1394 TRADE ASSOCIATION
Power Specification
Part 2: Suspend/Resume Implementation Guidelines

Revision 1.0

October 5, 1999

Sponsor

Energy Conservation Working Group
of the
1394 Trade Association

Approved by the

1394 Trade Association

Abstract: This specification provides informative text for the development of process and procedures specific to IEEE 1394a-2000 suspend/resume mechanisms with the intent to promote adequate, uniform and cost-effective power conservation management for suspend/resume compliant 1394 devices.

Keywords: 1394, Power Consumer, Power Producer, Suspend, Resume, Power Conservation, Low Power

1394 Trade Association
2350 Mission College Blvd., Santa Clara, Calif. 95054 USA

1394 Trade Association Specifications are developed within Working Groups of the 1394 Trade Association. Participants on the committees serve voluntarily and without compensation. Most participants represent member organizations of the 1394 Trade Association, but the Working Groups are free to invite technical participation from anyone. The specifications developed within the Working Groups represent a consensus of the expertise represented by the participants.

Use of a 1394 Trade Association Specification is wholly voluntary. The existence of a 1394 Trade Association Specification does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the 1394 Trade Association Specification. Furthermore, the viewpoint expressed at the time a specification is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the specification. Users are cautioned to check to determine that they have the latest edition of any 1394 Trade Association Specification.

Comments for revision of 1394 Trade Association Specifications are welcome from any interested party, regardless of membership affiliation with the 1394 Trade Association. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise about the meaning of specifications in relationship to specific applications. When the need for interpretations is brought to the attention of the 1394 Trade Association, the association will initiate action to prepare appropriate responses.

Comments on specifications and requests for interpretations should be addressed to:

Editor, 1394 Trade Association
Regency Plaza, Suite 350
2350 Mission College Blvd.
Santa Clara, Calif. 95054
USA

1394 Trade Association Specifications are adopted by the 1394 Trade Association without regard to whether or not their adoption may involve patents on articles, materials or processes. Such adoption does not assume any liability to any patent owner nor does it assume any obligation whatever to the parties adopting the specification documents.

1 Suspend/Resume

1.1 Table of Contents

1.1 Table of Figures

1.2 List of Tables

Table 1-1 – Document Revision Table

Date	Revision	Comment
01/15/98	0.00	• Initial draft adapted from Suspend/Resume proposal draft revision 0.10 and Draft 1.4 of IEEE P1394a; submitted to limited distribution for initial comment;

Date	Revision	Comment
04/10/98	0.01	• Corrected some spelling errors; made corrections to bus signal figures; changed name references to disabling ports and the port it is connected to; clarified some corner cases; published to PCWG Web site
05/10/98	0.70	• Aligned document revision number to indicate percentage complete (realizing, of course, the number may not be anywhere close to being accurate - but does seem to serve as a factor to motivate people to read the document!).
06/10/98	0.71	• Changed document to reflect sponsor change (ECWG versus AWG)
07/11/98	0.72	• Began task of checking grammar, punctuation & spelling. Task aborted to begin a general overhaul of the entire document.
09/04/98	0.73	• Added Scope section; Modified Overview section; Broke Suspend/Resume Section into smaller, more succinct subject areas.
09/28/98	0.74	• Added implementation example (clause 9.0); Changed Trade Association address, etc. in Document Template to reflect latest information; added suspend propagation figure in 5.2.2.2, updated glossary to include 'Z' definition, made further breakouts in section 5.2.2.2 to cover Bias handshake protocol during the suspend process, aborted suspends and suspend fault conditions; made various editorial corrections.
11/30/1998	0.79	• Added “Reserved” command to table 5-3 • Changed heading title for clauses 5.3, • Added technical text to clauses 5.3, 6.11(Resume), 6.12 (Port Status Change), 7.1.2 – Resume to Resume Collisions, 7.1.3 – Resume to Suspend Collisions, 7.1.4 – Suspend/Resume Command Packets sent to a Port connected to a 1394-1995 PHY, 7.1.5 – Resume a command packets sent to a port connected to a powered off port, 8 – Suspend Manager, 9 – Implementation Guidelines • Deleted clause 1.2 – Resume (Exit from a Suspend-Connected State) • Added Clause 6.2 – Resume (Exit from a Suspend-Connected State)
12/01/98	0.80	• Added Implementation Block Diagram • Clarified text in “Implementation Guidelines” Clause • Made changes to interface circuitry diagram and timing diagram in “Implementation Guidelines” clause
01/04/99	0.90	• Updated table of contents and copyright notice in all footers • Corrected TA address in section one footer • Editorial changes to Clause 9 – Implementation Guidelines
09/03/99	.92	• Updated Document number, references to P1394a changed to IEEE 1394a-2000; updated LinkOn signal conditioning circuit and its associated Timing Diagram; incorporated all Ballot Review Comments

2 Glossary of Terms

The glossary below provides definitions, unique in some instances, for terms used in the discussion of suspend and resume mechanisms, protocol, and process:

Active Port	A connected, enabled port that observes *bias* and is capable of detecting all Serial Bus signal states and participating in the reset, tree identify, self-identify and normal arbitration phases.
Bias	Output signal (*Bias_detect)* of a comparator whose inputs originate from the TPB pair pins on the connector. The comparator detects incoming TpBias from the connected peer port.
Bias_Handshake	Time to drive or detect TpBias low during the handshake between a suspend initiator and target; also the time permitted a resuming port to generate TpBias after detecting bias (~16384/BASE_RATE). There are minimum and maximum times defined for ***Bias_Handshake.***
Boundary Node	A node with two or more ports, at least one of which is active and another suspended.
Bus Manager	provides (among other functions) bus and port/node suspend/resume management support
Child Port	a port in a node connected further from the 1394 root node
Clear	logic 0, not set, a logic low state
Data Prefix time	refer to MIN_DATA_PREFIX in IEEE 1394a - currently equal to 0.14 µs. When used, the term "***Data Prefix time***" refers to a period of time no less than MIN_DATA_PREFIX.
$Detect_{min}$	a period of time equal to 8,192 SCLKs - approximately 166.67 µs
Disable Initiator	a port that asserts the TX_DISABLE arbitration state to its connected peer port
Disabled Port	A port configured to neither transmit, receive, repeat Serial Bus signals or propagate reset. A disabled port shall be reported as disconnected in a PHY's self-ID packet(s).
Disable Respondent	a port that receives the RX_DISABLR arbitration state from its connected peer port
Disconnected port	A port whose connection detect circuitry detects no peer PHY port at the other end of a cable. It is not important whether the peer PHY is powered or whether the port is enabled.
Isolated node	A node without active ports; the node's ports may be disabled, disconnected or suspended in any combination.
OHCI	an acronym for Open Host Controller Interface
Peer Port	a port connected to the other end of the cable
Parent Port	a port in a node connected closer to the 1394 root node
Private Node	a node which either does not provide CSR or that has a value in its CSR identifying it as a node whose power state is not to be managed by any other node - a Private Node manages its own power state
Public Node	a node which has include a value in its CSR space that identifies it as a node that requires its power states to be managed by another node

