The senior border needs to make sure that the B cloud concatenations don't go on forever. If a link is present on the border, the PHY can know when a cycle start is expected and can grab control back with a asynchronous request of CYCLE_START priority. If no link is present, the border will constantly use CYCLE_START to return control back to the Legacy cloud after each and every asynchronous packet transmission.

## 13.7.4 Bus reset

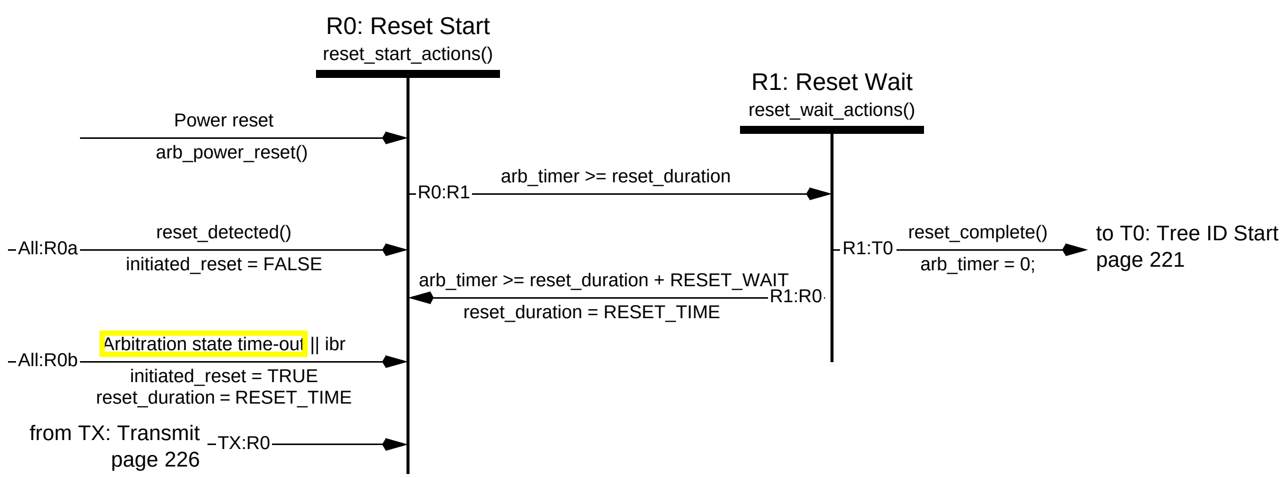


**Figure 13-8—Bus reset state machine**

NOTE—all C-code function names listed here are hypertext links in electronic versions of this specification

### 13.7.4.1 Bus reset state machine notes

On power reset, PHY register values and internal variables are set as specified in this section; in particular all ports are marked disconnected. A solitary node transitions through the reset, tree identify and self-identify states and enters A0: Idle as the root node.

**Transition All:R0a.** This is the entry point to the bus reset process if the PHY senses BUS_RESET on any active or resuming port. This transition shall be made in preference to any other transition that might be simultaneously eligible. The initiation of a bus reset cannot occur until a state's actions have been completed.

**Transition All:R0b.** This is the entry point to the bus reset process if this node is initiating the process. This happens under the following conditions:

a) Serial Bus management makes a PH_CONTROL.request that specifies a long reset;
b) The PHY detects a disconnect on its parent port; or
c) The PHY stays in any state (except A0:Idle during the time that `idle_arb_state_timeout` is FALSE , T0:Tree-ID Start, or a state that has an explicit time-out) for longer than MAX_ARB_STATE_TIME. This condition is not explicitly represented in either the state diagrams or the C code. When a local request is pending (from either the link or the PHY), then `idle_arb_state_timeout` is set TRUE . This transition is taken ifþthe PHY stays in A0:Idle for MAX_ARB_STATE_TIME after `idle_arb_state_timeout` is set TRUE .

With the exception of the last condition, the initiation of a bus reset cannot occur until a state's actions have been completed.

