Test Plan

## IP TEST

### Top overview

Below is the overview on chip top to show the test architecture.

Basically all function signals go through the IOMUX for FUNC. But for test signals, they go through IOMUX for DFT and then IOMUX for FUNC.

There is reserved datapath for test signals in IOMUX for FUNC.



The test signals including IP dedicated jtag signals and other necessary control status signals for testing purpose.

Detailed definition and pin mapping please check the below excel file:

### DDR PHY

Apart from the loopback test modes, there are various other PUB test modes that are used to facilitate the testing of various features of the PHY. These test modes include ATPG scan mode, delay line oscillator mode, and PLL test mode. The guidance provided in this chapter for silicon testing and characterization of Digital Delay Lines (DDLs) and Phase Locked Loops (PLLs) can be applied to either product characterization and/or production testing. This chapter describes the following:

❖“ATPG Test Mode” on page 358

❖“Loopback BIST Mode” on page 359

❖“Delay Element Testing” on page 362

❖“Analog/Digital Test Observability” on page 365

❖“Burn-in Recommendations” on page 368

❖“Bypass Mode” on page 368

#### ATPG Test Mode

All possible registers in the PHY blocks are put on scan chains. Each scan chain is based on an internal functional clock group with no mixing of clock edges. Different functional clocks, which have the same edge and have their clock trees functionally balanced, are treated as one clock group. All scan chains have lockup latches at their outputs.

The table below shows the characteristics of the PHY scan chains. There is a separate scan enable (atpg\_se[\*]) for each clock domain, even when that domain is local to sub-logic in the tile. Scan pins are intentionally left open in the RTL. The atpg\_si and atpg\_so (scan in/out) are treated as ports at the top-level for all the hard IP instances.

Chip scan insertion is not done in RTL. It is done in implementation. The scan-insertion tools for the hard-IP will create the scan chains for those hard IP in the gate-level netlist, but not in the RTL. For the PUB, the customer should add the scan insertion and stitching.

Each scan chain has a maximum length of 100 registers. Scan chains belonging to the same group or clocked by the same edge of the clock may be concatenated externally, to build up longer, balanced chains, if required. Since all scan chains use a positive clock edge, it is possible to concatenate all the chains together.

Two additional features are generally implemented internally to increase test pattern coverage without additional effort or actions:

❖Reset inputs of infrequently used asynchronous flops are forced off during scan test (using atpg\_mode).

❖Clock gaters are forced on during test clock strobing (using atpg\_se).

**Note** In ATPG Mode (atpg\_mode input pin == 1), all IO drivers are forced to maximum drive strength.

#### Loopback BIST Mode

The PHY contains features to support loopback testing functionality. Once placed in loopback mode, stimulus patterns may be either generated by the core, or by utilizing a Firmware image to activate the hardware training state machines. The recommended procedure for loopback is as follows:

❖Follow the PHY Initialization procedure to bring up VDD, VDDA and VDDQ

❖Follow the PHY initialization procedure to Start Clocks and Reset the PHY

❖Load the microcontroller memory with the Loopback BIST firmware

❖Lock the DfiClk to the desired Loopback Frequency

❖Execute the Loopback BIST Firmware:

✦Initialize the Firmware Message Block with the required test parameters. The content to be written is test dependent, so the documentation provided with each firmware BIST image should be consulted.

✦Enable access to the internal CSRs by setting the MicroContMuxSel CSR to 1. This allows the microcontroller unrestricted access to the configuration CSRs.

✦Begin execution of the firmware by setting the MicroReset CSR to 4’b0000.

✦Poll mailbox until the microcontroller signals loopback completed

✦Enable access to the internal CSRs by setting the MicroContMuxSel CSR to 0. This allows the microcontroller unrestricted access to the configuration CSRs.

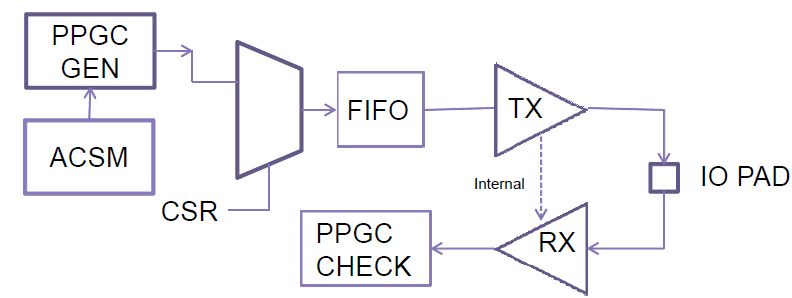
✦Read the Firmware Message Block to obtain the results from the Loopback BIST

Data signals are optionally looped back internally or at the at the I/O pad. If I/O pad loopback is selected, it is important that these points are correctly externally terminated. Data byte DQ/DM and DQS lanes are loop backed to their respective receivers. Address/command lanes contain dedicated built-in loopback logic. A PRB7 generator and self-synchronizing checkers are implemented for this purpose, along with dedicated test receivers.

A typical Data PHY loopback configuration sequence utilizing the hardware training state machines is shown in the following figure.

**Note** The following example is for information only. The loopback BIST firmware executes the sequence once loaded and activated.

Figure 5-1 Data Loopback

****

The portion of the Firmware dedicated to Loopback BIST performs the following functions

❖ The PPGC is configured to supply a test pattern. In the case of data loopback, a pre-configured PRBS23 or PRBS16 can be selected as the stimulus.

❖ The test mode configuration registers are configured for data loopback.

❖ The Tx and Rx delay are set s accordingly

❖ DTSM configured to check for errors

❖ ACSM configured to perform a sequence of multiple writes.

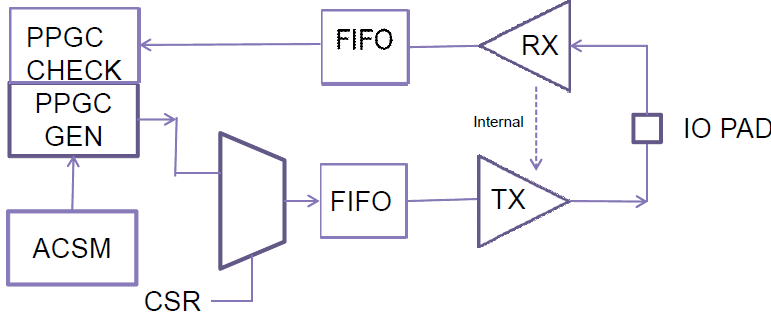
❖ ACSM loopback mode enabled.

❖ Initiate ACSM run sequence.

❖ On completion, check the accumulated error counter for each lane. A typical Address/Command PHY loopback configuration sequence is a follows.

**Note**The following example is for information only. The loopback BIST firmware executes the sequence once loaded and activated.

Figure 5-2 AC Loopback



❖ACSM configured to perform multiple writes to advance PPGC.

❖ACSM Enabled for either half rate or full rate AC PRBS mode.

❖Initiate ACSM run sequence.

❖On completion, check the error counter for each lane.

The portion of the Firmware dedicated to Loopback BIST performs the following functions

❖ The PPGC is configured to supply a test pattern. In the case of data loopback, a pre-configured PRBS23 or PRBS16 can be selected as the stimulus.

❖ The test mode configuration registers are configured for data loopback.

❖ The Tx and Rx delay are set s accordingly

❖ DTSM configured to check for errors

❖ ACSM configured to perform a sequence of multiple writes.

❖ ACSM loopback mode enabled.

