This Suspend/Resume proposal represents work in progress for the 1394 Power Management Task Group (operating under assignment from the 1394 Trade Association Architecture Working Group). By its very nature, these mechanism may/will have a direct impact on PHY silicon. Though this work has not yet been completed, content contained herein should provide the reader with detail of the suspend/resume mechanisms sufficient to understand suspend and resume on a port-to-port as well as a bus wide basis.

The Power Management Task Force offers this draft (revision 0.05) as the current definitive document for 1394 Suspend/Resume and request your feedback (comments/questions/suggestions) vial E-Mail addressed to: "list@p1394pm.org".

Steve Bard
p1394pm Acting Editor

6.7 Port Suspend

Port suspend mechanisms provide a facility for implementing a power conservation state while maintaining a port-to-port connection between a (possible) active bus segment and a (possible) inactive bus segment. While in this state, a port is unable to transmit or receive data transaction packets. However, a suspended port is capable of generating or receiving a resume event - restoring the port-to-port connection to normal, full-power, operation.

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

Suspend: a power state in which power consumption is as low as possible without turning power off - normal data processing is not possible;
Resume: a transition process from suspend to full power in which normal data processing is possible;
"Connected and active"; a port which is not in a suspend state and has a connection to a non-suspended port on another node;
Suspend initiator: an active port in a node which has been selected to initiate suspend notification to an active port in another node to which it is connected;
Suspend target: an active port in a node which receives suspend notification from a port in another node to which it is connected;
Resume initiator: a suspended port in a node which has been selected to initiate a resume event to a suspended port in another node to which it is connected and to any other ports in its node which are in a suspend state;
Resume target: a suspended port in a node which detects a resume event from a suspended port in another node to which it is connected;
"***Suspend Connection***:" a port-to-port connection between two nodes in which both ports are in a suspend state;
Suspend Domain: two or more nodes joined via port-to-port connections in which all ports may be in a suspended state;
Bus Manager: provides (among other functions) suspend mechanism management;
Boundary Node: a node with one or more connected and active ports and one or more suspend connections;
Set: logic 1, not clear, a logic high state;
Clear: logic 0, not set, reset, a logic low state;
Notify$_{hold}$: A period of time defined as 8,192 PHY clock - approximately 166.67 µs;
Bias$_{hold}$; A period of time defined as 8,192 PHY clock - approximately 166.67 µs;
Detect$_{min}$: A period of time defined as 8,192 PHY clocks - approximately 166.67 µs;
time$_{res\_notify}$: A period of time defined as 16,384 PHY clocks - approximately 333.33 µs;

6.7.1 Suspended PHY Port Behavior

A very low-level current (not to exceed 30 µamps) is supplied from TPA* to TPB* through a cable connection between two suspended ports on separate PHY's. The TPA* side of the port-to-port connection's current source is connected to the input of a voltage-sensing circuit. If the cable connection is removed, the voltage input to the sensing circuit will rise, generating a disconnect notification on its output.

An informative example follows (actual circuit implementation may vary):

Current is supplied through a cable connection to a destination port's TPB* through a 100 kilohm resistor connected between an *always on* power rail and the output of the disabled TpBias generator. The input to a schmidt trigger buffer/inverter is connected to the junction of the 100 kilohm resistor and the TpBIAS generator output. This mechanism will create a signal (active logic low) on the output of the buffer/inverter when the cable connection between the target and initiator is opened (see figure 6.11). Note: The output of the buffer/inverter is monitored only when a port is in a suspended state **-** e.g. TpBIAS generator (among other circuitry) has been disabled and its output is in a high impedance state.

While in the suspended state, a port is configured as follows (refer to section 6.7.16 for a detailed description of a port's ***Status*** and ***Control*** registers):

— Only circuitry associated with the port's voltage-sense and **Port_Status** comparator is active;
— The port's connection status (***Con***) bit is clear;
— The port's ***Disable*** bit is clear (i.e. port enabled);
— The port's ***Suspend*** bit is set (i.e. port is suspended);

6.7.2 Suspend Model and Process Overview

The suspend mechanism allows pairs of directly-connected ports to be placed into a low-power, suspended state. Any active port may be selected to initiate suspend state notification. Note: A port with its ***Suspend*** bit already set or its ***Disable*** bit set will ignore selection to be a suspend notification initiator. A port can be selected to become a suspend notification initiator by the Bus Manager or the link (associated with the PHY in which the port resides) via an extended PHY packet of type "PHY register write" (000001b). The port selected becomes the suspend initiator. The initiator notifies the port to which it is connected (the target) to enter into the suspend state.

A port is selected to become a suspend initiator when its ***Suspend*** bit becomes set (bit 1 of its ***Control*** register).