**Transition TX:R0.** This is the entry point to the bus reset process if this node is initiating an arbitrated (short) reset. If arbitration succeeded and the isbr_OK variable is set, there is no packet to transmit and the shot bus reset commences immediately.

 

**State R0:Reset Start.** The node sends a BUS_RESET signal whose length is governed by reset_time. In the case of a standard bus reset, this is long enough for all other bus activity to settle down (RESET_TIME is longer than the worst case packet transmission plus the worst case bus turn-around time). SHORT_RESET_TIME for an arbitrated (short) bus reset is significantly shorter since the bus is already in a known state following arbitration.

**Transition R0:R1.** The node has been sending a BUS_RESET signal long enough for all its connected neighbors to detect it.

**State R1:Reset Wait.** The node sends out IDLEs, waiting for all its active ports to receive IDLE or PARENT_NOTIFY (either condition indicates that the connected PHYs have left their R0 state).

**Transition R1:R0.** The node has been waiting for its ports to go idle for too long (this can be a transient condition caused by multiple nodes being reset at the same time); return to the R0 state again. This time-out period is a bit longer than the R0:R1 time-out to avoid a theoretically possible oscillation between two nodes in states R0 and R1.

**Transition R1:T0.** All the connected ports are receiving IDLE or PARENT_NOTIFY (indicating that the connected PHYs are in reset wait or starting the tree ID process).