❖ Initiate ACSM run sequence.

❖ On completion, check the accumulated error counter for each lane. A typical Address/Command PHY loopback configuration sequence is a follows.

Note The following example is for information only. The loopback BIST firmware executes the sequence once loaded and activated.

Note The following example is for information only. The Loopback BIST Firmware executes the sequence once loaded and activated.

❖ACSM configured to perform multiple writes to advance PPGC.

❖ACSM Enabled for either half rate or full rate AC PRBS mode.

❖Initiate ACSM run sequence.

❖On completion, check the error counter for each lane.

#### Delay Element Testing

There is a facility for testing the delay macros for linearity of delay time with changes in delay configuration.

There are two types of delay elements in the design:

1. LCDL used for address/command lane delay, write DQ/DQS delay, RxEn delay, and RxClkDly. (Read LCDL as “local calibrated delay line”. LCDLs are also sometimes called DLLs or DDLs.)

2. BDL used for read DQ delay. (Read BDL as “bit delay line”. BDLs are uncalibrated.) Each DQ/DBI lane has a BDL. The maximum delay BDLs can provide is one-eighth the maximum delay of the LCDLs.

An important property of this test logic is that “pclk”, the SDR UI-period clock generated by the PLL, is not used (that is, the PLL does not need to be locked). The only clock required is the input DfiClk which is used as the PHY's csr clock. Conceivably, a brief LCDL linearity test could be run during PLL locking

The delay provided by an LCDL consists of two parts:

1. A non-configurable (that is, unavoidable) delay that we call "insertion\_delay" or "zerodelay" and

2. A configurable delay that we call "Phase[8:0]" on the LCDL wrappers and "dly\_sel" on the LCDL/BDL macro.

The total delay is given by:

❖LCDL\_delay = (dly\_sel[8:0] \* stepsize) + zerodelay;

❖BDL\_delay = (dly\_sel[5:0] \* stepsize) + zerodelay;

Where the LCLD/BDL time constants zerodelay and stepsize are technology dependent, and vary with instance, process, temperature, and voltage.

The zerodelay is typically 200ps, and the stepsize is typically 5ps per unit dly\_sel.The preliminary smallest zerodelay is 90ps for the FFF corner at 0.99 V.

**Note** It is convenient to speak of the units of dly\_sel as being "bufs" (or buffers) even though that is not exactly how the LCDL is designed, when doing measurement unit conversion between dly\_sel and register{TxDq,TxDqs,RxEn,RxClk}Dly which are denominated in integer/fractional UI. Because of this units of stepsize could said to be ps/buf.)

Single-delay macros are tested with the following counters:

1. One down-counter clocked by DfiClk to provide an absolute time scale (readable as Register DlyTestCntDfiClk), and

2. A second up-counter clocked by a ring oscillator (RO) comprised of one delay element ("cell under test") and an inverter.

The RO counter is readable from Register DlyTestCntRingOsc{Db<$db>/Ac}[15:0]. The period of the SDR RO clock is 2\*LCDL\_delay.

The test might fail because of a design problem of the LCDL, or an individual instance might fail because of a "stuck-at", short, or open-circuit fault of dly\_sel or other control signals.

The objective of the linearity test is to show that the delay of the delay cells, and hence the period of the ring-oscillator (Tro) is linear with the dly\_sel (RegisterLcdlCalPhase[8:0]):

❖Tro = a \* dly\_sel + b;

The runtime for testing a single step is:

❖runtime = Register DlyTestCntDfiClkIV \* Tdficlk = Register DlyTestCntRingOsc \* Tro;

or, solving for Tro

❖Tro = Register DlyTestCntDfiClkIV \* Tdficlk / Register DlyTestCntRingOsc;

By commutivity

❖a \* dly\_sel + b = Register DlyTestCntDfiClkIV \* Tdficlk / Register DlyTestCntRingOsc;

Now solve for a and b. An alternative is to calculate the difference step-by-step shown below. One iteration of the test (ie "the inner loop") consists of the following:

❖The dly\_sel value is configured with dly\_sel\_i, by writing Register LcdlCalPhase[8:0]=i and pulsing Register LcdlCalPhaseUpdate.

❖The RO up-counter is zeroed and the DfiClk down-counter is loaded with initial value Register DlyTestCntDfiClkIV[15:0] by pulsing Register DlyTestCntInit.

❖On the deassertion of Register DlyTestCntInit, the DfiClk down counter starts to count down.

❖When Register DlyTestCntDfiClk[15:0] reaches zero the value of the RO counter DlyTestCntRingOsc{Db<$db>/Ac}[15:0]

❖is read, and the first of two samples, DlyTestCntRingOsc\_i[15:0], is acquired.

❖Now repeat the above with dly\_sel\_j = dly\_sel\_i + 1 (or Register LcdlCalPhase[8:0]=i+1) to get a second sample,

❖DlyTestCntRingOsc\_j[15:0];

The difference in the RO counter terminal values is the unnormalized stepsize for that pair of dly\_sel values.

The stepsize\_i in ps units between dly\_sel=i and dly\_sel=j=i+1 may be calculated as:

❖PeriodRO\_i(ps) = (Register DlyTestCntDfiClkIV[15:0] \* period(ps,DfiClk))/( Register DlyTestCntRingOsc\_i[15:0])

❖PeriodRO\_j(ps) = (Register DlyTestCntDfiClkIV[15:0] \* period(ps,DfiClk))/( Register DlyTestCntRingOsc\_j[15:0])

❖stepsize\_ji = PeriodRO\_j - PeriodRO\_i

A complete test (for a single delay macro) would march through all 511 values of dly\_sel\_j[8:0] = [1...511].Each resulting stepsize\_j must lay within a tolerance band, for the linearity test to pass.The accuracy of the measurement depends on the length of time that an iteration is allowed to run;larger values Register DlyTestCntDfiClkIV[15:0] will result in better accuracy and affects the tolerance band.

One idea for a "good enough" quick test is to sample only the ten values of dly\_sel={0,1,2,4,8,16,...256}.Note that this test mechanism can be used to "calibrate" the BDL (or the LCDL for that matter), where "calibrate" means to find the dly\_sel that causes a one-quarter UI delay for the BDL or a one UI delay for the LCDL, using the fact that the period of DfiClk is four UI. But there is a quicker way to do this calibration; see the related topic of LcdlCal.

Since this test uses hardware that is not used by mission-mode operation, a RO counter is shared for the testing of a group of delay macros. Each data byte "mac" modules (dwc\_ddrphydbyte\_top.v) has 25 delay macros but is allocated only one RO counter. So a build with nine data bytes has nine RO counters.

All of the delay macros in all of the address/command anib "mac" module (dwc\_ddrphyacx4\_top.v) share one RO counter; in a 12 anib build there are 24 delay macros. In 9-byte, 48-addr/cmd lanes build, there are 10 RO counters, and 10 delay macros can be tested in parallel. A complete test of such a build would require 25\*511 iterations. The time to complete is proportional to Register DlyTestCntDfiClkIV[15:0].

