# Contents

# Introduction

A hardware demonstration design, targeting the UltraScale Kintex KCU105 or Zynq UltraScale+ MPSoC ZCU102 Rev 1.0 or later evaluation platforms, can be generated and exercised using the scripts, source and GUI provided in this download.

This is a complete FPGA design, incorporating transmit and receive JESD204B cores and a JESD204 PHY core containing GTH transceivers. The transceivers are setup for PMA or cable loopback to allow the operation of the JESD204B cores to be explored. The JESD204 PHY core can be dynamically configured to line rates between 1G and 12.5G.

Using Xilinx Vivado Design Suite 2017.3. This design can be implemented and can be downloaded to the KCU105 or ZCU102 evaluation board. The data output from the RX core can be monitored using Xilinx Vivado Hardware Manager.

This design is controlled using an included GUI that connects to the KCU105 or ZCU102 using Ethernet. The GUI allows the line rate, link parameters and test patterns to be dynamically changed. The GUI also supports running eyescan (KCU105 only) on the transceivers and provides a simple 2D heat map of BER results.

# Demo Package contents

The JESD204_uhwd.zip file contains the following folders:

- ***script*** This folder contains the scripts used to generate the UltraScale Hardware Demo for the KCU105 platform.
- ***hdl*** This folder contains all the required source code for design blocks not sourced from the Vivado IP catalog.
- ***constraints*** This folder contains the design constraints required for the design.
- ***sw_src*** This folder contains the source code and a pre-generated ELF files for the software running on the Microblaze processor that is included in the design.
- ***GUI*** This folder contains the GUI used to control this hardware demo.
- ***bitstream*** This folder contains pre built bitstreams for the KCU105. These bitstreams were built using the source and scripts from this demo.

# Overview of the KCU105 Hardware Demonstration Design

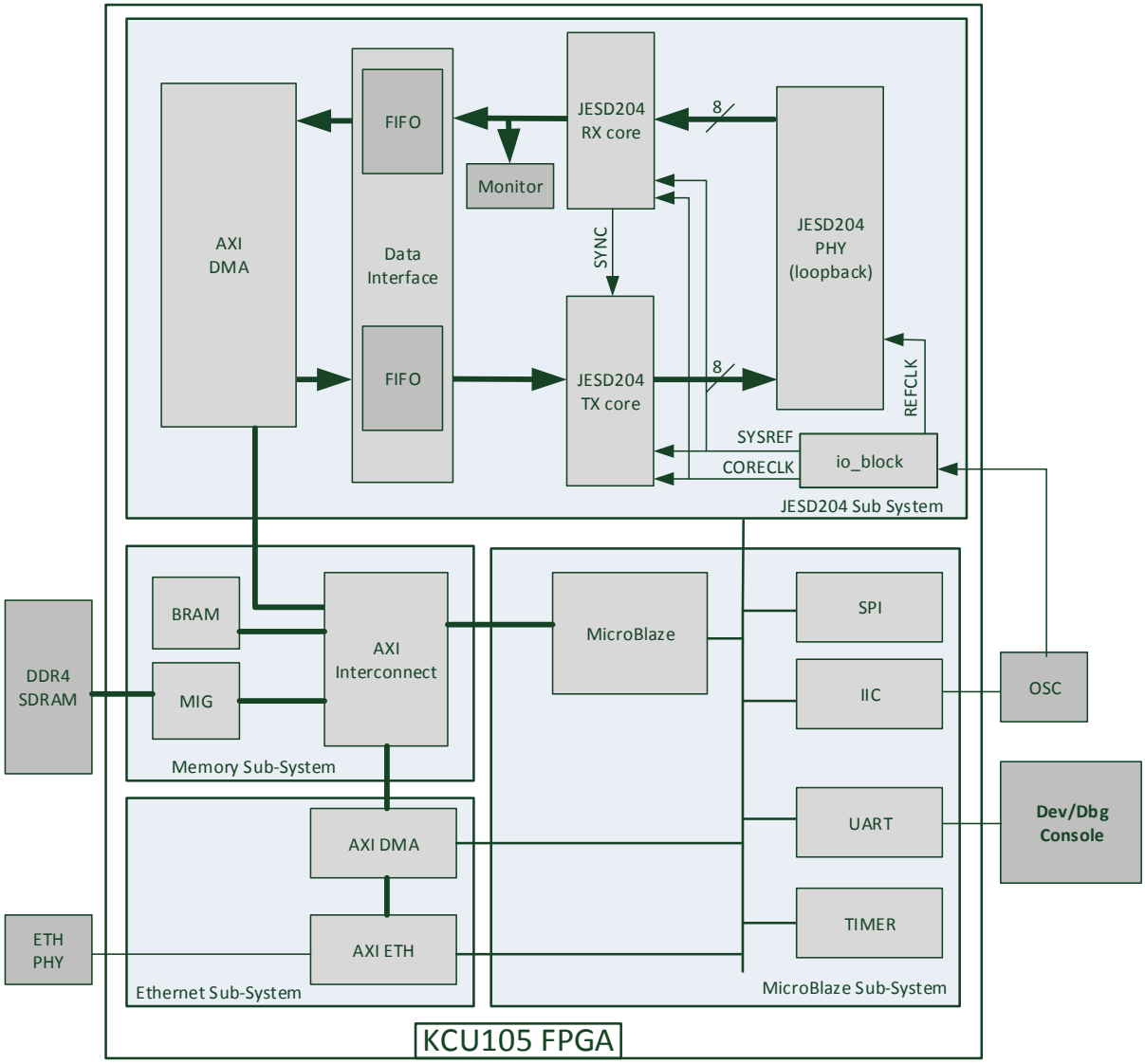


Figure 1 UltraScale Hardware Demonstration Design

Figure 1 illustrates the KCU105 UltraScale hardware demonstration design which consists of the following:

- JESD204 Subsystem
  - Transmit JESD204B core.
  - Receive JESD204B core.
  - JESD204B PHY core. This allows the transceivers to be shared between the transmit and receive JESD204B cores.
  - Data Interface. This custom IP enables playback and capture of data through the JESD204B cores.
  - AXI DMA. This provides DMA access to DDR4 memory for playback and capture of data.
  - Monitor Block. This custom IP is used to de-map the received data into sample data which can then be monitored using an ILA in the Vivado Hardware manager.

- IO Block. This custom IP is used to provide IO buffers and is also used to generate CORECLK and SYSREF from the externally supplied REFCLK.

