: A quantity of data equal to 2^{40} bytes.

transaction: An exchange between a requester and a responder that consists of a request and a response subaction. The request subaction transmits a Serial Bus transaction such as quadlet read, block write or lock, from the requesting node to the node intended to respond. Some Serial Bus commands include data as well as transaction codes. The response subaction returns completion status and sometimes data from the responding node to the requesting node.

unified transaction: A transaction in which the request and response subactions are completed as an indivisible sequence. Between the initiation of the request and the completion of the response, subactions by nodes other than the requester or the responder are blocked. A transaction that is not unified is called a split transaction.

unit: A component of a Serial Bus node that provides processing, memory, I/O or some other functionality. Once the node is initialized, the unit provides a CSR interface that is typically accessed by device driver software at an initiator. A node may have multiple units, which normally operate independently of each other.

unit architecture: The specification of the interface to and the behaviors of a unit implemented within a Serial Bus node.

3.1.3 Abbreviations

The following are abbreviations that are used in this specification:

CSR	Control and status register
CRC	Cyclical redundancy checksum
EUI-64	Extended Unique Identifier, 64-bits

3.2 Notation

The following conventions should be understood by the reader in order to comprehend this specification.

3.2.1 Numeric values

Decimal, hexadecimal and, occasionally, binary numbers are used within this specification. By editorial convention, decimal numbers are most frequently used to represent quantities or counts. Addresses are uniformly represented by hexadecimal numbers. Hexadecimal numbers are also used when the value represented has an underlying structure that is more apparent in a hexadecimal format than in a decimal format. Binary numbers are used infrequently and generally limited to the representation of bit patterns within a field.

Decimal numbers are represented by Arabic numerals without subscripts or by their English names. Hexadecimal numbers are represented by digits from the character set 0 – 9 and A – F followed by the subscript 16. Binary numbers are represented by digits from the character set 0 and 1 followed by the subscript 2. For the sake of legibility, binary and hexadecimal numbers are separated into groups of four digits separated by spaces.

As an example, 42, 2A₁₆ and 0010 1010₂ all represent the same numeric value.

3.2.2 Bit, byte and quadlet ordering

This specification uses and extends the facilities of Serial Bus, IEEE Std 1394-1995, and therefore uses the ordering conventions of Serial Bus in the representation of data structures. In order to promote interoperability with memory buses that may have different ordering conventions, this specification defines the order and significance of bits within bytes, bytes within quadlets and quadlets within octlets in terms of their relative position and not their physically addressed position.

Within a byte, the most significant bit, *msb*, is that which is transmitted first and the least significant bit, *lsb*, is that which is transmitted last on Serial Bus, as illustrated below. The significance of the interior bits uniformly decreases in progression from *msb* to *lsb*.

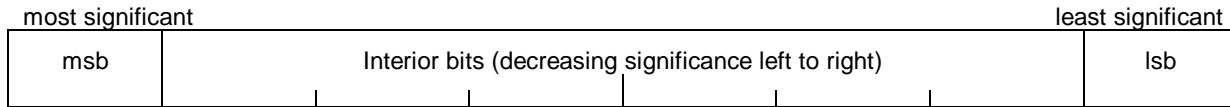


Figure 3-1 Bit ordering within a byte

Within a quadlet, the most significant byte is that which is transmitted first and the least significant byte is that which is transmitted last on Serial Bus, as shown below.

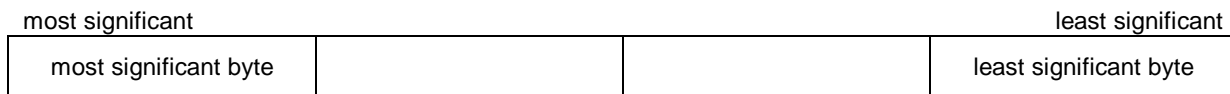


Figure 3-2 Byte ordering within a quadlet

Within an octlet, which is frequently used to contain 64-bit Serial Bus addresses, the most significant quadlet is that which is transmitted first and the least significant quadlet is that which is transmitted last on Serial Bus, as the figure below indicate.

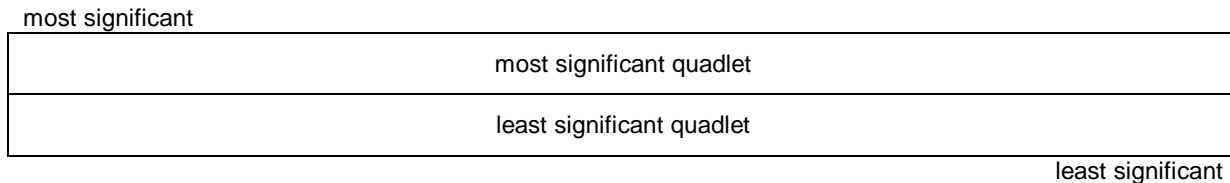


Figure 3-3 Quadlet ordering within an octlet

Increasing Serial Bus addresses for quadlets correspond to increasing addresses on other buses bridged to Serial Bus, but the correlation of addresses is problematical when block transfers take place that are not quadlet aligned or not an integral number of quadlets. In such cases, no assumptions may be made about the ordering (significance within a quadlet) of bytes at the unaligned beginning or fractional quadlet end of such a block transfer, unless an application has knowledge (outside of the scope of this specification) of the ordering conventions of the other bus.

3.2.3 Register specifications

This specification precisely defines the format and function of control and status registers, CSR's. Some of these registers are read-only, some are both readable and writable and some generate special side effects subsequent to a write.

In order to precisely define CSR's, their bit fields, their initial values and the effects of read, write or other transactions, the format illustrated in the figure below is used.

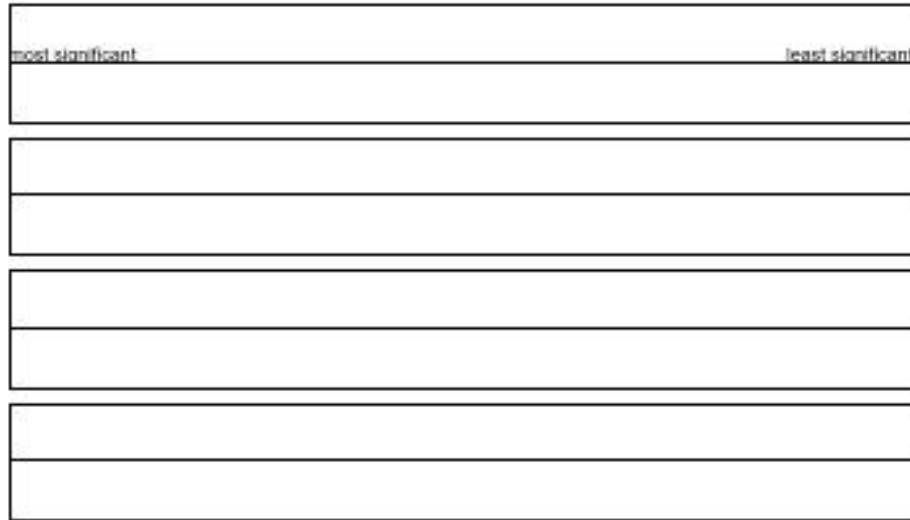


Figure x – CSR specification example

The register definition contains the names of register fields. The names are intended to be descriptive, but the fields are defined in the text; their function should not be inferred solely from their names. However, the following register definition field names have defined meanings.

Name	Abbreviation	Definition
bus-dependent	bus-depend	The meaning of the field shall be defined by the bus specification, in this case IEEE Std 1394-1995
reserved	r	The field is reserved for future definition (see definitions)
unit-dependent	unit-depend	The meaning of the field shall be defined by the company or organization responsible for the unit architecture
vendor-dependent	vendor-depend or v	The meaning of the field shall be defined by the node's vendor

CSR's shall assume initial values upon the restoration of power (a power reset) or upon a write to the node's RESET_START register (a command reset). If the power reset values differ from the command reset values, they shall be separately and explicitly defined. Initial values for register fields may be described as numeric constants or with one of the terms defined for the register definition. Values for register fields subsequent to a reset may be described in the same terms or as defined below.

Name	Abbreviation	Definition
unchanged	x	The field retains whatever value it had just prior to the power reset, bus reset or command reset.

In addition to numeric values for constant fields, the read values returned in response to a quadlet read transaction may be specified by the terms below.