#### Quick Reference for Delay Test Registers

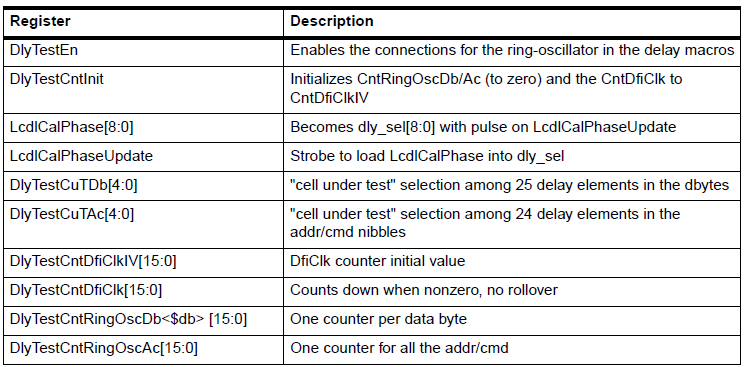
Each delay cell has two inputs from the PUB, DlyTestClkEn,DlyTestClkIn, and one output to the PUB, DlyTestClkOut

The selection of the cell under test and the stitching together of the RO and connection to the clock of the RO counter are done in the PUB.

Implementation should note that the DlyTestClkOut when wired in a one-delay-cell RO will have a period as short as 2\*zerodelay\_min (320ps assuming a 20% faster than zerodelay\_typ=200ps). This clock is used to clock a 16b counter in the PUB. All other logic in the PUB operates at a period of 4\*UI\_min = 1176 ps (for 3400MHz UI clock).

There is an optional configuration that provides a larger clock period. If DlyTestEoC does not equal DlyTestSoC, then the implementation RO clock period is 2\*(zero\_delay\_min\_EoC + LCDL\_delay\_SoC), where the LCDL\_delay\_SoC may have a fixed dly\_sel sufficient to have the minimum RO clock period meet implementation requirements.

Table 5-1 Delay Test Registers

****

#### Analog/Digital Test Observability

Table 5-2 Digital Observation on BP\_A1LN12

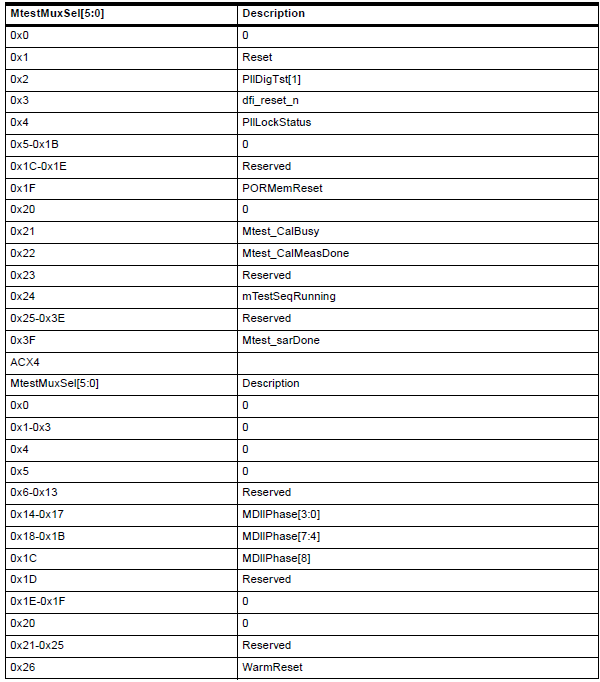
****

Table 5-2 Digital Observation on BP\_A1LN12

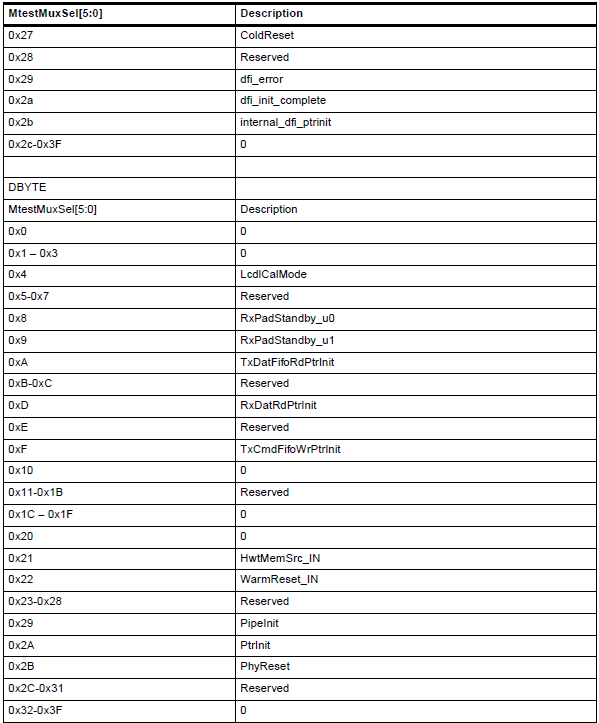
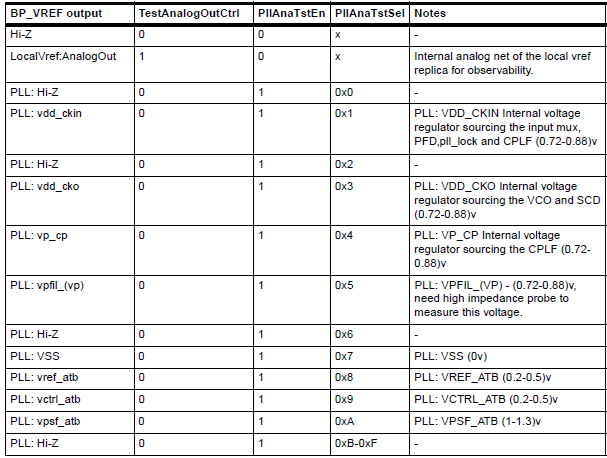
****

Table 5-3 Analog Observation on BP\_VREF

****

#### Burn-in recommendations

During burn-in testing, the recommended action is to run a combination of “Loopback BIST Mode” and “ATPG Test Mode” This will ensure maximum toggling of internal nodes. The recommendation is to alternate equally between the two modes throughout the burn-in testing process:

❖For 50% of the duration, activate Loopback BIST Mode

✦Loopback BIST Mode is enabled through loading and running of the Loopback BIST Firmware Image

❖For 50% of the duration, activate ATPG Test Mode

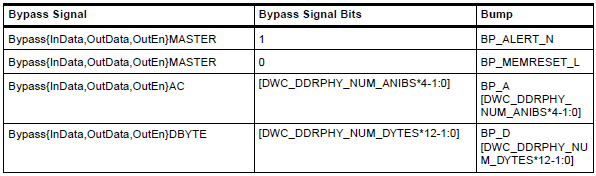
✦Operate in Scan Shift Mode

✦Activate all Scan Chains simultaneously

#### Bypass Mode

The table below shows the Bypass/Flyover mapping from the Bypass Signal to the Bumps. Boundary scan is added by the customer by connecting the various Bypass signals {ModeEn, OutEn, OutData, InData} {AC, DAT, MASTER} in the customer wrapper code to create the desired boundary-scan order and behavior. The PHY architecture/RTL is fully flexible with respect to this ordering.

Table 5-4 Bypass/Flyover Mapping from the Bypass Signal to the Bumps

****

##### Bypass Mode Rules

The following rules apply to the Bypass Mode control signals:

❖All Bypass signals must be zero in mission mode

❖All Bypass signals must stay zero until after PwrOkIn goes from 1-->0

❖All Bypass signals must be zero before PwrOkIn goes from 0-->1

Due to the design of BP\_MEMRESET\_L, the following rules apply to that I/O:

❖BP\_MEMRESET\_L does not support tristate

✦Generally BumpData = BypassModeEn && BypassOutData

✦This pin does not accept the JTAG HIGHZ command

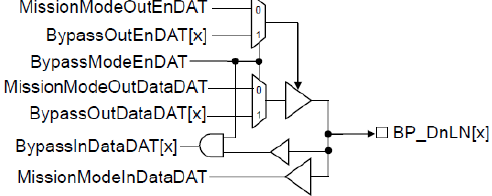
❖BP\_MEMRESET\_L is output-only

✦Generally BypassInData = BypassModeEn && BypassOutData

##### Bypass Mode Architecture

The functional architecture of the Bypass/Flyover controls is shown in the following figure.