## 13.7.5 Tree ID

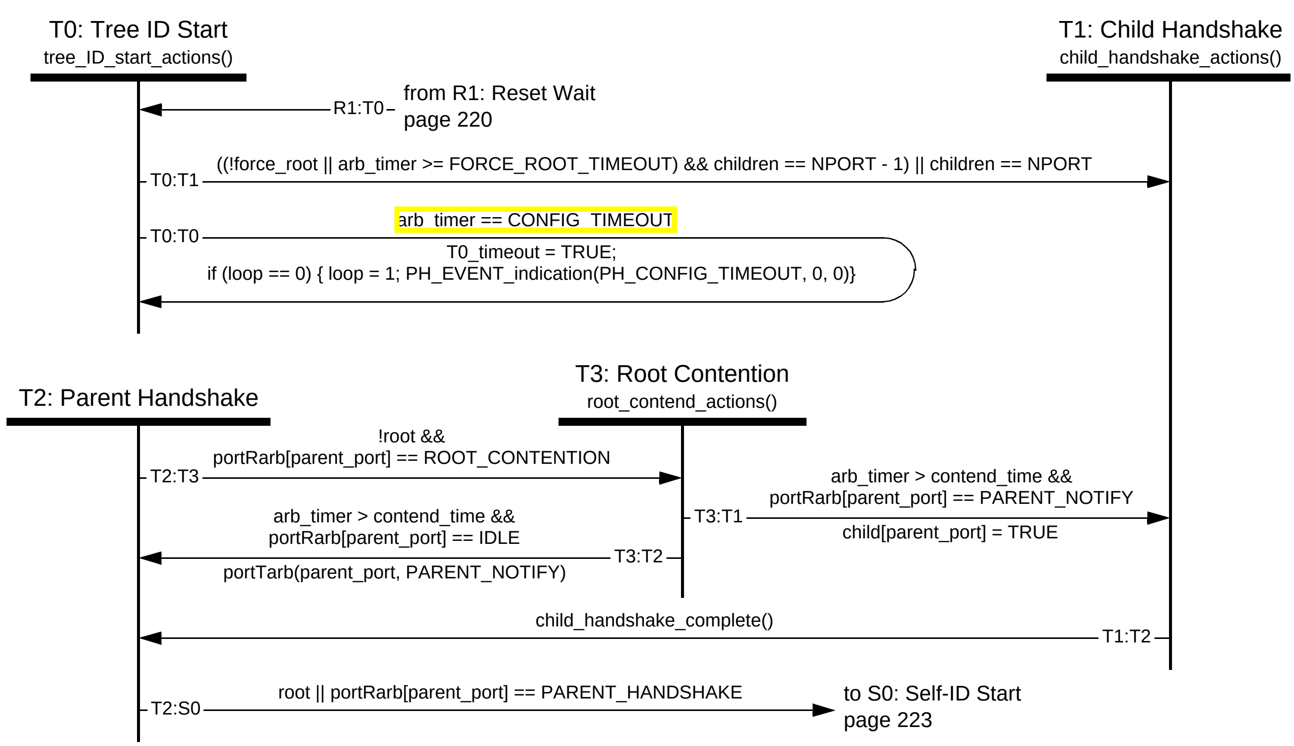


**Figure 13-9—Tree-ID state machine**

NOTE—all C-code function names listed here are hypertext links in electronic versions of this specification

### 13.7.5.1 Tree ID state machine notes

**State T0: Tree-ID Start.** In this state, a node waits up to a CONFIG_TIMEOUT period to receive the PARENT_NOTIFY signal from at least all but one of its active ports. When PARENT_NOTIFY is observed, that port is marked as a child port.

**Transition T0:T0.** If there's a loop on the bus consisting entirely of DS connections, then a configuration timeout occurs, setting the T0_timeout flag. All active ports operating in Beta mode are forced back into the "untested" state. This may indeed directly result in the bus initialization completing, or may allow the loop free build process to set appropriate Beta mode ports into the loop disabled state, allowing a fresh bus reset to complete.

**Transition T0:T1.** If a node detects the PARENT_NOTIFY signal on all of its ports, or all but one of its ports, it knows it is either the root or a branch; it can start the handshake process with its children. Leaf nodes (those with only one connected port) or root nodes (PARENT_NOTIFY on all ports) take this transition immediately. If the force_root flag is set, the test for the “all but one port” condition is delayed long enough that all other nodes on the bus will have transitioned at least to state T1 so all the ports should then be receiving the PARENT_NOTIFY signal (this extra delay is the FORCE_ROOT_TIMEOUT value).

**State T1: Child Handshake.** All ports that have been labled as child ports transmit the CHILD_NOTIFY signal. This allows the nodes attached to this node’s child port(s) to transition from T2 to S0. Leaf nodes have no children, so they exit this state immediately via transition T1:T2. If all ports are labeled as child ports, then the node knows it is the root.

**Transition T1:T2.** When all of a node’s children stop sending PARENT_NOTIFY, it then waits until it sees the CHILD_HANDSHAKE signal on all of its child ports. It then knows they have all transitioned to the self-ID start state, so the node can now handshake with its own parent.

**State T2: Parent Handshake.** At this point, a node is waiting to receive PARENT_HANDSHAKE signal (for DS connections, this is the result of the node sending PARENT_NOTIFY and its parent sending CHILD_NOTIFY). This step is bypassed if the node is root (receiving PARENT_NOTIFY on all connected ports). Another way this state can be exited is if it receives the ROOT_CONTENTION signal from its parent.

**Transition T2:S0.** When the node receives the PARENT_HANDSHAKE signal, it starts the self-ID process by sending the IDLE signal (see State S0: Self-ID Start, below). It also takes this transition if it is root, since it doesn’t have a parent.

**Transition T2:T3.** If a node receives a PARENT_NOTIFY signal on the same port that it is sending a PARENT_NOTIFY signal, the merged signal is interpreted as ROOT_CONTENTION. This can only happen for a single pair of nodes if each bids to make the other node its parent.

**State T3: Root Contention.** At this point, both nodes back off by sending an IDLE signal, starting a timer, and picking a random bit. If the random bit is one, the node waits longer than if it is a zero. When the timer has expired, the node samples the contention port once again.

**Transition T3:T2.** If a node receives an IDLE signal on its proposed parent port at the end of the delay, it once again sends the PARENT_NOTIFY signal. If the other node is taking longer it takes the T3:T1 transition and allows this node to exit state T2 via the self-ID start path. Otherwise the two nodes again see the ROOT_CONTENTION signal and repeat the root contention process with a new set of random bits..

**Transition T3:T1.** If a node receives a PARENT_NOTIFY signal on the proposed parent port at the end of the delay it means the other node has already transitioned to state T2, so this node returns to state T1 and becomes the root.