Name	Abbreviation	Definition
last write	w	The value of the field shall be either the initial value or, if a write or lock transaction addressed to the register has successfully completed, the value most recently stored in the field. ¹
last update	u	The value of the field shall be that most recently updated by the node hardware or software. An updated field value may be the result of a write effect to the same register address, a different register address or some other change of condition within the node.

The effects of data written to the register shall be specified by the terms below.

Name	Abbreviation	Definition
effect	e	The value of the data written to the field may have an effect on the node's state, but the effect may not be immediately visible by a read of the same register. The effect may be visible in another register or may not be visible at all.
ignored	i	The value of the data written to the field shall be ignored; it shall have no effect on the node's state.
stored	s	The value of the data written to the field shall be immediately visible by a read of the same register; it may also have other effects on the node's state.

Reserved fields within a register shall be explicitly described with respect to initial values, read values and write effects. Initial values and read values shall be zero while write effects shall be ignored. CSR's that are not implemented, either because they are optional or they fall within a reserved address space, shall abide by these same conventions if a successful completion response is returned for a read, write or lock transaction.

4 Power state management model (informative)

The purpose of power state management is to conserve power and maximize device availability for end-users on the 1394 Serial Bus. In order to provide a model that may be implemented generically, the control mechanisms are abstracted into 'power states.'

While any entity on the serial bus could provide the described control, the intention is that an intelligent computing device such as a personal computer could best implement the control, and would offer the most benefit through closer integration of various PC applications with Serial Bus devices.

The power state management model is described in this section.

4.1 The Generic Device Power Management Model (informative)

In the device power management architecture, system software directs power management and integrates the activities of other components. Applications automatically take advantage of devices' power management capabilities using abstract application interfaces that work with any device, regardless of its bus interface.

4.1.1 Power Policy and Policy Owners

The decisions that determine how to save energy, when to turn-off a device, and when to enable a device to wakeup the system are referred to as the *power policy*. In the architecture, power policy is based on end-user preferences, application needs, and hardware capabilities.

¹ For clarity, read values for a field in a register that accepts lock transactions may be described as *last successful lock* rather than *last write*. However, the abbreviation in both cases remains *w*. Similar liberties may be taken with the use of *conditionally stored* in place of *stored* when the action occurs as the result of a lock transaction, but the corresponding one-letter abbreviation, *s*, is also unchanged.

The implementation of power policy is distributed across the Serial Bus, with different *policy owners* for different devices. Device power management on the Serial Bus is a problem of managing collections of devices, each with different power policy owners. A Serial Bus node is a collection of power-manageable devices: specifically, the PHY, the Link (including configuration information and wake event logic), and each of the Units within the node.

The policy-owner for a particular device is that node that has the best knowledge of the device's usage from the perspective of the end-user and the applications.

4.1.2 Power Control

If the policy owner is outside the node, the path between the policy owner and the Unit must not be in a higher power-saving state than the Unit. The path between the policy owner and a node Unit is made up of the node's Link, the node's PHY, and the PHY port the policy owner is connected to. Several examples can illustrate this fundamental rule of power control:

If any Unit in a node is in the D0 state (fully operational) then the node's Link power state must be L0, the node's PHY power state must be H0, and the power state of the PHY port the policy owner is connected to must be P0.

If any Unit on a node is in the D2 state: the node's Link must be in either the L0 or L2 state; the PHY must be in the H0 or H2 state; and the port the policy owner is connected to must be in the P0 or P2 state.

If all the Units in a node are in the D3 (fully OFF) state, then the node's Link power state may be L0, L2, or L3. The node's PHY power state may be H0, H2, or H3 and each of the node's ports may be in the P0, P2, or P3 state.

For more information, see section 4.2, "Unit Power States," and section 4.3, "Bus Power States."

If the policy owner resides inside the node, the relationships of the power states between the devices collected in the node is completely up to the resident policy owner.

4.2 Unit power states (informative)

This specification defines four power states for Serial Bus Units, as shown by the table below.

Unit power state	Operational condition	Comments
D0	Fully operational	All unit context is maintained and all unit functionality is available.
D1	Unit-dependent	The power consumption in this state is less than D0. The time required to make a transition to D0 is less than the time required to transition to D0 from D2. All unit operational context is preserved but all unit functionality may not be available.
D2	Unit-dependent	The power consumption in this state is less than in D1. The latency required to make the transition to either D0 or D1 is less than the time to make the transition to either D0 or D1 from D3. Unit operational context may not be preserved.
D3	Not operational	The unit uses negligible power and its power source may be turned-off.

Unit power states consume incremental power above and beyond the power consumption of the PHY and Link, and reflect incremental operational functionality that the Unit provides.

D0 is mandatory and shall be implemented by all units that conform to this specification. The other unit power states are optional. If any other Unit power state is implemented, the only required transitions are from that power state to D0 and from D0 to that power state. For example, if a Unit implements D0, D2, and D3, then the required transitions are D0 to D2, D2 to D0, D0 to D3, and D3 to D0.

The specific details of the Unit power states are defined in the relevant Unit Architecture Specification. Whether or not a unit power state is implemented, as well as the power consumed in each implemented state, may be determined by an examination of configuration ROM (for more information, see section 8.5.1, “Unit_Power_State entry”).

4.3 Link, PHY, and port power states (informative)

The power states supported by this specification are based on definitions contained in IEEE Std. 1394a, but are defined here in terms of their software-visible characteristics and capabilities. The Port, PHY, and Link states listed below are related to each other, and to Unit power states defined in section 4.2, in a strict relationship defined by this specification (unless the node itself is the power policy owner for the Unit).

Note: The default states, corresponding to legacy unit, Link, PHY, and port power states, are shaded in the tables below [TBD]

Table 1. Link Power States

1394 Link PM State	1394 Unit PM State	Actions to Unit	Actions from Unit	Description
L0	D0, D2, or D3	Any 1394 Transaction	Any 1394 Transaction Resume Signaling	Link ON (Active) LPS ON
L2	D2 or D3	None	Resume Signaling	LINK ON (Active) LPS OFF
L3	D3	None	None	Link OFF LPS OFF

The Power Manager may always use the LINK-ON bit in a node’s *self_ID* packet to determine the current Lx state the node.

If the Link is currently in L2 or L3, the Power Manager may write LINK-OFF into the STATE-SET bit of a PHY packet and transmit it to the node.

If the Link is currently in L0, the Power Manager may write LINK-ON into the STATE-SET bit of a PHY packet and transmit it to the node. If the wake enable bit is set in any of the Unit registers for the node, the Link will go into L2; otherwise, it will go into L3.

Table 2. PHY Power States

1394 PHY PM State	1394 Link PM State	Actions to Unit	Actions from Unit	Description
H0	L0, L2, or L3	Any 1394 Transaction	Any 1394 Transaction	Ports Active PHY core ON
H2	L2 or L3	Resume Signaling	Resume Signaling	Ports suspended PHY core OFF
H3	L3	None	None	Ports disabled PHY core OFF

Table 3. Port Power States

1394 Port PM State	1394 PHY PM State	Actions to Unit	Actions from Unit	Description
P0	H0	Any 1394	Any 1394 Transaction	Port active

		Transaction		
P2	H0 or H2	Resume Signaling Resume Packet	Resume Signaling	Port suspended
P3	H0, H2, or H3	None	None	Ports disabled

Note: The L1, H1, and P1 power states are reserved for a future specification.

4.4 Power Policy Owners and the Power States (informative)

The policy owner makes all state change decisions, and power state change commands may be sent by the policy owner. This section describes the policy owners for each of the power states: the Unit power states D0, D1, D2, and D3; the Link power states L0, L2, and L3; the PHY power states H0, H2, and H3; and the port power states P0, P2, and P3.

4.4.1 Unit Power Policy

Units must accept and act on power state commands from the current policy owner. Units may ignore power state change commands from any source other than the current policy owner. The protocol established between an application or driver and the Unit, for example, SBP-2, determines the current policy owner.

Note that different Units on a node may have different power policy owners.

4.4.2 Link Power Policy

For all manageable nodes, the power policy owner of the Link power states (L0, L2, and L3) is either the Power Manager or the node itself. The goal of Link power policy is to conserve as much power as possible while meeting the power-state needs of the units on each node, and ensuring that communications pathways on the bus remain available when needed.

For the rules that determine whether the Power Manager or the node itself is the power policy owner for a manageable node, see section 5.2.2, “Power Policy Ownership Arbitration.”

4.4.3 PHY Power Policy

The PHY power state transitions are automatic, based on the states of the ports and the Link attached to the PHY. For more information, see the P1394a specification.