Figure 5-3 Bypass/Flyover Controls

****

#### IO Cell DC Parametric Characterization

To perform IO Cell Parametric Characterization, the PHY provides several options for input and output stimulus and measurements. Through use of the CSR configurations and the Bypass signals, the user may perform Voltage Input High/Low (VIH/VIL) and Voltage Output High/Low (VOH/VOL) characterization on DQ and AC receivers and drivers, respectively.

#### 5.1.8.1 VIH/VIL Testing

##### Testing the BP\_ALERT\_N Receiver

The test should be configured in flyover mode. Put the BP\_ALERT\_N receiver is test mode by setting the following:

1. PwrOk==1

2. atpg\_mode==0

3. BypassModeEnMaster==1

4. CSR MemAlertControl

a. MALERTRxEn==1

b. MALERTVrefLevel=='d64 (can be any value , but preferably 64-70)

5. For a particular VrefDAC CSR setting :: slowly sweep BP\_ALERT\_N from VSS to VDDQ ; note the voltage on BP\_ALERT\_N and the related logical value on BypassInDataMASTER[1].

6. Then change the VrefDAC CSR setting to another value and repeat steps 1-5

##### Testing a DQ Receiver

The voltage on BP\_D0LN0 should be swept to test the Receiver instance u\_DWC\_DDRPHYDATX4X2\_0.txrxdq0.RX and observe when the output BypassInDataDAT [0] toggles, program the VrefDAC CSR setting to a chosen value (for example ‘d64).

The test should be configured in flyover mode. Put the BP\_D0LN0 receiver in test mode by setting the following:

1. PwrOk==1

2. atpg\_mode==0

3. BypassModeEnMaster==1

4. BypassModeEnDat == 1

5. CSR writes: cwriting to csrs UcMemResetControl,PorControl,DqDqsRcvCntrl1 to power up the receiver

a. UcMemResetControl - UcDctSane should be set to “1”

b. PorControl - PllDllLockDone should be set to “1”

c. DqDqsRcvCntr - DfeCtrl 3:2 == 01

d. Reserved 15:14 == 01 //DFE0 enabled

e. DqDqsRcvCntrl1 - RxPadStandbyEn should be set to “1”

6. For a particular VrefDAC CSR setting :: slowly sweep BP\_D0LN0 from VSS to VDDQ ; note the voltage on BP\_D0LN0 and related logical value on BypassInDataDAT[0].

7. Then change the VrefDAC CSR setting to another value (for example ‘d69) and repeat steps 1-5.

#### 5.1.8.2 VOH/VOL Testing

##### DQ Pull Up Testing

1. Full PHY bring up, run calibration at the background (continuous calibration).

2. Set the device in Flyover mode. ✦BypassOutDataDAT[0] = 1'b1 ✦BypassOutEnDAT[11:0] = 12'hfff

3. Set the desired effective resistance on the BP\_D0LN0 node. Refer to the “Output Conditions” section of the PHY Databook for the list of effective resistances at select voltage points.

4. Sweep pull\_up impedance csr , TxEqImpedanceDq[5:0] with these values ✦6'h01, 6'h02, 6'h03, 6'h06, 6'h07, 6'h0A, 6'h0B, 6'h0E, 6'h0F, 6'h1A, 6'h1B, 6'h1E, 6'h1F, 6'h3A, 6'h3B, 6'h3E, 6'h3F.

5. Measurements to be taken on the BP\_D0LN0 node.

##### DQ Pull Down Testing

1. Full PHY bring up, run calibration at the background (continuous calibration).

2. Set the device in Flyover mode. ✦BypassOutDataDAT[0] = 1'b0 ✦BypassOutEnDAT[11:0] = 12'hfff

3. Set the desired effective resistance on the BP\_D0LN0 node. Refer to the “Output Conditions” section of the PHY Databook for the list of effective resistances at select voltage points.

4. Sweep pull\_down impedance csr , TxEqImpedanceDq[11:6] with these values

✦6'h01, 6'h02, 6'h03, 6'h06, 6'h07, 6'h0A, 6'h0B, 6'h0E, 6'h0F, 6'h1A, 6'h1B, 6'h1E, 6'h1F, 6'h3A, 6'h3B, 6'h3E, 6'h3F

5. Measurements to be taken on the BP\_D0LN0 node.

##### AC Pull Down Testing

1. Full PHY bring up, run calibration at the background (continuous calibration).

2. Set the device in Flyover mode.

✦BypassOutDataAC[0] = 1’b0;

✦BypassOutEnAC[0] = 1'b1;

3. Set the desired effective resistance on the BP\_A0LN0 node. Refer to the “Output Conditions” section of the PHY Databook for the list of effective resistances at select voltage points.

4. Sweep pull\_down impedance csr , ATxImpedance[9:5] with these values (00000, 00001, 00011, 00111, 01111, 11111).

5. Measurements to be taken on the BP\_A0LN0 node.

##### AC Pull Up Testing

1. Full PHY bring up, run calibration at the background (continuous calibration).

2. Set the device in Flyover mode.

✦BypassOutDataAC[1] = 1’b1;

✦BypassOutEnAC[0] = 1'b1;

3. Set the desired effective resistance on the BP\_A0LN0 node. Refer to the “Output Conditions” section of the PHY Databook for the list of effective resistances at select voltage points.

4. Sweep pull\_up impedance csr , ATxImpedance[4:0] with these values (00000, 00001, 00011, 00111, 01111, 11111).

5. Measurements to be taken on the BP\_A0LN0 node.

#### VDDQ Quiescent Current Measurement

The state of the PHY at the de-asserting edge of the WarmReset sequence is very close to the conditions required to perform VDDQ Quiescent Current Measurements. After successfully completing the Power-on and Reset sequence (see “PHY Initialization Details” on page 490), the following additional registers should be programmed

**Caution** No other registers should be programmed other than those specified in order to ensure the correct measurement

❖Disable the PHY Master Vref Generator by writing:

✦Address 32’h0200B2 Data 16’h0000 //DWC\_DDRPHYA\_MASTER0\_VrefInGlobal\_p0

In this state, the PHY drives the following signals statically to zero. The state of these pins are provided for information only, this state does not affect the measurement.

❖BP\_MEMRESET\_L = 1’b0 //DRAM RESET#

❖BP\_A pins that can funtion as CKW:

✦3ACX::BP\_A[1:0]

✦6ACX::BP\_A[13:12,1:0]

✦10ACX::BP\_A[21:20,1:0]

✦12ACX::BP\_A[47:46,3:0]

#### Input Pin Leakage Measurement

To perform Input Pin Leakage measurements, the following PHY top-level input signals must be set:

❖BypassModeEnAC = 1’b1

❖BypassOutEnAC[(`DWC\_DDRPHY\_NUM\_ANIBS\*4)-1:0] = All zeros

❖BypassModeEnDAT = 1’b1

❖BypassOutEnDAT[(`DWC\_DDRPHY\_NUM\_DBYTES\*12)-1:0] = All zeros

❖BypassModeEnMASTER = 1’b1 ❖BypassOutEnMASTER[1] = 1’b0

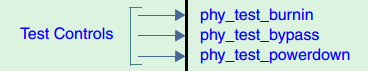
All PHY I/O pins will be tristated in this mode and will support the input pin leakage measurements.

