Serdes Diag User Guide

User Guide

Applies to Product Release: 01.00.00.07
Publication Date: June, 2017

Texas Instruments, Incorporated
20450 Century Boulevard
Germantown, MD 20874 USA

Contents

This tool demonstrates how to configure and use the SERDES Diag APIs on KeyStone II devices. The Diag APIs currently support C66 architecture. There are 3 tests included in the package:

BER Test: The example finds out the **optimal TX coefficients** (CM, C1, C2) by performing Bit Error Rate (BER) sweeps on the Serdes where both EVM0 and EVM1 sends the transmit pattern (for example PRBS 31) and both EVM0 & EVM1 detects the sequence and performs BER calculations across the desired serdes using the CPU. The BER plot can be viewed using the BERDiagramViewer.jar program.

Note: BER plot doesn't support PCIe Serdes. PCIe serdes BER results can be viewed in the CCS output log.

EYE Test: The example performs on chip **eye measurement** allowing the user to "see" the eye opening of the receive data after equalization. Both EVM0 and EVM1 sends the transmit pattern (for example PRBS 31) and both EVM0 & EVM1 detects the sequence and performs EYE measurements across the desired serdes using the CPU. The EYE plot can be viewed using the EYEDiagramViewer.jar program.

Note: EYE test doesn't support PCIe Serdes.

PRBS Test: The example finds out the **optimal RX coefficients** (Attenuation and Boost) by sending a PRBS 31 pattern and training the receiver to get the best RX coefficients. The PRBS adaptation implementation is required for all interfaces using 5Gbps or higher data rates. The output log displays these values.

Note: PRBS test doesn't support PCIe Serdes.

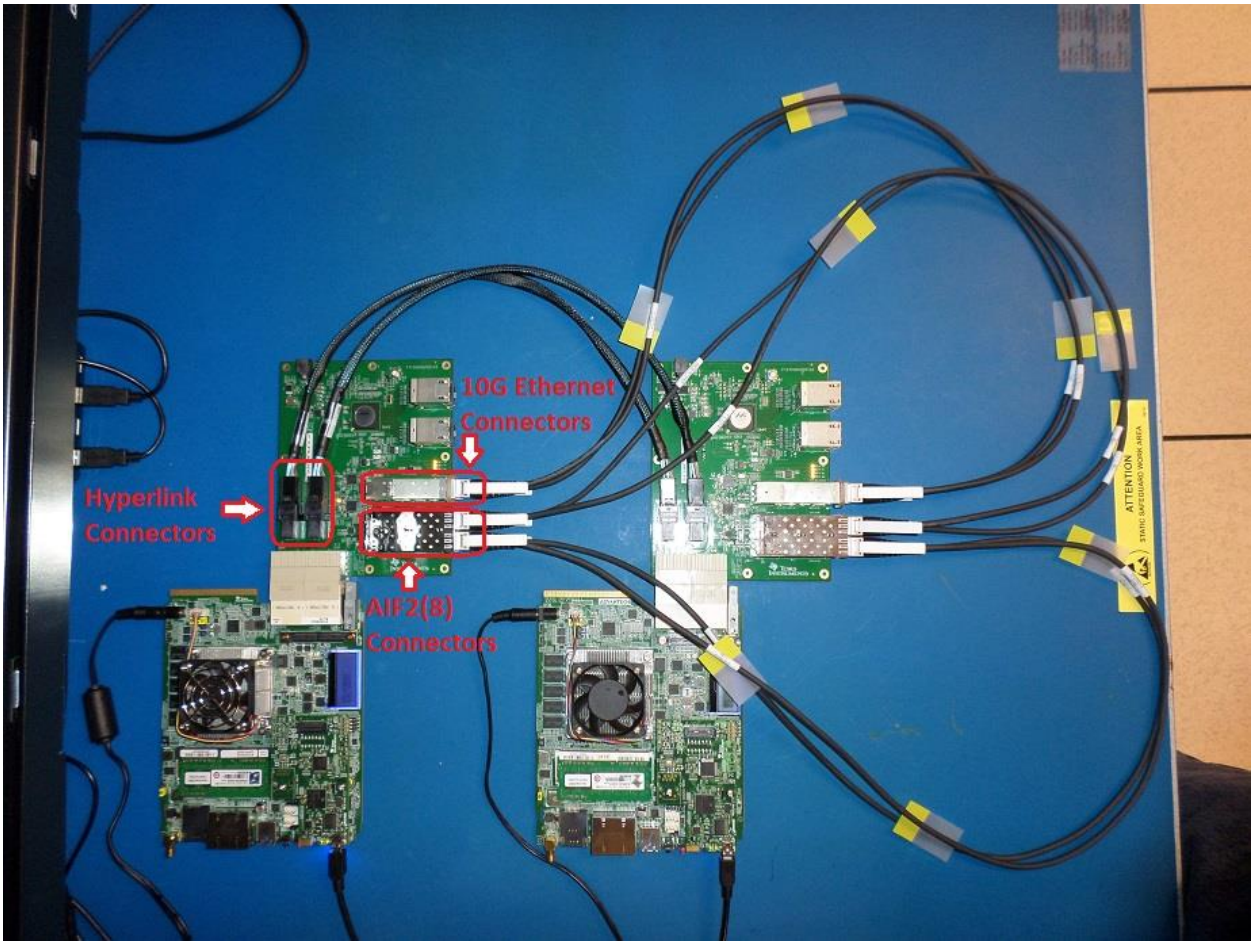
Table to determine breakout card needed for specific interfaces supported by EVM for Serdes Diag testing:

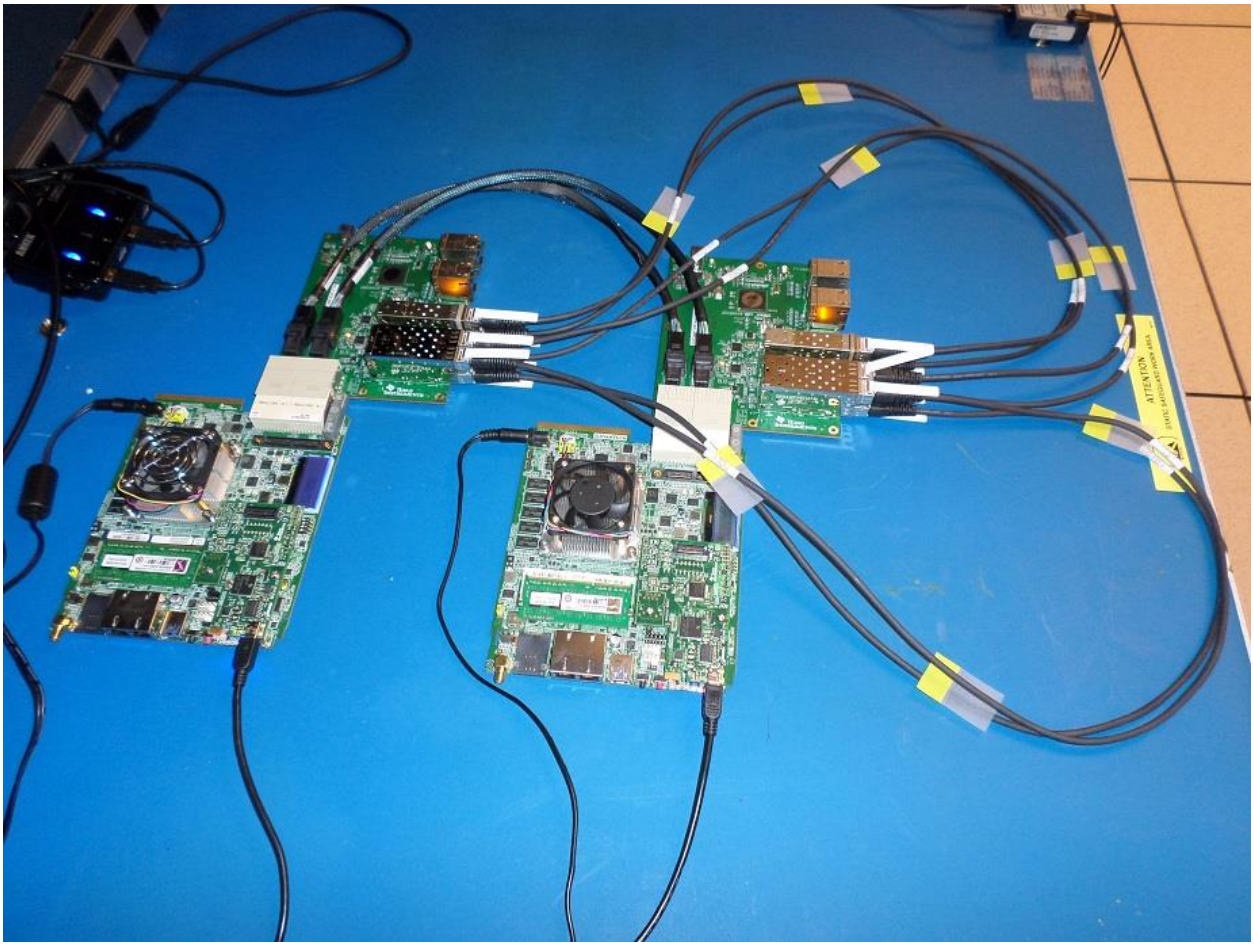
EVM Interface and Break-Out card to use (N/A = Interface not available)							
	PCIE	**Hyperlink**	**SRIO**	**AIF2(4)**	**SGMII**	**10G ETHERNET**	**AIF2(8)**
K2HK	DUAL-BOC	RTM-BOC	DUAL-BOC	DUAL-BOC	DUAL-BOC	RTM-BOC	RTM-BOC
K2E	DUAL-BOC	RTM-BOC	N/A	N/A	DUAL-BOC	RTM-BOC	N/A
K2L	DUAL-BOC	N/A	N/A	N/A	N/A	N/A	N/A

1. Example setup:

- 2 K2HK or K2E or K2L EVMs (Diag APIs support all K2K, K2H, K2L and K2E platforms)
- 2 RTM Break Out Cards (BOCs)
- 2 AMC Break Out Cards
- For Hyperlink test: 1 HyperLink Cable connected from Port 0 of EVM0 to Port 0 of EVM1 (via BMC)
- For 10GE test: 1 10GE SFP Cable connected from Port 0 of EVM0 to Port 0 of EVM1 (via BMC)
- For SGMII test: SGMII board-board tests cannot be run on EVM since BER pattern cannot be transmitted over Copper cables. SGMII tests will be single board internal loopback only.
- For AIF2 test: 1 AIF2 SFP Cable connected from Port 0 of EVM0 to Port 0 of EVM1 (via AMC)
- 2 Texas Instruments XDS2xx USB Onboard JTAG Emulators connected to a single CCS Host Machine
- Both EVMs are configured in No Boot/Sleep Mode.