- Microblaze Subsystem
  - Microblaze Processor.
  - AXI Lite Peripheral Interconnect structure.
  - AXI IIC. This is used to control the SI570 programmable oscillator on the KCU105. This oscillator is used to supply REFCLK to the JESD204B Subsystem.
  - AXI UART. This can be used for development / Debug.
  - AXI Timer. This timer is required by the TCP protocol implementation of the LwIP software stack.
  - AXI SPI. Unused in this demo design.

- Ethernet Subsystem
  - AXI 1G/2.5G Ethernet Subsystem. This IP is configured for 1G and uses SGMII to communicate with the Ethernet Phy on the KCU105.
  - AXI DMA. This provides DMA access to memory for the AXI Ethernet Subsystem.

- Memory Subsystem
  - MIG. This block provides the DDR4 Memory interface. The local bus is 512bits wide and runs at 300MHz which equates to a memory clock of 1.2GHz for the DDR4 memory. The MIG core is also used to generate the 142MHz clock used to drive the MicroBlaze processor and the AXI peripherals as well as a 25Mhz clock for the DRP interface on the transceivers.
  - AXI interconnect. This is a multi-layered structure to separate and optimise the routing and resources used by the two data bus structures (512bit 300MHz and 32bit 150Mhz).
  - Block Rams for processor memory and buffer descriptor memory for the JESD204 block AXI DMA.

## Overview of the ZCU102 Hardware Demonstration Design

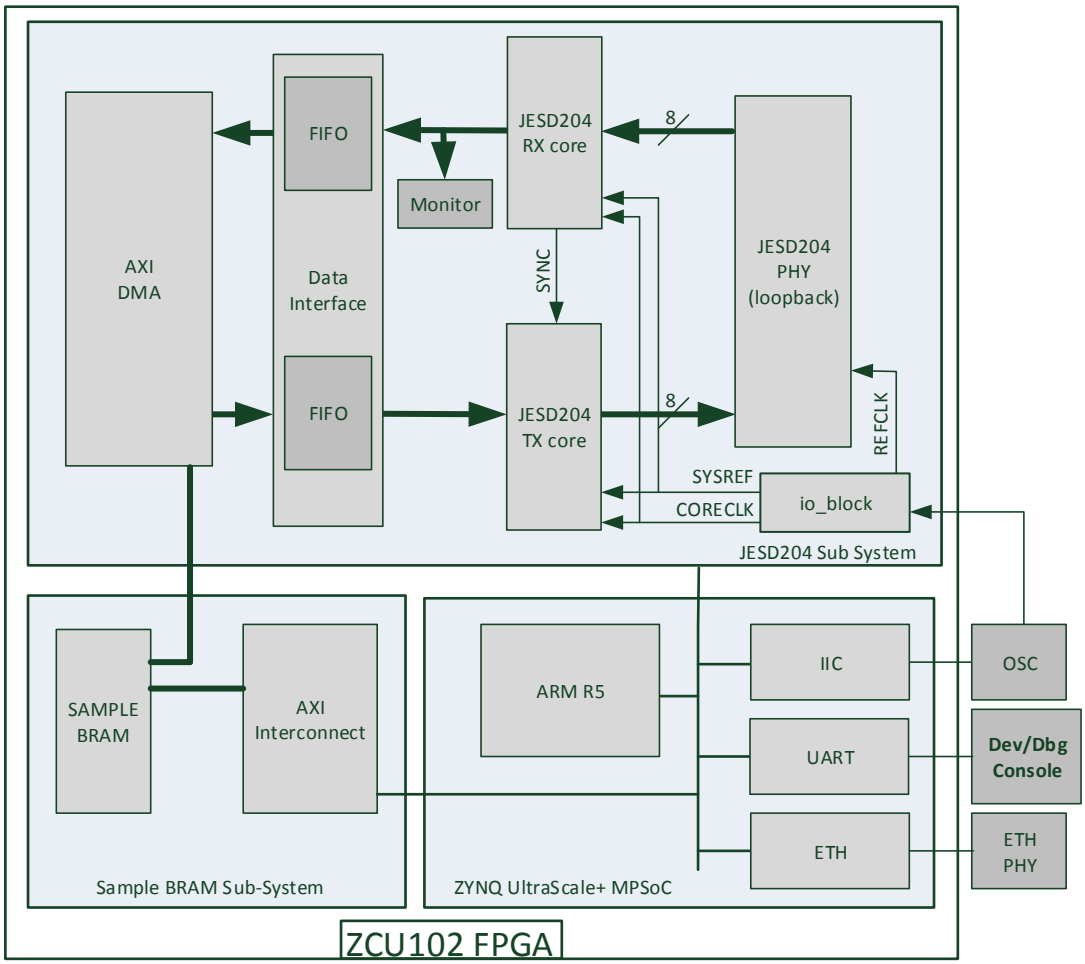


Figure 2 UltraScale Hardware Demonstration Design

Figure 2 illustrates the ZCU102 UltraScale+ MPSoC hardware demonstration design which consists of the following:

- JESD204 Subsystem
  - Transmit JESD204B core.
  - Receive JESD204B core.
  - JESD204B PHY core. This allows the transceivers to be shared between the transmit and receive JESD204B cores.
  - Data Interface. This custom IP enables playback and capture of data through the JESD204B cores.
  - AXI DMA. This provides DMA access to DDR4 memory for playback and capture of data.
  - Monitor Block. This custom IP is used to de-map the received data into sample data which can then be monitored using an ILA in the Vivado Hardware manager.
  - IO Block. This custom IP is used to provide IO buffers and is also used to generate CORECLK and SYSREF from the externally supplied REFCLK.

- ZYNQ UltraScale+ MPSoC
  - ARM R5 Processor.

- AXI UART. This can be used for development / Debug.
- AXI Timer. This timer is required by the TCP protocol implementation of the LwIP software stack.
- Ethernet
- Block Rams for processor memory and buffer descriptor memory for the JESD204 block AXI DMA.

- Sample BRAM Subsystem
  - AXI interconnect. This is a multi-layered structure to separate and optimise the routing and resources used by the two data bus structures (512bit 300MHz and 32bit 150Mhz).

# Generating the Hardware Demonstration Design

All the files required to create the hardware demonstration design are located in the download. The following steps must be carried out to successfully generate the hardware demo project.