RESET_DETECT	refer to RESET_DETECT in IEEE 1394a - currently defined as 80.0 milliseconds minimum and 85.3 milliseconds maximum
Resume Fault	a resume initiator port configuration in which the port ***Fault*** bit is one and ***Bias*** bit is zero. This condition occurs when a resume target fails to assert TpBias to its resume initiator. A port in this configuration may have the same power consumption characteristics as a port in a suspend state
Resume initiator	a port in a suspend state selected to assert TpBias to its connected peer (thus initiating a resume process).
Resume target	a port (whose fault bit is zero) in an isolated node which, while in a suspend state, detects *Bias* from the peer port to which it is connected and causes all other suspended ports in its node to become resume initiators.
Resuming Port	A previously suspended port which has observed *Bias* or has been instructed to generate TpBias. In either case, the resuming port engages in a protocol with its connected peer PHY in order to reestablish normal operations and become active.
Root Node	the highest priority 1394 node (always is the cycle master) that has all of its connected ports identified as Child Ports
RX_DISABLE_NOTIFY	Arbitration line state in which Arb_A = 1, Arb_B = Z; Enter the suspended state and initiate short bus reset on all other active ports. The peer PHY port will be disabled.
RX_SUSPEND	Arbitration line state in which Arb_A = '0,' Arb_B = '0'; Exchange TpBias handshake with the peer PHY and place the port into the suspended state. Also initiate suspend (i.e., propagate TX_SUSPEND) on all other active ports.
Set	logic 1, not clear, a logic high state
Short_Reset Interval	See SHORT_RESET_TIME in IEEE 1394 - currently equal to 1.30 μs minimum and 1.40 μs maximum
Suspend Connection	a port-to-port connection between two nodes in which both ports are in a suspend state
Suspend Domain	One or more isolated nodes connected to other isolated nodes via suspended connection(s). Two isolated nodes are part of the same suspended domain if there exists between them a physical connection. A boundary node connected to one or more suspended domain is not part of the suspended domain(s).
Suspend Fault	a suspend initiator port configuration in which the port ***Fault*** bit is one and ***Bias*** bit is one. This condition may occur when a suspend target fails to stop asserting TpBias to its suspend initiator (i.e. the suspend initiator continues to detect *Bias*). A port in this configuration must have the same power consumption characteristics as a port in a suspend state
Suspend initiator	An active port that transmits the TX_SUSPEND signal and engages in a protocol with its connected peer PHY to suspend the connection.
Suspend Manager	A term synonymous with Power State Manager (Part 3 of the 1394 Trade Association Power Specification)
Suspended node	An isolated node with at least one port that is in a suspend power state

Suspend port	A connected port not operational for normal Serial Bus arbitration but otherwise capable of detecting both a physical cable disconnection and received bias.
Suspend target	A suspended port that observes the RX_SUSPEND signal. A suspend target requests all of the PHY's other active ports to become suspend initiators while the suspend target engages in a protocol with its connected peer PHY to suspend the connection.
TX_DISABLE_NOTIFY	Arbitration line state in which Arb_A = Z, Arb_B = 1; Request the peer PHY port to enter the suspended state. The transmitting port will be disabled.
TX_SUSPEND	Arbitration line state in which Arb_A = '0,' Arb_B = '0'; Request the peer PHY to handshake TpBias and enter the suspended state. The request is also propagated by the peer PHY to its other active ports.
Z	A high-impedance electrical state - neither set (1) nor clear (0)

NOTE: All timing values are to be interpreted as the *maximum* time allowed for an operation to complete, a response to occur, or a stimulus to be applied. In actual application, all operations are expected to complete in the shortest time possible, responses are to be provided in the earliest time period possible, and stimulus shall be applied as soon as feasible.

3 Scope

This specification provides informative data pertaining, specifically, to the implementation and application of 1394 suspend/resume mechanisms (as defined in High Speed Serial Bus Specification IEEE 1394a. Portions of that standard have been incorporated into this specification. In instances where this specification is in conflict with IEEE 1394a, the IEEE document shall be the governing standard.

The implementation and guidelines suggested within this specification are representations of possible usage models and should be considered when implementing 1394a compliant silicon in a system where power conservation is of paramount concern.

Information contained in this specification should not be interpreted as a definitive source of all possible usage models. The incorporation of suspend/resume using 1394a mechanisms may vary for specific platform implementations.

The link (host controller) must also support low power mechanisms. However, this specification will not address any low power implementations of the link device (e.g. this specification will address low power management from the PHY perspective only).

The low power mechanisms describe in this specification deal with how a port is placed into a low power suspend state in either a port to port connection or on a bus wide basis. How a port resumes is also discussed. In addition, the effect of using the low power mechanisms has on the various portions of the PHY (e.g. the PHY/Link interface, the PHY port interface, and the PHY core) will be discussed.

Finally, methods of facilitating notification of various port events will be presented.

This document does not discuss the protocol or process by which a node gains power state control of a port, a set of ports in a node, or a set of nodes. Granting and denying a node ownership of power state control for a port, a set of ports, or a set of nodes is discussed in the 1394 Trade Association Power Specification, Part 3: Power State Management.

4 Overview

The purpose of the suspend/resume (low power) mechanisms in the PHY silicon is to provide a means of implementing 1394 interconnectivity in an environment in which power consumption is to be limited as

often as possible due to the limited amount of energy available. Suspend allows a 1394 interface to be put into a power-conserving "low-power" state, while remaining sensitive to "wake events" that can cause the system to resume normal (full-power) operation. Most implementations of this nature are likely to be battery operated devices.

Port suspend/resume mechanisms provide a facility for implementing a port and PHY low power conservation state while maintaining a port-to-port connection between 1394 bus segments.

While in a low power state, a port is unable to transmit or receive data transaction packets. However, a port in a low power state is capable of detecting connection status change and an ability to detect incoming TpBias. In addition, depending on the configuration of specific bits in certain PHY registers, the system platform may be notified of port status change events - including (but not necessarily limited to) a status change from: suspend, resume, connect, disconnect, disable, enable, and fault.

A system implementation of a leaf node may find it necessary to place its single port into a low power state yet maintain the ability to return to a fully operational state. Using port disable and enable commands, such a system may suspend its port(s) (and the peer ports they are connected to) without affecting any other node in the 1394 topology.

A system may be functioning as a bus/power manager and, as such, may require the ability to place all nodes (and their ports) into a low power state when it becomes appropriate to do so. While a few scenarios of this nature will be discussed in this specification, it should not be assumed that the specification includes a complete and comprehensive selection of possible, necessary, or appropriate scenarios for which PHY port low-power mechanisms are used to conserve energy consumption.

A port will transition to a low power state when it becomes disconnected or, while still connected, detects a zero (low) value for Bias (the port status bit which, when one (high) indicates the connected peer port is asserting TpBias). The circumstances under which a port will cease to assert TpBias are discussed in greater detail in subsequent sections.

A port must have its ***Enable*** bit set to one before it may be in an active state. A disconnected port is, by default, in a low power state. Upon being connected to a peer port, a previously disconnected port will become active upon receiving a bus reset after its ***Bias*** bit is one (high).

When a connected port is in a low power state, it will become active upon receiving a bus reset after its ***Bias*** bit is one (high) state.

5 Port Suspend

A port begins the process of transitioning to a suspend state when, while active, it is selected as a suspend initiator via a PHY command packet or becomes a suspend target (e.g. detects a properly framed RX_SUSPEND arbitration state).

The detail which follows has been extracted from the IEEE P1394a 2.0 draft standard. It has been included here so as to provide information necessary to understand the more comprehensive discussion of the suspend/resume process later in this specification.

5.1 Port Register Map

The register map in the table below identifies a port's register bits. The suspend/resume process and mechanism specifically make use of the bits: ***Connected***, ***Bias***, ***Int_Enable***, ***Disable***, and ***Fault***.

Table 5-1 -- Port Register map

	Contents							
Address*	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
(1)000$_b$	Astat			Bstat		Ch	Connected	Bias	Disabled
(1)001$_b$	Negotiated_speed			Int_Enable	Fault	Rsrvd		
(1)010$_b$	Rsrvd							
(1)011$_b$	Rsrvd							
(1)100$_b$	Rsrvd							
(1)101$_b$	Rsrvd							
(1)110$_b$	Rsrvd							
(1)111$_b$	Rsrvd							

*Note: The most significant bit of the address (parenthesized) is used only by the link during a PHY register read/write (Lreq). The most significant bit is implied to be one when addressing a port via a PHY remote packet (access or command)..

The table below presents the definitions for each of the bits found in the port register map.