### 13.7.6 Self_ID

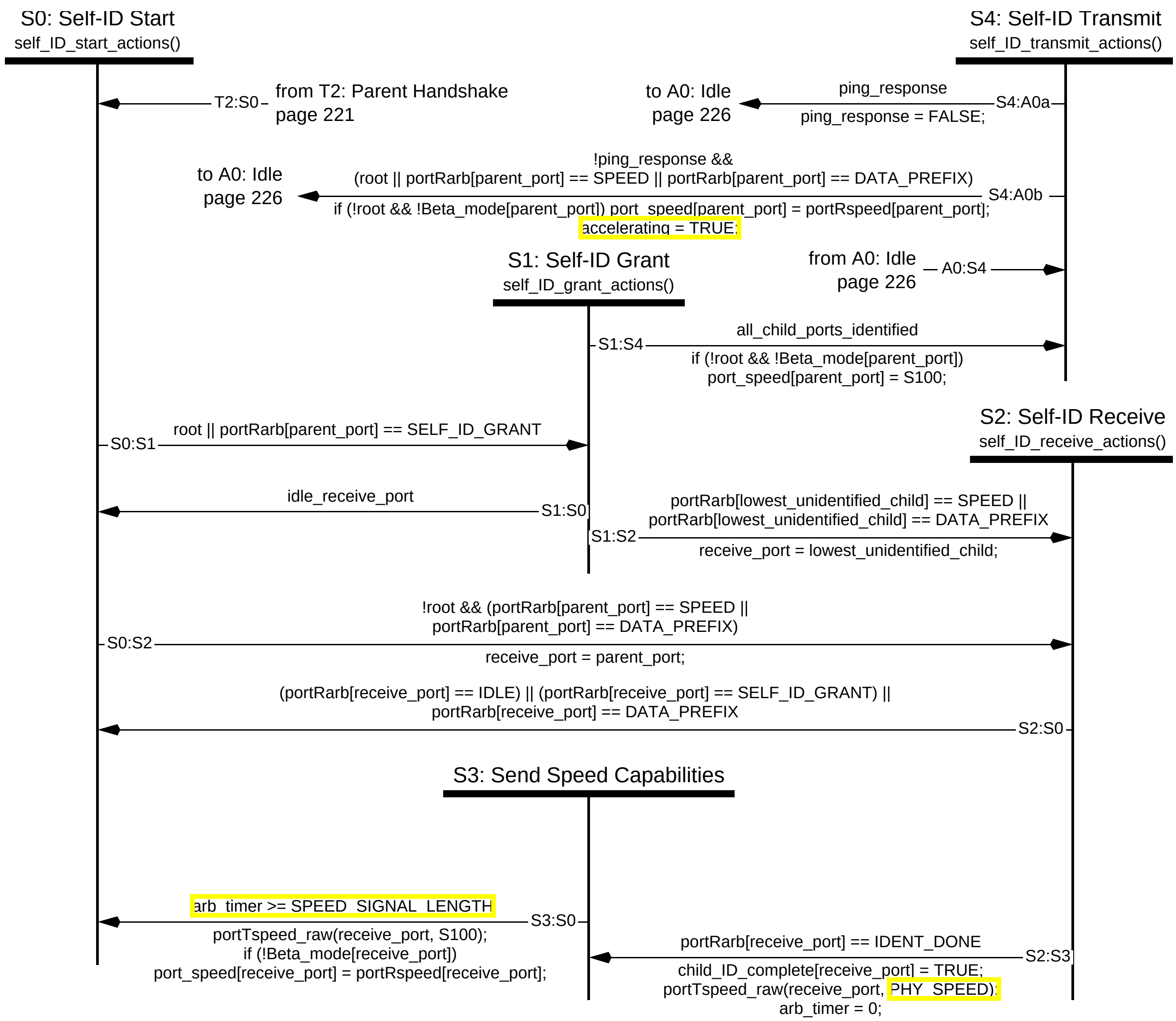


**Figure 13-10—Self-ID state machine**

NOTE—all C-code function names listed here are hypertext links in electronic versions of this specification

#### 13.7.6.1 Self ID state machine notes

**State S0: Self ID Start.** At the start of the self-ID process, the PHY is waiting for a grant from its parent or the start of a self-ID packet from another node. This state is also entered whenever a node is finished receiving a self-ID packet and all its children have not yet finished their self identification.