The Power Manager does not use the PHY power states.

4.4.4 Port Power Policy

Non-manageable nodes use the port power states (P0, P2, and P3) to manage themselves and to manage the ports that connect to them. For more information, see the P1394a specification, the Device Bay specification, and so on.

The Power Manager does not use the port power states.

4.5 1394 Bus Power Manager (informative)

The Bus Power Manager performs two basic functions:

Cable power distribution and power domain management (allocating and deallocating cable power from power providers in order to serve power consumers).

Link power state management for manageable nodes (in order to conserve system-wide power usage).

There is exactly one Power Manager on any given Serial Bus, and it is always located on the Bus Manager node. Bus Manager arbitration has been enhanced in 1394a specification with the addition of a Power Manager-capable bit in the Bus Info block. This allows a Power Manager-capable node to have priority in Bus Manager elections.

The *Specification for Power Management* (this specification) also defines a means for the current Bus/Power Manager to abdicate that responsibility, either turning it over to another Power Manager-capable node, or returning the bus to the non-power managed condition (no Power Manager present). For more information, see section 5.1.

4.6 Batteries and battery groups (informative)

REVIEWERS: Is this section needed???

Power policy is affected by whether a device is powered by a consumable, limited resource, or by an ‘unlimited’ resource. This specification adds a way for devices [TBD a Unit?] to expose
 Currently operating on AC power.
 Currently operating from one or more removable batteries.
 Currently operating from one or more permanently installed batteries.

This specification allows for batteries to be collected into groups. A battery group is implemented as a Unit on the 1394 node, and other Units may refer to a battery group as its power source. Although batteries may be installed or removed independently, a battery group is considered a single power source (when it provides power to one or more Units) or a power consumer (when charging).

Units, when they are battery-powered, may draw their power from one or more battery groups. Battery groups may supply power to one or more units. Configuration ROM entries permit the description of all these permutations. [TBD Is this promise made good on? No, I don’t think so. We need to revisit this.] For more information about these Configuration ROM entries, see section TBD.

5 Power manager

This section is the Power Manager specification; it describes how the Power Manager is determined subsequent to a Serial Bus reset and describes the functions of the Power Manager.

The Power Manager is a Bus Manager-capable node, as specified by IEEE Std 1394-1995, with additional capabilities and responsibilities defined by this specification.

5.1 Determination of the power manager

Since Power Manager capabilities are an extension of the Bus Manager, all Power Manager-capable nodes shall participate in contention to become the Bus Manager after a Serial Bus reset is observed.

Any Power Manager-capable node that does not become the Bus Manager shall examine the Configuration ROM of the Bus Manager to determine whether or not it is also a Power Manager. The *pmc* bit in the bus information block provides this information.

If the Bus Manager is not a Power Manager, a Power Manager-capable node shall attempt to become the Bus Manager as follows:

The candidate Power Manager shall set the abdicate bit in the Bus Manager's STATE_SET register. If the Bus Manager conforms to this specification, this has the effect of forcing the Bus Manager to ignore its own incumbency and to behave as if it were a challenger for the role of Bus Manager.

The candidate Power Manager shall initiate a Serial Bus reset.

Immediately upon completion of the self-identify process, the candidate Power Manager shall attempt to become the Bus Manager in accordance with the procedures in IEEE Std 1394-1995, with one exception. The candidate Power Manager shall not wait 125 ms before making a lock transaction to the BUS_MANAGER_ID register at the isochronous Resource Manager node, but shall attempt to become the Bus Manager immediately upon the completion of the self-identify process.

If a Power Manager-capable node does not become the Bus Manager, it shall transmit a PHY configuration packet with the R bit set to one, the root_ID field set to a value of 3F₁₆, and the T bit cleared to zero. The effect of this PHY configuration packet is to clear the force_root bit of all nodes to zero while leaving the gap_count at its present value. [REQUEST FOR REVIEW: why is this step here?]

The candidate Power Manager shall then initiate a Serial Bus reset and attempt to become the Bus Manager as described in step 3 above.

A Power Manager-capable node that becomes the Bus Manager is also the Power Manager. After first fulfilling the obligations of a Bus Manager specified by IEEE Std 1394-1995, the Power Manager shall perform the functions described below.

5.2 Power manager responsibilities

The major Power Manager responsibilities are:

Cable power distribution and power domain management (allocating and deallocating cable power from power providers in order to serve power consumers).

Link power state management for manageable nodes (in order to conserve system-wide power usage).

5.2.1 Managing power domains

A *power domain* is a region of the Serial Bus whose cable power is provided by a single power provider. The Power Manager must manage power domains

When the Power Manager is just coming onto the bus.

When the Power Manager is already on the bus and devices come and go.

After a bus reset event, before the Power Manager may respond to any requests addressed to its POWER_CHANGE register, it must anticipate the amount of power required in each power domain when all manageable nodes in that domain are fully operational. For a definition of "manageable node," see section 5.2.1.1, "Definition of Manageable Nodes."

After a bus reset, the Power Manager must perform the following actions. Note that these actions are based on the rule that if a node consumes Serial Bus power then its Link must be off immediately after it is attached to the bus.

The Power Manager must parse the Serial Bus topology into disjoint power domains. To do this, the Power Manager may obtain all the information it needs from the "repeat power" bit in the POWER_CLASS field of each node's *self_ID* packet. The repeat power bits may show the limits of the power domains. The Power Manager may also read the power provider field in each node's configuration ROM, if it is available. Information read from configuration ROM always takes precedence over information other information used to parse the disjoint power domains on the Serial Bus.

The Power Manager must calculate the amount of power available in each power domain.

Within each power domain, the Power Manager must determine the total amount of power required to turn on the Link of each manageable node in the domain. The Power Manager does this by analyzing the self ID packets broadcast by the

manageable nodes. In its determination, the Power Manager must also reserve 3 watts for every node port in the domain that is not currently consuming power or is occupied by a self-powered device.

If the amount of power available within the power domain is not sufficient to enable all manageable nodes, the Power Manager shall enable none of the nodes *at this time*. Later, applications may send Link-on change requests to the Power Manager for one or more of the nodes in the domain (for more information about Power Management Link-on requests, see section 7).

Otherwise, if the power available within the domain is adequate, the Power Manager may transmit Link-On packets to all manageable nodes whose Link and higher layers have not yet been enabled. The Power Manager must ensure that for every node to which a Link-on packet is transmitted, that node's entry in the Node Usage Table is set to zero (for more information about the Node Usage Table, see section TBD). If, after a reasonable period of time, one or more of these Node Usage Table entries has not been incremented, the Power Manager may turn off the Link for that node.

Once this process is complete, the Power Manager may service power change requests addressed to its POWER_CHANGE register.

The Power Manager must maintain power consumption information across bus topology changes. Specifically, the values in the Node Usage Table entries must track with changes in the node IDs between bus resets. For example, if the Node 5 has a usage value of 2 before a bus reset and that node's ID is Node 7 after the bus reset, then the Power Manager must ensure the Node 7 entry has a value of 2.

It is suggested that the Power Manager coordinate transmitting the Link-on packets with other activities on the Serial Bus after a DFC event. The timing diagram below illustrates this.

Interval A is the period between the first bus reset and the second bus reset; during this time the Bus Manager is figuring out what the new gap count is. The Power Manager should not transmit Link-on packets during this time interval.

Interval B is the period during which devices are re-establishing connectivity and is an extremely busy time on the bus. The Power Manager should not transmit Link-on packets during this time interval.

Interval C is the time during which the Power Manager should transmit the Link-on packets, and as close to the beginning of this time period as possible.

It is suggested that the time interval between receiving a Link-on packet at a node, and when the Link becomes available be less than 1 second. Note that if a node's Link is not available at the time of the second bus reset, the node should generate a bus reset when its Link does become available.