Table 5-2 – Port Register Map Bit Field Definitions

Register (1)000_2

Field	Size	Type*	Power Reset Value	Description
Astat	2	Ro		TPA line state for the port: 00_2 = invalid 01_2 = 1 10_2 = 0 11_2 = Z
Bstat	2	Ro		TPB line state for the port (same encoding as Astat)
Ch	1	Ro		If set, the port is a child, else a parent. The meaning of this bit is undefined from the time a bus reset is detected until the PHY transitions to state T1: Child Handshake during the tree identify process (see 4.4.2.2 in IEEE Std 1394-1995).
Connected	1	Ro		If equal to one (set) the port is connected - else disconnected.
Bias	1	Ro		If equal to one (set) incoming TpBias is detected.
Disabled	1	Ro		If equal to one, the port is disabled. The value of this bit subsequent to a power reset is implementation dependent. If a hardware configuration option is provided, a single option may control the power reset value for all ports.

Register (1) 001_2

Field	Size	Type*		Description
Negotiated_speed	3	Ro		Indicates the maximum speed negotiated between this PHY port and its immediately connected port; the encoding is the same as for the *xspd* bit in self-ID packet 7 (see clause 6.2.1 of the IEEE P1394a, Draft 2.0, specification).
Int_Enable	1	rw	0	Enable port event interrupts. When set to one, the PHY shall set Port_event to one if any of Connected, Bias, Disabled or Fault (for this port) change state.
Fault	1	rw	0	Set to one if an error is detected during a suspend or resume operation. A write of one to this bit clears it to zero.
Rsrvd	3	Ro		reserved - must not be used or interpreted as having a meaningful value

Register: (1) 010_2 through (1) 111_2

Field	Size	Type*		Description
Rsrvd	8	Ro		Reserved - must not be used or interpreted as having a meaningful value

5.2 Suspend/Resume Overview

The suspend mechanism allows pairs of directly-connected ports to be placed into a low-power, suspended state. Any active port may transition to a suspend state when:

1. it is addressed with a *command packet* (see clause 5.2.1) with the ***cmnd*** field set to a value of two (initiate suspend - i.e., become a suspend initiator). The command packet may have been transmitted by the PHY's local link or by some other node.

2. it detects a properly framed RX_SUSPEND
3. it is in a PHY in which another active port in the same PHY detects a properly framed RX_SUSPEND
4. it detects a properly framed RX_DISABLE_NOTIFY,
5. it no longer detects ***Bias***

5.2.1 Command Packet

A port addressed with a command packet which has a ***cmnd*** value of two (010_2) becomes a suspend initiator. The format for the command packet is illustrated by the figure and table which follow:

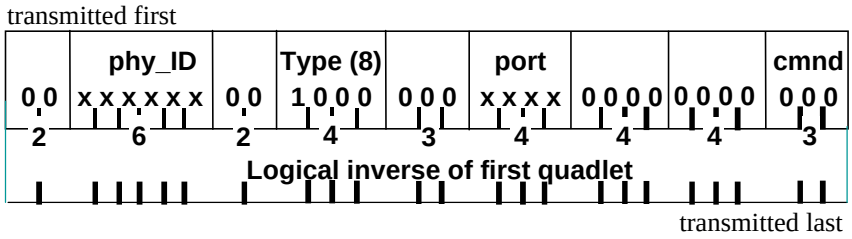

Figure 5-1 Command packet format

Table 5-3 Command packet field definitions

Field	Comment
phy_id	Physical node identifier of the destination of this packet
type	Extended PHY packet type (8 indicates command packet)
port	This field selects one of the PHY's ports
cmnd	Command: 0 NOP 1 Transmit TX_DISABLE_NOTIFY then disable port 2 Initiate Suspend (become a suspend initiator) 3 Reserved 4 Clear the port's Fault bit to zero 5 Enable port 6 Resume Port 7 Reserved

5.2.1.1 Confirmation Packet

Upon receipt of the command packet, the node in which the addressed port resides, will respond with a confirmation packet. The format and content of the confirmation packet is described and illustrated in the figure and table which follow:

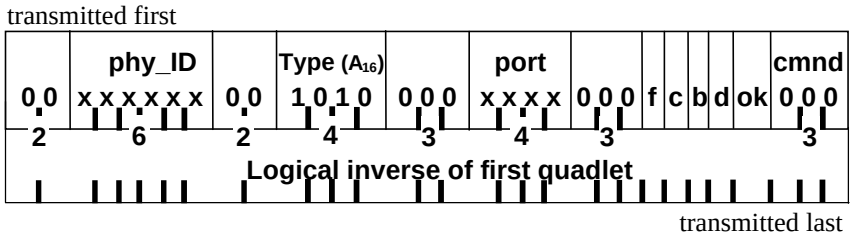

Figure 5-2 Confirmation Packet Format

Table 5-4 Confirmation Packet Field Definitions

Field	Comment
phy_id	Physical node identifier of the source of this packet
type	Extended PHY packet type (A_{16} indicates remote confirmation packet)
port	This field shall specify the PHY port to which the command action is to be applied
fault	Abbreviated as "*f*" in the figure above, this bit is the current value of the Fault bit from PHY register 101_2 for the addressed port.
connected	Abbreviated as "*c*" in the figure above, this bit is the current value of the Connected bit from PHY register 1000_2 for the addressed port.
bias	Abbreviated as "*b*" in the figure above, this bit is the current value of the BIAS bit from PHY register 1000_2 for the addressed port.
disabled	Abbreviated as "*d*" in the figure above, this bit is the current value of the Disabled bit from the PHY register 1000_2 for the addressed port.
ok	This bit will be set (one) if the command was accepted by the PHY and zero otherwise.
cmnd	The cmnd value (from the preceding remote command packet) with which this confirmation packet is associated.

5.2.2 Command Packet/Confirmation Packet Bus Signals

Following *idle* (ack_gap) from the confirmation packet, the suspend initiator generates DATA_PREFIX followed by a short bus reset to all other active ports in its node while generating the arbitration line signal TX_SUSPEND to the suspend target (the suspend target receives RX_SUSPEND).

Note that while DATA_PREFIX is received by all active ports in the suspend initiator node, the RX_SUSPEND arbitration signal is received by the suspend target.

The DATA_PREFIX is followed by a short bus reset on the active ports of the suspend initiator node, causing the creation of an active domain (which no longer includes the suspend initiator nor the "*tree*" connected to the suspend initiator). During the self-ID phase, suspended ports (or ports in the process of changing state from active to suspend) are reported as "not active" (not connected in 1394-1995 terms) by the PHY in which they reside.

The figures below illustrate bus signals at the input of the suspend target port and all other active ports in the node containing the suspend initiator that are transmitted during the suspend process. The top pair of event sequences illustrates the bus signals seen by the suspend target. The bottom pair of event sequences illustrates the signals transmitted from all other active ports in the node containing the suspend initiator.

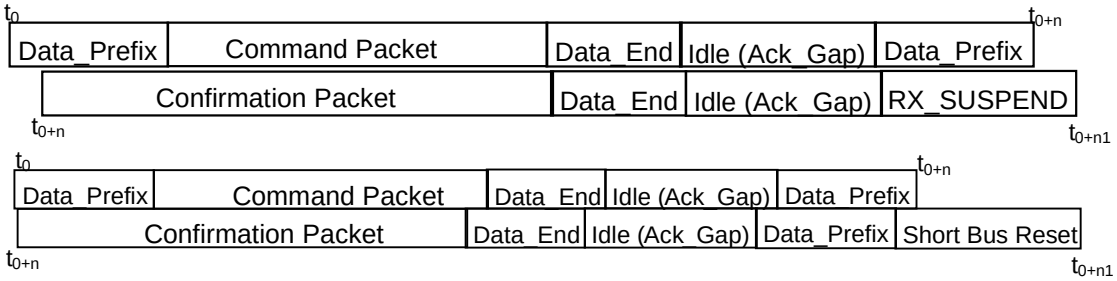

Figure 5-3 Bus Signals During the Suspend Process

The node containing the suspend target, upon detecting the RX_SUSPEND arbitration state, causes all of its active ports to become suspend initiators - generating TX_SUSPEND to their connected peer ports (causing all active ports to propagate the received suspend request).