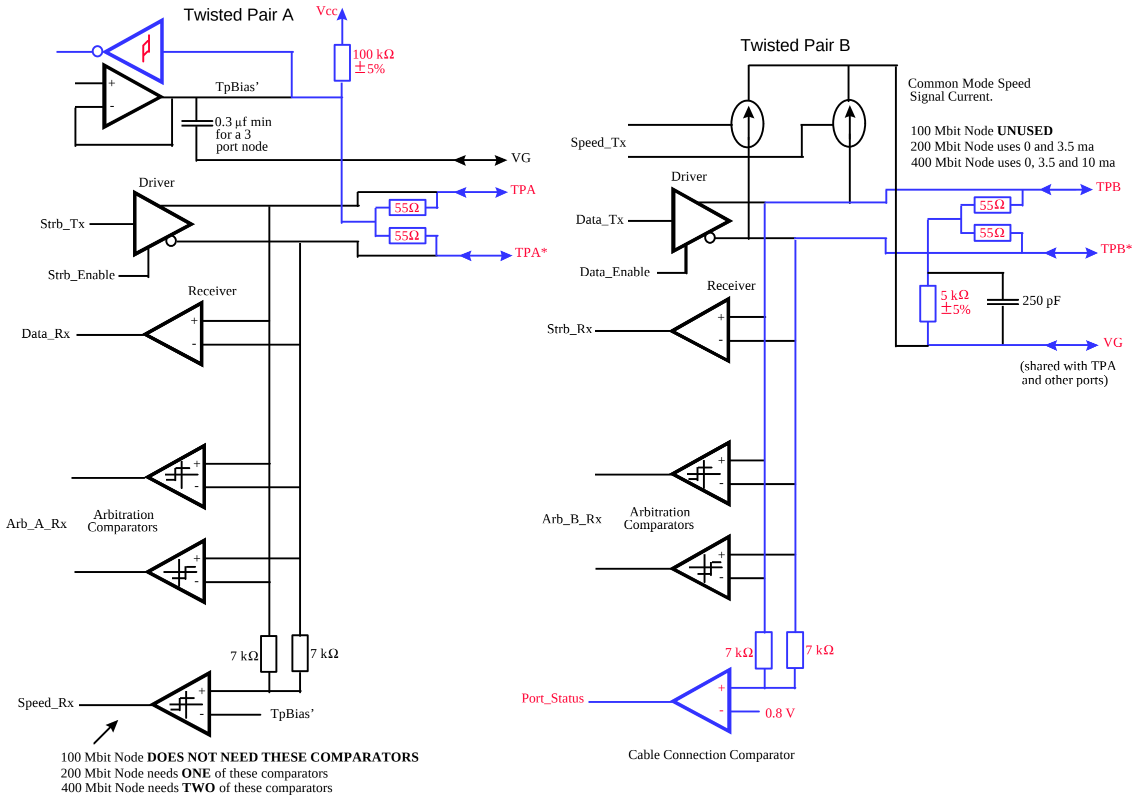
A suspend domain is created when three (or more) nodes are joined via a port-to-port connection in which each port is in a suspended state. Two nodes with a port-to-port connection in which both ports are in a suspend state constitutes a ***suspend connection*** and can be the attach point between an active (i.e. not suspended) domain and a suspend domain.

Suspended ports connected within a suspend domain can be restored to normal operation via a "resume" event (within the constraints outlined in proceeding sections). A resume event may be initiated by any port in a suspend state.

A connected port in a suspend state which detects a resume event becomes a resume target and will propagate the resume event to all other connected and suspended ports in its node (within the constraints outlined in proceeding sections).

6.7.3 Port Suspend Circuitry

Figure 6.11 shows the cable media signal interface configuration - which includes circuitry required to be active in a suspended-connected state.



100 Mbit Node **UNUSED**
200 Mbit Node uses 0 and 3.5 ma
400 Mbit Node uses 0, 3.5 and 10 ma

100 Mbit Node **DOES NOT NEED THESE COMPARATORS**
200 Mbit Node needs **ONE** of these comparators
400 Mbit Node needs **TWO** of these comparators

Figure 6-11 – Cable media signal interface configuration with Suspended-Connected Circuitry

TpBias is used by both the suspend initiator and suspend target to establish either a suspend connection or generate a resume event. Connect and disconnect state information is obtained via the output of voltage sense circuitry whose input is connected to the TpBias generator output. A leakage current source (also attached to the TpBias generator output) is delivered into a current sink on the TPB* signal line of a suspended port on another node through a cable connection. When one suspended port or the other is disconnected from the cable, a rise in voltage occurs on the voltage sensing circuit input - generating disconnect notification. The input to the voltage sense circuitry will remain in a low state while the current sink is attached through the cable connection. Connect notification results in a resume event and the port's ***Con*** bit is set - among other things (outlined in proceeding sections). Disconnect notification results in the port's ***Con*** bit being cleared - a resume event is **not** generated.

6.7.4 PHY Register Addressing

PHY registers are accessed using an extended PHY packet with the following structure:

0 0	phy_ID	0 0	type	port	reg	value
2	6	2	6	5	3	8

Figure 6-12 – PHY Control Packet

The fields in this packet are defined as follows:

phy_ID: 6-bit node address of the ports PHY;

type: 6-bit field identifying the type of PHY-register operation (000000b=read, 000001b=write);

port: 5-bit port-select value. Values 0-26_{10} inclusive select a specific port in the node; values 27_{10} through 30_{10} are reserved; value 31_{10} selects the 8 node registers;

reg: 3-bit register select field used to address a particular port (or node) ***Control*** or ***Status*** register. For port values of 0-26_{10} inclusive: register 000b (***Status***) and 001b (***Alternate Status***) are read-write for link access (as appropriate) but are read-only for an extended PHY packet command; a read from 010b (***Control***) or 011b (***Clear Control***) returns the current value of the port's ***Control*** register; writes to 010b (***Control***) set corresponding bits (as specified in the 'value' field) in the port's ***Control*** register; writes to 011b (***Clear Control***) clears the corresponding bits (as specified in the 'value' field) in the port's ***Control*** register. Extended PHY packet accesses to any node registers (i.e. port=31_{10}) will return the content of the node control register accessed (e.g. a read only operation of node registers 0 through 8_{10});

value: 8-bit field which will either return the contents of the selected register being read or contain a pattern of bit(s) to be set or cleared in the selected register during a write.

When a valid extended PHY packet of either type read or write is sent to a node, the node PHY returns a response packet formatted as shown in figure 6.13

0 0	phy_ID	0 0	111111	port	reg	current value
2	6	2	6	5	3	8

Figure 6-13 – PHY-Access Response Packet

The fields in this packet are defined as follows:

phy_id: node ID of node generating the response packet;

type: 111111b;

port: port register value (same as the port register value provided in the extended
PHY
packet for which the response is being generated);

reg: register value (same as the register value provided in the extended PHY packet for which the response is being generated);

value: the 8 bit value currently contained in the register addressed by the port and reg fields. If an invalid (e.g. reserved) register is addressed via the port and reg fields, the value returned will be zero for all bits.