Here’s an example showing 2 K2HK EVMs with 2 RTM BOCs connected using 10GE, Hyperlink and AIF2 B8 cables.





2. Tools included (which can be configurable):

- serdes_dss_ber_**k2e/k2l/k2h/h2k**.bat -> Invokes serdes_dss_ber_**k2e/k2l/k2h/h2k**.js to launch DSS and run BER test
- serdes_dss_ber_**k2e/k2l/k2h/h2k**.js -> DSS script for running BER test
- serdes_dss_eye_**k2e/k2l/k2h/h2k**.bat -> Invokes serdes_dss_eye_**k2e/k2l/k2h/h2k**.js to launch DSS and run EYE test
- serdes_dss_eye_**k2e/k2l/k2h/h2k**.js -> DSS script for running EYE test
- serdes_dss_prbs_**k2e/k2l/k2h/h2k**.bat -> Invokes serdes_dss_prbs_**k2e/k2l/k2h/h2k**.js to launch DSS and run PRBS test
- serdes_dss_prbs_**k2e/k2l/k2h/h2k**.js -> DSS script for running PRBS test
- serdes_diag.h -> Provides APIs for Serdes Diag

3. Prerequisites:

- Running the serdes_dss_xxx.bat requires a dual board target configuration file (ccxml). Instructions on how to create it can be found in the section below. The serdes_dss_xxx.js file should be updated to reference the location of this newly created ccxml.

- Running the serdes_dss_xxx.bat requires that the serdes_diag_xxx_K2XC66ExampleProject project be built and also generate an output .out file. The path where the .out is located needs to be reflected in the serdes_dss_ber.js
- Each platform (K2K, K2E, K2L, K2H) has a CCS example project associated with it. The parameters that can/should be modified depend on the user's board setup. The configurable parameters are listed below. **Generating the .out files is a prerequisite**.

4. GIT Access:

- The serdes diag code is also available on TI GIT.
- git clone git://git.ti.com/keystone-rtos/serdes_diag.git
- mkdir diag under existing pdk_keystone2_3_01_xx_xx/packages/ti directory.
- Copy the serdes_diag folder into the pdk_keystone2_3_01_xx_xx/packages/ti/diag directory.

5. Example Software Flow & Configurability

- serdes_diag_test_main.c contains the main function. It initializes all the serdes peripherals using the serdes_diag_test_init( ) API. The **serdes_diag_test_phy_type** should be specified in serdes_diag_platform.h in order to run the BER/EYE test for that specific SERDES.
- Each platform contains its own serdes_diag_platform.h under test\k2x\c66\bios directory. All the serdes initialization variables should be set here in this file. The user can modify it according to their needs.
- serdes_diag_test.c contains the APIs which call the BER utility. Configurable parameters are listed.

6. Steps to create CCS example projects for Serdes Diag

- Once the PDK is installed, the serdes example projects can be created by running the **pdkProjectCreate.bat** found at the top level of pdk_xxxx\packages folder. Examples on how to create projects for each platform and each modules are shown in the pdkProjectCreate.bat. Instructions can also be found here: http://processors.wiki.ti.com/index.php/Rebuilding_The_PDK#PDK_Example_and_Test_Project_Creation
  Example for creating K2K Serdes Project:
  pdkProjectCreate.bat K2K all little serdes_diag dsp "C:\ti\pdk_<soc>_<version>\packages"

7. Steps to run the Board-Board BER Diag Example using DSS:

- Step 1:
  Import the serdes_diag_ber_K2XC66ExampleProject into CCS and build it. serdes_diag_test_phy_type variable is set to SERDES_HYPERLINK by default. The configurable parameters should be set in serdes_diag_platform.h according to the PHY type selected.

- Step 2:
  Run serdes_dss_ber_k2k/k2e/k2h/k2l.bat to setup the PDK_INSTALL_PATH (This variable should be pointed to the user's local PDK directory) and launching the DSS session to run the BER example. The testcase is driven by the DSS script, serdes_dss_ber_k2k/k2e/k2h/k2l.js.

From a windows machine, you can run the script directly from the command window as below:

- serdes_dss_ber_k2k/k2e/k2h/k2l.bat

The testcase uses CCSv6 by default. The serdes_dss_ber_k2k/k2e/k2h/k2l.bat can be modified to point to CCSv5 as well.

For example: C:\ti**ccsv5**\ccs_base\scripting\bin\dss.bat can be used in the serdes_dss_ber_k2k/k2e/k2h/k2l.bat file instead of ccsv6.

- Step 3:
  The DSS uses synchronization between devices using flags. When the dss_test_complete_flag is seen as 0x5A5A5A5A, it is still running the test. The test duration (ber_init_params.bertesttime = 1000000000; \\1 sec) mentioned in serdes_diag_test.c is the duration of the test per iteration. Depending on the CM, C1, C2 sweep iteration range (start and end values set in serdes_diag_test.c), it can take atleast bertesttime x CM x C1 x C2 x 1/DSP freq.
  For example, if bertesttime is 10000000000 cycles, CM, C1 and C2 range is 0-4 each and DSP freq is 1 GHz, the total test time would be 10000000000 x 4 x 4 x 4 x (1/1000000000) = atleast 640 seconds to finish the test.

```
Turning off L1 Data Cache.
Turning off L1 Data Cache.
Turning off L2 Cache.
Turning off L2 Cache.
AIF2 Serdes Init Complete
AIF2 Serdes Init Complete
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
```

- Step 4:
  Once the DSS session shows Serdes Diag Test Complete (dss_test_complete_flag becomes 0xA5A5A5A5) as shown below, the final output log .txt files for both devices can be found at the same location as the .out directory.