1) Unzip and open the hardware demo folder.
2) Open Vivado and confirm the version is 2017.3
3) Change the Vivado working directory to the root of the unzipped folder. Changing directory in Vivado is achieved by simply typing "cd <directory>" at the tcl console.
4) Choose to build for either KCU105 or ZCU102 by typing either of the following in the vivado tcl console:
   set BOARD "KCU105"
   Or
   set BOARD "ZCU102"
5) Type the following in the tcl console "source ./script/build_it.tcl" This will run the main script that builds the design.

The build_it.tcl script does the following:

a) Creates a new Vivado project for the design.
b) Creates projects for the included hdl blocks and packages them as IP for use in the IPI block diagram of main design.
c) Builds the design in IPI.
d) Synthesizes the design.
e) Implements the design.
f) Generates bitstream's for the designs.
   For the KCU105 two bitstream are generated:
   "**KCU105_XLOOP_STATIC.bit**" which has the Ethernet interface set to use a static IP address.
   "**KCU105_XLOOP_DHCP.bit**" which has the Ethernet interface set to accept an IP address from a DHCP server. See Setup Ethernet Interface later in this document for detailed instructions.
   For the ZCU102 a single bitstream "**mySystem_wrapper.bit**" is generated. The choice between static and dynamic IP address is made when you choose which software to download and run on the ARM processor in the design.

After running the build_it.tcl script two new folders will have been created.

- local_IP_<BOARD>_XLOOP
  This folder contains:
  - The projects that were created to package the RTL blocks from the hdl directory into IP for use in IP Integrator.
  - A repository containing all the packaged IP blocks.

- prj_<BOARD>_XLOOP
  This folder contains the Vivado project for the BOARD that has been built.

# Running the Hardware Design Using Vivado

## Prerequisites for Running the Demonstration

- KCU105 or ZCU102 Evaluation Platform
- Vivado 2017.3 toolset
- Micro-USB cable for USB JTAG connection
- Micro-USB cable for USB UART connection
- Ethernet Cable

## Setup Ethernet Interface

A choice must be made on how to run the demo based on your local network and any relevant security policies. The choices are detailed as follows:

- Static IP address. With static IP address the design running on the BOARD is configured to use an IP address of 192.168.1.10 and a subnet mask of 255.255.255.0. In this setup, it is expected that the BOARD is directly connected to your PC and not to local network connection. See Setup for Static IP address. Note: If a network is available and it contains a DHCP server then it is recommended to use Dynamic IP Address over Static IP address.

- Dynamic IP Address. With Dynamic IP address, the design running on the BOARD is configured to communicate with a DHCP server to request an IP address and subnet mask. See Setup for Dynamic IP Address.

## Setup for Static IP address

- Connect an Ethernet cable between the Ethernet port on the host PC and the RJ45 Ethernet port on the BOARD.
- Set the host PC to use the wired Ethernet port and configure an IP address of 192.168.1.1

This can be achieved as follows:
Open the “Networks and Sharing Center” from “Control Panel”