6.7.5 Entry into Suspend

A port will initiate suspend notification when its port ***Control*** register is written to with bit 1 (***Suspend***) of the value field set from either a Bus Manager or the link associated with the node in which the port resides. Each method is described in the following sections.

Once a suspend connection state has been established between a suspend initiator and a suspend target, both the initiator and target nodes are required to have only their connect/disconnect voltage sense circuitry and **Port_Status** comparator active.

While in a suspend state, a port must not consume more than 1.65 milliwatts.

When a port detects its incoming TpBias decrease to below 0.4 volts, it will set its own ***Suspend*** bit and drive its outgoing TpBias to below 0.4 volts. After driving a low voltage TpBias for a time interval of $\textbf{\textit{Bias}}_{\textbf{\textit{hold}}}$, the port will configure its TpBias generator output to a high impedance state, thereby allowing the voltage-sense circuitry input connected to the output of the TpBias generator to detect a port connection. A connection will exist if the input to the voltage sense circuitry is held low - a result of a cable connection to a TPB* current sink path on a port of another node.

A port will clear its ***Con*** bit when its ***Suspend*** bit is set. The node in which the port resides that detected a loss of incoming TpBias will generate a short reset on all of its connected and active ports.

A port which has its ***Suspend*** bit set will not respond to or generate a bus reset or subsequent suspend notifications. A “*suspended*” port will not drive its TpBias.

6.7.5.1 Suspend Initiator selected via an Extended PHY Packet

An extended PHY packet of type ‘PHY Register Write” to a node’s port ***Control*** register with bit 1 (***Suspend***) of the data value field set will select that port as a suspend initiator. The port connected to the suspend initiator becomes the suspend target.

Immediately upon receiving the extended PHY packet, the suspend initiator responds with a PHY response packet which includes the value contained in the initiator’s ***Control*** register.

Upon completion of sending the PHY response packet, the initiator asserts **TX_DATA_END** to all of its connected and active ports followed by a short reset to all connected and active ports except the target, which is sent a **TX_SUSPEND_NOTIFY** (Arb_A_Tx = ‘Z,’ Arb_B_Tx = ‘1’).

The initiator holds **TX_SUSPEND_NOTIFY** to the target for a time of $\textbf{\textit{Notify}}_{\textbf{\textit{hold}}}$, waiting for the target to respond by driving its TpBias to the initiator to below 0.4 volts. Upon seeing the low voltage TpBias from the target, the initiator completes the suspend connection protocol process by setting its ***Suspend*** bit, clearing its ***Con*** bit, and driving TpBias below 0.4 volts to the target for a time of $\textbf{\textit{Bias}}_{\textbf{\textit{hold}}}$. After driving a low voltage TpBias to the target for the specified time, the initiator disables the output of its TpBias generator (generator output goes to a high impedance state). All non-essential circuitry for the initiator is placed into the lowest power state possible (only connect/disconnect voltage sense circuitry and the **Port_Status** comparator are required to remain active during when a port is in a suspend state).

If the initiator does not detect an incoming low voltage TpBias during $\textbf{\textit{Notify}}_{\textbf{\textit{hold}}}$ time of assertion of **TX_SUSPEND_NOTIFY** to the target, the initiator will, after placing the output of it’s TpBias generator in a high impedance state, sample the output of its voltage-sense circuitry. If the voltage-sense circuitry output is in a logic high state (indicating a port connection), the initiator sets its ***Disable*** bit, other wise the initiator remains in this state (i.e. ***Suspend*** bit set, ***Disable*** bit clear, ***Con*** bit clear) until a connection is detected at which time a resume event will be generated.

A port which has both its ***Disable*** and ***Suspend*** bits set indicates a condition in which suspend notification between an initiator and a target did not complete properly.

If a connected port with both its ***Suspend*** and ***Disable*** bits set (as the result of a suspend notification failure) should become disconnected, its ***Disable*** bit will clear (its ***Suspend*** bit will remain set) - thereby allowing a resume event to occur when a new connection is made.

6.7.5.2 Suspend Initiation from the Link

A link selects a port to become a suspend initiator by generating an extended PHY packet of type “PHY Register Write” to its own PHY. The PHY packet will have the node ID of the PHY associated with the link. The PHY packet will address the port ***Control*** register of the port to be selected as a suspend initiator with bit 1 (***Suspend***) in the data value field set.

The suspend initiator node will arbitrate the bus. When the node gains control of the bus, it generates an extended PHY packet of type “PHY register write” with its own node ID, port ID, and control register value with bit 1 of the data field set - as if it were going to set its own ***Suspend*** bit. This notifies all other nodes on the bus (at least those that are interested) of the event. The Bus Manager (if not the node that generated the PHY packet) understands that it did not send the PHY packet, therefore, it knows the only other condition which would result in such a packet is when the suspend initiator request came from a nodes own link. The Bus Manager performs the same internal processing (if any) as it would have if it had sent the PHY packet.

The suspend initiator asserts **TX_DATA_END** to all of its connected and active ports followed by a short reset to all connected and active ports except the target, which is sent a **TX_SUSPEND_NOTIFY** (Arb_A_Tx = ‘Z,’ Arb_B_Tx = ‘1’).

The suspend connection state process now continues as previously outlined.

6.7.5.3 Target Response to Suspend Request

When an initiator generates **TX_SUSPEND_NOTIFY** (Arb_A_Tx = ‘Z,’ Arb_B_Tx = ‘1’) to the target, it will be received as an **RX_IDENT_DONE**. Under normal operating circumstances, the target would receive an **RX_IDENT_DONE** as the result of a child PHY completing transmission of its self-ID. In this circumstance, an event has **not** occurred in which the target would be expecting to receive an **RX_IDENT_DONE** from a child PHY, therefore, the target interprets the *out-of-context* **RX_IDENT_DONE** as a suspend notification event from an initiator.