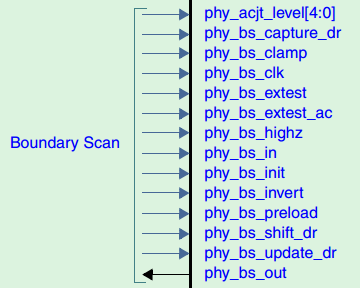
The PHY Output-only Pin BP\_MEMRESET\_L does not support input pin leakage measurements (or the JTAG HIGHZ command). The PHY Analog pins {BP\_VREF, BP\_ZN, BP\_ZN\_SENSE} do not support input pin leakage measurements.

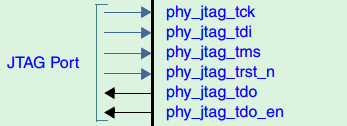
**Note** This state is typically achieved when the customer has connected the PHY Bypass\* signals to a TDR that implements the JTAG HIGHZ command.

### PCIE PHY

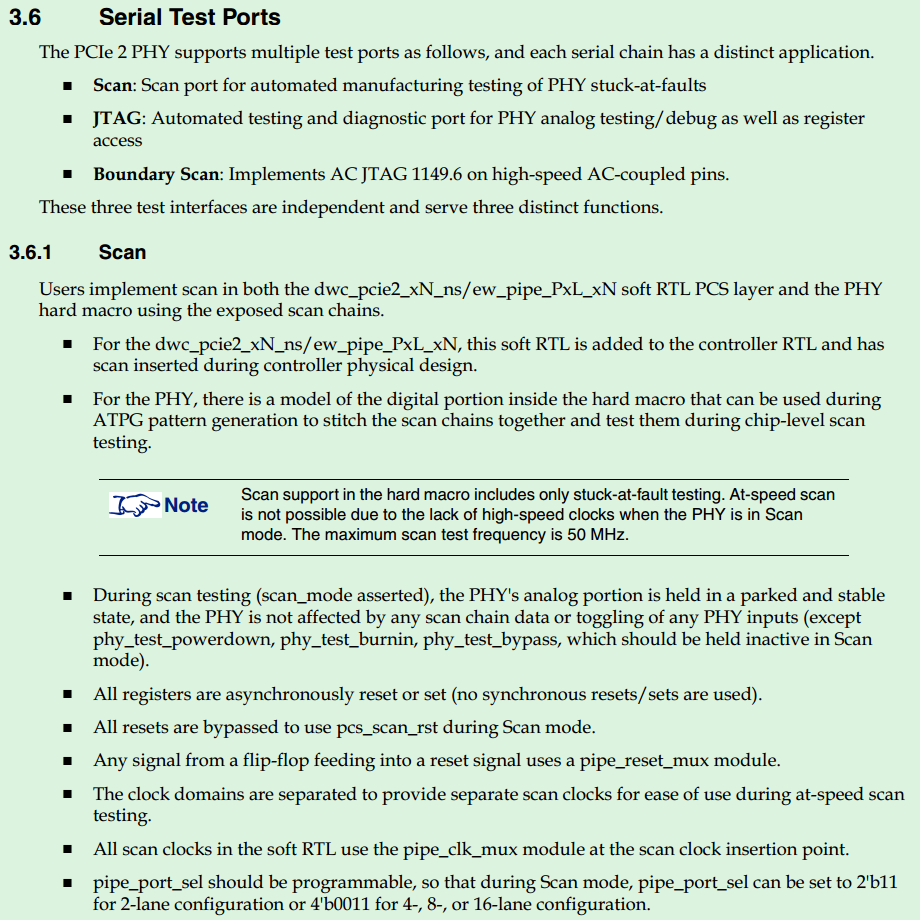
#### TEST PIN DIAGRAM

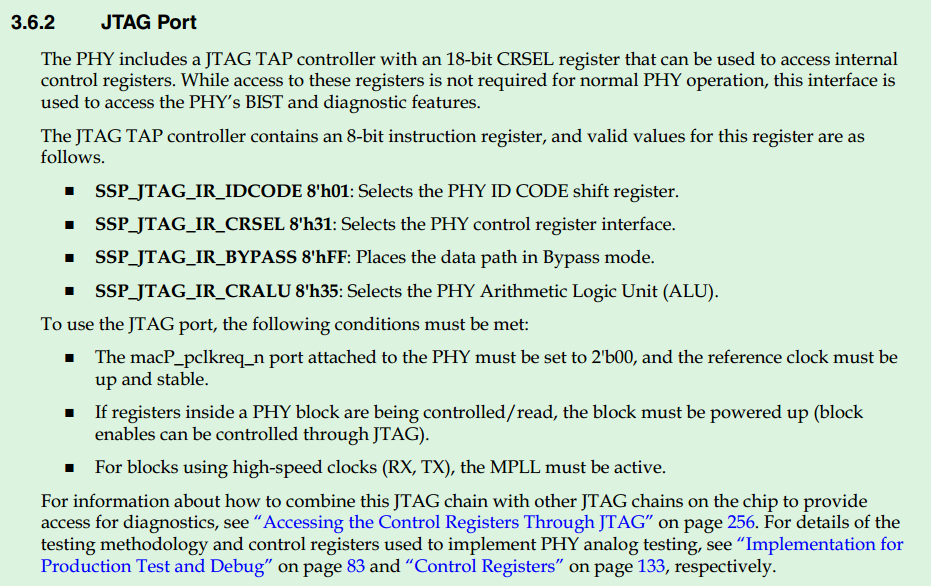


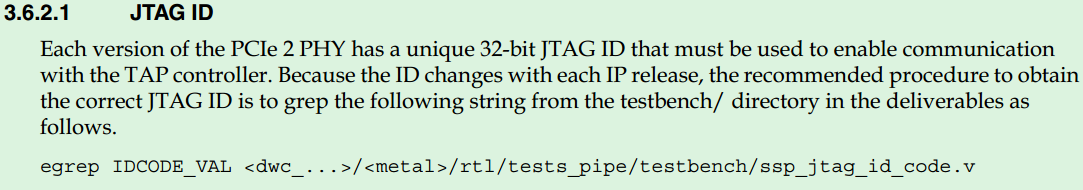


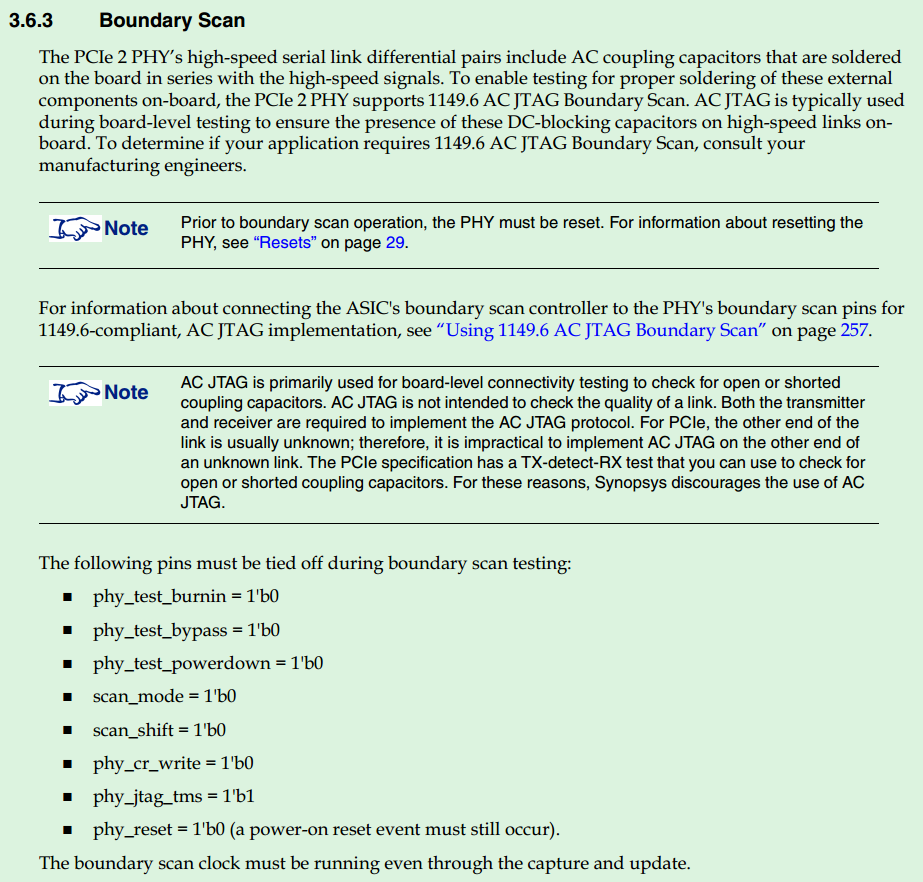


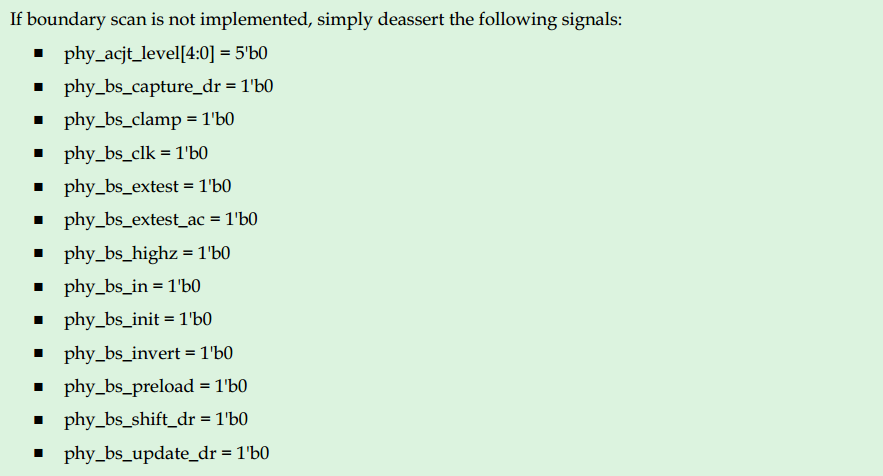