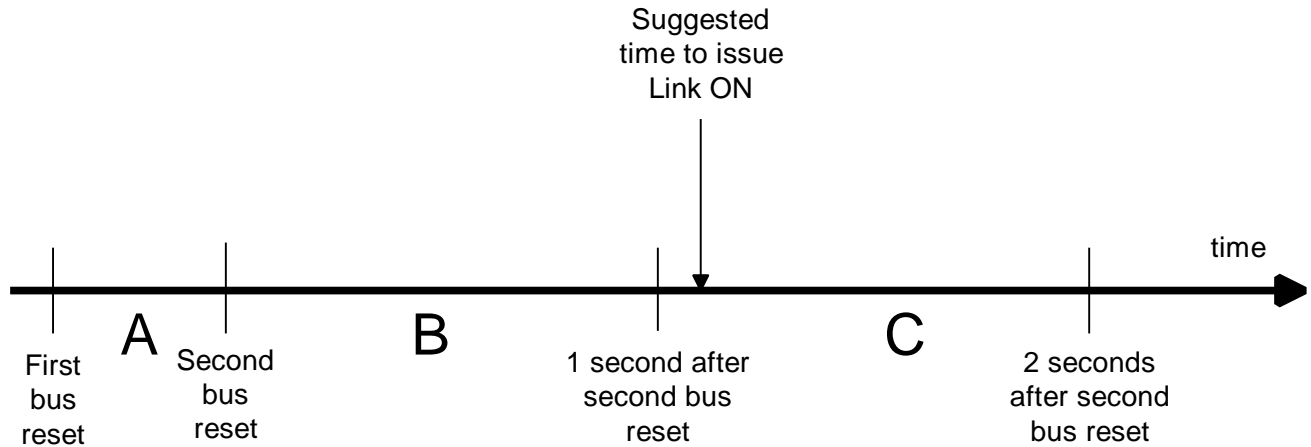


Figure 5-1 Link ON Timing Diagram

5.2.2 Manageable and Non-Manageable Nodes

The Power Manager/Bus Manager is not involved in power management of 1394 ports, of PHYs, or of Links in Device Bay subsystems or embedded nodes. This is true for both types of Device Bay subsystems (that is, subsystems controlled by a USB-based Device Bay Controller and subsystems controlled by an ACPI-based Device Bay Controller).

The following illustration shows an example network of connections between ports on 1394 PHYs (the PHYs are shown as dark boxes).

Node “A” is a PC chassis with a USB-based Device Bay Controller (DBC) and a 2-port PHY. One port is connected to an embedded node (for example, a native 1394 HDD) and the other port is connected, through a walkup connector, to one of the ports on node “B.”

Node “C” is a remote Device Bay chassis containing a PHY with four ports. Two ports are connected to bays, one port is connected, through a walk-up connector, to node E, and one port is connected, through a walkup connector, to node “B.”

Node “D” is a PC motherboard with two Device Bay bays, an ACPI-based DBC, and a 4-port PHY. Two of the ports are connected to bays, one port is connected to an embedded node (for example, a native 1394 HDD) which is then connected to another embedded node, and the fourth port is connected to a walk-up connector that is connected to a walk-up connector on node “B.”

The important point here is that only nodes “B” and “E” are manageable nodes from the point of view of the Power Manager software (which is assumed to be running on PC “A” in this example).

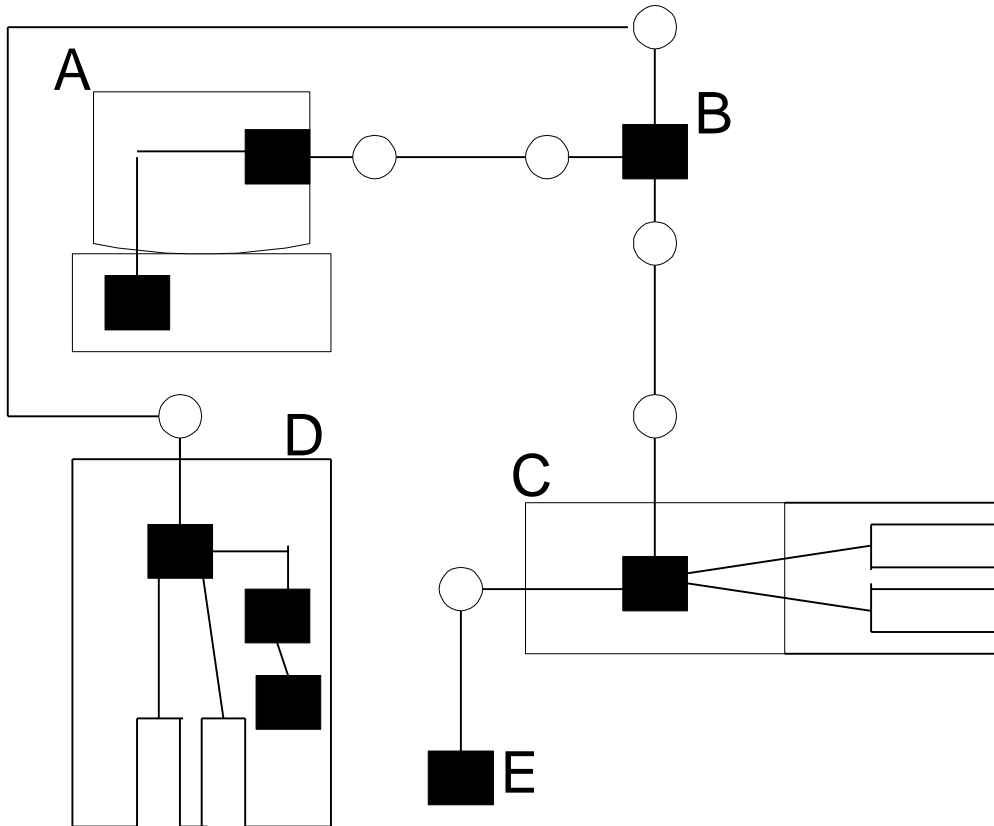


Figure 5-2 Example showing manageable nodes

5.2.3 Notifications and the Power Manager

The Power Manager must receive two types of notifications:

In the context of a static power provider in a power domain, a power policy owner wants to reserve an amount of power. The behavior of the power provider in a power domain is dynamic; the power provider comes and goes. A special case of this is a laptop PC switching back and forth between being a power provider and a power consumer.

For more information, see section 5.2.3, Example Scenarios.

5.2.4 Power policy ownership arbitration

In this section, it is assumed that the Power Manager node is a computer that has interest in opportunistically turning Link power on and off at other nodes on the Serial Bus, in order to reduce power consumption over the entire Serial Bus. This sets the context for the power policy ownership rules, as follows.

Power policy ownership for each manageable node's Link power states may be at the Power Manager node or at the node itself. In general,

If, when a *self-powered* node is first connected to the Serial Bus with both its PHY and Link enabled, then the power policy owner is the node itself.

If, when a node is first connected its PHY is enabled and its Link is off, then the power policy owner is the Power Manager.

Example applications of these rules are, a native self-powered SBP-2 disk drive connects to the bus with its Link off, so the power policy owner is the Power Manager.

For another example, all bus-powered nodes connect to the bus with the PHY on and Link off. Note that an exception to this rule is a bus-powered node with a total power consumption of less than 3 Watts with both its Link and PHY on. In this case, there may be little or no advantage to turning the Link on and off to conserve power, so the power policy owner for this type of bus-powered node is the node itself.

For nodes for which it is the power policy owner, the Power Manager may recognize opportunities for changing Lx states on that node.

When Use Table entry is incremented from 0 to 1 (becomes non-zero), PM shall ensure that node's Link is enabled. When the target node's link becomes available, the device may be used by the requesting node.

To do this, the Power Manager must maintain a Use Table with 63 quadlet entries. The index into this table is the unique PHY ID that every node on the Serial Bus has after the bus topology operation is complete following a bus reset. All nodes on the bus have access to the Use Table, so it may be used by policy owners that are not the Power Manager.

The Power Manager must maintain power consumption information across bus topology changes. Specifically, the values in the Node Usage Table entries must track with changes in the node IDs between bus resets. For example, if the Node 5 has a usage value of 2 before a bus reset and that node's ID is Node 7 after the bus reset, then the Power Manager must ensure the Node 7 entry has a value of 2.

Nodes must use the Compare/Swap subrequest of the Lock request to modify the table so conflicts between more than one node attempting to access the same table entry at the same time are automatically resolved. The Power Manager is automatically notified each time a table entry value is changed.

Rules for using the table are:

Every time a node on the Serial Bus uses the target node, it must increment the target node's entry in the table. The using node may increment the entry once or once for every unit in the target node. "Using the target node" includes Turning on the target node's Link.

A PC node logging on to multi-initiator unit.

When the using device is done using the target node it must decrement the target node's entry in the table. If the using node incremented the table entry once for each unit in the using node when it began using the target node, it must decrement the entry that same number of times.