The target, upon seeing the *out of context* **RX_IDENT_DONE**, responds by setting the ***Suspend*** bit in all of the other connected and active ports (selecting them as suspend initiators) in its node. The target will drive its TpBias to the initiator to below 0.4 volts for a time of $\boldsymbol{Bias_{hold}}$ (waiting for the initiator to respond by driving its TpBias into the target to a voltage level below 0.4 volts). After driving a low voltage TpBias to the initiator for the $\boldsymbol{Bias_{hold}}$ time, the target disables the output of its TpBias generator (generator output goes to a high impedance state). All non-essential circuitry for the target is placed into the lowest power state possible (only connect/disconnect voltage sense circuitry and the **Port_Status** comparator are required to remain active during when a port is in a suspend state).

As soon as the target sees its incoming TpBias drop to below 0.4 volts, the target completes its portion of the suspend connection protocol by setting its ***Suspend*** bit and clearing its own ***Con*** bit.

If the target does not detect a low voltage TpBias from the initiator during the $\boldsymbol{Bias_{hold}}$ time of asserting of its low voltage TpBias to the initiator, the target will, if the voltage level on the input to the target’s voltage-sense circuitry produces a logic high output (indicating a port connection), the target sets it’s ***Disable*** bit, other wise, the target remains in this state (i.e. ***Suspend*** bit set,

Disable bit clear, ***Con*** bit clear) until a connection is detected at which time a resume event will be generated.

A port which has both its ***Disable*** and ***Suspend*** bits set indicates a condition in which suspend notification between an initiator and a target did not complete properly.

If a connected port with both its ***Suspend*** and ***Disable*** bits set (as the result of a suspend notification failure) should become disconnected, its ***Disable*** bit will clear (its ***Suspend*** bit will remain set) - thereby allowing a resume event to occur when a new connection is made.

A port can be pre-configured to not respond to suspend notification (thereby preventing it from propagating suspend notification into other connected and active ports in it's node). The method described is the recommended suspend propagation block mechanism:

Assume a bus topology as in Figure 6-14 (below):

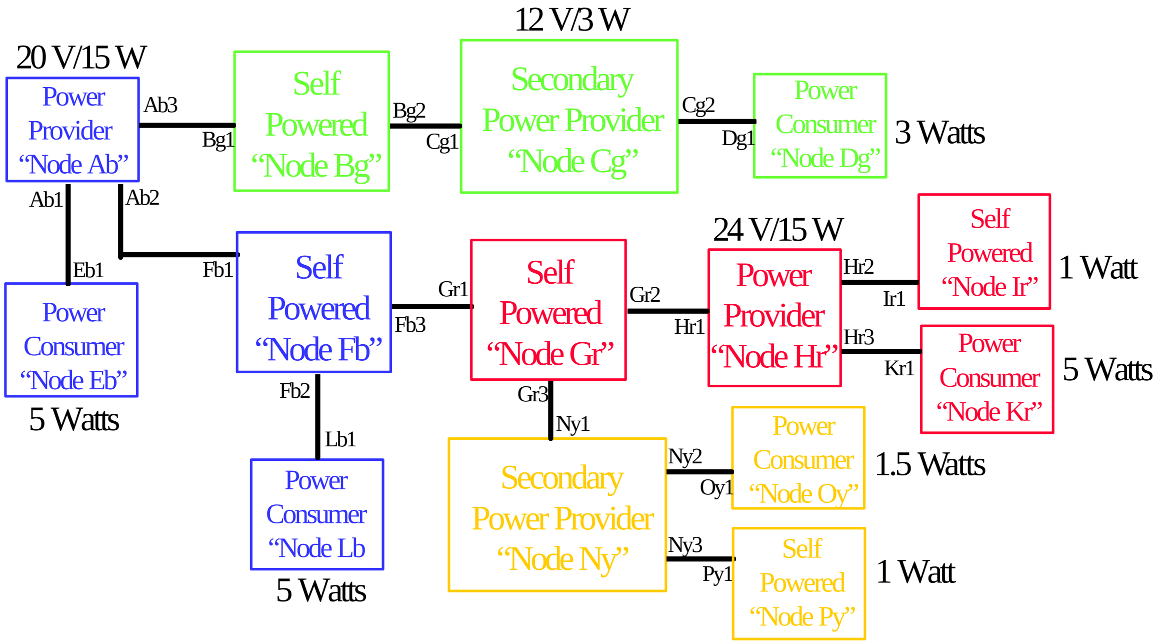


Figure 6-14 – Example Suspend/Resume Domain Topology

Further, assume node Bg is the Bus Manager and desires to create a suspend domain consisting of nodes Gr, Hr, Ir, and Kr. Nodes Ny, Oy and Py are not to be suspended as are nodes Lb, Fb, Eb, Ab, Bg, Cg, and Dg.

The Bus Manager begins by sending a PHY register write packet to node Gr, port Gr3, setting port Gr3's ***Disable*** bit. Port Gr3 immediately drops its TpBias to port Ny1, setting the ***Suspend*** bit in Ny1. Port Ny1 responds by dropping its TpBias to port Gr3, setting the ***Suspend*** bit in Gr3. Both nodes Gr and Ny assert a short reset to the other connected and active ports in their domains (Gr2, Gr1, and Ny3, Ny2 respectively). An active domain consisting of nodes Ny, Oy, and Py now exists. Port Ny1 of node Ny is in a suspend state.

The Bus Manager node (Bg) sends another PHY register write packet to Node Gr, port Gr3, clearing Gr3's ***Disable*** bit (port Gr3's ***Suspend*** bit remains set). This enables port Gr3 to recognize a resume event from port Ny1 if port Ny1 ever generates a resume event. If the Bus Manager never wants Gr3 to respond to a resume event from port Ny1, the Bus Manager never clears port Gr3's ***Disable*** bit.

The Bus Manager node (Bg) sends a PHY register write packet to node Fb, port Fb3, selecting it as a suspend initiator. The suspend propagates into and through node Gr into nodes Hr, Ir, and Kr.