#### Serial Test Ports



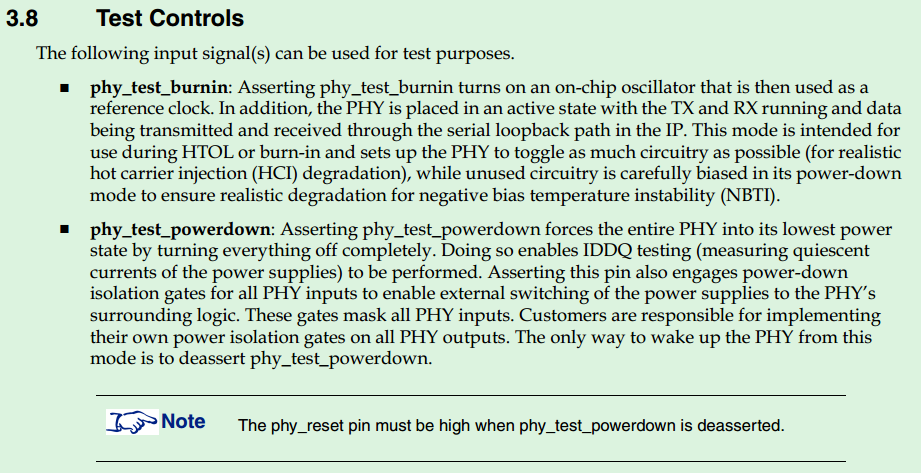


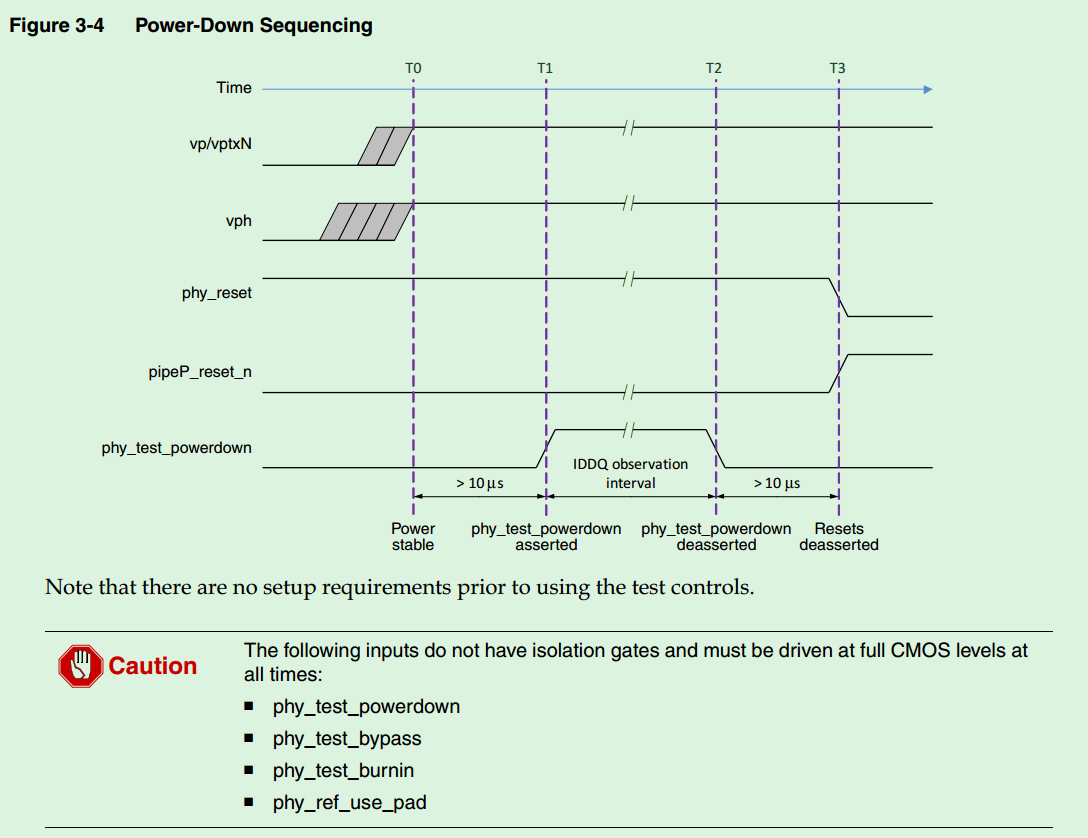






#### Test controls





### USB PHY

#### Test Strategy Overview

The USB 3.0 PHY is designed to minimize the effort required to get the finished product chip intoproduction with minimum test time and minimum test development. Software is provided with the IP tocreate test vectors that utilize the PHY’s Built-In-Self-Test (BIST) capability and JTAG port to test the PHY inproduction. Test development time is minimized by placing sufficient BIST hardware in the IP to compareanalog values with high and low limits in the IP and simply scan out a pass or fail result for easy vectorcomparison on a digital tester.

