**This procedure enables ODI through the Avalon-MM interface, allowing you to view the eye.**

1. Disable the background calibration if the device is H-Tile production and background calibration is enabled:
   1. Set 0x542[0] to 0x0
   2. Read 0x481[2] until it becomes 0x0
2. Request access to the Native PHY’s AVMM bus:
   1. Set 0x000 to 0x02
   2. Read 0x481[2] until it becomes 0x0
3. If the RX adaptation is manual mode (footnote 1), set 0x148[0] to 0x1. Otherwise skip this step.
4. Set 0x169[6] to 0x1 to enable the counter to detect error bits.
5. Set 0x168[0] to 0x1 to enable the serial bit detector for ODI.
6. If the DFE is enabled (footnote 2):
   1. Set 0x169[2] to 0x1 to enable DFE speculation.
   2. Set 0x149[5:0] to 0x07 to read the DFE tap signs.
   3. Read register 0x17F[6] and store it as DFE\_tap1\_sign.
7. If the DFE is disabled(footnote 2):
   1. Set register 0x169[2] to 0x0 to disable DFE speculation.
8. Users can tradeoff between the ODI run time and BER resolution by setting the number of bits to count before stopping at each horizontal/vertical point combination. Set {0x169[1:0],0x168[5]} to:
   1. Count up to 2^16: 0x0
   2. Count up to 10^6: 0x1
   3. Count up to 10^7: 0x2
   4. Count up to 10^8: 0x3
   5. Count up to 3 x 10^8: 0x4
   6. Count up to 10^9: 0x5
   7. Count up to 2^32: 0x6
9. Set register 0x158[5] to 0x1 so the user has control of the serial bit comparator.
10. Set register 0x12D[4] to 0x0 to disable the path from the DFE to the AVMM testmux.
11. If the device is H-Tile production, configure the ODI bandwidth for the datarate by setting register {0x145[7],0x144[7]} to:

|  |  |
| --- | --- |
| Datarate | Setting |
| > 25Gbps | 0x0 |
| > 16Gbps to 25Gbps | 0x1 |
| > 10Gbps to 16Gbps | 0x2 |
| =< 10Gbps | 0x3 |

1. If the device is not H-Tile production, configure the ODI bandwidth for the datarate by setting register {0x145[7],0x144[7]} to:

|  |  |
| --- | --- |
| Datarate | Setting |
| > 20Gbps | 0x0 |
| > 12.5Gbps to 20Gbps | 0x1 |
| > 6.5Gbps to 12.5Gbps | 0x2 |
| =< 6.5Gbps | 0x3 |

1. Set register 0x144[6:4] to 0x0 to set the ODI phase interpolator to 128 steps.
2. Set register 0x140[5:3] to 0x0 to disable the ODI test pattern generator.
3. Reset the serial bit comparator control logic by setting 0x13C[0]to 0x0.
4. Release reset on the serial bit comparator control logic by setting 0x13C[0] to 0x1.
5. Set register 0x171[4:1] to 0xB to configure the AVMM testmux to read out the ODI counter values.

To save time, you can sweep the horizontal eye opening across 128 phase steps with  
the vertical phase set to zero. This helps determine the extent of the eye opening.  
Then, you can perform a refined horizontal or vertical eye sweep to capture the two dimensional eye diagram.