An interesting observation: if the Bus Manager wanted to configure ALL remaining nodes to a suspend state (since domain ‘y’ is active and port Gr3 will now respond to a resume event from port Ny1) the Bus Manager could generate a PHY register write packet to node Kr, port Kr1, selecting it as the suspend initiator. The suspend event would propagate through all of the ‘r’ domain and on into the ‘b’ and ‘g’ domains.

6.7.6 Resume (Exit from a Suspended-Connected State)

A connected port in a suspend state will generate a resume event when the ***Suspend*** bit in its ***Port Status*** register is cleared or when a node receives an extended PHY packet of type ‘***RESUME***’ (000010b - see section 6.7.14).

The ***Suspend*** bit in a suspended port may be cleared via an extended PHY packet of type “PHY Register Write” from either the Bus Manager of the link associated with the PHY whose port is to be resumed. NOTE: A PHY write to a port ***Control*** register with bit 1 set in the data value field will set the port’s ***Suspend*** bit, while a PHY write to a port ***Clear Control*** register with bit 1 set in the data value field will clear the port’s ***Suspend*** bit.

A port which generates a resume event becomes a resume initiator. A resume event is generated when the resume initiator asserts TpBias to the suspend port to which it is connected.

A connected port in a suspend state which detects TpBias asserted on its TPB/TPB* pair, becomes a resume target.

A resume target responds to a resume initiator by asserting its TpBias to the resume initiator upon which it then clears its own ***Suspend*** bit and the ***Suspend*** bit of all other ports in its node which do not have their ***Disable*** bit set (selecting them as resume initiators).

If a resume initiator does not detect TpBias from its resume target within a time interval of $\textbf{\textit{Detect}}_{min}$ (measured from the time the resume initiator’s clocks become stable), the resume initiator will set its own port ***Disable*** bit and no longer drive TpBias to the resume target. The resume initiator will continue to have its ***Suspend*** bit clear.

A port which has its ***Disable*** bit set and its ***Suspend*** bit clear may be an indication that it is a port which was not successful in getting a proper response from a resume target or a port which became disabled as a direct result of a PHY register write (see section 6.7.15) to its ***Clear Control*** register with bit 2 in the value field set.

A port with its ***Disable*** bit set and its ***Suspend*** bit clear will not respond to any connect or disconnect event. In addition, it will not respond to or generate bus resets, suspend or resume events.

When a resume target exists in a node which has one or more connected and active ports, the PHY in which the resume target resides will not propagate any activity from the active bus segment into the resume target. The PHY does, however, assert **TX_REQUEST** from all of its resume targets to all of their resume initiators. **TX_REQUEST** is asserted within one debounce interval of the resume target asserting TpBias to its resume initiator. The PHY will wait two debounce intervals (measured from the time the resume target detected TpBias from its resume initiator) and then begin arbitration for control of the active bus (for the purpose of generating an arbitrated short reset to all of its connected and active ports as well as to all of its resume targets). If the PHY detects a short reset on any of its resume targets before the expiration of the 2 debounce delay, it will generate a long reset on all of its connected and active ports as well as all of its resume targets.

A resume initiator which detects a **TX_REQUEST** from its resume target will propagate the **TX_REQUEST** to all other ports in its node which may be in the process of resuming.

A resume initiator will wait five debounce intervals (measured from the time the resume target cleared its ***Suspend*** bit in its own ***Control*** register) after which it will generate a long reset - providing it has not seen detected a reset its resume target during the delay period..

A five debounce delay interval provides an opportunity for a resume target (which may be connected to a boundary node as far 16 hops away from the original resume initiator) to request an opportunity to arbitrate for control of an active bus (for the purpose of generating an arbitrated short reset) to which it may have an active port connection.

The farthest resume target node from a resume initiator (16 hops) will have TpBias restored to all of its suspended ports in approximately 170 milliseconds (assuming resume event propagation takes as long as 10 milliseconds per node and a single debounce interval of approximately 85 milliseconds, the farthest resume target will see TpBias restored in about two debounce intervals).

The farthest resume target node from a resume initiator will (as previously described), assuming it has one or more active port connections to an active bus, begin arbitrating for control of the active bus for the purpose of generating an arbitrated short reset. Arbitration may take as long as two debounce intervals, therefore, the farthest resume target node may not be able to generate a short reset until the resume initiator has entered into its fifth debounce interval. If a bus reset does not occur after five debounce intervals, the resume initiator will assume there is no connection to an active bus segment, therefore, the resume initiator will asserts a long reset.

When multiple resume target nodes are asserting **TX_REQUEST**, the node which first wins bus arbitration with its active bus segment will generate a short reset to its resume initiator.

A resume initiator will always propagate a bus reset into any other port in its node which is in the process of resuming.

When a bus reset is detected by another resume target node which also had asserted **TX_REQUEST**, that node will generate a long reset on both the active segment to which it is attached and its resume initiator.

6.7.6.1 Resume Propagation

A resume event will not propagate through an active port (e.g. a resume event in one suspend domain will not propagate into another suspend domain through an active boundary node).

A port which has its ***Disable*** bit set will not respond to or generate bus resets, suspend notification or resume events. A "*disabled*" port will not drive its TpBias.

A connected port in a suspend state will generate a resume event when the ***Suspend*** bit in its ***Port Status*** register is cleared or when a node receives an extended PHY packet of tpe '***RESUME***' (000010b - see section 6.7.14).

The ***Suspend*** bit in a suspended port may be cleared via an extended PHY packet of type "PHY Register Write" from either the Bus Manager of the link associated with the PHY whose port is to be resumed. NOTE: A PHY write to a port ***Control*** register with bit 1 set in the data value field will set the port's ***Suspend*** bit, while a PHY write to a port ***Clear Control*** register with bit 1 set in the data value field will clear the port's ***Suspend*** bit.

A port which generates a resume event becomes a resume initiator. A resume event is generated when the resume initiator asserts TpBias to the suspend port to which it is connected.

