PORT SUSPEND/RESUME ACTIONS

Draft 0.41 Sept. 8, 1997

P1394A/97-0053R0

```
void suspend_initiator_actions()   {
      port_timer = 0;
      queued_suspend_initiator = FALSE;      // clear the suspend request
      wait((port_timer >= NOTIFY_HOLD) || (STATUS_A.bias == 0));
      if (STATUS_A.bias == 0)  {              //TpBias driven low by target
            CONTROL_SET.suspend = 1;        // set suspend when target TPBias low
            TpGen = LOW;                    // drive TpBias
            wait_time (BIAS_HOLD);
      }
 }


void suspend_target_actions()  {
      queued_suspend_target = FALSE;
      port_timer = 0;                        // start the timer
      TpGen = LOW;                           // drive TpBias low to Initiator
      wait ((port_timer >= BIAS_HOLD) || (STATUS_A.bias == 0));
                                             // wait for initiator to drive TpBias
      if (STATUS_A.bias == 0)       // handshake with initiator
            CONTROL_SET.suspend = 1;        // enter suspended state
}

void suspend_failed_actions ()  {
      power_down = TRUE;                     // turn off bias generator; this will disconnect 1995 PHY
}


void disconnected_actions ()  {
      power_down = TRUE;                     // turn off bias generator
      CONTROL_SET.suspend = 1;
}

void suspended_actions () {
      TpGen = LOW;                           // drive TpBias low and then release
      wait (con_status == TRUE);             // wait to eliminate false disconnect
      power_down = TRUE;                     // turn off bias generator
      CONTROL_SET.suspend = 1;
}
```

```
void resume _actions()  {
      queued_resume = FALSE;  // clear the resume request
      power_down = FALSE;                              // connect detector is invalid
      TpGen = HIGH;                                    // drive TpBias to resume target
      port_timer = 0;                                  // start timer
      wait ( (port_timer >= DETECT_MIN) || (STATUS_A.bias == 1));
                                                                  // wait for target to drive TpBias
      if (STATUS_A.bias == 1)                 // resume succeeded
            CONTROL_CLEAR.suspend = 1;
}



void resume_failed_actions ()  {
      TpGen = LOW;                            // drive TpBias low and then release
      wait (con_status == TRUE);              // wait to eliminate false disconnect
      power_down = TRUE;                      // return to suspend state
}


void connected_actions ()  {
}


void disabled_actions ()  {
      TpGen = LOW;                            // drive TpBias low and then release
      wait (con_status == TRUE);              // wait to eliminate false disconnect
      power_down = TRUE;                      // Stop driving TpBias
}
```