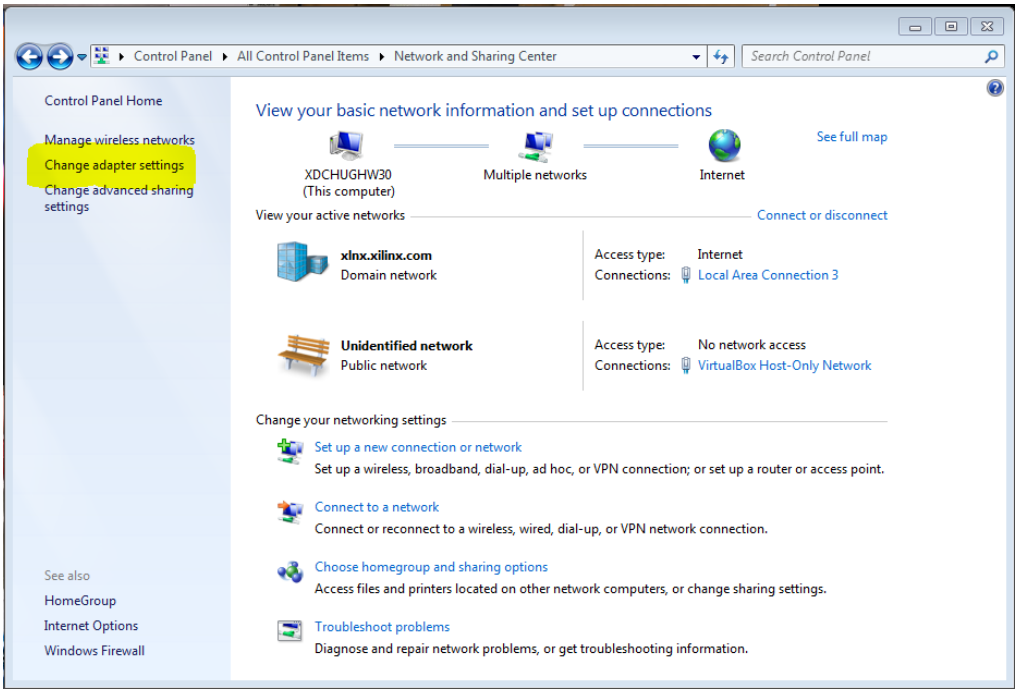


Figure 3 Network and Sharing Center

Click on “Change adapter settings” Highlighted yellow in Figure 3

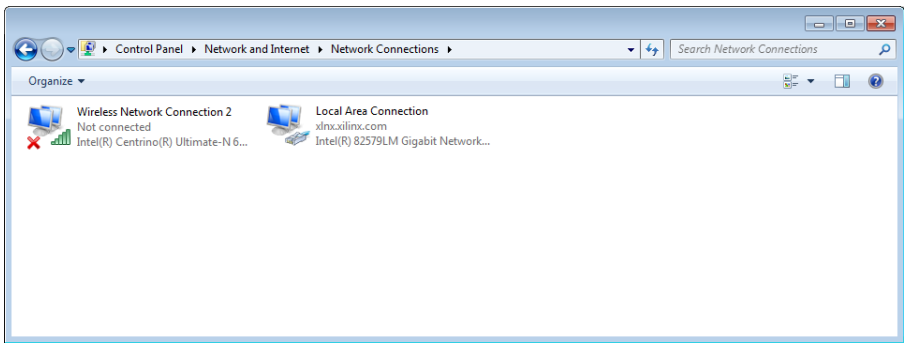


Figure 4 Network Connection

Right click on your “Local Area Connection” icon and choose properties

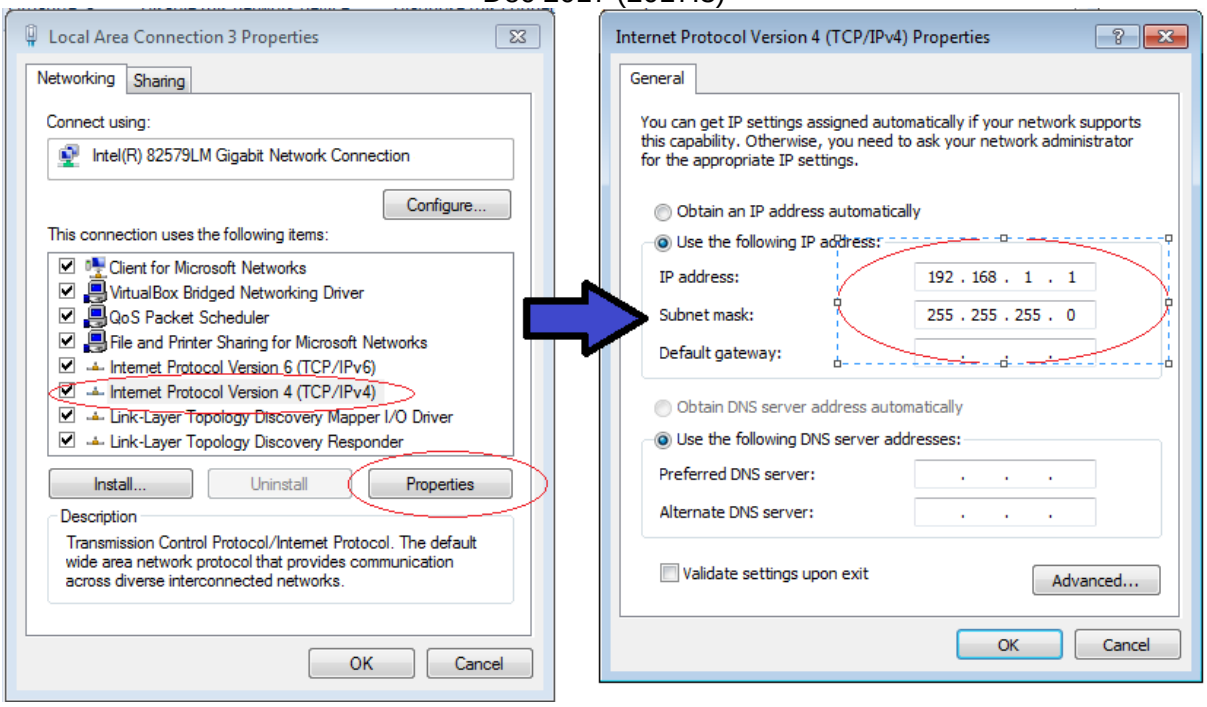


Figure 5 TCP/IPv4 properties

Then edit the Properties for “Internet Protocol Version 4 (TCP/IPv4)” as shown in Figure 5.

OK all the changes.

Notes:

- You may need to temporarily disable any wireless networking interfaces.
- You may need to temporarily disable any firewall that may be used on your PC network interface.

## Setup for Dynamic IP Address

- Connect an Ethernet cable between the Ethernet port on your local network and the RJ45 Ethernet port on the BOARD.

Notes:

- Do not connect directly your PC.
- The BOARD, the DHCP server and your PC must all be on the same network.
- You may need to temporarily disable any firewall that may be used on your PC network interface.

## Configuring the FPGA KCU105

- Connect a USB port on the host PC to the USB-JTAG port on the evaluation board.
- Connect a USB port on the host PC to the USB UART port on the evaluation board. See the KCU105 User guide for details about the Silicon Labs CP2105GM dual USB-to-UART Bridge interface on the KCU105. You must ensure the Silicon Labs drivers are installed.
- Configure a terminal application (Tera Term, Hyperterm, PuTTY etc) to communicate with the COM port assigned to the Silicon Labs CP210x standard interface as shown in Figure 6.

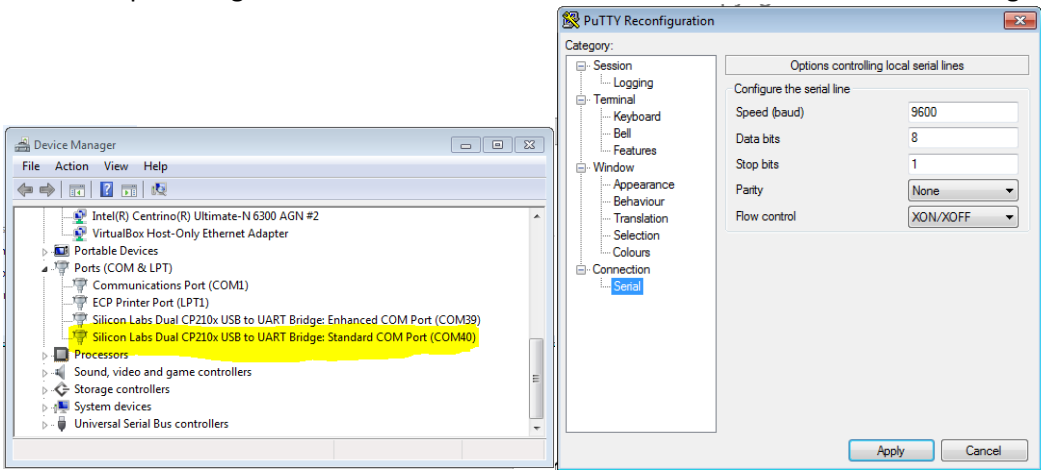


Figure 6 Status console UART Setup

- Power up the evaluation board.
- Open the Vivado Hardware Manager from the Flow Navigator in the project that was built.
- Open a new hardware target
- Select the targeted FPGA, and click on next. Leave all other options on the default values.
- Right click on the device that should have appeared in the GUI and select '*program device*' from drop down menu.
- Locate the bitstream file. Two bitstream files will have been generated. You must choose the correct file based on your chosen network configuration (Static IP or DHCP assigned).
    1. ./prj_MyKcu105_XLOOP/prj_MyKcu105_XLOOP.runs/impl_1/KCU105_XLOOP_STATIC.bit
    2. ./prj_MyKcu105_XLOOP/prj_MyKcu105_XLOOP.runs/impl_1/KCU105_XLOOP_DHCP.bit

- Click ok. This action will program the device.
- After programming completes. The on board firmware will negotiate the Ethernet link speed and initialise the platform.
- Take note of the BOARD IP address output from the UART to the status console (see Figure 7). This IP address must be used when connecting with the GUI.