Before decrementing the target node's table entry the using node must request that the target node's units go into the D3 (power off) state. The policy owner must always inform the PM whether the device actually changed states. For an example of the target being a multi-initiator node, see section 5.3, "Example Scenarios."

Whenever a node's table entry value is zero, its Link may be turned off.

5.3 Example Scenarios

Example scenarios in this section are:

Policy owner is device driver on PC node that is logged on to a "single-initiator" hard disk node.

A "multi-initiator" hard disk node has itself as the policy owner and is logged on to by multiple PC nodes.

A "multi-initiator" hard disk node has itself as the policy owner and is logged on to by a PC for exclusive use.

A laptop PC switches from being a power provider to being and a power consumer.

5.3.1 Single-initiator hard disk node scenario

TBD

5.3.2 Multiple PC nodes logged on to a multi-initiator hard disk node

TBD

5.3.3 PC node logged on to a multi-initiator hard disk node with exclusive use

TBD

5.3.4 Laptop PC switches for power provider to power consumer

Rules for a laptop participating in this scenario are:

In its Configuration ROM, the laptop must be either a power consumer or a power provider, not both.

When it switches between being a power consumer and a power provider, the laptop must change its power consumer or power provider setting in its Configuration ROM, its “generation bit”, the POWER_CLASS field in its *self-ID* packet, and cause a bus reset.

6 Power management procedures

This section describes how to use the power management facilities defined in the preceding sections.

6.1 Device Attachment

The steps required to support power management when a node is attached to the Serial Bus are listed below in two sequences. First, the steps that must always take place to support basic power management are listed. Then the optional steps for supporting remote wake-up are listed.

6.1.1 Basic Power Management Support at Device Attach

When a node is attached to the Serial Bus, it must draw less than three watts to power its PHY. During the *self_ID* process, the Power Manager determines, from the POWER_CLASS field in the *self_ID* packet, how much power it takes to turn on the device’s Link. The Link must be turned on before the device’s ROM may be read and the device enumerated.

If there is enough power to turn-on the Link, the Power Manager turns it on (puts the Link into L0).

The Power Manager reads the Node_Power_Level entry in the Node power directory in configuration ROM. This provides a more granular value for node power consumption than the POWER_CLASS field in the *self_ID* packet. The Power Manager then updates its power allocation for this node accordingly. [REVIEWERS: Is this step necessary?]

If a power policy owner for a Unit is located in a node anywhere on the Serial Bus, the owner may read the Unit’s ROM to determine the supported power states (by reading the Unit_Power_State entries in the Unit power directory). The driver/application may also determine at this time if the unit is capable of causing a wakeup, and from which Unit power state (Dx state) this capability is available.

To power the unit up to D0, the policy owner for the Unit notifies the Power Manager (using the POWER_CHANGE register in the Power Manager node) to request a power state that consumes more power (that is, a transition to D0). If there is enough power, the Power Manager OK’s this request (RESP_COMPLETE) and allocates the full amount of power required for D0, as determined by reading the Unit_Power_State entry for power state D0 in the Unit power directory. If there is not enough power available, the Power Manager fails the request (RESP_CONFLICT_ERROR) and does not allocate the power.

Assuming the request is OK’d, the policy owner issues the command to the Unit to change the power state.

When the policy owner decides to put the Unit into a low-power state (that is, a transition **from** D0), the policy owner issues the command to the Unit to change the power state.

The policy owner notifies the Power Manager that the Unit is in a low-power state, using the POWER_CHANGE register in the Power Manager node.

The Power Manager may de-allocate the power saved by reducing the unit's power state. Note that whether or not the Power Manager does this is implementation-specific, as deallocation may not be desirable due to difficult user scenarios.

When all the Units on a node are put into lower power states and are no longer being used, the Power Manager may attempt to save even more power by reducing the node's Link power state. As above, the Power Manager may de-allocate the saved power.

6.1.2 Remote Wakeup Support

A Unit's power policy owner may also choose to use the Unit's wakeup feature. The policy owner will typically do this when it is waiting for some specific event to occur on the Unit, and either the policy owner's node or the Unit (or both) is put into a low-power state to save power during the wait.

A policy owner may determine whether a unit supports this feature (by reading the *wake-up* field in the Unit_Power_State entry for the low power state from which wake-up is to occur). If the Unit does support this feature, the policy owner may enable the feature by setting the *wake-up* bit in the Unit_Power_State register on the Unit.

The policy owner then sets the Unit into the appropriate unit power state (Dx state), as indicated by the ROM. For more information about setting the Unit power state, see section 6.2.

The Power Manager may attempt to save more energy at the node, based on the lower unit power state. If possible, the Power Manager may lower the node's Link power state (the node's Link power state must not be lower than the power state of any Unit in the node). The Power Manager must also ensure that this Link power state may support wakeup signaling [TBD how does the Power Manager do that?]. For more information about setting the Link power state, see section 6.2. [TBD Is this consistent with the Use Table???

When the Unit detects a wakeup event, the Unit causes resume signaling to be signaled on all of its node's ports. Resume signaling causes every PHY on the bus to return to the H0 state (Links may or may not be on, depending on the policy for that node).

After the bus reset that signifies the end of the resume process, the policy owner checks to see if its requested wakeup event was the cause of this resume, by checking the UNIT_POWER_STATE register. If so, it turns on the Unit (sets it to the D0 state) so it may handle the event using normal 1394 bus transactions. The policy owner uses the procedures described before to accomplish this.

6.2 Power state change

An application that wishes to effect a change in a Unit's power state communicates that request to the Power Manager by means of a quadlet write request to the POWER_CHANGE register.

As specified in section 7.1.1, the value of the quadlet write contains the following information:

The identity of the Serial Bus node that contains the unit (the physical ID of the node that contains the unit).

The address of the Unit's power management registers.

The desired power state for the Unit (D0, D1, D2, or D3).

Upon receipt of a quadlet write request at the POWER_CHANGE register, the Power Manager performs the following actions:

The Power Manager shall determine the difference in power consumption between the Unit's current power state and the desired power state.

The Power Manager shall determine, according to *phy_ID*, the power domain to which the Unit's node belongs.

If the difference determined in step 1 indicates a decrease in the unit's power consumption, the Power Manager may increase the record of the power available in the specified domain by the amount of power no longer consumed by the Unit. The Power Manager then returns a RESP_COMPLETE response for the write transaction to the POWER_CHANGE register.

Otherwise, if the difference indicates an increase in the Unit's power consumption, the Power Manager shall examine the power available in the specified domain. If there is sufficient power available in the domain, the Power Manager shall decrease the record of the power available in the specified domain by the amount of additional power consumed by the unit and return a RESP_COMPLETE response for the write transaction to the POWER_CHANGE register. If insufficient power is available within the domain, the Power Manager shall return a RESP_CONFLICT_ERROR response for the write transaction to the POWER_CHANGE register.

In all cases, the original requester is responsible for effecting the power change if it receives a RESP_COMPLETE response code, or for aborting the power change if it receives a RESP_CONFLICT_ERROR response code.

Note that there is considerable implementation flexibility for the Power Manager. A sophisticated Power Manager may maintain a record of each Unit's power consumption requirements at each supported power state and key this to the *csr_offset* of the Unit's power management registers. Another Power Manager might read the Unit's configuration ROM in order to determine the difference between the Unit's current power state and the requested power state.

6.3 Wake-up

Applications that control devices on Serial Bus are assumed to execute as computers, set-top boxes, or analogous equipment that may itself be subject to power control. If an external event, such as a ring indication in the case of a FAX or modem device or a user activation of an "instant on" power switch, requires an application response it may be necessary to first "awaken" the application.

Wake-up is defined in IEEE Std. P1394a.

7 Power management registers

This section defines several different types of registers, all of which are necessary to support power management on Serial Bus:

Registers required at the power manager.

Registers required for any unit that supports power management.

The clauses that follow describe the registers in detail.

7.1 Power manager registers

The Power Manager is a Bus Manager-capable node that implements extensions to the standard registers defined by IEEE Std 1394-1995. Specifically, the Power Manager shall implement the *abdicate* bit as defined in IEEE Std. P1394a. In addition, Power Managers implement a new register, the POWER_CHANGE register. These registers shall lie within initial units space and shall be located at or above address FFFF F001 0000₁₆ within the node's 48-bit address range.

The relative relationship of the POWER_CHANGE register is fixed within a contiguous block of quadlets, as defined by the table below. The base address of the Power Manager node's set of power management registers is obtained from the Power_Management_Registers entry in the root_directory (for more information, see section 8.1.3).