The figure below is provided to make more clear the process which occurs during suspend propagation.

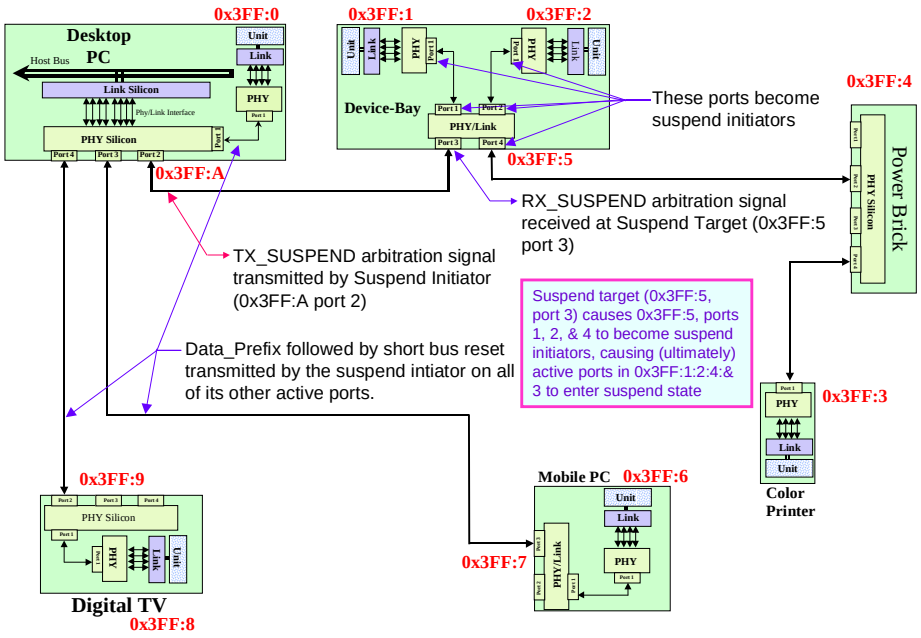

Figure 5-4 Suspend Propagation through an active bus

5.2.2.1 *Bias* Handshake Protocol during Suspend Process

The suspend target acknowledges receipt of RX_SUSPEND by driving TpBias low to the suspend initiator. TpBias is driven low for a period of time no greater than **BIAS_HANDSHAKE$_{MAX}$**. While driving its TpBias low, the suspend target is monitoring ***Bias*** (incoming TpBias from the suspend initiator) - waiting for it to go to zero.

After sending TX_SUSPEND to the suspend target, the suspend initiator monitors ***Bias*** (incoming TpBias from the suspend target) - waiting for ***Bias*** to go to zero.

When, during the BIAS_HANDSHAKE wait period, the suspend initiator detects ***Bias*** equal to zero, the suspend initiator will drive the output of its TpBias low until the PHY silicon internal ***connect_detect*** circuitry becomes active (after which the suspend initiator disables its TpBias generator - placing its output into a high impedance state).

When the suspend target detects ***Bias*** equal to zero, it will place its TpBias generator into a high impedance state. At this point, both the suspend initiator and the suspend target transition into a low-power suspend connected state. The only circuitry required to remain powered during this state is the connect detect circuitry and the incoming TpBias detection circuitry.

5.2.2.2 Suspend Initiator Aborts "Suspend"

When the suspend initiator (for whatever reason) aborts the suspend process, the suspend target will continue to detect ***Bias*** after the expiration of **BIAS_HANDSHAKE$_{MAX}$**. The suspend target will assert TpBias to the suspend initiator as well as cause all of the ports in its node that may be in the process of suspending to also assert TpBias to their suspend targets (causing them to, essentially, become resume initiators) - the suspend process is aborted.

5.2.2.3 Suspend Fault

When a suspend initiator continues to detect ***Bias*** from the suspend target after the expiration BIAS_HANDSHAKE$_{MAX}$, the suspend initiator will set its ***Fault*** bit and then drive its TpBias to the suspend target low until the PHY silicon internal ***connect_detect*** circuitry becomes active. The suspend initiator then places the output of its TpBias generator in a high impedance state. The suspend initiator enters into a suspend state as a suspend-fault condition (e.g. ***Bias*** = ***Fault*** = one).

5.3 Single Connection (port-to-port) Suspend

It may be necessary to suspend a specific port without affecting other nodes in the 1394 bus topology. This may be accomplished by using the port disable and enable mechanisms. First, the disable initiator port is disabled (which will result in its connected peer port becoming suspended) and then, after ***Bias*** transitions to a low (zero) state (and is maintained there) the port is then enabled (thus allowing it to be "resumed" when and if the connected peer asserts TpBias.

A single port-to-port connection may only be suspended between two 1394a PHY nodes. In certain implementations, a 1394-1995 PHY device may discontinue generating TpBias when it no longer sees TpBias. A single port-to-port connection may be suspended when connecting to nodes with this type of implementation.

However, typically, a 1394-1995 PHY implementation continues to generate TpBias as long as the PHY is powered. If the "*disable port*" method is used in an attempt to establish a port-to-port connection when a 1394a port is connected to this type of typical 1394-1995 implementation the 1394a port will resume when the it is enabled - the 1394a port will detect ***Bias*** equal to one and, thus, become a resume target. Software/firmware should read the status of ***Bias*** before enabling the 1394a port. If ***Bias*** is set to one, the port should not be enabled. Instead, software/firmware should set the disabled ports INT_ENABLE bit to one. The system will then be notified when the disabled port status changes. A disabled port may generate a PHY status packet with PHY Interrupt bit set when its ***Connected*** bit changes state. Upon receiving a PHY interrupt due to port status change on the disabled port, software/firmware may then elect to enable the port.

When a 1394a port is connected to a previously-disabled port on another node, the newly-connected port will detect the connection to the disabled port. This will cause the newly-connected 1394a port to assert its TpBias in an attempt to resume the disabled port. Because the port being connected to is disabled, it will not participate in the resume "***bias*** handshake protocol". The newly connected 1394a port will return to a suspend state with its fault bit set to one.

A port which has been disabled will not respond to the detection (or loss) of ***Bias***, suspend, resume, or fault clear commands (whether from the local link or a remote node). A disabled port will not pass through, respond to, or generate bus arbitration states - including bus reset. A disabled port will respond to a port enable command (either from the local link or through another active port in the same node). The connect-detect logic for a disabled port remains active, in addition to circuitry necessary to support, when enabled, port interrupt (providing link notification when the disabled port's ***Connected*** bit changes state) - all other circuitry for a disabled port is not required to be powered.

A port becomes disabled when it receives a disable port command (either from the host through the link or from a node on the 1394 cable bus). It may also be disabled via an Lreq from the host link which directly sets the ports ***Disable*** bit.

When a port's ***Disable*** bit is directly set by the host link (via an Lreq) the port does not arbitrate for the bus, generate or participate in any disable notification protocol with its connected peer port - the port being disabled discontinues driving TpBias to its connected peer port by disabling its TpBias generator (causing its output to enter into a high impedance state). No other action is taken in this instance (other than link notification as the result of the port's ***Int_Enable*** bit being set (one).

When disabling a port by addressing it with a command packet, the node generating the command packet must arbitrate for the bus. The figure below illustrates the bus signals seen by all active ports (except the port being disabled and the port it is connected to) on the bus as the result of a port in a node receiving a port disable command:

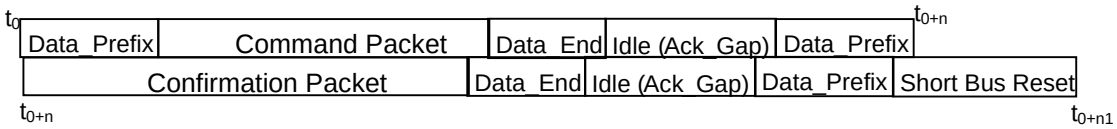

Figure 5-5 Bus Signals seen by all active ports except port being disabled

The port addressed with the port disable command is referred to as the disable initiator port and is the port which will be disabled. The port connected to the disable initiator port is referred to as the disable respondent port.