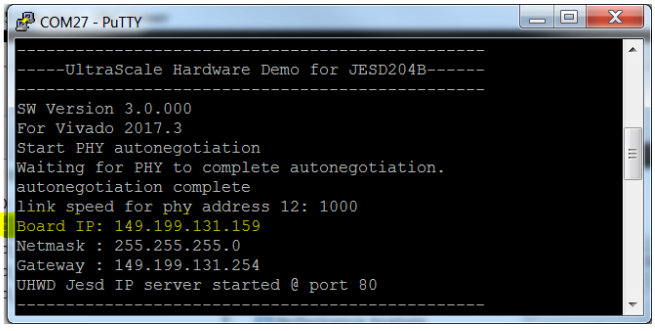


Figure 7 Board IP Address

## Configuring the FPGA ZCU102

- Connect a USB port on the host PC to the USB-JTAG port on the evaluation board.
- Connect a USB port on the host PC to the USB UART port on the evaluation board. See the ZCU105 User guide for details about the Silicon Labs CP2108 Quad USB-to-UART Bridge interface on the ZCU102. You must ensure the Silicon Labs drivers are installed.
- Configure a terminal application (Tera Term, Hyperterm, PuTTY etc) to communicate with the COM port assigned to the Silicon Labs CP210x Uart Bridge interface 0 as shown in Figure 8.

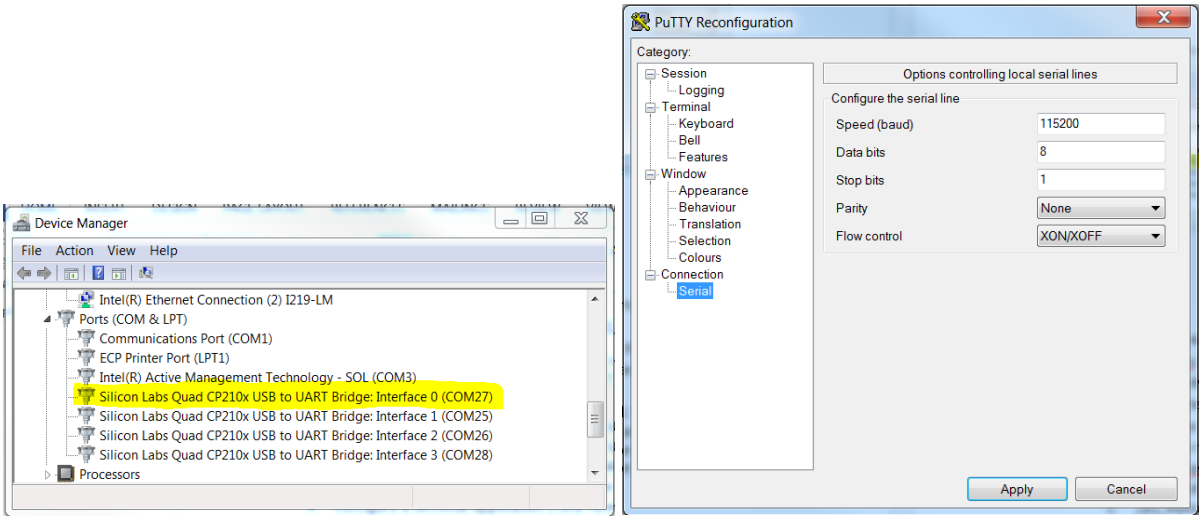


Figure 8 Status console UART Setup

- Power up the evaluation board.
- Open the Vivado Hardware Manager from the Flow Navigator in the project that was built.
- Open a new hardware target
- Select the targeted FPGA, and click on next. Leave all other options on the default values.
- Right click on the device that should have appeared in the GUI and select '*program device*' from drop down menu.
- Locate the bitstream file. A single bitstream file will have been generated.
- Click ok. This action will program the device.
- After programming completes. Procede to downloading the SW to the ARM processor using the SDK.

## Download software to ZCU102 using SDK

- Ensure the FPGA has been programmed successfully using the instructions for configuring the FGPA ZCU102 in this document.
- Launce the SDK from the file menu in Vivado.
- Once the SDK is open you should see two prebuild app projects with associated bsp projects
    1. "uhwd_static" Containing the ARM software built with a static IP address
    2. "uhwd_dhcp" Containing the ARM software built to obtain an IP address from a DHCP server.

- Select one or other of the app projects and chose "Run Configurations" from the "Run" menu.

- When the run configurations dialog appears choose “Run or Debug program using system debugger”
- When the system debugger pane opens ensure to select “Reset RPU” and “PL Powerup” as shown in Figure 9