A connected port in a suspend state which detects TpBias asserted on its TPB/TPB* pair, becomes a resume target.

A resume target responds to a resume initiator by asserting its TpBias to the resume initiator upon which it then clears its own ***Suspend*** bit and the ***Suspend*** bit of all other ports in its node which do not have their ***Disable*** bit set (selecting them as resume initiators).

If a resume initiator does not detect TpBias from its resume target within a time interval of ***Detect***$_{min}$ (measured from the time the resume initiator's clocks become stable), the resume initiator will set its own port ***Disable*** bit and no longer drive TpBias to the resume target. The resume initiator will continue to have its ***Suspend*** bit clear. The resume event will not propagate through this disabled port.

A port which has its ***Disable*** bit set and its ***Suspend*** bit clear may be an indication that it is a port which was not successful in getting a proper response from a resume target or a port which became disabled as a direct result of a PHY register write (see section 6.7.15) to its ***Clear Control*** register with bit 2 in the value field set.

A port with its ***Disable*** bit set and its ***Suspend*** bit clear will not respond to any connect or disconnect event. In addition, it will not respond to or generate bus resets, suspend or resume events.

A resume initiator which detects a **TX_REQUEST** from its resume target will propagate the **TX_REQUEST** to all other ports in its node which may be in the process of resuming.

A resume initiator will always propagate a bus reset into any other port in its node which is in the process of resuming.

6.7.7 Detach Detection During Suspend

A disconnect between two connected ports that are in a suspend state is detected when the TPA* current source current sink through the cable connection is lost as a result of the cable being disconnected. When the cable connection is removed from one port or the other, a rise in voltage on the input to a port's voltage sense circuitry occurs. The voltage sensing circuit generates connect and disconnect notification. Connect notification results in a resume event on both ports and both port's ***Con*** bits are set. Disconnect notification does not result in a change to the port's ***Con*** bit (already cleared as a result of the port entering into a suspend state), in addition, a resume event is **not** generated as a result of a disconnect while a port is in the suspend state and the port's ***Suspend*** bit remains set.

6.7.8 Connect Detection During Suspend

A node will power up with all of its ports in a suspend state. When a port powers up it may or may not have a connection to a port on another node. This section describes the behavior for both instances.

6.7.8.1 New Connect Detection During Suspend

A node port with no connection, when it powers up, will have all ports configured as shown in figure 6-15:

				Bit name & State				
Register	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
000b	Astat		Bstat		Ch	Con	Bias	rsrvd

(Status)	X	X	X	X	X	0	X	
010b	Chg_int_in	Suspend	Disable	Enab_token	rsrvd			
(Control)	X	1	0					

Figure 6-15 – Port Power-up Configuration

When a port has a new connection made to it, the output of its voltage sense circuit (see figure 6-11) will transition from a low state to a high state. When the port sees this transition, it will sample its TPB/TPB* pair for the presence of a TpBias.

If the newly connected port detects an incoming TpBias on its TPB/TPB* pair, its ***Con*** bit will set. The port will assert TpBias on it's TPA/TPA* pair, clear its ***Suspend*** bit and generate a resume event to all other ports in its node which are suspended but not disabled by clearing their ***Suspend*** bits (selecting them as resume initiators). Resume events are propagated through the node's resume initiator's as previously outlined.

If an incoming TpBias is not detected on its TPB/TPB* pair, the newly connected port will assert TpBias on its TPA/TPA* pair to its connected port for a time period of ***Bias***$_{hold}$ after which it will sample for TpBias on its TPB/TPB* pair. If it does not detect TpBias, it will set its ***Disable*** bit and no longer drive TpBias. Its ***Con*** bit will still be clear. If, during the ***Bias***$_{hold}$ time, TpBias is detected on its TPB/TPB* pair, its ***Con*** bit will set, its ***Suspend*** bit will clear and it will generate a resume event to all other ports in its node which are suspended but not disabled by clearing their ***Suspend*** bits (selecting them as resume initiators). Resume events are propagated through the node's resume initiator's as previously outlined.

6.7.8.2 Existing Connect Detection During Suspend

When a port in a node (with a connection to a port on another node) powers up, the output of its voltage sense circuit will be high (indicating a connection condition).

The port will sample its TPB/TPB* pair for the presence of a TpBias. If it detects an incoming TpBias on its TPB/TPB* pair, its ***Con*** bit will set. The port will assert TpBias on it's TPA/TPA* pair, clear its ***Suspend*** bit and generate a resume event to all other ports in its node which are suspended but not disabled by clearing their ***Suspend*** bits (selecting them as resume initiators). Resume events are propagated through the node's resume initiator's as previously outlined.

If an incoming TpBias is not detected on its TPB/TPB* pair, the newly connected port will assert TpBias on its TPA/TPA* pair to its connected port for a time period of ***Bias***$_{hold}$, after which it will sample for TpBias on its TPB/TPB* pair. If it does not detect TpBias, it will set its ***Disable*** bit and no longer drive TpBias. Its ***Con*** bit will still be clear. If, during the ***Bias***$_{hold}$ time, TpBias is detected on its TPB/TPB* pair, its ***Con*** bit will set, its ***Suspend*** bit will clear and it will generate a resume event to all other ports in its node which are suspended but not disabled by clearing their ***Suspend*** bits (selecting them as resume initiators). Resume events are propagated through the node's resume initiator's as previously outlined.

6.7.9 Port Power Loss During Suspend

When power is lost, all connected ports continue to provide a current sink to their connected ports that are in a suspend state and no event notification will be provided to the suspended port connection.

A port which had an active connection to a port on another node will no longer assert TpBias to its connected port. If the connected port is a 1995 PHY it will see the loss of TpBias as a disconnect event and respond accordingly.

If the connected port is an "a" PHY, the connected port will see a drop in its incoming TpBias and will respond as outlined in section 6.7.5.

6.7.10 Reset Propagation

A reset will not propagate into and/or through a port with either its ***Suspend*** bit or ***Disable*** bit set.