The disable initiator port generates Data_Prefix followed by a bus reset on all other active ports in its node. The disable initiator port generates the TX_DISABLE_NOTIFY line arbitration state to the disable respondent port while it is generating Data_Prefix to all other active ports in its node. The bus reset creates a new bus topology for active ports in the disable initiator node that will exclude the disable initiator port and any nodes connected to it via the disable respondent port.

Immediately after transmitting the TX_DISABLE_NOTIFY arbitration signal to the disable respondent port, the disable initiator port will drive the output of its TpBias generator low until ***connect_detect*** becomes valid, after which the disable initiator port will disable its TpBias generator, causing its output to go to high impedance.

The figure below illustrates the bus signals seen by disable respondent port:

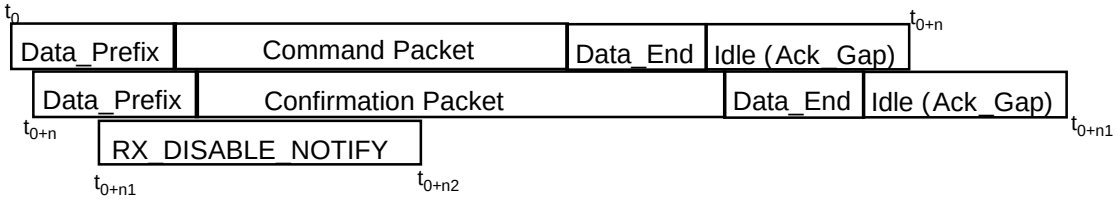

Figure 5-6 Bus Signals received by Disable Respondent from Disable Initiator

When a disable respondent port detects the RX_DISABLE_NOTIFY arbitration state, it will assert Data_Prefix followed by a bus reset on all other active ports in its node (i.e. it will not repeat the TX_DISABLE_NOTIFY arbitration state - it replaces the RX_DISABLE_NOTIFY arbitration line state with

Data_Prefix). Note: the disable respondent port does not need to arbitrate for the bus to assert the bus reset. Bus arbitration has already been awarded to the disable initiator and the disable respondent port is acting as a signal repeater with the unique ability to substitute Data_Prefix followed by a bus reset for the RX_DISABLE_NOTIFY arbitration line states - which, by the way, are the same signals transmitted to all other active ports in the disable initiator ports node.

It should be noted that if the disable respondent port is losing its parent port, the reset generated will be a long reset, otherwise, a short bus reset will be generated.

5.4 Suspending - sudden *Bias* Loss

A port will suspend due to sudden loss of ***Bias*** while a connection to its peer port is maintained.

When an active port detects its ***Bias*** transition from one to zero (for whatever reason - perhaps the connected port powers off) it will generate a reset on all active ports in its node. In addition, it will drive the output of its TpBias generator to a low state until ***connect_detect*** becomes active, after which it will disable its TpBias generator (whose output will transition to a high impedance state). The port will enter into the suspend state while ***Connected*** is still one (indicating a cable connection continues to exist). If ***Connected*** transitions to zero the port will enter into the disconnected state.

6 Resume & Port Status Change Notification

There are occasions when it is desirable to notify the link of a PHY port status change - either when one or more ports in the PHY begin a resume process or when the status of a specific port(s) change. A change in port status occurs when any one or more of a ports ***Bias***, ***Fault***, ***Connected***, or ***Disable*** bits change state.

Software (or firmware) on the host side of the link must configure the PHY to provide notification to the host when a port status change occurs. The host may receive notification of port status change activity when the link is no longer active (powered-off or otherwise in some non-functional power state).

Software on the host side of the link may configure the WATCHDOG bit in the PHY to one. This will cause the PHY to generate a system notification event when the PHY detects a resume event on any port.

In addition, 1394 hardware and system software/firmware implementation may be such so as to facilitate complete suspend/resume for the node/unit from an external point of control (through the PHY port connection). The PHY may be configured to generate link notification via appropriate setting/clearing of the PHY WATCHDOG bit and/or individual port INT_EN bits. Details regarding this configuration can be found in the 1394 Trade Association Power Specification, Part 3: Power State Management

Host notification of port activity may come from the link or from the PHY (depending upon specific hardware implementations). Host notification may come from the link when the link is active (e.g. LPS is asserted to the PHY) when the PHY generates a PHY interrupt status packet as the result of a port event.

Host notification of port activity may come from the PHY, for example, when a hardware implementation conditions the LinkOn output signal from the PHY to generate a host interrupt (in a PC, for example, an SMI, SCI, PME or some other wake/notify event alerting system software/firmware of the link on signal from the PHY).

When the host recognizes the PHY port event, host system software/firmware would executes tasks appropriate for servicing the PHY port event - including, but not limited to: powering on/up the link and/or other hardware mechanisms required to support full 1394 data transactions.

6.1 Types of port change notification

There are several different types of activity which may occur on a port to cause notification - all require the PHY to be conditioned to cause notification for a particular port activity.

6.1.1 Resume

When host system software/firmware has set the RESUME_INT in the PHY to one, host notification will occur when the PORT_EVENT bit in the PHY transitions from zero to one. When RESUME_INT is set to one, PORT_EVENT will transition from zero to one when any port's BIAS bit transitions from zero to one. If the port is disabled, however, transitions of the port's BIAS bit will not result in a transition of the PORT_EVENT bit. NOTE: There are specific configurations in which PORT_EVENT will transition even when the port is disabled - these configurations will be discussed in a subsequent section.

The PORT_EVENT bit will remain set to one until host system software/firmware clears it to zero.

There are two specific link power scenarios of interest when a PHY port resumes (as either a resume target or a resume initiator). The first, and simplest, is when the link is active (i.e. LPS is asserted to the PHY and the Link_active bit in the PHY is set to one). In this instance, the PHY will generate a PHY interrupt packet to the link. The link will, in turn, notify system software/firmware of the PHY interrupt event - the software/firmware must enumerate the source of the PHY interrupt via appropriate PHY register reads.

The second scenario is one in which the link is not active (e.g. LPS is not asserted, the PHY Link_active bit has been cleared to zero, or, perhaps, the link is not powered) The PHY will generate a LinkOn signal when the PHY PORT_EVENT bit is set to one or when any port ***Bias*** bit transitions from zero to one when the PHY ***RESUME_INT*** bit is set to one. The LinkOn signal will continue to be generated until the PHY detects LPS from the Link AND the PHY Link_active bit is set one.

It should be noted that, in some implementations, the PHY may generate a PHY interrupt status packet when LPS becomes asserted when the PHY Link_active bit is set to one and while PORT_EVENT continues to be one. This condition may occur when software/firmware sets to one the PHY Link_active bit prior to clearing PORT_EVENT to zero (to acknowledge receipt of the PHY link on signal). The fact that PORT_EVENT is set to one and LPS is asserted and the Link_active bit is one, will cause the PHY to generate an interrupt status packet. Software/firmware always clear the PORT_EVENT prior to setting the Link_active bit to one if this condition is to be avoided.

It should also be noted, in a situation in which the PORT_EVENT occurs just as the link is going inactive (e.g. removing LPS to the PHY while Link_active is set to one) the PHY may transmit an interrupt status packet to the link without the link receiving it properly - therefore, system software/firmware may not be made aware of the PORT_EVENT until much later when, after transmitting the interrupt status packet, the PHY detects the absence of LPS while PORT_EVENT is one and, therefore, generates a LinkOn signal. The LinkOn signal will continue to be asserted until LPS is asserted to the PHY and PORT_EVENT is cleared to zero (note: once LPS is asserted, the PHY will generate another PHY interrupt status packet to the link because the Link_active bit is still set. Software/firmware should always clear the Link_acive bit to zero prior to removing the LPS from the PHY - LPS is deasserted when the link receives a soft reset).