#### ASIC Pin Requirements to Enable Production Testing

To enable ATE testing, the only requirements on the ASIC-side pins are access to the four JTAG pins and thefollowing settings.

■ Reset must be deasserted.

■ The reference clock must be running, properly configured using ref\_use\_pad, and enabled usingref\_ssp\_en.

■ The test\_powerdown\_ssp and test\_powerdown\_hsp signals must be set to 1’b0.

■ The test\_burnin signal must be set to 1’b0.

#### Integration Requirements for usbmix

After POR (power on reset), ref\_use\_pad is configured properly. And test\_powerdown\_ssp , test\_powerdown\_hsp and test\_burnin are tie to 0 when ioring\_ipp\_ind\_test\_mode is 1. And ref\_ssp\_en is asserted whenioring\_ipp\_ind\_test\_mode is 1.

So the integration requirements for usbmix testing is

(1) usb1\_ateresetdeasserted

(2) ioring\_ipp\_ind\_test\_modeasserted

#### usbmix test ports from PAD

|  |  |  |
| --- | --- | --- |
| **Signals** | **IO** | **Description** |
| usb1\_atereset | I | USB1 ATE reset |
| usb1\_LOOPBACKENB | I | USB1 Loopback Test Enable |
| usb1\_VATESTENB0 | I | USB1 Analog Test Pin Select |
| usb1\_atereset | I | USB2 ATE reset |
| usb1\_LOOPBACKENB | I | USB2 Loopback Test Enable |
| usb1\_VATESTENB0 | I | USB2 Analog Test Pin Select |

#### Boundary scan ports of usbmix top

|  |  |  |
| --- | --- | --- |
| **Signals** | **IO** | **Description** |
| sjc\_usb3\_ac\_bs\_capture\_dr | I | Boundary Scan CaptureDR Signal. |
| sjc\_usb3\_ac\_bs\_clamp | I | Boundary Scan Clamp Signal. |
| sjc\_usb3\_ac\_bs\_clk | I | Boundary Scan clock. |
| sjc\_usb3\_ac\_bs\_extest | I | Boundary Scan EXTEST Signal. |
| sjc\_usb3\_ac\_bs\_extest\_ac | I | Boundary Scan AC EXTEST Signal. |
| sjc\_usb3\_ac\_bs\_highz | I | Boundary Scan High-Impedance Signal. |
| sjc\_bsr\_tdo | I | Boundary Scan Chain Input. To usb1 phy bs\_in |
| sjc\_usb3\_ac\_bs\_init | I | Boundary Scan Initialize Signal. |
| sjc\_usb3\_ac\_bs\_invert | I | Boundary Scan Invert Signal. |
| sjc\_usb3\_ac\_bs\_preload | I | Boundary Scan Preload Signal. |
| sjc\_usb3\_ac\_bs\_shift\_dr | I | Boundary Scan ShiftDR Signal. |
| sjc\_usb3\_ac\_bs\_update\_dr | I | Boundary Scan UpdateDR Signal. |
| usb2\_usb3\_phy\_bs\_out | O | Boundary Scan Chain Output. From usb2 phy bs\_out |

#### Scan ports

|  |  |  |
| --- | --- | --- |
| **Signals** | **IO** | **Description** |
| pcs\_scan\_mode | I | PCS Scan Mode |
| pcs\_scan\_shift | I | PCS Scan Shift |
| pcs\_scan\_rst | I | PCS Scan Reset |
| pcs\_scan\_refclk | I | PCS Scan Reference clock |
| pcs\_scan\_clk | I | PCS Scan Clock |
| pcs\_scan\_pclk | I | PCS Scan PCLK |
| scan\_mode | I | Scan Mode |
| scan\_shift | I | Scan Shift |
| scan\_rst | I | PHY Scan Reset |
| scan\_clk | I | Scan Clock |
| scan\_in[39:0] | I | Scan Mode Input Data |
| scan\_out[39:0] | O | Scan Mode Output Data |
| SCANNSI | I | Scan Data In for Negative-Edge-Triggered Scan Chain |
| SCANNLNSO | O | Non-Lock-Up Scan Data Out for Negative-Edge-Triggered Scan Chain |
| SCANNSO | O | Scan Data Out for Negative-Edge-Triggered Scan Chain |

#### JTAG Port

|  |  |  |
| --- | --- | --- |
| **Signals** | **IO** | **Description** |
| jtag\_tck | I | JTAG clock |
| jtag\_tdi | I | JTAG input port |
| jtag\_tms | I | JTAG State Machine Control |
| jtag\_tdo | O | JTAG output port |
| jtag\_tdo\_en | O | JTAG Output enable |

#### Boundary scan ports of phy

|  |  |  |
| --- | --- | --- |
| **Signals** | **IO** | **Description** |
| acjt\_level | I | 1149.6 Receiver Sensitivity Level Control. Cfg by register |
| bs\_capture\_dr | I | Boundary Scan CaptureDR Signal. From usbmix top input |
| bs\_clamp | I | Boundary Scan Clamp Signal. From usbmix top input |
| bs\_clk | I | Boundary Scan clock. From usbmix top input |
| bs\_extest | I | Boundary Scan EXTEST Signal. From usbmix top input |
| bs\_extest\_ac | I | Boundary Scan AC EXTEST Signal. From usbmix top input |
| bs\_highz | I | Boundary Scan High-Impedance Signal. From usbmix top input |
| bs\_in | I | Boundary Scan Chain Input |
| bs\_init | I | Boundary Scan Initialize Signal. From usbmix top input |
| bs\_invert | I | Boundary Scan Invert Signal. From usbmix top input |
| bs\_preload | I | Boundary Scan Preload Signal. From usbmix top input |
| bs\_shift\_dr | I | Boundary Scan ShiftDR Signal. From usbmix top input |
| bs\_update\_dr | I | Boundary Scan UpdateDR Signal. From usbmix top input |
| bs\_out | O | Boundary Scan Chain Output |

#### BIST Capabilities

The USB 3.0 PHY provides a lot of BIST capabilities on the SS function. For more details, refer to [1].

The USB 3.0 PHY provides a lot of BIST capabilities on the HS function. For more details, refer to [1].

ATE testing using the USB 3.0 PHY’s BIST function is through the JTAG port or the 16-bit parallel interfaceof the PHY.

##### Guidelines for Running BIST

To ensure that BIST operates correctly, follow these guidelines.

■ Do not connect any capacitive loads or resistive terminations on the D+ and D– lines.  
■ Use one of the following termination options while running BIST:  
❑Run BIST in two stages, so that HS BIST is run with 45-Ω termination-to-ground, while FS and LSBIST are run with no capacitive loads or resistive terminations on the D+ and D– lines.  
❑Use a 130-Ω termination-to-ground for all three BIST modes. Note that this value might need tobe adjusted, because it is dependent on the length of the load-board traces.

##### Diagnostic Features

In addition to the ATE capability, the USB 3.0 PHY can use the JTAG port to extract an eye diagram of thesignal at the input of the receiver. This feature is helpful for debug or initial system power-up.