**Transition S0:S1.** If a node is the root, or if it receives a SELF_ID_GRANT signal from its parent, it enters the Self-ID Grant state.

**Transition S0:S2.** If a node receives a SPEED or DATA_PREFIX signal from its parent, it knows that a self-ID packet is coming from a node in another branch in the tree.

**State S1: Self-ID Grant.** This state is entered when a node is given permission to send a self-ID packet. If it has any unidentified children, it sends a GRANT signal to the lowest numbered of those, unless it is acting as the proxy for that port, in which case it transmit the proxy self_ID packet. All other connected ports are sent a DATA_PREFIX signal to warn them of the start of a self-ID packet.

**Transition S1:S2.** When the PHY receives a SPEED or DATA_PREFIX signal from its lowest numbered unidentified child, it enters the Self-ID Receive state.

**Transition S1:S0.** If the PHY transmitted a proxy self_ID packet, then it transitions back to S0.

**Transition S1:S4.** If there are no more unidentified children, it immediately transitions to the Self-ID Transmit state.

**State S2: Self-ID Receive.** As data symbols are received from the bus they are passed on to the link layer as PHY data indications. Note that multiple self-ID packets may be received in this state. The parent PHY must also monitor the received speed signal whenever IDENT_DONE is received from the child. Because of resynchronization delays in repeating the packet, the parent PHY may not complete retransmission of the packet data and data end signal for up to 144 ns or more (as specified by PHY_DELAY) after the start of the IDENT_DONE signal. Since the child sends its speed signal for no more than 120 ns from the start of the IDENT_DONE signal, the parent could miss the speed signal from the child if it entered S3 before completing a speed signal sample. If the PHY gets an IDENT_DONE signal from the receiving port, it flags that port as identified and, if the port is operating in DS mode, starts sending the speed capabilities signal. It also starts the speed signaling timer

**Transition S2:S0.** When the receive port goes IDLE, gets a SELF_ID_GRANT or observes DATA_PREFIX for a unconcatenated packet it enters the Self-ID Start state to continue the self-ID process for the next child. The last case guards against a possible failure to observe IDLE.

**Transition S2:S3.** The port has received an IDENT_DONE from the child.

**State S3: Send Speed Capabilities.** If a node is capable of sending data at a higher rate that S100 and the receiving port is operating in DS mode, it transmits on the receiving child port its speed capability signals for a fixed duration SPEED_SIGNAL_LENGTH. The parent PHY must also monitor the received speed-signal whenever IDENT_DONE is received from the child.

**Transition S3:S0.** When the speed signaling timer expires, and the receive port is operating in DS mode, any signals sent by the child have been latched, so it is safe to continue with the next child port. This is also the point at which negotiated_speed in the port register map is set for DS mode operation

**State S4: Self-ID Transmit.** This state may be entered either as part of the self-identify process or as the result of the receipt of a PHY ping packet. In the latter case, any pending Legacy link requests are cancelled and the set of self-ID packets are transmitted. When state S4 is entered as part of the self-identify process, the actions are more complex, as described below.

At this point, all child ports have been flagged as identified, so the PHY can now send its own self-ID packet. When a non-root node is finished, it sends a IDENT_DONE signal while simultaneously transmitting a speed capability signal to its parent and IDLE to its children. The speed capability signal is transmitted for a fixed time duration of SPEED_SIGNAL_LENGTH. Simultaneously it monitors the bus for a speed capability transmission from the parent. The highest indicated speed is recorded as the speed capability of the parent. The root node just sends IDLE to its children. Note that the children will then enter the Idle state described in the next clause, but children on DS ports will never start arbitration since an adequate arbitration gap will never open up until the Self-ID process is completed for all nodes.