6.1.2 Port Status Change

PORT_EVENT may be configured to transition from zero to one even though the PHY RESUME_INT bit is zero. In this configuration, one or more PHY ports will have been configured to generate a PORT_EVENT upon a status change. A PHY port is configured to generate a PORT_EVENT when its INT_ENABLE bit has been set to one.

When a ports INT_EMABLE is set to one, the PHY PORT_EVENT bit will set to one whenever any of a port's ***Connected***, ***Bias***, ***Disabled*** or ***Fault*** bits change state. If, however, a port has been disabled prior to the port's INT_ENABLE bit being set to one, PORT_EVENT will not set to one for any port status bit except ***Connected***. Note: some early implementations of 1394a PHY devices may not set PORT_EVENT to one when a disabled port's ***Connected*** bit transitions from one to zero - even though INT_ENABLE may be set to one. All 1394a PHY implementations will set PORT_EVENT to one when a ports ***Connected*** bit transitions from zero to one.

6.2 Resume (Exit from a Suspend-Connected State)

A resume event is generated when a port asserts TpBias to the suspended port to which it is connected. The port which first asserts TpBias functions as a resume initiator. The port which first detects assertion of TpBias from a resume initiator functions as a resume target. Neither port will enter an active state until it detects a bus reset.

Unless it is a boundary node, the node in which the resume target resides will cause any of its other suspended ports to become resume initiators.

A resume initiator in a boundary node will wait three times the ***RESET_DETECT*** interval (approximately 240 milliseconds) to detect a bus reset before arbitrating the bus to apply a short reset.

A resume initiator in a node other than a boundary node will wait seven times the RESET_DETECT interval (approximately 560 milliseconds) to detect a bus reset before issuing a long reset itself.

Nodes containing resume targets only do not issue bus resets.

7 "*Corner*" Cases

There are specific situations in which it may not be intuitively obvious as to how a node will behave when starting, completing, or in the process of suspending or resuming and the node receives some other instruction (whether via a remote command packet or Link Request - LReq). This section discusses each of the known corner cases and outlines the expected behavior.

7.1.1 Suspend to Suspend Collisions

Suspend to suspend collisions occur when a port in a PHY is participating in the suspend process (either as an initiator or a target) and another port in the same PHY is selected as either a suspend initiator or a suspend target prior to the first suspend process completing.

One corner case occurs when a port in a PHY has been selected as a suspend initiator and, for whatever reason, another port in the same node becomes a suspend initiator (from either the link or a remote node) before the first suspend initiator has completed its process.

One other corner case is when a PHY in the process of suspending all of its ports (because one of them had become a suspend target) and the link endeavors to select one of the PHY ports (or a port remote to the PHY) as a suspend initiator prior to the existing suspend process completing.

7.1.2 Resume to Resume Collisions

These collisions occur when a suspended port in a PHY has been selected as a resume initiator and another (suspended) port in the same node becomes a resume target. This situation will occur when a suspend domain on an existing bus is the process of resuming and a new device is attached to previously disconnected port in the resuming suspend domain. When this occurs, the newly connected device becomes part of the resuming suspend domain - the existing resuming ports are not affected. The newly attached device will become active upon completion of the resume process.

In the instance where the selected resume initiator is in a boundary node which contains other suspended ports and other disconnected ports, when a device becomes attached to a disconnected port the other suspended ports do NOT become resume initiators. The newly connected device and the port to which it has been connected participate in the TpBias handshake protocol discussed in clause 6.1.1.

7.1.3 Resume to Suspend Collisions

These type of collisions occur when a port in a node is in the process of either suspending or resuming and another port in the same PHY is selected to perform the opposite function.

In the instance where ports in the node are in the process of suspending and a previously suspended port becomes a resume target, the suspending ports will complete the suspend process after which all newly suspended port will become resume initiators.

In the instance where ports in a node are in the process of resuming and an active port becomes a suspend target, the suspend target does not respond to RX_SUSPEND (it ignores the arbitration state as if it has not been received). This precludes the suspend target from propagating the suspend request. The suspend target does not suspend.

7.1.4 Suspend/Resume Command Packets sent to a Port connected to a 1394-1995 PHY

A suspend command packet sent to a 1394a port connected to a 1394-1995 port will, in most instances, result in the 1394a port entering into a suspend state with its ***Fault*** and ***Bias*** bits set to one. In instances where an implementation of a 1394-1995 node results in the 1394-1995 port no longer asserting TpBias when it detects a loss of TpBias, the 1394a port will enter into a normal suspend state after the 1394-1995 port stops transmitting TpBias (i.e. the 1394a port will detect the loss of incoming TpBias and its ***Fault*** and ***Bias*** bits will clear to zero.

When connected to a 1394-1995 port which is driving TpBias, a suspended 1394a port which has its ***Fault*** bit set will resume (clearing its ***Fault*** bit to zero) when the 1394a port receives a resume command.

7.1.5 Resume command packets sent to a port connected to a powered off port

A 1394a port connected to a powered off port may be in a suspend state. It will return to a suspend state with its ***Fault*** bit set when it endeavors to process a resume command.

8 Suspend Manager

The "***Suspend Manager***" is a component of the bus Power Manager and, as such, is described in greater detail in Part 3 of the 1394 Trade Association Power Specification (Power State Management). The information provided here has been included in this specification as an introduction and an overview of the details provide in Part 3. The reader is encouraged to read Part 3 of the Power Specification for a better understanding of the terms, procedures, and protocols used in an implementation of the Suspend manager.

The Suspend Manager develops a bus topology of Private and Public nodes. A Private node is a node which maintains private ownership for the power state of all its own PHY ports, the link to which the PHY is attached and all of the units that may reside on the host link (at least those that have been exposed to the 1394 cable bus via 1394 CSR contents). A Private node excludes any other node on the cable bus from suspended or disabling any of its ports. A Private node is identified by: a) an appropriately identified value in its CSR space; b) no provision for power state management in CSR space.

A Public node is a node which allows other nodes on the 1394 cable bus to place any or all of its ports into a suspend or disabled state. A Public node does not change the power state of any of its ports, link, or units – a Public node may only change power state when directed to do so by another node. A Public node may provide notification to another node on the 1394 cable bus (the node that has responsibility for managing the power states of the Public node) that it is able to have its power state changed. A Public node must be identified by an appropriately identified value in its CSR space.

A Public node may be divided into two classes: 1) Direct and 2) Indirect. A Direct Public node is one that all power policy decisions for its power state has been abdicated to another node on the 1394 cable bus. An Indirect Public node maintains ownership for its own power policy, however, it will accept requests from other nodes on the 1394 cable bus to change its power state within the bounds of its own power policy.

Leaf nodes (e.g. single port devices) are typically configured as Public Direct nodes. A leaf node typically abdicates to a separate node on the 1394 cable bus the ownership of its power policy. The Power Policy Owner (PPO) of a node manages the nodes various low power states.. The PPO for a leaf node registers

with the Power State Manager as a Public Direct or Public Indirect Power Policy Owner for the leaf node. The PPO also registers with the leaf node. The leaf node has a register set that identifies its PPO and the Power State Manager. A node that registers as the Public Indirect PPO for a node has the capability to accept and manage power state requests from other nodes that may be sharing the leaf node being managed. The Public Indirect PPO serves as the proxy for all power state management requests from other nodes to the leaf node.

When another node has been granted node power policy ownership status, the power policy ownership may or may not extend to all components of the node (e.g. all ports, link and units). The node must be designed to enable host notification of PHY port status changes. When the node is directed by the PPO to suspend (or resume) via an active PHY port connection, the software/firmware responds with appropriate action.

A Public Direct node shall set its PHY ports INT_ENABLE bit to one. A node must have a source of power to its PHY when the node is in any managed low power state (physical OFF is not considered a "managed" low power state). A PHY responds to a port resume event by generating link notification either a PHY status packet with PHY interrupt set or by generating a LinkOn signal.