**The following are the steps to scan the horizontal opening**

1. Set register 0x144[2] to 0x0 and 0x156[0] to 0x0 to capture the zero crossing for the bottom half of the eye.
2. Set register 0x143[7:2] to 0x0 to set the vertical step to 0.
3. If the DFE is disabled (footnote 2), set register 0x14D[0] to 0x0.
4. If the DFE is enabled (footnote 2), set register 0x14D[**0**] to 0x1 to select positive speculation and:
   1. If DFE\_tap1\_sign = 0, set 0x156[1] to 0x1.
   2. If DFE\_tap1\_sign = 1, set 0x156[1] to 0x0.
5. If the targeted device is H-Tile production or H-Tile ES3:
   1. Set register 0x100[4] to 0x1.
   2. Set register 0x0[7:0] to 0x01 to request PreSICE to calibrate the ODI samplers.
   3. Read 0x481[2] until it become 0x0.
6. Set register 0x171[4:1] to 0xB to configure the AVMM testmux
7. Define a floating point array with 128 members called ODI\_error\_count and set all the values to 0.
8. Set register 0x157[3:2] to 0x2 to capture the odd eye. Repeat steps **9** after setting register 0x157[3:2] to 0x1 to capture the even eye.
9. Create an integer called horizontal\_phase and set it to 1. Repeat steps **10** to **23** while incrementing horizontal\_phase until it becomes 128.
10. Set register 0x145[6:0] to the encoded phase in table 95 (for example, 0x71 for horizontal\_phase=1).
11. Set 0x168[2] to 0x0 to reset the serial bit counter.
12. Set 0x168[2] to 0x1 to release reset on the serial bit counter.
13. Set 0x149[5:0] to 0x1C to be able to read the ODI status.
14. Read 0x17E[1] until it becomes 0x1 to indicate the ODI has received the selected number of bits and has completed.
15. Set 0x149[5:0] to 0x1B to be able to read out the number of ODI error bits.
16. Read register 0x17E[7:0] and save it as an integer ODI\_count\_A.
17. Set 0x149[5:0] to 0x1A to be able to read out the number of ODI error bits.
18. Read register 0x17E[7:0] and save it as an integer ODI\_count\_B.
19. Set 0x149[5:0] to 0x19 to be able to read out the number of ODI error bits.
20. Read register 0x17E[7:0] and save it as an integer ODI\_count\_C.
21. Set 0x149[5:0] to 0x18 to be able to read out the number of ODI error bits.
22. Read register 0x17E[7:0] and save it as an integer ODI\_count\_D.
23. ODI\_error\_count[horizontal\_phase] = ODI\_count\_A \* 2^24 + ODI\_count\_B \* 2^16 + ODI\_count\_C \* 2^8 + ODI\_count\_D + ODI\_error\_count[horizontal\_phase]
    1. If the device is not H-Tile production, the ODI\_error\_count may be greater than the actual count by 1.
24. Scan through the ODI\_error\_count array and find the phases that have no errors and determine the left eye opening and right eye opening. Store the phases as left\_phase and right\_phase.

**To scan both the horizontal and vertical phases:**

Sweep the horizontal and vertical phases to get a two-dimensional eye diagram.

Note: The horizontal phase steps (left\_phase and right\_phase) correspond to zero BER  
in the previous sweep. The phase steps with no BER may wrap around, for example,  
from phase 110 to phase 20.