  Example: packages\MyExampleProjects\serdes_diag_ber_K2KC66ExampleProject\Debug\ soc_device0_ber_scanout.txt

```
Saving results to file: soc_device0_ber_scanout.txt
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
Saving results to file: soc_device1_ber_scanout.txt


dss_test_complete_flag_device0: 0xA5A5A5A5
dss_test_complete_flag_device1: 0xA5A5A5A5
Serdes Diag Test Complete
```

- Step 5:
  Once the .txt files are generated, use the BER Diagram Viewer tool (located at serdes_diag/tools/ber_diagram_tool) explained in Section 10 to view the BER plot.

a. Configurability:

- serdes_dss_ber_k2k/k2e/k2h/k2l.js
  - Path to the ccxml file location
  - Path to the .out files
  - Type of emulators connected to device0 and device1
  - Additional GEL scripting to program PLL and DDR initialization can be added.

- serdes_diag_platform.h contains platform specific settings
  - Ref clock configurations
  - Link Rates
  - Number of Lanes, lane mask
  - Additional serdes phy types can be added to the existing setup if needed.

- serdes_diag_test.c
  - Serdes TX Coefficients sweep range (CM, C1 and C2)
  - Serdes TX Voltage Regulator range and TX Attenuator Range.
  - Number of lanes
  - Test Time
  - Time out Duration
  - PRBS Test Pattern
  - Loopback type (Non Loopback, FEP Loopback)

8. Steps to run the Board-Board EYE Diagram Example using DSS:

- Step 1:
  Import the serdes_diag_eye_K2XC66ExampleProject into CCS and build it. serdes_diag_test_phy_type variable is set to SERDES_HYPERLINK by default. The configurable parameters should be set in serdes_diag_platform.h according to the PHY type selected.

- Step 2:

Run serdes_dss_eye_k2x.bat to setup the PDK_INSTALL_PATH (This variable should be pointed to the user's local PDK directory) and launching the DSS session to run the EYE example. The testcase is driven by the DSS script, serdes_dss_eye_k2x.js.

From a windows machine, you can run the script directly from the command window as below:

- serdes_dss_eye_k2x.bat

The testcase uses CCSv6 by default. The serdes_dss_eye_k2x.bat can be modified to point to CCSv5 as well.

For example: C:\ti**ccsv5**\ccs_base\scripting\bin\dss.bat can be used in the serdes_dss_eye.bat file instead of ccsv6.

```
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
Finished Serdes_Diag_EyeMonitor for lane0!
Beginning Serdes_Diag_EyeMonitor for lane1...
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
Finished Serdes_Diag_EyeMonitor for lane0!
Beginning Serdes_Diag_EyeMonitor for lane1...
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
```

- Step 3:
  Once the DSS session shows Serdes Diag Test Complete, the final output log .txt files for both devices can be found at the same location as the .out directory.

  Example: packages\MyExampleProjects\serdes_diag_eye_K2KC66ExampleProject\Debug\ soc_device0_eye_scanout_lane0.txt

```
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
Finished Serdes_Diag_EyeMonitor for lane1!
Eye Scans complete!
dss_test_complete_flag_device0: 0xA5A5A5A5
dss_test_complete_flag_device1: 0x5A5A5A5A
Serdes Diag Test Complete
```

- Step 4:
  Once the .txt files are generated, use the EYE Diagram Viewer tool (located at serdes_diag/tools/eye_diagram_tool) explained in Section 11 to view the EYE diagram.

a. Configurability:

- serdes_dss_eye_k2k/k2e/k2h/k2l.js
    - Path to the ccxml file location
    - Path to the .out files
    - Type of emulators connected to device0 and device1
    - Additional GEL scripting to program PLL and DDR initialization can be added.

- serdes_diag_platform.h contains platform specific settings
    - Ref clock configurations
    - Link Rates
    - Number of Lanes
    - Additional serdes phy types can be added to the existing setup if needed.

- serdes_diag_test.c
    - Serdes TX Coefficients sweep range (CM, C1 and C2)
    - Serdes TX Voltage Regulator range and TX Attenuator Range.
    - Number of lanes
    - Test Time
    - Time out Duration
    - PRBS Test Pattern

9. Steps to run the Board-Board PRBS Test Example using DSS:

- Step 1:
  Import the serdes_diag_prbs_K2XC66ExampleProject into CCS and build it. serdes_diag_test_phy_type variable is set to SERDES_HYPERLINK by default. The configurable parameters should be set in serdes_diag_platform.h according to the PHY type selected.

- Step 2:
  Run serdes_dss_prbs_k2x.bat to setup the PDK_INSTALL_PATH (This variable should be pointed to the user's local PDK directory) and launching the DSS session to run the PRBS example. The testcase is driven by the DSS script, serdes_dss_prbs_k2x.js.

  From a windows machine, you can run the script directly from the command window as below:

  - serdes_dss_prbs.bat

  The testcase uses CCSv6 by default. The serdes_dss_prbs.bat can be modified to point to CCSv5 as well.

  For example: C:\ti**ccsv5**\ccs_base\scripting\bin\dss.bat can be used in the serdes_dss_prbs_k2x.bat file instead of ccsv6.