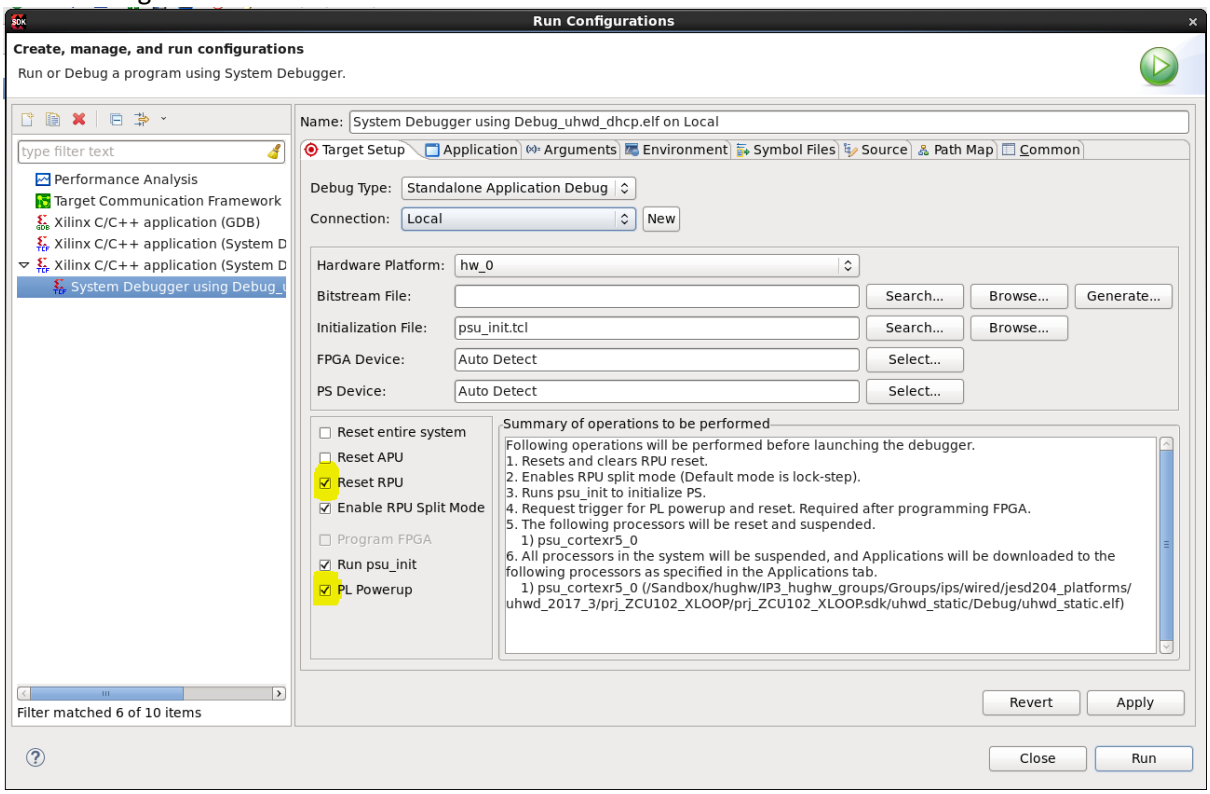


Figure 9 Run Configurations

- Hit Apply then Run.
- Take note of the BOARD IP address output from the UART to the status console (see Figure 10). This IP address must be used when connecting with the GUI.

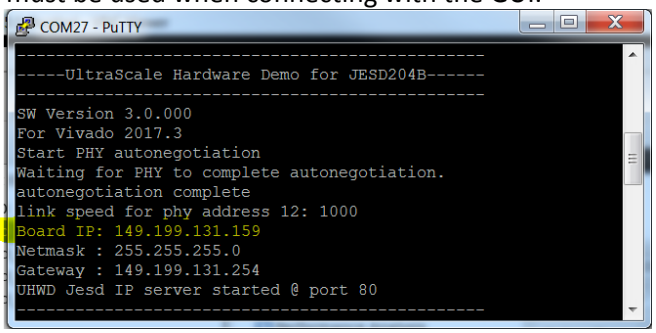


Figure 10 Run Configurations

## Monitoring the Device in the Vivado Hardware Manager

Once the device has been configured three entries will appear in the Vivado GUI under the device (SysMon, hw_ila_1, MIG_1(KCU105 only)).

On the KCU105. The MIG_1 Tab will be open by default. This should show successful calibration of the DDR4 memory. Double click on the SysMon entry and select all the check boxes in the New Dashboard dialog box that appears as, shown in Figure 11. This will create a dashboard that contains the FPGA System Monitor and the hw_ila_1 signals. The Run Trigger button can be pressed at any time to observe the data that is coming from the JESD204B receive core. There should also be a SysMon entry showing the temperature of the FPGA.

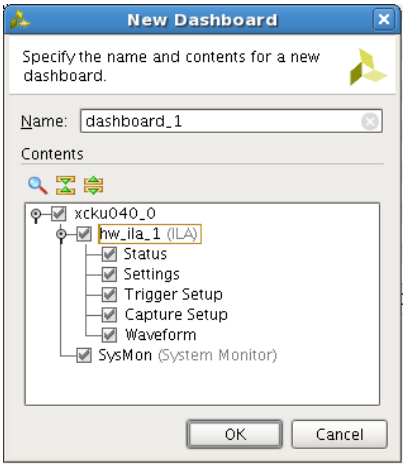


Figure 11 Dashboard Setup

On the ZCU102 the hw_ila_1 window will be open by default.

Analog waveforms may be observed in the dashboard by selecting the RX Monitor channel data and setting the waveform style to Analog, shown in Figure 12.

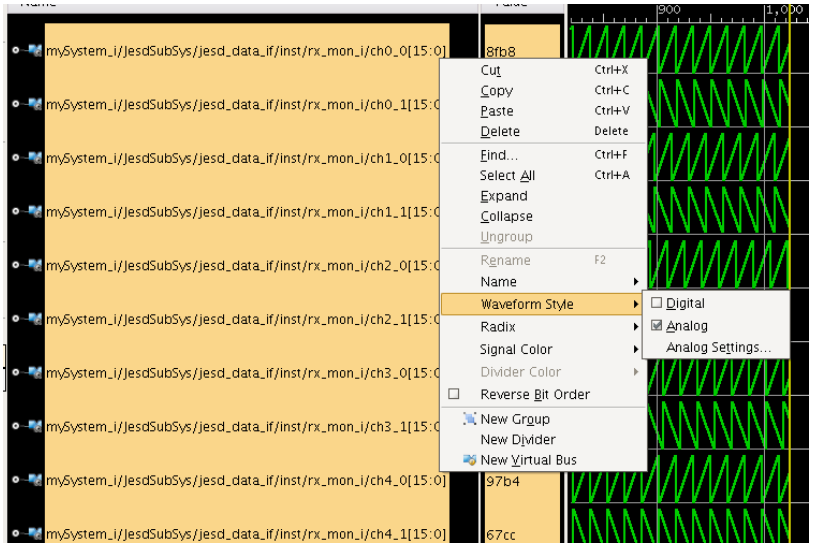


Figure 12 Analog waveform setup

## Setting Up the Control GUI

To launch the JESD204 UltraScale Hardware Demo GUI navigate to GUI directory and double click on uhwd_gui.exe. A window will appear asking for the device's IP address, Figure 13. The IP address you enter here must be the address you noted at step 6 of Configuring the FPGA. After entering the appropriate IP address and click Accept.

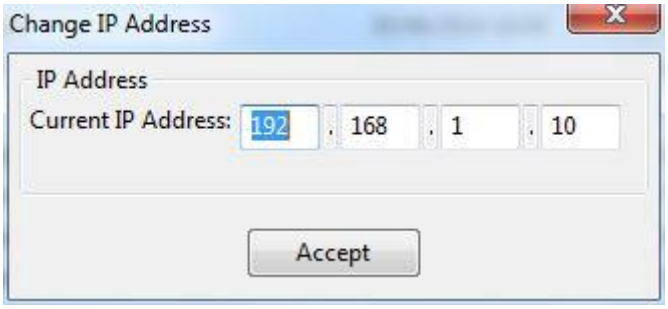


Figure 13 IP Address Input

If the BOARD is connected correctly to the PC and the IP address is correct, the GUI will launch. If not, an error message will appear stating that the BOARD could not be found (after clicking OK on the error message the GUI will open).