While transmitting the IDENT_DONE signal in the S4 state, the child monitors the received speed-signal from the parent. The child PHY then transitions to the A0:Idle state when it receives an DATA_PREFIX signal from its parent. The parent PHY will be in the S2:Self-ID-Receive state to receive the self-ID packet(s) from the child. When the parent PHY receives an IDENT_DONE signal from the child PHY, the parent transitions to the S3:Send-Speed-Capabilities state. In the S3 state, the parent transmits a speed-signal for 100-120 ns to indicate its own speed capability, and monitors the received speed-signal from the child. The highest indicated speed is recorded as the speed capability of the child. After transmitting its own speed-signal the parent PHY transitions to the S0:Self-ID-Start state.

**Transition S4:A0b.** The PHY then enters the Idle state described in the next clause when the self-ID packet has been transmitted, this is not a ping response, and if either of the following conditions are met:

a) The node is the root. When the root enters the Idle state, all nodes are now sending IDLE signals and the gap timers will eventually get large enough to allow normal arbitration to start.
b) The node starts to receive a new self-ID packet (SPEED or DATA_PREFIX). This will be the self-ID packet for the parent node or another child of the parent. This event shall cause the PHY to transition immediately out of A0:Idle into RX:Receive

If the parent port will be operating in DS mode, this is when the negotiated_speed field in the port register map for the parent port is set.