Relative Offset	Name	Required	Description

Architecture for Power Management on High Speed Serial Bus

00 ₁₆	POWER_CHANGE	Power Manager only (Optional)	Enables applications to request a power state (Dx) change for a Unit on a Serial Bus node or a Link power state (Lx) change on a node.
------------------	--------------	-------------------------------	----------------------------------------------------------------------------------------------------------------------------------------

7.1.1 POWER_CHANGE register

The POWER_CHANGE register provides a means by which policy owners notify the Power Manager of changes in the power state (Dx state) of a unit at a Serial Bus node. The POWER_CHANGE register is pointed to directly by the Power_Management_Register entry in the Root_Directory.

Policy owners must notify the Power Manager, by using this register, of
 All unit power state changes that will result in increased power reservation state.
 All times when it releases its reservation.

Notification must be made *before* a change to a higher state (which might require additional cable power) and *after* changes to a lower state (which may free-up some cable power).

If the Power Manager denies the request for the power change, then a RES_CONFLICT_ERROR shall be returned for the write request to this register. A RESP_COMPLETE, however, indicates that the Power Manager has granted permission for the power change and has made the necessary changes to cable power allocation. Note that in all cases, the requestor shall do the actual change of the unit's power state.

The format of the write-only Power Change register is shown below.

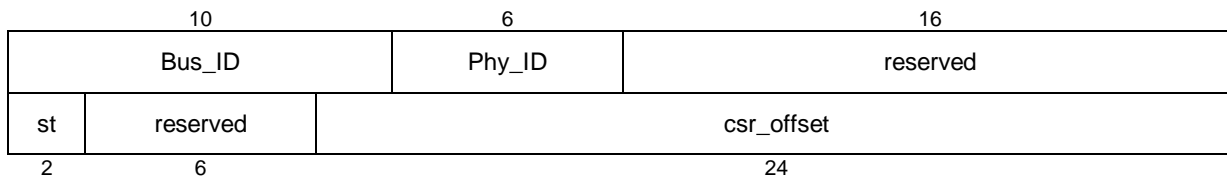


Figure 7-1 POWER_CHANGE register format

The *state* field (abbreviated as *st* in the figure above) shall specify the desired power state for the Unit (Dx state) or Link (Lx state).

The *phy_ID* field shall specify the physical ID of the node at which the unit resides. The physical ID of the node is necessary for the Power Manager to determine the power domain in which the node resides.

The *csr_offset* field shall specify the unit whose power state is to be changed. The value of *csr_offset* shall be equal to the field of the same name obtained from the Unit_Power_Management entry in the Unit's unit power directory. A *csr_offset* field value of zero means that this request is for the Link (Lx) power state of the node, not the Unit (Dx) power state.

7.1.2 Use table registers

The Use Table is an array of 63 read/write quadlet registers that directly follow the POWER_CHANGE register. If a Use Table entry contains a value of zero, then the corresponding node is not being used. If a Use Table entry is greater than zero, then the corresponding node is being used.

7.2 Unit register

Each Unit that supports power management shall have the set of control and status registers shown in the following table. These registers shall lie within initial units space and shall be located at or above address FFFF F001 0000₁₆ within the node’s 48-bit address range.

The relative relationship of these registers is fixed within a contiguous block of quadlets, as defined by the table below.

Relative offset	Name	Required	Description
00 ₁₆	Unit_Pwr_State	Mandatory	Reports the unit’s power state
04 ₁₆	Unit_Pwr_Control	Mandatory	Permits the unit’s power state to be managed
08 ₁₆	BATTERY_STATE	Optional	Reports the battery’s power status

The base address of each unit’s set of power management registers is obtained from the Unit_Power_Management entry in the unit’s unit power directory.

7.2.1 Unit_Pwr_State register

The Unit_Pwr_State register is a read-only register that provides information about the unit’s current power status. The format of the Unit_Pwr_State register is shown below.

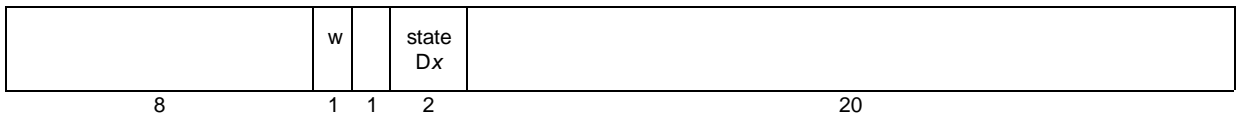


Figure 7-2 Unit_Pwr_State register format

The *wake_source* bit (abbreviated as *w* in the figure above) indicates whether or not the Unit is the source of a system wake event. A value of one (1) shall indicate that this Unit was the source of a wake-up event, while a value of zero (0) shall indicate that this Unit has not been the source of a wake-up event since this bit was last cleared.

The *state* field shall specify the Unit’s current power state (D0, D1, D2, or D3).

7.2.2 Unit_Pwr_Control register

The Unit_Pwr_Control register is a write-only register that permits the Power Manager or another application to alter the Unit's power state.

If the unit architecture for this Unit does not provide a means of controlling the unit's power state then this register shall be implemented.

If the unit architecture for this Unit does provide a means of controlling the unit's state then that method is preferred and this register should not be implemented.

The format of the Unit_Pwr_Control register is shown below.

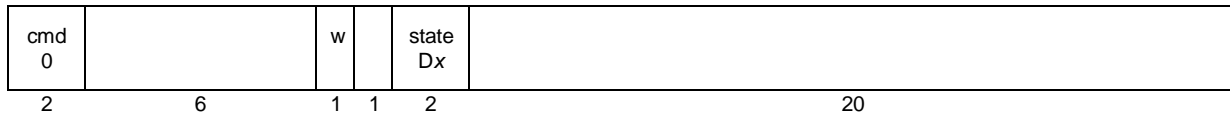


Figure 7-3 Unit_Pwr_Control register format

The *cmd* field shall specify a power management operation to be executed by the Unit, as defined by the following table.

Value	Name	Description
0	NO_OP	No operation
1	STORE_STATE	Stores the Unit power state field (0, 1, 2, or 3).
2	STORE_WAKE	Stores Wake enable/disable field.
3	STORE_BOTH	Stores both the Unit power state field (0, 1, 2, or 3) and the Wake enable/disable field.

The *wake_enable* bit (abbreviated as *w* in the figure above) indicates whether or not the Unit is enabled for a system wake event. A value of one (1) shall indicate that this Unit is enabled for a wake-up event, while a value of zero (0) shall indicate that this Unit is disabled for a system wake event.

The *state* field shall specify power state (D0, D1, D2, or D3) to which to set the Unit.

7.2.3 BATTERY_STATE register

REVIEWERS: Is this section needed???

The BATTERY_STATE register is a read-only register that provides information about the current condition of an individual battery. Batteries or groups of batteries may provide power to one or more units or may provide power to the node. The relationship between batteries, battery groups, units, and the Serial Bus node is described in more detail in clause TBD. The definition of the BATTERY_STATE register is given by the figure below.

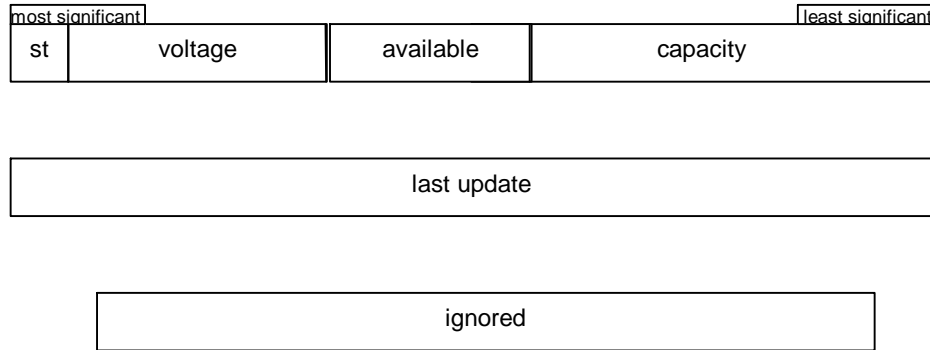


Figure x – BATTERY_STATE format

The *st* field shall specify the current state of the unit’s battery, as encoded by the values in the table below.

Value	Power source
0	No battery is installed or no information is available about the state of the battery.
1	Reserved for future specification.
2	A battery is installed and is providing power to the unit.
3	A battery is installed and is being charged.