# Operating the Hardware Demonstration

Once the GUI has been launched it will open in its default state which can be seen in Figure 14.

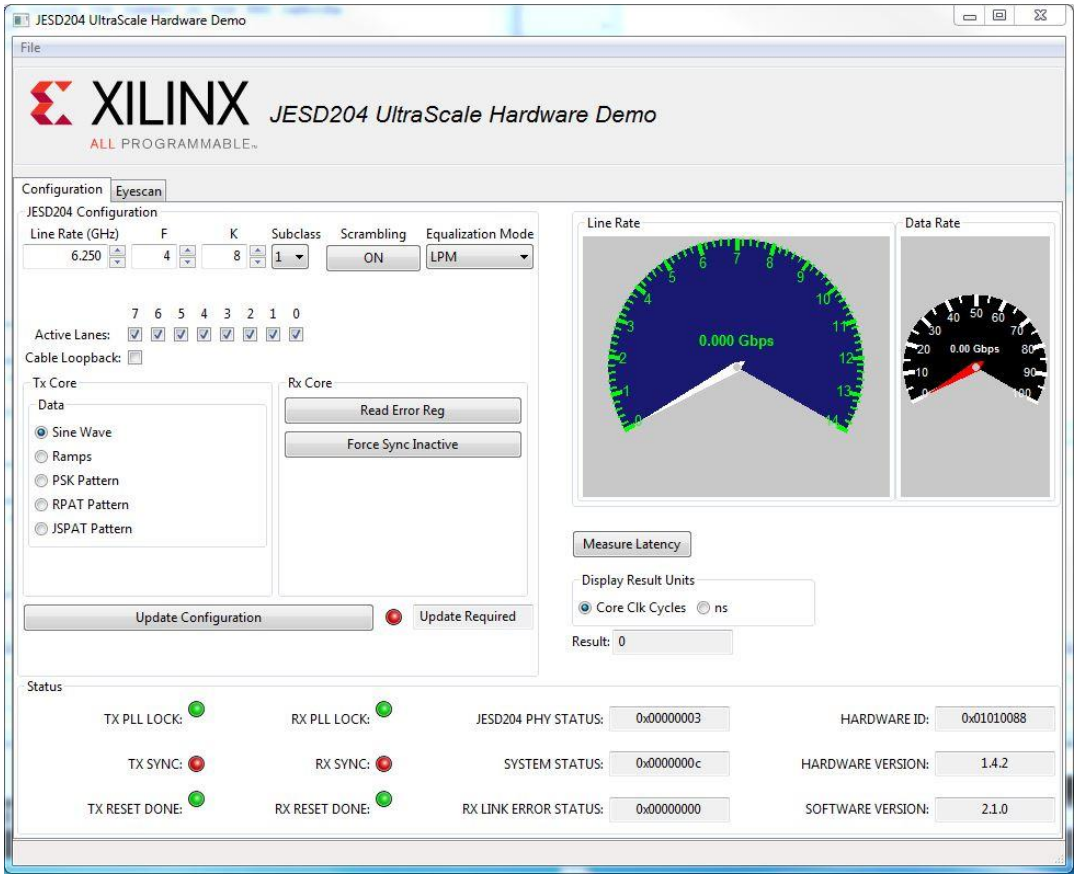


Figure 14 GUI Default State

The GUI under the Configuration Tab is split up into three major sections:

- JESD204 Configurations: from here, various JESD204 configurations can be set
  - Line Rate: Set Line rate of transceiver. Valid values from 1.0GHz to 12.5GHz
  - F: Octets per frame, any value from 1-256 can be selected
  - K: Frames per multiframe, any value from 1-32 can be selected
  - Subclass: JESD204 Subclass selection
  - Scrambling. A toggle button to enable or disable scrambling in the JESD204 core
  - Equalization Mode: Select between LPM and DFE
  - Active Lanes. Each check-box represents a lane, when selected that lane is active
  - Cable Loopback: When selected an external SMA cable loopback can be performed on lane 8. If no cables are present and this option is enabled the cores will not achieve SYNC
  - Under the Tx Core section, five different data patterns are available, a Sine Wave, a Ramp Pattern and a Phase Shift Keying modulation Pattern. The last two options, RPAT and JSPAT are Link test patterns and when enabled, the JESD204 cores will not achieve SYNC. These patterns can be used to run Eyescan(KCU105 only).
  - Under the Rx Core section, the '*Read Error Reg*' when pressed displays any errors that have occurred over the Link and the '*Force Sync Inactive*' toggle button forces the Tx JESD204 SYNC

input low. Both of these buttons can be pressed at any time and do not require the '*Update Configuration*' to be pressed.

- The '*Update Configuration*' when pressed updates the JESD204 configuration with the values selected in the GUI. Note, that until this button is pressed, any changes made to the configuration values in the GUI will not be passed into the cores. If a core setting has been modified, the LED located on the right of the button will turn RED, indicating that an update of the configuration is required.

- The right side of the GUI contains controls and displays for various features of the Hardware Demo:
  - The left speedometer displays the configured line rate of the Transceivers while the one on the right displays the aggregate data rate.
  - The '*Measure Latency*' toggle button enables a deterministic latency test witch counts the number of core clock cycles (or ns depending on GUI selection) required from the data entering the transmit JESD204 core to it exiting the receive JESD204 core. This test is available only when all 8 lanes are enabled. To run, enable the test and toggle the '*Force Sync Inactive*' button. The result should remain the same (Subclass 1 and 2 only) between re-synchronizations.
- The bottom part of the GUI displays various Status indicators based on the state of the hardware demo.

Once the '*Update Configuration*' is pressed the Tx and Rx SYNC LEDs should turn green and the GUI should match Figure 15.