- Step 3:
  Once the DSS session shows Serdes Diag Test Complete, the final output log .txt files for both devices can be found at the same location as the .out directory.

Example: packages\MyExampleProjects\serdes_diag_prbs_K2KC66ExampleProject\Debug\
soc_device0_prbs_scanout_lane0.txt

```
dss_test_complete_flag_device1: 0x5A5A5A5A
dss_test_complete_flag_device0: 0x5A5A5A5A
dss_test_complete_flag_device1: 0x5A5A5A5A
Finished Serdes_Diag_EyeMonitor for lane1!
Eye Scans complete!
dss_test_complete_flag_device0: 0xA5A5A5A5
dss_test_complete_flag_device1: 0x5A5A5A5A
Serdes Diag Test Complete
```

- Step 4:
  Once the .txt files are generated, the optimal RX coefficients are in the txt file.

b. Configurability:

- serdes_dss_prbs_k2k/k2e/k2h/k2l.js
  - Path to the ccxml file location
  - Path to the .out files
  - Type of emulators connected to device0 and device1
  - Additional GEL scripting to program PLL and DDR initialization can be added.

- serdes_diag_platform.h contains platform specific settings
  - Ref clock configurations
  - Link Rates
  - Number of Lanes
  - Additional serdes phy types can be added to the existing setup if needed.

- serdes_diag_test.c
  - Serdes TX Coefficients sweep range (CM, C1 and C2)
  - Serdes TX Voltage Regulator range and TX Attenuator Range.
  - Number of lanes
  - Test Time
  - Time out Duration
  - PRBS Test Pattern

10. Steps to run Single Board (Loopback) using CCS:

1. These steps below are applicable for any BER or PRBS or EYE tests for all platforms.
2. Import the serdes_diag_ber/eye/prbs_K2XC66ExampleProject into CCS and build it.
3. Set #define SERDES_DIAG_TEST_LOOPBACK_MODE **CSL_SERDES_LOOPBACK_ENABLED** in serdes_diag\test\k2x\c66\bios\serdes_diag_platform.h
4. If the test is a BER test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_BERTest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c
5. If the test is an EYE test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_EYETest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c

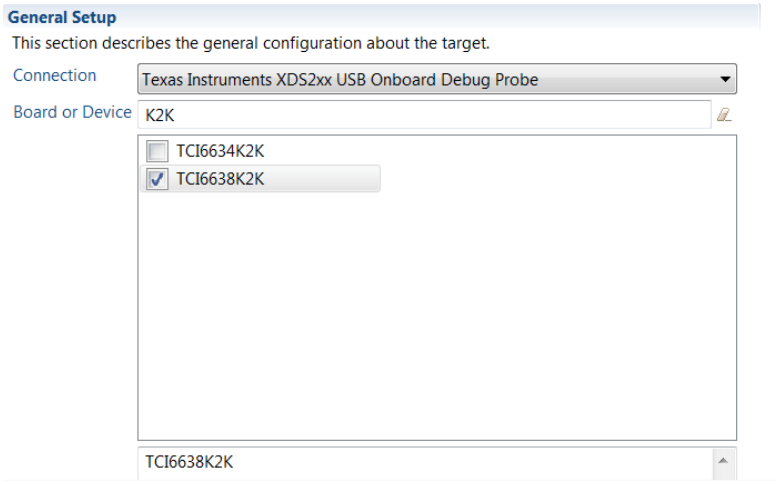
6. If the test is a PRBS test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_PRBSTest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c
7. Other configurable parameters can be set in serdes_diag_platform.h according to the PHY type selected.
8. Load .out in CCS and run

11. Steps to run Single Board (Non-Loopback) connected to external PHY using CCS:

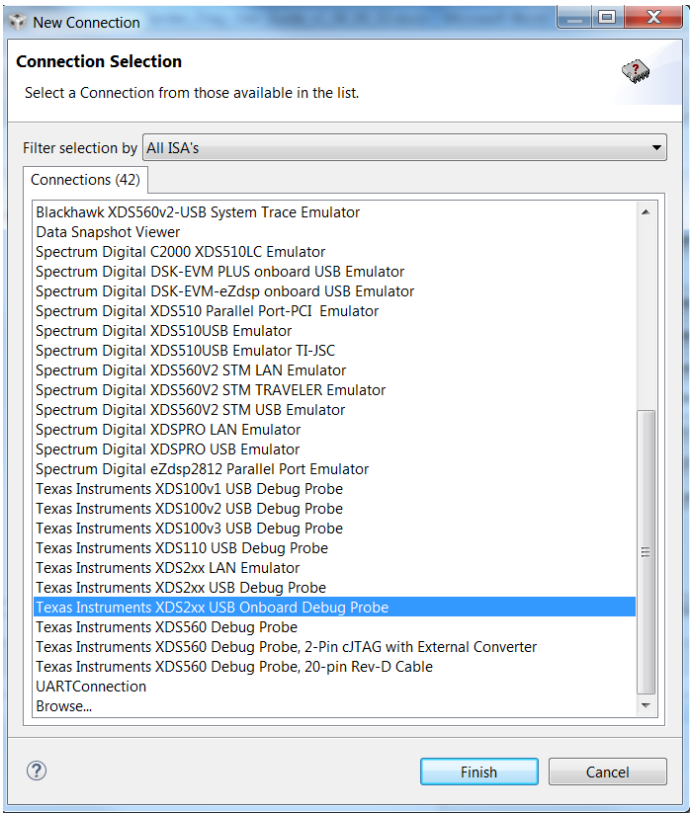
1. These steps below are applicable for any BER or PRBS or EYE tests for all platforms.
2. Import the serdes_diag_ber/eye/prbs_K2XC66ExampleProject into CCS and build it.
3. Set #define SERDES_DIAG_TEST_LOOPBACK_MODE **CSL_SERDES_LOOPBACK_DISABLED** in serdes_diag\test\k2x\c66\bios\serdes_diag_platform.h
4. If the test is a BER test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_BERTest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c
5. If the test is an EYE test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_EYETest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c
6. If the test is a PRBS test, set ber_init_params.loopback_type = **SERDES_DIAG_LOOPBACK**; in Serdes_Example_PRBSTest API in serdes_diag\test\k2x\c66\bios\serdes_diag_test.c
7. Other configurable parameters can be set in serdes_diag_platform.h according to the PHY type selected.
8. Load .out in CCS and run.