#### Scan

The PHY provides a Scan interface for implementing scan testing in both the hard macro PHY and the softPCS layer.

An ATPG model of the digital portion of the PHY is provided (in .ctl, .ctldb, .spf formats of directory /ux/m850D/INPUT/nxp/IP\_USB3/0819/USB3/usb3\_sspx1\_hspx1/scan/dft\_model), which you canuse during ATPG pattern generation to stitch the scan chains together and test them during chip-level scantesting.

**NOTE:** Scan support in the PHY includes only stuck-at-fault testing. At-speed testing is performedthrough BIST and ATE testing. The maximum scan test frequency is 100 MHz.

##### Scan Configuration

To enable scan testing, set the following signals:  
■ test\_burnin = 1'b0  
■ test\_powerdown\_ssp = 1'b0  
■ test\_powerdown\_hsp = 1'b0  
■ scan\_mode = 1'b1  
In Scan mode, all analog blocks are powered down. All digital outputs to the analog block are looped backto the digital portion to provide coverage on all analog/digital interfaces. When running scan, the vph andvptxN supplies are not required; only the analog vp supply is required.

##### PHY Scan Details

The PHY provides a single scan\_clk scan clock input. All clocks used in the PHY (either top-level inputs orinternally generated) are multiplexed with scan\_clk. The maximum scan\_clk frequency for both Shift andCapture modes is 50 MHz.

All output clocks from the PHY have the scan\_clk signal multiplexed onto them and do not require anadditional scan MUX on the output.

All resets used in the PHY (either top-level inputs or internally generated) are controlled in Scan mode byscan\_rst and phy\_reset.

The SuperSpeed and high-speed portions of the PHY contain a total of 40 scan chains.

The high-speed portion of the PHY contains a single negative-edge-triggered scan chain.

All scan chains terminate with lock-up latches with one exception. For the negative-edge-triggered scanchain, the PHY provides an additional non-lock-up latch output that does not contain an output lock-uplatch.

The PHY .lib files include all appropriate timing arcs to support Shift and Stuck-at Capture modes.

##### PCS Layer Scan Details

The PHY includes separate scan clock and reset inputs for each PCS layer clock domain to support at-speedscan testing of the PCS layer.

The following figure outlines the PCS scan clocking architecture.

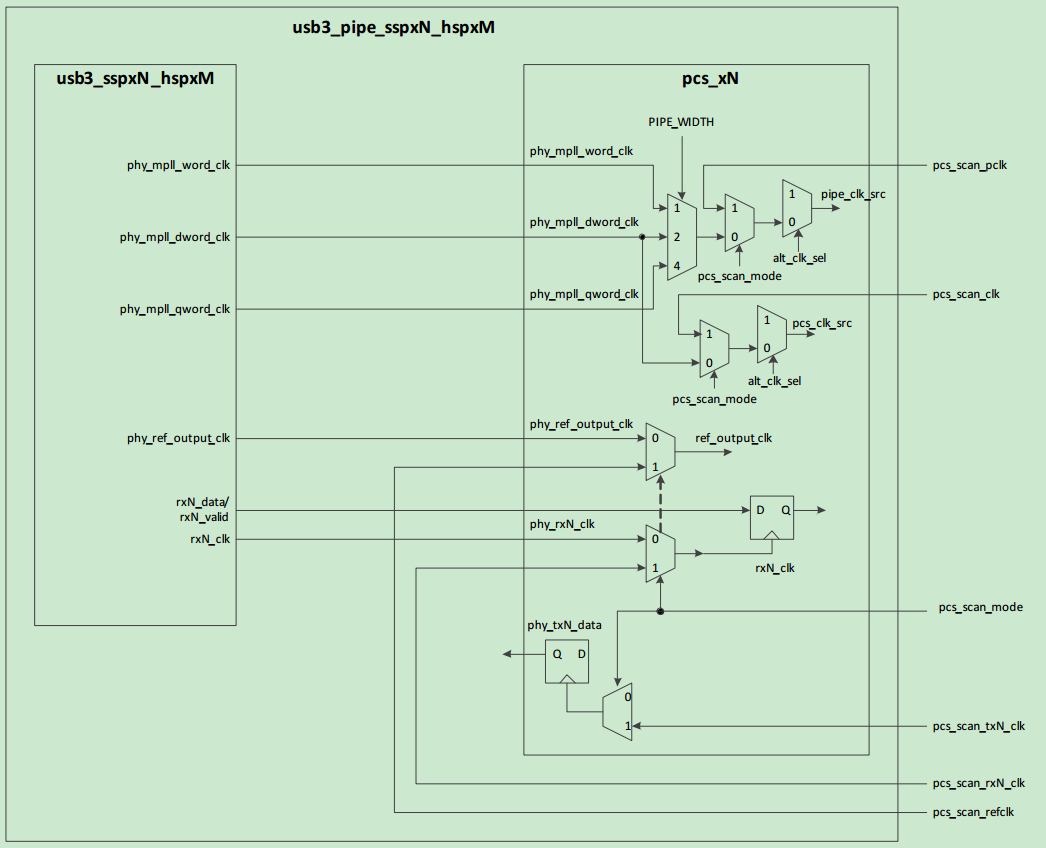
****

Figure 1PCS Scan Clocking Architecture

To support DFT insertion flows that do not include the PHY (that is, when running DFT insertion with thecontroller/subsystem and excluding the PHY), all primary input clocks to the PCS contain scan MUXesdespite that fact that all output clocks in the PHY already have scan multiplexed onto them.

**NOTE**: In Scan mode, the analog portion of the PHY is powered down; therefore, all at-speed scanclocks must be provided to the PCS layer from external clock sources.

To facilitate at-speed scan testing and to ensure coverage of the interface between the PHY and controller,you should connect the pcs\_scan\_mode, pcs\_scan\_shift, and pcs\_scan\_rst signals to the appropriatescan\_mode, scan\_shift, and scan\_rst signals in the controller.The following table lists the at-speed scan clocks and their target clock frequencies.

Table 1At-Speed Scan Clocks and Target Clock Frequencies

|  |  |
| --- | --- |
| **PCS Layer Input** | **PHY Input Clock Frequency (MHz)** |
| pcs\_scan\_clk | 250 |
| pcs\_scan\_pclk | 125 (32-bit PIPE interface) |
| pcs\_scan\_refclk | 100 |
| pcs\_scan\_rx<#>\_clk | 250 |
| pcs\_scan\_tx<#>\_clk | 250 |

To maximize coverage across all interfaces when implementing stuck-at fault testing only, tie all scan clocks(on the PHY, soft PCS, and controller) together to a single clock source.

##### ATPG Vectors

When running scan vector simulations, use the full gate-level netlist. The scan netlist provided in thedeliverables includes a scan model of the analog portion of the design only. When running scan vectorsimulations, use the full behavioral analog model within the gate-level netlist.

In addition, to prevent errors in the analog model when running scan vector simulations, define theDWC\_TETRAMAX define.

### Using 1149.6 AC JTAG Boundary Scan

The SuperSpeed serial link differential pairs for the USB 3.0 PHY include AC coupling capacitors that aresoldered on the board in series with the high-speed signals. To test for proper soldering of these externalcomponents on the board, the PHY supports IEEE standard 1149.6 AC JTAG Boundary Scan. As shown inthe following figure, the USB 3.0 PHY boundary scan includes its own I/O logic module (IOLM). This cell  
includes the boundary shift register and primary logic to support I/O testing of the differential transmitterand receiver pins.

PINMUX TABLE