6.7.11 PHY-Core Suspend/Resume Control

A PHY core will not enter into a suspend state as long as LPS is asserted by the link.

If a port's ***Chg_int_en*** bit is set, an interrupt event to the link (LinkOn) will assert each time a port's ***Control*** port state changes.
In this manner, a link can be notified each time a PHY port enters into a suspend or disabled state.

When all ports on a PHY are inactive (i.e. not connected, suspended, or disabled) a link can, at its option, remove LPS from the PHY.

When all ports on a PHY are inactive (i.e. not connected, suspended, or disabled) and LPS is removed by the link, the PHY will enter into a suspend state.

A suspended PHY maintains an ability to generate a LinkOn when one of its port's ***Control*** register bits changes state (assuming the port's ***Chg_int_en*** bit is set).

However, if a PHY's ***Res_Disable*** bit is set, the PHY will not generate a LinkOn to the link - regardless of the state of any of the PHY's port's ***Control*** register ***Chg_int_en*** bit.

The link can set the PHY ***Res_Disable*** bit through the PHY-link interface, however, ***Res_Disable*** cannot be set or cleared via a bus extended PHY register write packet.

6.7.11.1 PHY-Core Control from the Link

A link can instruct a PHY to enter a suspend state by setting the ***Suspend*** bit or ***Disable*** bit in all of the ports in the PHY (selecting each of them to be suspend initiators or forcing each of them to disable their TpBias generators - respectively). When the last port has entered into a suspend state, the link may then remove LPS from the PHY and the PHY will enter into a suspend state.

The software stack controlling the link must be cognizant of the timing constraints involved (i.e. LPS should not be removed before all ports in the PHY have completed entering a suspend state or have been successfully disabled).

One note of interest: if the link uses the ***Disable*** bit, once all ports have been disabled, the link could then (after some _TBD___ delay) make certain all ***Suspend*** bits in all ports of the PHY are set, after which the link clears all ***Disable*** bits in all ports of the PHY, followed by clearing the PHY ***Resume_Enable*** bit (bit 0 of PHY node register 0101b), and then remove LPS to the PHY. This would precondition all ports in the suspended PHY to respond to a resume event - resuming the PHY and asserting LinkON to the link (notifying the link of a resume event on the PHY).

6.7.12 Resume/Resume Collisions

When a port in a node is in the process of participating as a resume target as a result of detecting a resume event from a connected resume initiator and a suspended port in the same node as the resume target detects resume event from a second connected resume initiator (as opposed to a resume event generated from the current resume target in its own node), the second suspended port in the node will begin a resume target response to its resume initiator **and** assert **TX_REQUEST** on both resume initiators. When 4 debounce intervals have elapsed since the detection of the second resume event, the node will generate a reset (providing no other node has generated a reset). If node in which the resume targets reside receives a reset on the from the first resume initiator before the second resume event has been active for 2 debounce intervals, the reset is not propagated to the second resume initiator. If a reset is received from the second resume initiator before a reset is seen from the first resume initiator, the reset is propagated to the first resume initiator.

6.7.13 Suspend/Resume Collisions

If a resume event is detected by a resume target in a node containing a port recently selected as a suspend initiator or suspend target, the resume event will not be propagated into the resume target for a delay of ***time$_{res\_notify}$***. If the newly selected suspend initiator (or target) in the node happens to be the last port in the node to suspend, the nodes clocks will not enter into a suspend state for 1mS after detecting all ports in the node have their ***Suspend*** bit set.

6.7.14 Extended PHY Packet of type 'Resume'

A Bus Manager may send an extended PHY packet of type "***Resume***."

An extended PHY packet of type ***Resume*** takes the following format:

			type	port	reg	value
0 0	phy_ID	0 0	0 0 0 0 1 0	0 0 0 0 0	0 1 1	0 0 0 0 0 0 1 0
2	6	2	6	5	3	8

Figure 6-16 – Extended PHY Packet of type "*Resume*"

The fields in this packet are defined as follows:

phy_id: node ID of node generating packet;
type: 000010b (***Resume***);
port: port register value = 00000b;
reg: port ***Clear Control*** register value = 011b;
value: 00000010b (***Suspend*** bit position set);

A node which receives this PHY packet will clear the ***Suspend*** bit of all its connected ports that are in a suspend state (selecting them as resume initiators).

Each port selected as a resume initiator will generate a resume event to its connected port.

The node which receives the ***Resume*** PHY packet will wait five debounce times before asserting a reset on its resume initiators and connected and active ports (following the constraints outlined in section 6.7.6).

6.7.15 Port Disable

A port will be disabled when its port ***Control*** register is written to with bit 2 (***Disable***) of the value field set from an extended PHY packet of type 'PHY register write' from a Bus Manager or from the link associated with the node in which the port resides. Each method is described in the following paragraphs.

When a port has its ***Disable*** bit set as the result of an extended PHY register write packet from a Bus Manager, the port's node immediately responds with a PHY response packet which includes the value contained in the ***Control*** register of the port to be disabled.

Upon completion of sending the PHY response packet, the node in which the port to be disabled resides asserts **TX_DATA_END** to all of its connected and active ports followed by a short reset to all connected and active ports except the port to which the port to be disabled is connected, to which it asserts **TX_PRT_DISABLED** (Arb_A_Tx = 0, Arb_B_Tx = 0) and ceases to drive TpBias (i.e. its TpBias generator is turned off - output goes to a high impedance). The port being disabled asserts **TX_PRT_DISABLED** for a time period of ***Notify***$_{hold}$. The port on the far end (connected to the newly disabled port) receives **TX_PRT_DISABLED** as **RX_ROOT_CONTENTION**. When the port on the far end detects a loss of TpBias from the disabled port to which it is connected, it will responds as previously outlined.