12. Creating Dual EVM Target Configuration:

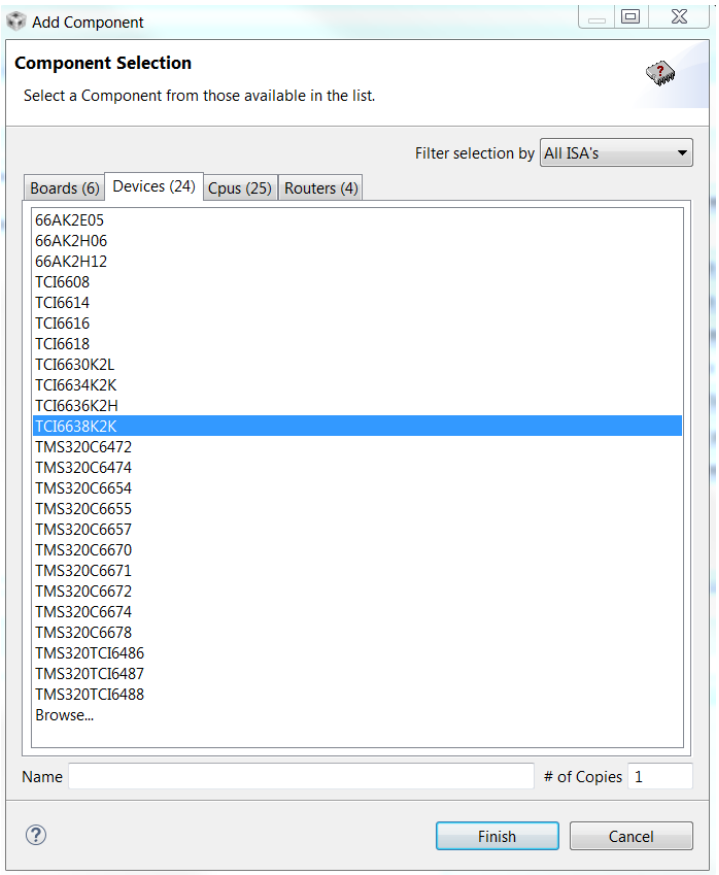
1. Select “new target configuration” in CCS. Select the connection type and the device type depending on the Emulator and the SoC on the target board.



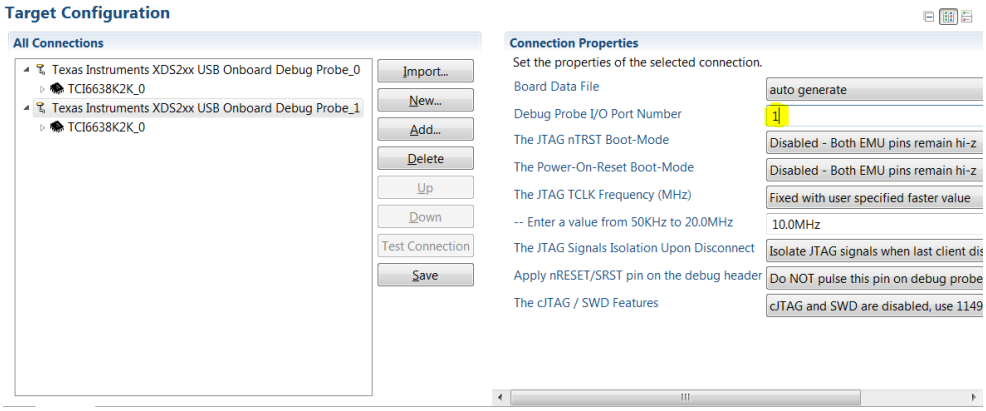
2. Select “save” and click on the Advanced setup “Target Configuration”.
3. Click on “New” to add a new emulator to the existing setup and select “Finish”.