### 13.7.7 Arbitration

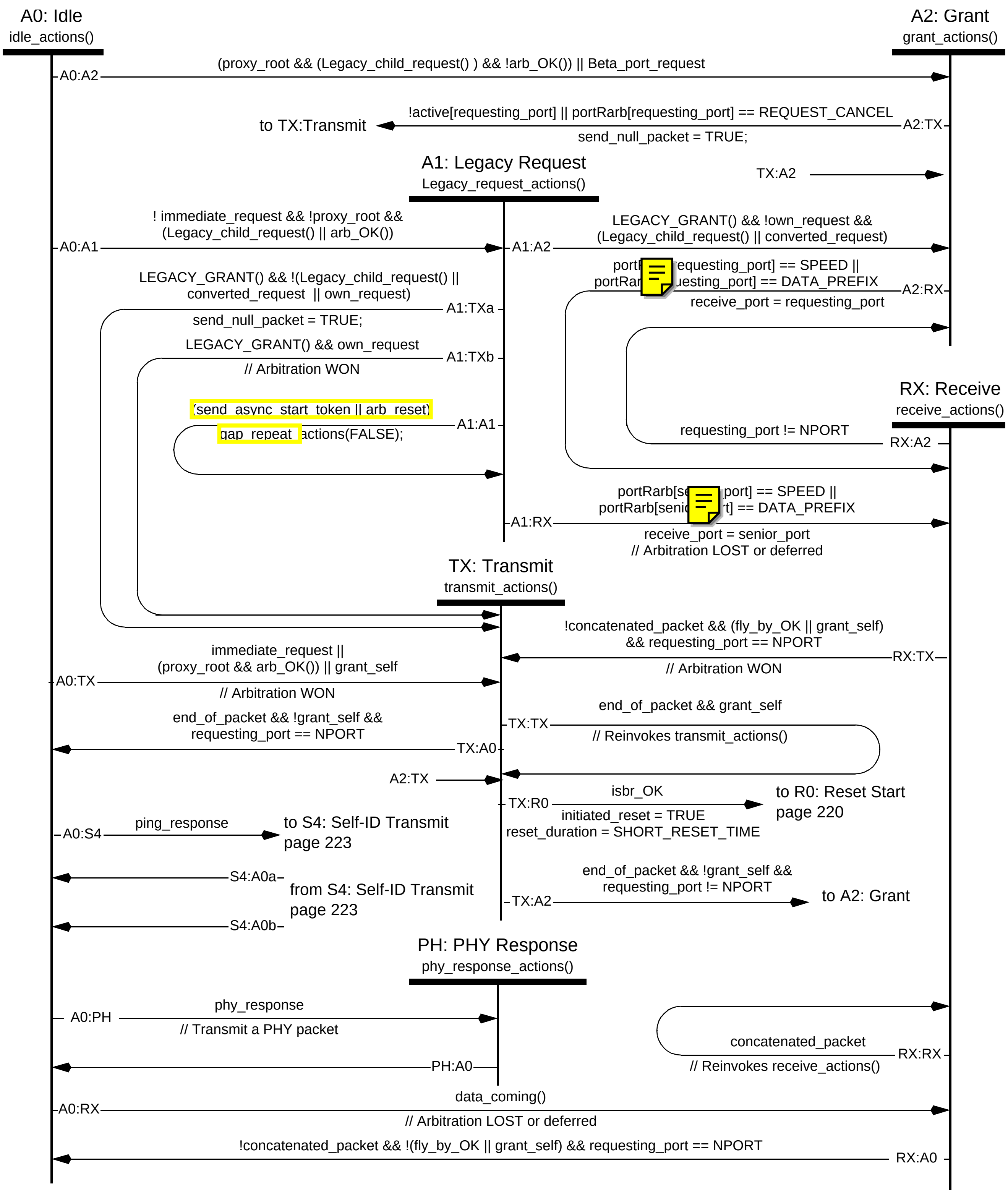


**Figure 13-11—Border arbitration state machine**

NOTE—all C-code function names listed here are hypertext links in electronic versions of this specification

### 13.7.7.1 Arbitration state machine notes

**State A0: Idle.** All inactive nodes stay in the idle state until an internal or external event. All DS ports transmit the IDLE arbitration signal while Beta ports follow the Beta mode request repeating rules (see clause 13.7.3.2). Transitions into this state from states where idle was not being sent reset an idle period timer.

**Transition A0:A1.** If the PHY has a queued request (other than an immediate request) from its own link (and it is a Legacy link) or receives an REQUEST signal from one of its children on a DS port (and is not the root or senior border), it passes the request on to its parent. The `arb_OK()` function qualifies asynchronous requests according to the time elapsed since A0: Idle was last entered. In particular, notice that the test for a subaction gap is performed for a single value (equality), not a greater than comparison. If arbitration were to be initiated at other times between the detection of a subaction gap and an arbitration reset gap, some nodes could mistakenly observe an arbitration reset gap.

**Transition A0:A2.** If, on the other hand, the PHY receives a REQUEST signal from one of its children on a DS port, has no queued requests from its own Legacy link and is the root, it starts the bus grant process. It will also start the bus grant process if a request is received on a Beta port (but not its attached link) and the node is BOSS and a grant can be given (the last subaction is complete).

**Transition A0:PH.** If an extended PHY packet (other than the ping packet) has been received, a response is required.

**Transition A0:TX.** If the PHY has a queued request from its own Legacy link and it is the root or senior border, or the PHY has a queued request from its own Beta link and it either receives a grant token or it is BOSS and the last subaction is complete, or if the PHY has a queued immediate request (generated during packet reception if the link layer needs to send an acknowledge); then the PHY notifies the link layer that it is ready to transmit and enters the Transmit state.

**Transition A0:S4.** In response to the receipt of a PHY “ping” packet, the variable ping_response is set TRUE and a transition is made to the Self-ID Transmit State to send the self-ID packet(s).

**Transition A0:RX.** if a PHY starts receiving a packet, it immediately enters the Receive state.

**State A1: Legacy Request.** A PHY has received a Legacy request and must forward that request towards the root. The PHY sends a REQUEST signal to its parent and a data prefix to all its connected children. This will signal all children to get ready to receive a packet.

**Transition A1:A1.** If the PHY must send an asynchronous start or arbitration reset token on a Beta port, then it sets a flag to guarantee that DS ports will see the appropriate gap (the actual gap generation is done by the actions taken by the various states).

**Transition A1:A2.** If the PHY receives a GRANT signal from its parent and the requesting child is still making a request, the PHY grants the bus to that child.