When a port is selected by its associated link to be disabled, the node in which the port to be disabled resides will arbitrate the bus. When the node gains control of the bus, it generates an extended PHY packet of type "PHY register write" with its own node ID, port ID, and control register value with bit 2 of the data field set - as if it were going to set its own Disable bit. This notifies all other nodes on the bus (at least those that are interested) of the event. The Bus Manager (if not the node that generated the PHY packet) understands that it did not send the PHY packet, therefore, it knows the only other condition which would result in such a packet is when the port disable command came from a nodes own link. The Bus Manager performs the same internal processing as it would have if it had sent the PHY packet.

At the conclusion of the PHY packet, the node in which the port to be disabled resides asserts **TX_DATA_END** to all of its connected and active ports followed by a short reset to all connected and active ports except the port to which the port to be disabled is connected, to which it asserts **TX_PRT_DISABLED** (Arb_A_Tx = 0, Arb_B_Tx = 0) and ceases to drive TpBias (i.e. its TpBias generator is turned off - output goes to a high impedance). The port being disabled asserts **TX_PRT_DISABLED** for a time interval of ***Notify***$_{hold}$. The port on the far end (connected to the newly disabled port) receives **TX_PRT_DISABLED** as **RX_ROOT_CONTENTION**. When the port on the far end detects a loss of TpBias from the disabled port to which it is connected, it will responds as previously outlined.

If the node containing the port to be disabled is not able to gain control of the bus (via arbitration) for three debounce intervals, the node will assert a long reset to all connected and active ports in its node and will asserts **TX_PRT_DISABLED** (Arb_A_Tx = 0, Arb_B_Tx = 0) and ceases to drive TpBias (i.e. its TpBias generator is turned off - output goes to a high impedance). The port being disabled asserts **TX_IM_DISABLED** for a time interval of ***Notify***$_{hold}$. The port on the far end (connected to the newly disabled port) receives **TX_IM_DISABLED** as **RX_ROOT_CONTENTION**. When the port on the far end detects a loss of TpBias from the disabled port to which it is connected, it will responds as previously outlined.

A port which has been disabled as the result of a PHY packet (either from its associated link or from a Bus Manager) cannot have its ***Disable*** bit cleared as the result of a disconnect or a connect event. The ***Disable*** bit can only be cleared by a PHY packet command (from either its associated link or from a Bus Manager via a connected and active port in the node which the disabled port resides).

A port which has been disabled as the result of a failed suspend/resume negotiation may have its ***Disabled*** bit cleared as the result of a disconnect event.

A disabled port with a connection to another port will have the following ***Control*** and ***Status*** register configuration:

				Bit name & State				
Register	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
000b	Astat		Bstat		Ch	Con	Bias	rsrvd
(Status)	X	X	X	X	X	0	X	
010b	Chg_int_in	Suspend	Disable	Enab_token	rsrvd			
(Control)	X	Y	1					

Figure 6-17 – Configuration for a connected and disabled port

Note: the ***Con*** bit will be set regardless of the state of the disabled port's incoming TpBias. If the port to which the disabled port is connected is a 1995 PHY, the disabled port's ***Suspend*** bit will be clear. If the port to which the disabled port is connected is an “a” PHY, the disabled port’s ***Suspend*** bit will be set.

If the port became disabled as the result of the receipt of a valid PHY packet, the disabled status of a port will not change when a connection to the disabled port is removed. If, however, the port became disabled as the result of a failed suspend/resume negotiation, and the connection to a disabled port is removed, the disabled port’s ***Control*** and ***Status*** register configuration changes to:

				Bit name & State				
Register	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
000b	Astat		Bstat		Ch	Con	Bias	rsrvd
(Status)	X	X	X	X	X	0	X	
010b	Chg_int_in	Suspend	Disable	Enab_token	rsrvd			
(Control)	X	1	0					

Figure 6-18 – Configuration for a recently disconnected port that was disabled

The port is no longer disabled. Bus notification is not required because the PHY has previously reported in its self-ID that this port was disconnected (a port’s ***Disabled*** bit (when set) forces its ***Con*** bit to a cleared state).

The port will now generate a resume event when a new connection is made.

When a new connection is now made to the (no longer disabled) port, the port’s ***Control*** and ***Status*** register configuration changes to:

				Bit name & State				
Register	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
000b	Astat		Bstat		Ch	Con	Bias	rsrvd
(Status)	X	X	X	X	X	1	X	
010b	Chg_int_in	Suspend	Disable	Enab_token	rsrvd			
(Control)	X	0	0					

Figure 6-19 – Configuration for a recently connected port

The port’s ***Con*** bit becomes set and its ***Suspend*** bit clears - selecting the port as a resume initiator. The resume event will propagate to any suspended port in the node in which the port that has just had a new connection made to it resides. All other connected and active ports will **not** see, an otherwise normal, connect event bus reset. The ports connected to an active bus will eventually see a reset as a result of the resume propagation process.

If there are no other suspended ports in the node in which the port resides that has just had a new connection attached, a normal connect event bus reset will be asserted on all connected and active ports.

6.7.16 Port Register Definitions

Figure 6-20 is a table which identifies the bits located in a ports registers.

				Contents				
Address	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1000b	Astat		BStat		Ch	Con	Bias	rsrvd
1001b	Negotiated_speed			rsrvd				
1010b	Chg_int_en	Suspend	Disable	Enab_token	rsrvd			
1011b	rsrvd							
1100b	rsrvd							

1101b								
1110b								
1111b								

Figure 6-20 – Port *Control*, *Clear Control*, *Status*, and *Alternate Status* Register map

The behavior associated with a port's ***Suspend*** and ***Disable*** bits has been described exhaustively in preceding sections.

The ***Chg_int_en*** (*change interrupt enable)* bit when set results in the PHY generating a LinkOn each time one of the following bits in change state: ***Con***, ***Suspend***, or ***Disable***.

If the ***Chg_int_en*** bit is clear, the PHY will not generate a LinkOn as a result of a state change of the previously stated bits.

Bits indicated as ***rsrvd*** must not be used or interpreted as having a meaningful value.