4. Click on the newly added emulator and Select “Add”. Select the appropriate device under the “Devices” tab.



5. Edit the Emulator I/O Port Number to 1 for the second connection as shown in yellow below.



6. Click save and exit.

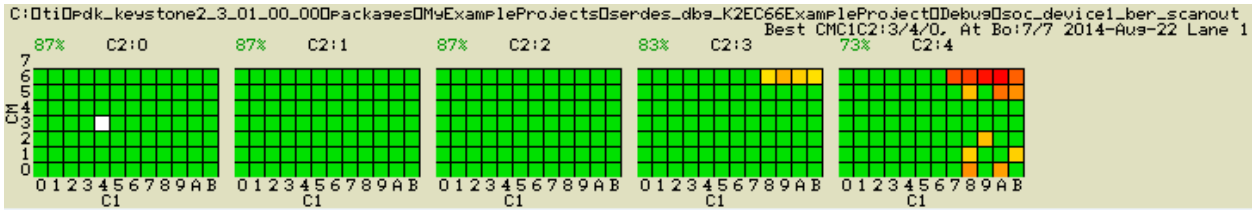
13. BER Diagram Viewer

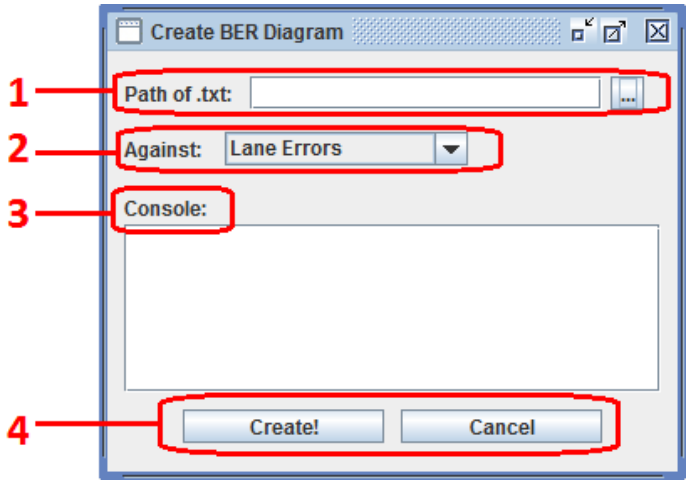
Contents:

BER Diagram Viewer (BerDiagramViewer.jar) is located at serdes_diag/tools/ber_diagram_tool. The purpose of this program is to create a BER diagram from an On-Chip Byte-Error Rate .txt file generated by the Serdes. This program was created as a distributable and interactive interface to quickly generate BER diagrams. This program was written in Java using Eclipse IDE, a free download from https://www.eclipse.org/. This program was compiled with Java version 7 update 60 build 1.7.0_60-b19 using the runtime environment JavaSE-1.6 (jre6). Users must have Java version 7 update 60 or above to run this program, which can be freely downloaded at https://java.com/en/download/manual.jsp. The user is required to have the Java runtime environment installed to run this program.
Note: The program currently supports 32 bit version of Java.

User Interface

1. File Path
    a. Enter the full path of the BER .txt file(s) generated by the Serdes_Diag_BERTest( ) API.
    b. Column order does not matter in the calculations.
2. Dependent variable
    a. Choose to plot against Lane Errors, Attenuation, or Boost
3. Console Window
    a. After the diagram is generated, information about the output file, FIR coefficients, and Attenuation/Boost is displayed in the console window.
4. Create the BER Diagram, or exit the program
5. BER Diagram image (*.bmp) is created at the same location as the .txt file.
6. Color coding is used to determine different BER ranges.
    a. **Green** denotes good operating range (~0 bits in error)
    b. **Yellow** denotes errors in the range of 1-64.
    c. **Yellowish-Orange** denotes errors in the range of 64-512.
    d. **Orange-Red** denotes errors in the range of 512-4096.
    e. **Red** denotes errors in the range of 4096-32768.
    f. **Blue** denotes errors >32768 or if the BIST valid is not detected on the receiver.
7. The **best TX coefficients** are denoted by a **white dot** in the diagram shown below. The image also shows CMC1C2 values associated with the white dot.


C:\ti\pdk_keystone2_3_01_00_00\packages\MyExampleProjects\serdes_dbg_K2EC66ExampleProject\Debug\soc_device1_ber_scanout
Best CMC1C2:3/4/0, At Bo:7/7 2014-Aug-22 Lane 1
87% C2:0
87% C2:1
87% C2:2
83% C2:3
73% C2:4
CM
C1


Create BER Diagram
1
Path of .txt:
2
Against:
Lane Errors
3
Console:
4
Create!
Cancel

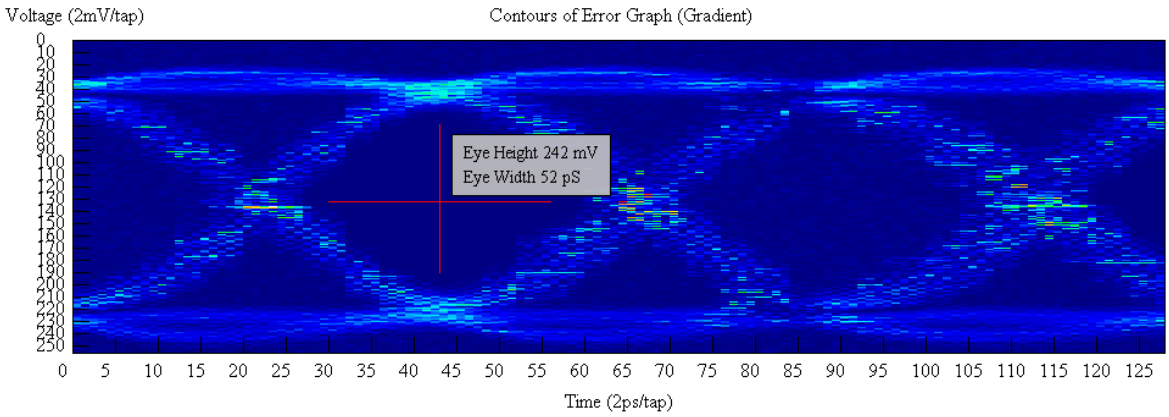
14. Eye Diagram Viewer

Contents:

Eye Diagram Viewer (EyeDiagramViewer.jar) is located at serdes_diag/tools/eye_diagram_tool. The purpose of this program is to create an eye diagram from an eye scan .txt file from Serdes Diag API Serdes Diag_EyeMonitor(). This program was created as a no-license-required replacement for an existing MATLAB script with the same purpose. This program was written in Java using Eclipse IDE, a free download from https://www.eclipse.org/. This program was compiled with Java version 7 update 60 build 1.7.0_60-b19 using the runtime environment JavaSE-1.6 (jre6). Users must have Java version 7 update 60 or above to run this program, which can be freely downloaded at https://java.com/en/download/manual.jsp. The user is not required to have the Java runtime environment installed to run this program.

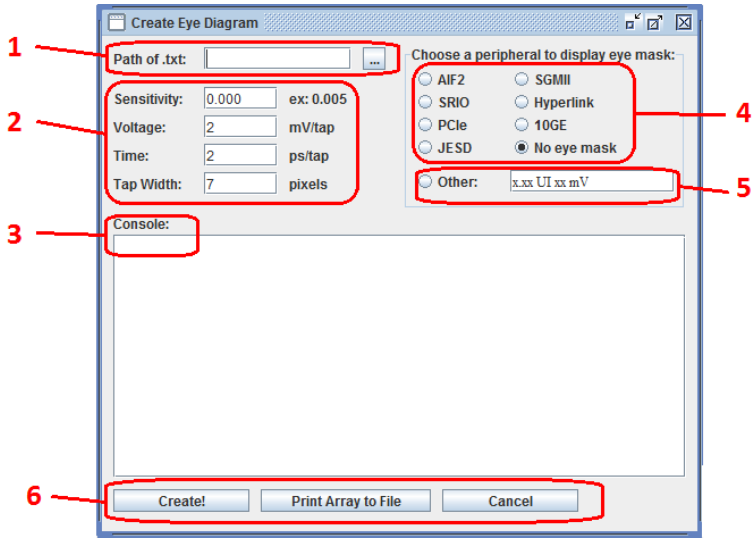
Steps to Generate the Eye Diagram

1. Read in the .txt file generated by the SERDES Eye Diagram API
2. Find the eyes
    a. Sweeps through the data looking for zero regions. If the sensitivity > 0.0, then the program looks for regions that are less than (maximum*sensitivity).
    b. Continues sweeping through the data; when it finds another eye, the program compares the two and keeps the larger one. If the new eye has a significantly (+10%) larger vertical axis than the old eye, then the new eye becomes the largest. If they are within 10% of each other, then the eye with the longest horizontal axis becomes the largest. (See Calculations Class > findTheLargerEye())
3. Draw the diagram
    a. Load in the color map and assign values to the different colors
    b. Find the color that corresponds to the data point and draw it on the diagram
    c. Find the eye mask in EyeMaskSummary.txt and draw it on diagram
    d. Draw the eye crosshair
    e. Draw the axes and labels
4. Saves the diagram to the same directory as the input file using the same name + ".png"



User Interface

1. File Path
    a. Enter the full path of the eye scan .txt file either by typing in the textbox, or by hitting the browse button. Multiple file selection is supported.
2. Required Options
    a. The sensitivity option is used to manipulate the eye size calculation. If the sensitivity is greater than 0.0, then the program looks for regions that are less than (maximum*sensitivity) rather than exclusively zero regions.
    b. Select how many mV/tap and how many ps/tap that matches the eye scan .txt file.
    c. The “Tap Width” option stretches the eye scan image horizontally.
3. Console window
    a. After the diagram is generated, information about the image, eye, and mask (optional) is displayed here.
4. Eye Mask selection
    a. Choose the peripheral that was used to create the eye scan .txt file. This is used in selecting the proper mask dimensions from the EyeMaskSummary.txt file. If multiple eye scan .txt files are selected, the program will apply this option to all files on the list.
5. Custom Eye Mask
    a. If a custom mask is desired, enter its dimensions in place of the x’s. The custom dimensions are parsed by spaces, so a custom dimension can have more digits than the number of x’s in the textbox.
6. Create the eye diagram, print the processed data to a .csv file, or exit the program.


Create Eye Diagram
1
Path of .txt:
...
2
Sensitivity:
0.000
ex: 0.005
Voltage:
2
mV/tap
Time:
2
ps/tap
Tap Width:
7
pixels
Choose a peripheral to display eye mask:
AIF2
SGMII
SRIO
Hyperlink
PCIe
10GE
JESD
No eye mask
4
Other:
x.xx UI xx mV
5
3
Console:
6
Create!
Print Array to File
Cancel

Known Issues

1. If inaccurate crosshairs, try increasing the sensitivity
    a. Usually 0.005 will cause the desired result
    b. Found SGMII to need 0.005 sensitivity

2. Eye height and width calculations only take into account the length of the red crosshair. If the eye extends beyond the red crosshair (or off the page), then this extra distance is not included in the eye width/height calculation.

3. If the eye mask is detecting an incorrect lane rate, check to see if there is an entry for the desired lane rate in EyeMaskSummary.txt. If no entry exists, you must use the “other” option in the GUI.

4. The files ColorMapTable.txt and EyeMaskSummary.txt must be in the same directory as the .jar file

5. A blank eye diagram indicates that all of the values in the .txt file were constant