When *st* has a value of two or three, the remainder of the fields in the BATTERY_STATE register shall contain valid information.

The *voltage* field indicates the peak voltage, in decivolts, of the battery.

The *available* field shall specify the remaining capacity of the unit’s battery as a percentage of *capacity*.

The *capacity* field shall specify the maximum capacity, in watt-hours, of the battery when fully charged.

8 Configuration ROM

All nodes that implement the power management facilities defined by this specification shall implement general format configuration ROM in accordance with ISO/IEC 13213:1994 and IEEE Std 1394-1995. General format configuration ROM is a self-descriptive structure as illustrated below. The bus information block and root directory are at fixed locations; all other directories and leaves are addressed by entries in their parent directory.

The figure below shows how the general ROM format may accommodate a diversity of directory and leaf entries in a tree structure. This specification establishes two new classes of directories, the node power directory and the unit power directory, shown shaded in gray. A node that supports power management or implements one or more units that support power management shall implement the configuration ROM entries described in the clauses that follow.

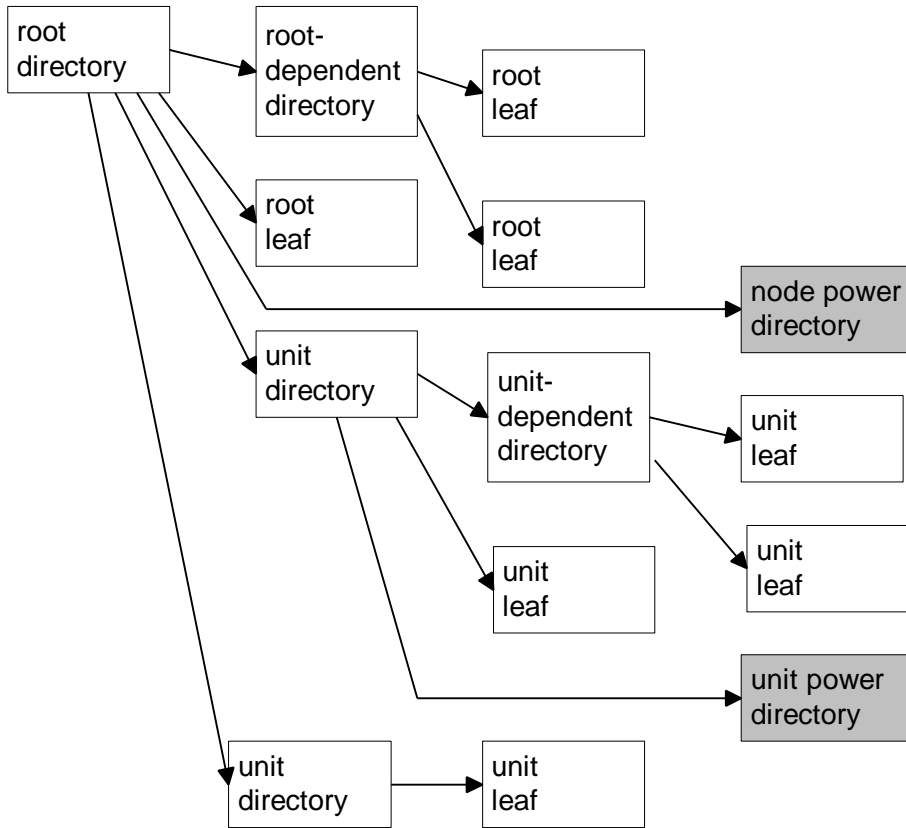


Figure 8-1 Configuration ROM hierarchy

8.2 Root directory entries

The following descriptions use several common fields. The field *key_type* (abbreviated *kt*) is as defined in ISO/IEC 13213: 1994.

<i>kt</i>	Meaning
0	immediate value
1	CSR offset
2	indirect offset of leaf
3	indirect offset of directory

The field *key_value* is as defined in ISO/IEC 13213: 1994; *key_value* taken together with *key_type* may be referred to simply as *key* and specifies the meaning of the directory entry.

8.2.1 Node_Power_Directory entry in the Root_Directory

Nodes that comply with the *Specification for Power Management* must use a Node_Power_Directory entry in the Root_Directory.

A Node_Power_Directory entry is placed in the Root_Directory to describe the location within configuration ROM of a node power directory for a node (for a description of the node power directory, see section 8.3). There shall be no more than one Node_Power_Directory entry in the Root_Directory.

The format of this entry is illustrated below.

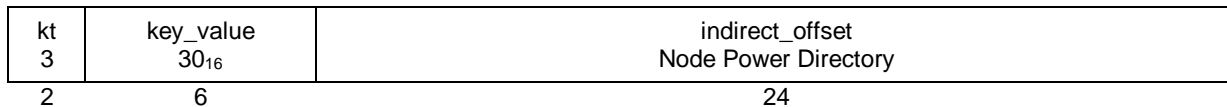


Figure 8-3 Node_Power_Directory entry format

F0₁₆ is the concatenation of *key_type* and *key_value* for the Node_Power_Directory entry.

The *indirect_offset* field specifies the number of quadlets from the address of the Node_Power_Directory entry in the Root_Directory to the start of the node power directory within configuration ROM.

8.2.2 Unit_Power_Directory entry in the Root_Directory

The Unit_Power_Directory entry is optional, but must be used in the unit directories of power manageable units. Units that comply with the *Specification for Power Management* must use a Unit_Power_Directory entry in the Root_Directory. The power consumption for non-power manageable units is included in the power consumption for the link power state(s).

A Unit_Power_Directory entry is placed in the Root_Directory to describe the location within configuration ROM of a unit power directory for a unit (for a description of the unit power directory entries, see section TBD). There must be no more than one Unit_Power_Directory entry in a unit directory.

The format of this entry is illustrated below.

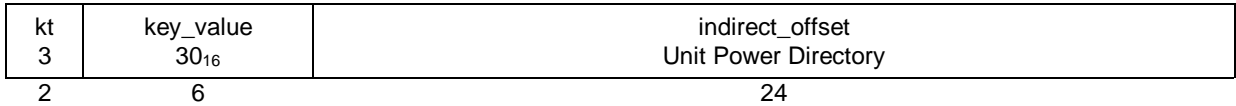


Figure 8-4 Unit_Power_Directory entry format

F0₁₆ is the concatenation of *key_type* and *key_value* for the Unit_Power_Directory entry.

The *indirect_offset* field specifies the number of quadlets from the address of the Unit_Power_Directory entry in the Root_Directory to the start of the node power directory within configuration ROM.

8.2.3 Power_Management_Registers entry in the Root_Directory

Power Manager-capable nodes must use a Power_Management_Registers entry in the Root_Directory. A Power_Management_Registers entry must be used only in Power Manager-capable nodes and is useful only in the current Power Manager node.

The Power_Management_Registers entry specifies the base address of the node’s power management registers. The Power Management Registers are described in section [points to Power Change Register, Node Use Table, etc.].

The format of a Power_Management_Registers entry is illustrated below.

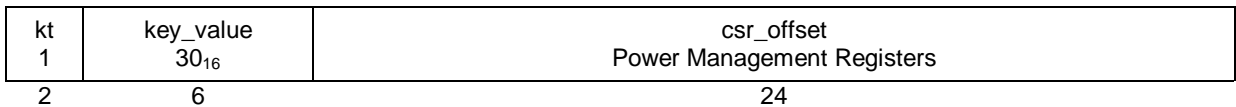


Figure 8-5 Power_Management_Registers entry

70₁₆ is the concatenation of *key_type* and *key_value* for the Power_Management_Registers entry.

The *csr_offset* field contains the quadlet offset, from the base address of initial register space, FFFF F000 0000₁₆, to the base address of the power management registers for the Power Manager-capable node.

8.3 Node power directory entries

The following descriptions use several common fields. The field *key_type* (abbreviated *kt*) is as defined in ISO/IEC 13213: 1994.

<i>kt</i>	Meaning
0	immediate value
1	CSR offset
2	indirect offset of leaf
3	indirect offset of directory

The field *key_value* is as defined in ISO/IEC 13213: 1994; *key_value* taken together with *key_type* may be referred to simply as *key* and specifies the meaning of the directory entry.

The field *pw_type* defines the type of node that the entry describes:

<i>pw_type</i>	Meaning
0	Manageable node
1	Device Bay node
2	Embedded node
3 - 7	Reserved

The 1394 Power Manager is only interested in *pw_type* zero, the power managers for the other node types may utilize the other entries.