When a node has been assigned a PPO, the peer port to which the node is connected is assigned the same PPO. If the peer port connection has a previously assigned PPO, the current node requesting PPO grant will be denied. The Power/Suspend Manager never grants a PPO request for a PHY port which would be different than the PPO for the PHY port connected to the port for which the request is being made.

A node (and its connected peer port) that does not have an assigned PPO is managed by the Power/Suspend manager.

9 Implementation Guidelines

There are probably as many ways to implement low-power designs using 1394a PHY silicon as there are vendors which manufacture 1394a compliant silicon. The implementation guideline presented here is one which, when used in all implementations, has a tendency to eliminate as many interoperability issues as a common implementation is able.

The low-power mechanisms incorporated into 1394a silicon require software (or firmware) to stimulate them in order to realize the full benefit available. Therefore, this section will also include guidelines to the developer of software/firmware components. These guidelines are not intended to resolve all issues which might occur within a system incorporating low-power mechanisms. However, using these guidelines, the designer may find information needed to incorporate a complete low power implementation.

The presumption of these guidelines is that the system host within a given node (incorporating the link silicon) will want to maintain control and configuration of all ports contained in the PHY attached to its local link. With this in mind, the reader must understand that system notification is required when any port event of interest occurs.

System notification may be realized by a LinkOn signal from the PHY (when the link is not active*) or when the PHY creates and sends to the link a 4-bit PHY interrupt status packet. In the latter case, it is incumbent upon system software/firmware to properly process the PHY interrupt status packet.

*NOTE: "Link not active" defined as when the logical and of the LPS signal from the link and the PHY register Link_active bit is zero.

System notification may be realized on a port by port basis and/or all port basis.

Notification on an all port basis is enabled when the PHY register bit RESUME_INT is set to one (high). When one, the PHY register bit RESUME_INT will enable system notification to occur when any port on the PHY detects Bias transition from zero to one (a resume event). When the link is not active, the PHY will assert a LinkOn signal when the PHY PORT_EVENT bit becomes set to one.

The PHY PORT_EVENT bit will set to one when any port ***Bias*** bit transitions from zero to one AND the PHY register bit RESUME_INT is set to one.

If the link is active (both the PHY Link_active bit is one and the LPS signal from the link is asserted), the PHY will create a 4-bit PHY interrupt status packet and send it to the link across the PHY/Link interface when the PHY PORT_EVENT bit is set to one. System software/firmware must process the information from the Link. NOTE: As of this writing, not all link designs recognize PHY status packets which are sent to the link after the PHY status packet sent subsequent to a bus reset (containing the node-ID).Notification on a port by port basis is enabled when the INT_ENABLE for a port has been set to one. When a ports INT_ENABLE bit is set to one, the PHY PORT_EVENT bit will set (ultimately causing system notification as outlined in these guidelines) when one or more of the following occur:

1) Bias bit transitions
2) Fault bit transitions
3) Disabled bit transitions
4) Connected bit transitions

Other events (not related to power management mechanisms) which will cause the PHY to assert the LinkOn signal when the link is not active are: Loop, power fail, and time-out.

System event notification may occur in one of two ways:

1) When the PHY Link_active bit is set to one, and LPS has previously been asserted (and recognized by the PHY to be asserted), the PHY PORT_EVENT bit will be set to one (as a result of one of the previously mentioned four events). The PHY will transmit a 4-bit PHY interrupt status packet to the link over the PHY/Link interface. The PHY status packet will have bit 3 set (one) - indicating a PHY interrupt event. Software/firmware may have previously disabled the PHY interrupt mask in the OHCI/Link IntMask register (bit position 19) thus creating a system interrupt from the OHCI controller when the link receives the PHY interrupt status packet. Software/firmware must poll the various registers within the PHY to determine the cause/source of the PHY interrupt. Software/firmware must reset the PORT_EVENT bit to zero to clear the PHY interrupt event. If LPS becomes de-asserted or the PHY Link_active bit is cleared to zero (e.g. the link becomes not active) while PORT_EVENT continues to be set to one, the PHY will cause its link on signal pin to be asserted. The LinkOn signal will continue to be asserted until the link become active and PORT_EVENT has been cleared to zero.

   When the link is not active, and the PHY PORT_EVENT bit sets to one, the PHY will cause its link on signal to be asserted. The PHY will continue to assert its LinkOn signal until the link becomes active and PORT_EVENT is cleared to zero by software/firmware. When the link becomes active, software/firmware must take into account that the PHY will transmit a PHY interrupt status to the link if the software/firmware has not cleared the PHY PORT_EVENT bit to ZERO before the PHY detects an active link condition - which will be the case if software/firmware does not clear the PORT_EVENT bit to zero before setting the Link_active bit to one (the PHY/Link interface cannot become active until AFTER the PHY has detected LPS). Software/firmware may decide to mask the PHY interrupt in the OHCI/Link IntMask register (bit position 19) prior to asserting LPS.

   In OHCI 1.1 implementations, the LinkOn signal from the PHY may be configured to cause a PCI power management event (PME). The PCI PME# signal may be used to "wake" the system or some component of the system.

   In OHCI 1.0 (or non-OHCI implementations), the LinkOn signal from the PHY may be the stimulus used to create a 1394 power management event. The actual power management event may be an interrupt signal to an embedded controller or a state transition on a wake signal pin on the platform core logic chip set.

   In any event, system software/firmware shall respond appropriately to the 1394 power management event.

   The block diagram below provides a high level view of the connection between the various components.

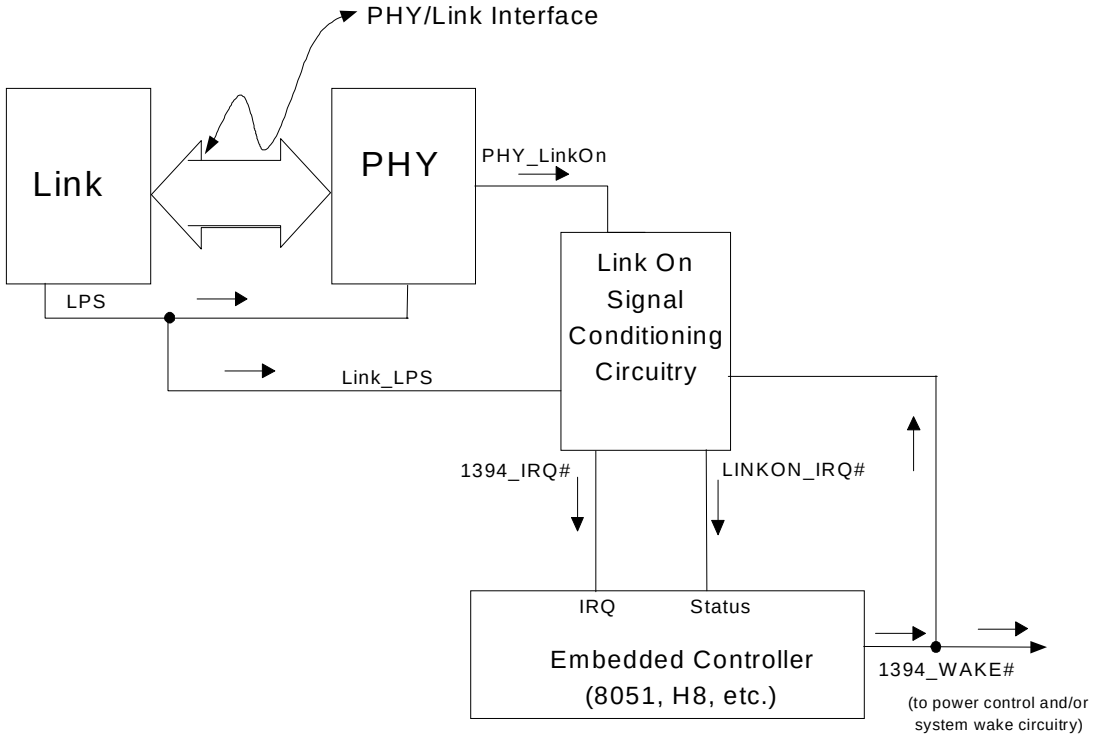

Figure 9-1 - Implementation Block Diagram

Upon power-on reset, the signals ***LINKON_IRQ#*** and ***1394_IRQ#*** will be in a logic high (one) state. The signal ***1394_IRQ#*** is the signal used to generate the 1394 power management event notification to the platform. The signal ***LINKON_IRQ#*** is the 1394 power management event status and is asserted low - zero (active low - zero). ***LINKON_IRQ#*** is asserted upon the first trailing edge of the ***PHY_LinkOn*** signal. As ***LINKON_IRQ#*** transitions from one to zero (asserts), the signal ***1394_IRQ#*** is asserted low (zero).