**Transition A1:RX.** If the PHY receives a DATA_PREFIX or SPEED signal from its parent, then it knows that it has lost the arbitration process and prepares to receive a packet. If the link layer was making the request, it is notified.

**Transition A1:TXa.** If the PHY receives a GRANT signal from its parent and the requesting child has withdrawn its request, the PHY goes to Transmit state to send a null packet.

**Transition A1:TXb.** If the PHY receives a GRANT signal from its parent and the link layer has an outstanding request (asynchronous or isochronous), the PHY notifies the link layer that it can now transmit and enters the Transmit state.

**State A2: Grant.** During the grant process, the requesting child is sent a GRANT signal and the other children are sent a SPEED and/or a DATA_PREFIX so that they will prepare to receive a packet.

**Transition A2:RX.** If the DATA_PREFIX or SPEED signal is received from the requesting child, the grant handshake is complete and the node goes into the Receive state.

**Transition A2:TX.** If the requesting child withdraws its request (the granting PHY sees a REQUEST_CANCEL signal) or is no longer active, the PHY goes directly to the Transmit state to send a null packet.

**State PH: PHY Response.** When the node has received an extended PHY packet (other than the ping packet) that requires a response, this state takes the appropriate actions, builds a remote reply or remote confirmation packet and transmits the packet from all active ports.

**Transition PH:A0.** After transmitting the PHY response packet, return unconditionally to idle.

**State RX: Receive.** When the node starts the receive process, it notifies the link layer that the bus is busy and starts the packet receive process described below. Outstanding fair and priority requests from a Legacy link are cancelled—immediately if arbitration enhancements are globally disabled, otherwise by the receipt of any packet other than an acknowledgement—and the link will have to reissue them later. Requests by Beta links are not cancelled. Note that the packet received could be a PHY packet (self-ID, link-on or PHY configuration), acknowledge, or normal data packet. PHY configuration and link-on packets are interpreted by the PHY, as well as being passed on to the link layer.

**Transition RX:A0.** If transmitting node stops sending a packet and no additional packet is concatenated to it (the received signal is DATA_END or IDLE) and the PHY cannot concatenate a queued packet of its own (fly-by arbitration is disabled or there is no queued packet), the bus is released and the PHY returns to the idle state.

**Transition RX:A2.** tbd.

**Transition RX:RX.** If a packet ends but another packet is concatenated to it (indicated by receiving DATA_PREFIX on the receiving port), the receive process is restarted.

**Transition RX:TX.** If fly-by arbitration is enabled and an acknowledge or isochronous packet ends, a queued request may be granted.

**State TX: Transmit.** Unless an arbitrated (short) bus reset has been requested, the transmission of a packet starts by the node sending a SPEED and DATA_PREFIX signal, and then sending PHY clock indications to the link layer. For each clock indication, the Link sends a PHY data request. The clock indication – data request sequence repeats until the Link sends a DATA_END. Concatenated packets are handled within this state whenever the Link sends at least one data bit followed by a DATA_PREFIX. The arbitration enable flag is cleared if this was a fair request.

**Transition TX:A0.** If the link layer has finished sending a packet and is not requesting concatenation of another packet and the node cannot grant any pending Beta requests (not the end of a subaction), the PHY returns to the Idle state.

**Transition TX:A2.** If the link layer has finished sending a packet and is not requesting concatenation of another packet and there is no queued request from its own Beta link and if the PHY can grant a request from one of its Beta ports (the previous packet was the end of a subaction), the PHY transitions to the Grant state to send a GRANT signal to a requesting Beta port.

**Transition TX:R0.** If arbitration succeeded and the isbr_OK variable is set, there is no packet to transmit. The PHY transitions to the Reset start state to commence a short bus reset.

**Transition TX:TX.** If the link layer has finished sending a packet and is requesting concatenation of another packet or if the node can grant a pending Beta request from its own link (the PHY is BOSS and the previous packet was the end of a subaction), the PHY returns to the Transmit state to send the new packet.