1. Create a 130\*130 2D floating point array called ODI\_error\_count and initialize it to be 0.
2. Create a 130\*130 2D floating point array called ODI\_pattern\_count and initialize it to be 0.
3. If DFE is disabled (footnote 2), repeat steps **5** to **36** two times:
   1. In the first iteration, set 0x144[2] to 0x0 and 0x14D[0] to 0x0.
   2. In the second iteration, set 0x144[2] to 0x1 and 0x14D[0] to 0x1.
4. If the DFE is enabled (footnote 2), repeat steps **5** to **36** four times:
   1. In the first iteration, set 0x144[2] to 0x0 and 0x14D[0] to 0x0.
      1. If DFE\_tap1\_sign = 0, set 0x156[2] to 0x0.
      2. If DFE\_tap1\_sign = 1, set 0x156[2] to 0x1.
   2. In the first iteration, set 0x144[2] to 0x0 and 0x14D[0] to 0x1.
      1. If DFE\_tap1\_sign = 0, set 0x156[2] to 0x1.
      2. If DFE\_tap1\_sign = 1, set 0x156[2] to 0x0.
   3. In the first iteration, set 0x144[2] to 0x1 and 0x14D[0] to 0x0.
      1. If DFE\_tap1\_sign = 0, set 0x156[2] to 0x0.
      2. If DFE\_tap1\_sign = 1, set 0x156[2] to 0x1.
   4. In the first iteration, set 0x144[2] to 0x1 and 0x14D[0] to 0x1.
      1. If DFE\_tap1\_sign = 0, set 0x156[2] to 0x1.
      2. If DFE\_tap1\_sign = 1, set 0x156[2] to 0x0.
5. If the targeted device is H-Tile production or H-Tile ES3:
   1. Set register 0x100[4] to 0x1.
   2. Set register 0x0[7:0] to 0x01 to request PreSICE to calibrate the ODI samplers.
   3. Read 0x481[2] until it becomes 0x0.
6. Set register 0x171[4:1] to 0xB to configure the AVMM testmux.
7. Set register 0x157[3:2] to 0x2 to capture the odd eye. Repeat step **8** after setting 0x157[3:2] to 0x1 to capture the even eye.
8. Set integer variable vertical\_phase to 0 and repeat steps **9** to **11** while incrementing vertical\_phase until it reaches 126.
9. If the vertical phase < 0x3F:
   1. set 0x156[0] to 0x1 to capture the top half of the eye.
   2. Set register 0x143[7:2] to 0x3F – vertical phase.
10. If the vertical phase >= 0x3F:
    1. set 0x156[0] to 0x0 to capture the bottom half of the eye.
    2. Set register 0x143[7:2] to vertical phase – 0x3F.
11. Set an integer horizontal\_phase to be left\_phase – 10 and repeat steps **12** to **36** while incrementing horizontal\_phase until it reaches right\_phase + 10.
    1. If right\_phase < left\_phase, ie the eye is wrapped around, then increment right\_phase by 0x80.
12. If horizontal\_phase is < 1, then increment it by 0x80.
13. If horizontal\_phase > 128, then decrement it by 0x80.
14. Set register 0x145[6:0] to the encoded horizontal\_phase.
15. Set 0x168[2] to 0x0 to reset the serial bit counter.
16. Set 0x168[2] to 0x1 to release reset on the serial bit counter.
17. Set 0x149[5:0] to 0x1C to be able to read the ODI status.
18. Read 0x17E[1] until it becomes 0x1 to indicate the ODI has received the selected number of bits and has completed.
19. Set 0x149[5:0] to 0x1B to be able to read out the number of ODI error bits.
20. Read register 0x17E[7:0] and save it as an integer ODI\_count\_A.
21. Set 0x149[5:0] to 0x1A to be able to read out the number of ODI error bits.
22. Read register 0x17E[7:0] and save it as an integer ODI\_count\_B.
23. Set 0x149[5:0] to 0x19 to be able to read out the number of ODI error bits.
24. Read register 0x17E[7:0] and save it as an integer ODI\_count\_C.
25. Set 0x149[5:0] to 0x18 to be able to read out the number of ODI error bits.
26. Read register 0x17E[7:0] and save it as an integer ODI\_count\_D.
27. ODI\_error\_count[horizontal\_phase][vertical\_phase] = ODI\_count\_A \* 2^24 + ODI\_count\_B \* 2^16 + ODI\_count\_C \* 2^8 + ODI\_count\_D ~~+~~ ODI\_error\_count[horizontal\_phase][vertical\_phase]
    1. If the device is not H-Tile production, the ODI\_error\_count may be greater than the actual count by 1.
28. Set 0x149[5:0] to 0x17 to be able to read out the number of ODI **pattern** bits.
29. Read register 0x17E[7:0] and save it as an integer ODI\_pattern\_A.
30. Set 0x149[5:0] to 0x16 to be able to read out the number of ODI **pattern** bits.
31. Read register 0x17E[7:0] and save it as an integer ODI\_pattern\_B.
32. Set 0x149[5:0] to 0x15 to be able to read out the number of ODI **pattern** bits.
33. Read register 0x17E[7:0] and save it as an integer ODI\_pattern\_C.
34. Set 0x149[5:0] to 0x14 to be able to read out the number of ODI **pattern** bits.
35. Read register 0x17E[7:0] and save it as an integer ODI\_pattern\_D.
36. ODI\_pattern\_count[horizontal\_phase][vertical\_phase] = ODI\_pattern\_A \* 2^24 + ODI\_pattern\_B \* 2^16 + ODI\_pattern\_C \* 2^8 + ODI\_pattern\_D + ODI\_pattern\_count[horizontal\_phase][vertical\_phase]
37. The BER at the horizontal phase and vertical phase = ODI\_error\_count[horizontal\_phase][vertical\_phase]/ODI\_Pattern\_count[horizontal\_phase][vertical\_phase]

**How to disable the ODI:**

After ODI is complete, disable ODI with the following procedure.

1. Set register 0x168[0] to 0x0.

2. Set register 0x158[5] to 0x0 to disable ODI control from the AVMM interface.

3. If the RX adaptation mode is manual (footnote 1), set 0x148[0] to 0x0 to disable the adaptation

logic and save power.

4. Enable the background calibration if the device is H-Tile production and users want to enable background calibration:

* 1. Set 0x542[0] to 0x1

=================================================

**(footnote 1)** To determine RX adaptation mode, read bit[5] of register 0x161.

RX Adaptation is in Manual Mode when 0x161[5] = 1

**(footnote 2)** To determine DFE mode, read bit[6] of register 0x161.

DFE is disabled when 0x161[6] = 1