The embodiment of this circuit assumes ***1394_IRQ#*** will connect to an IRQ signal pin on an embedded controller. The embedded controller IRQ signal pin may be the only IRQ signal pin available and may, therefore, be shared by other sources of interrupts to the embedded controller. The signal ***LINKON_IRQ#*** is used as an interrupt status signal - identifying the source of the interrupt as a 1394 wake event in implementations for which the interrupt signal pin is stimulated by more than one source. Thus, when ***1394_IRQ#*** asserts, the embedded controller firmware interrupt service routine would qualify the source of the interrupt by polling a register containing the various interrupt status signals - determining the source of this interrupt to be the 1394 Link On signal from the PHY. The firmware then responds to the 1394 interrupt by asserting ***1394_WAKE#*** low (zero). This signal could be used to power up appropriate portions of the platform which have been placed into a lower energy conservation state (or even powered off).

The assertion of ***1394_WAKE#*** will deassert the ***1394_IRQ#*** signal. In addition to causing ***1394_WAKE#*** to be asserted, the embedded controller firmware would notify system software/firmware of the need to have the link controller assert the LPS signal to the 1394 PHY (enabling the PHY/Link interface). It is assumed that system software/firmware would have cleared the PHY Link_active bit to zero previously (just before it previously had removed the ***LPS*** signal from the PHY).

Once ***LPS*** has been asserted by the link controller to the 1394 PHY, the first rising edge of ***LPS*** will cause the signal ***1394_IRQ#*** to be asserted again. This results in the embedded controller receiving a second interrupt - notifying it that the link has asserted the ***LPS*** signal, and that the embedded controller should now deassert the ***1394_WAKE#*** signal (which it does). Noting that the ***LPS*** signal is (most likely) a differentiated signal, on the first occasion when ***LPS*** is high coincident with ***1394_WAKE#*** being high, both J-K flip flops will be preset, causing both ***1394_IRQ#*** and ***LINKON_IRQ#*** to deassert.

The LinkOn signal from the PHY is a digital signal of a minimum frequency of 4 MHz (a maximum of 8 MHz) with a duty cycle of not less than 40% and not more than 60%. This signal must be the source of an interrupt for state change notification when the link is not active.

The circuit diagram below embodies a method for both converting the LlinkOn signal to a single pulse event as well as providing a signal which can be used to identify the source of the interrupt.

The presented circuit is not to be interpreted as the best embodiment of the endeavor, but has been designed in this manner to make as clear as possible to show the necessary signal conversions.

INPUTS					OUTPUTS	
$\overline{PRE}$	$\overline{CLR}$	CLK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H†	H†
H	H	↓	L	L	Q_0	$\overline{Q_0}$
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	TOGGLE	
H	H	↓	X	X	Q_0	$\overline{Q_0}$

†This configuration is nonstable, that is, it will not persist when either PRE or CLR returns to theinactive (high) level

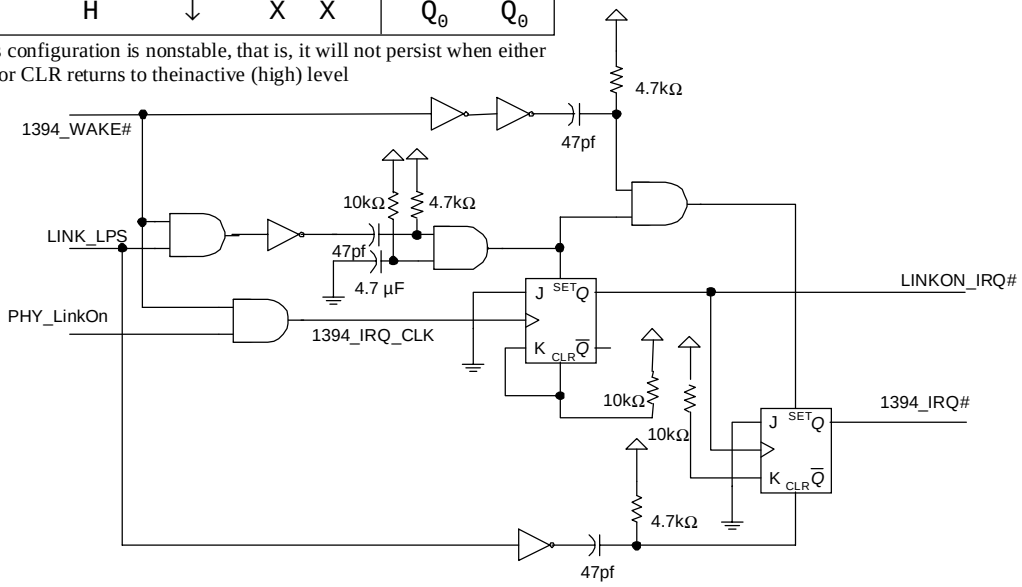

Figure 9-2 - Link On Signal Conditioning Circuit

The following timing diagram shows the relationship of the various signals.

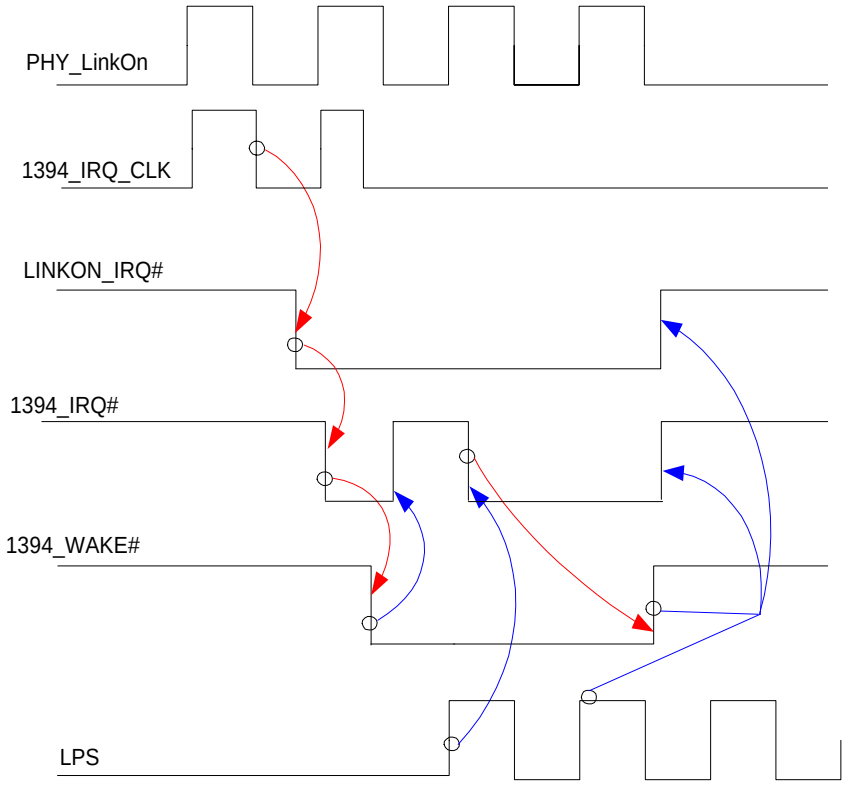

Figure 9-3 - Conditioning Circuit Signal Timing Relationships

Software/firmware shall be responsible for enabling the LPS signal from the Link device to the PHY device upon detection of the ***1394_WAKE#*** signal.