The *continuous* field (abbreviated *c*) specifies whether the power being described is either the peak ($c = 0$) or continuous ($c = 1$) power. There should be an entry for both peak and continuous for each *state* being described.

The *state* field specifies the power state being described, either L0, L2, or L3 for node (Link) power state, or D0, D1, D2, or D3 for Unit power state. There should be an entry for each implemented power state.

The *voltage* field specifies a voltage value in tenths of a volt (decivolts). The precise meaning of this field depends on the *key_value*.

The *power* field specifies a power value in tenths of a watt (deciwatts). The precise meaning of this field depends on the *Key Value*.

8.3.1 Node_Power_Management entry

The Node_Power_Management_Entry must be used for all nodes that provide power to the 1394 cable. The entry provides the address of the register(s) that control the power provider.

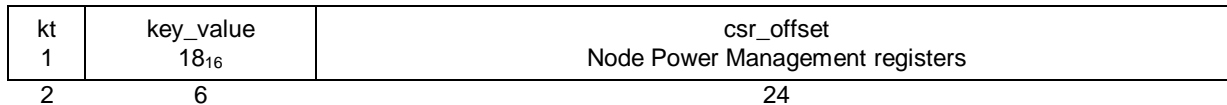


Figure 8-6 Node_Power_Management entry

58₁₆ is the concatenation of *key_type* and *key_value* for the Node_Power_Management entry.

The *csr_offset* field contains the quadlet offset, from the base address of initial register space, FFFF F000 0000₁₆, to the base address of the power management registers for the node.

8.3.2 Link_Power_State entry

A Link_Power_State entry describes the power consumption for each implemented Lx state in the node. A self-powered node should report the *Voltage* and *Power* fields as zero.

Note: If the device is designed to operate in multiple environments then there may be entries for more than one *pw_type*. Which entries are used will be decided by the determination of the *pw_type* of the node, see section TBD.

kt	key_value		r	c	state	voltage	power
0	0	pw_type	0	1	Lx	decivolts	deciwatts
2	3	3	1	1	2	9	11

Figure 8-7 Link_Power_State entry

The value of the *continuous* field (abbreviated as *c* in the above figure) must be set to one since only continuous power consumption is reported for the Link.

The *voltage* field for nodes that are not self-powered specifies the minimum voltage at which the link operates in this *state* (measured at the connector). For a *pw_type* value of zero this is defined as the minimum voltage at which the node variable *cable_power_active* is true, where *cable_power_active* is defined by 1394 (for 1394-1995 this is 7.5 volts, thus *voltage* should be 75).

Note: A *pw_type* zero node must tolerate the maximum output voltage for a cable power source (for 1394a this is 33 volts).

The *power* field for nodes that are not self-powered specifies the maximum power consumption of the node when in the specified *state*. This is defined as the total power consumption measured at the node’s connector minus the consumption of each of the node’s power manageable units as specified by the Unit_Power_State entry for each unit’s current state.

Note: The Unit power state is restricted to be numerically greater than or equal to the Link power state. Thus for a node with a single Unit the value reported here for link state L2 would be the maximum of:

- Total power consumption when Unit state is D2 minus the power consumption reported for Unit state D2
- Total power consumption when Unit state is D3 minus the power consumption reported for Unit state D3

8.3.3 PHY_Port_PwType entry

The PHY_Port_PwType entry identifies PHY ports on this node that specify the *PwType* of nodes connected to them. These entries should only be present in self-powered nodes that either control the power supply to the specified ports or pass 1394 cable power to the port (*PwType* zero). The *PwType* of a node is determined as described in section TBD. Nodes that have these entries must explicitly specify a *PwType* for each of their PHY ports. Nodes that do not have these entries are assumed to only have ports of the same *PwType* as the node.

kt	key_value		reserved	port_map															
0	2	pw_type	0	0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
2	3	3	8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 8-8 PHY_Port_PwType entry

The *port_map* field is a bit map with one bit corresponding to the PHY ports of this node that may only be connected to nodes of the specified *PwType*.

8.3.4 Power_Source_State entry

The Power_Source_State entry for a non-zero *pw_type* is reserved for use by the specification for the *pw_type*.

The `Power_Source_State` entry for a `pw_type` value of zero occurs only in nodes that provide 1394 cable power. These entries specify to the Power Manager the amount of power provided in each implemented power provider (PPx) state implemented by the power provider node.

kt	key_value		r	c	state	voltage decivolts	power deciwatts
0	1	pw_type	0	1	PPx		
2	3	3	1	1	2	9	11

Figure 8-9 `Power_Source_State` entry

voltage is the minimum guaranteed voltage in decivolts at which power is supplied.

power is the minimum guaranteed power that may be used in deciwatts. Any power that may be consumed by this node's link and PHY should not be included here; however, this value may assume that any of the node's units that draw from this source are in their lowest power consumption state (usually D3).

Note: If the *power* and *voltage* specified indicate a current of more than the maximum output current per port specified by 1394 then the node shall have a current limiter on each PHY port of *pw_type* zero. This ensures that the maximum output current per port is not exceeded.

The *continuous* field indicates if the values for *voltage* and *power* represent peak (zero) or continuous (one) power. Both peak and continuous power must be specified for each implemented PPx state.

8.3.5 Node_Power_Characteristics entry

The `Node_Power_Characteristics` entry specifies characteristics of a node. The `Node_Power_Characteristics` entry is required only if any of the *node_power_characteristics* are non-zero.

kt	key_value	node_power_characteristics	
0	19 ₁₆	L	reserved
2	6	1	23

Figure 8-10 `Node_Power_Characteristics_Entry`

The *link_policy_owner* field (abbreviated *L* in the above figure) specifies the power policy owner for the link power. A field value of zero means that the Power Manager is the policy owner and a value of one means that this node is the policy owner.

8.4 Unit power directory entries

Units that comply with the *Specification for Power Management* must implement a unit power directory. This directory provides a means to describe the power characteristics of the unit and the location of the registers used to control its power status.

8.4.1 Unit_Power_Management entry

The `Unit_Power_Management_Entry` points to the Unit Power Management registers described in section 7.2. The format of this entry is shown in the figure below.

kt	key_value	csr_offset
1	18 ₁₆	Unit Power Management registers
2	6	24

Figure 8-11 `Unit_Power_Management_Entry`

58₁₆ is the concatenation of *key_type* and *key_value* for the Unit_Power_Management entry.

The *csr_offset* field shall contain the quadlet offset, from the base address of initial register space, FFFF F000 0000₁₆, to the base address of the power management registers for the target. All unit registers shall be located at or above address FFFF F001 0000₁₆; therefore the value of *csr_offset* shall not be less than 4000₁₆.

8.4.2 Unit_Power_State entries (new version)

The Unit_Power_State entry specifies the power requirement of the unit in the specified Dx state. There are entries for all implemented Dx states for both peak and continuous power requirements.

Self-powered nodes should report both *voltage* and *power* field values as zero.

Units on nodes that are cable power sources may either

Report zero for *voltage* and *power* if the unit's Dx state does not affect the power available to the cable, or

Report the *voltage* supplied by the source and the amount of *power* by which the available cable power is reduced.

Note: If the device is designed to operate in multiple environments then there may be entries for more than one *pw_type*. Which entries are used will be decided by the determination of the *pw_type* of the node, see section TBD.

The format of a Unit_Power_State entry is shown below.

kt	key_value		w	c	state	voltage	power
0	0	pw_type			Dx	decivolts	deciwatts
2	3	3	1	1	2	9	11

Figure 8-12 Unit_Power_State entry

For *pw_type* zero, *voltage* specifies the minimum voltage that must be supplied at the node's connector for this unit to operate in this state. For other *pw_type* values the meaning of *voltage* is described in the specification appropriate to the *pw_type*.

The field *wakeup* (abbreviated *w*) specifies whether or not the unit may be a wakeup source when the *state* being described is selected. *w* is set to one if the unit may be a wakeup source while in this *state*.

8.4.3 Battery_Group entry

REVIEWERS: Is this section needed???

The Battery_Group entry is an entry in the unit power directory that optionally specifies the unit's power source(s) when powered from batteries. There may be multiple Battery_Group entries within a unit power directory. The format of the Battery_Group entry is specified in clause TBD.

One battery group may power more than one unit within a node. In this case, the Battery_Group entries within the unit power directories would all reference the same battery group.