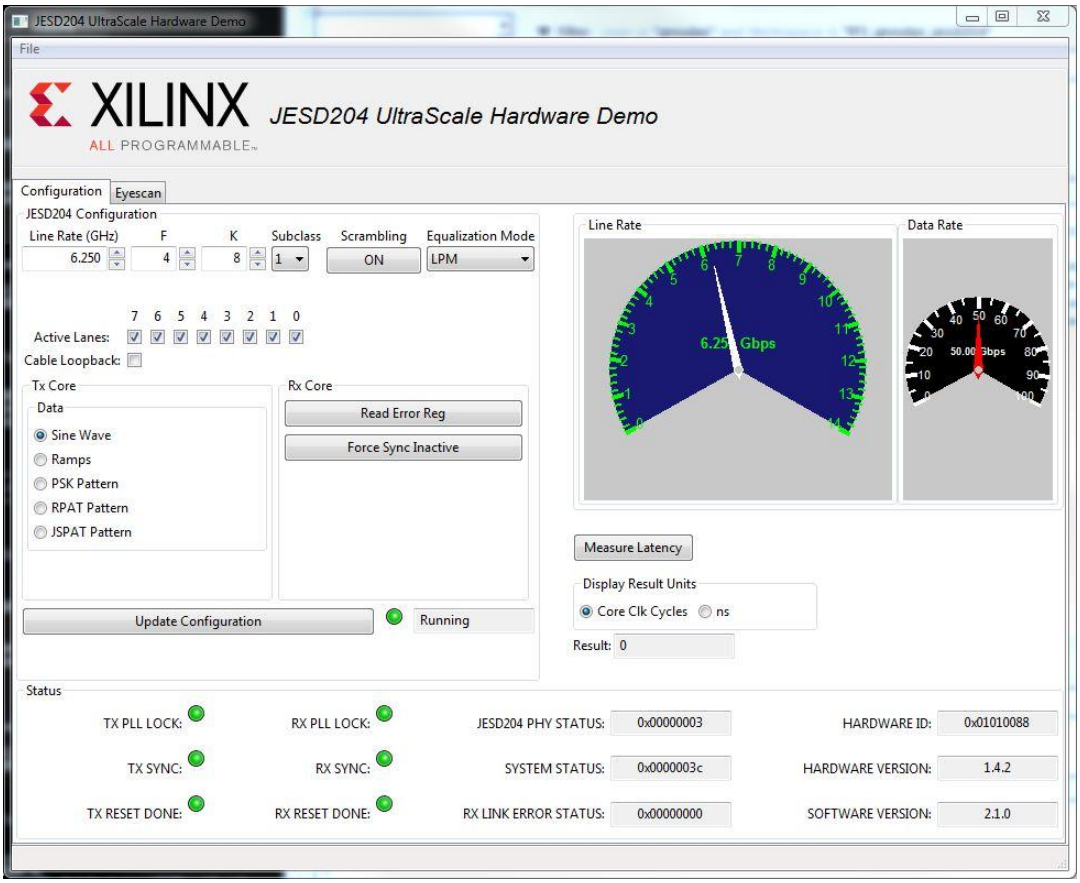


Figure 15 GUI After Configuration

## Running Eyescan (KCU105 only)

In the second tab of the GUI (which is only displayed if connected to a KCU105), Figure 6, Eyescan can be run on any enabled lane. To start the data sampling required to display the BER heat map, select the correct lane tab depending on the lane wanting to run Eyescan and then click on Start Eyescan. Once clicked the button will grey out until the process is complete. Note that while Eyescan is running, changing the configuration of the JESD204 cores is disabled (Figure 17). Eyescan can be run in parallel on multiple lanes. The time required to complete the Eyescan depends on the line rate and equalization mode selected in the Configuration tab.

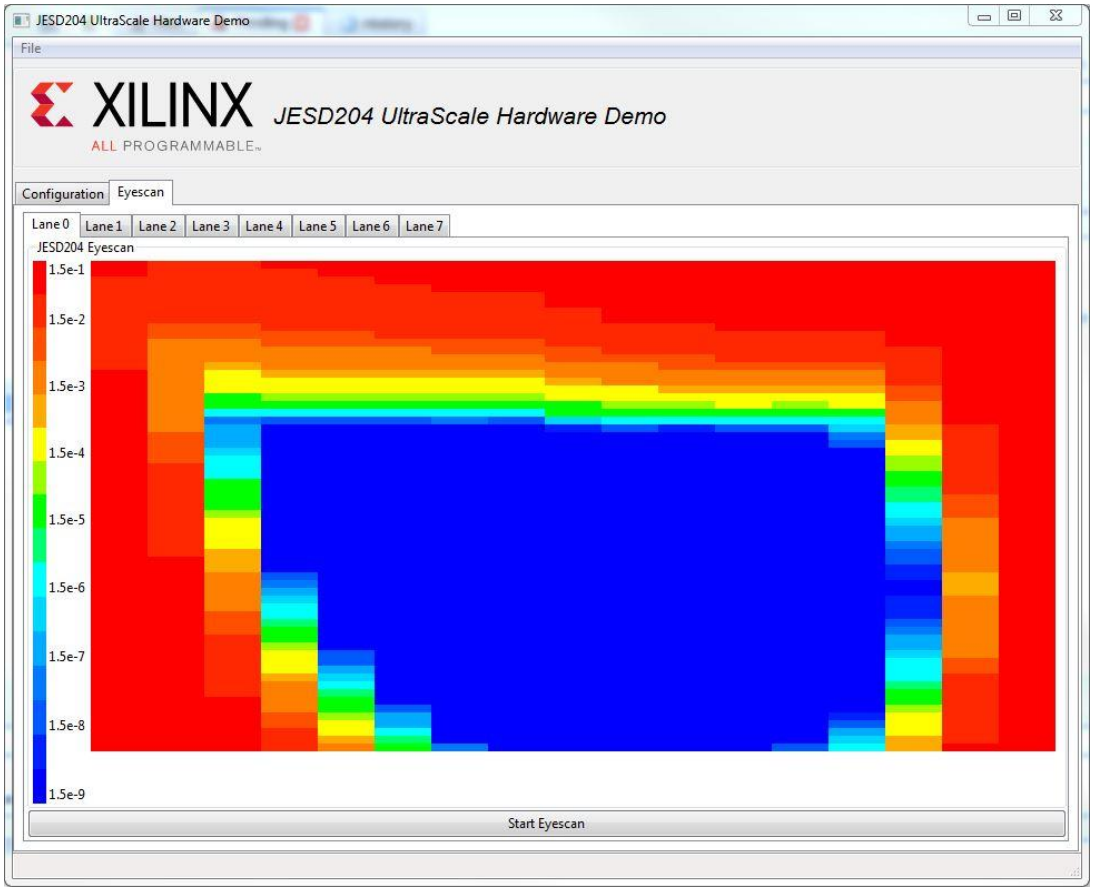


Figure 16 Eyescan Result

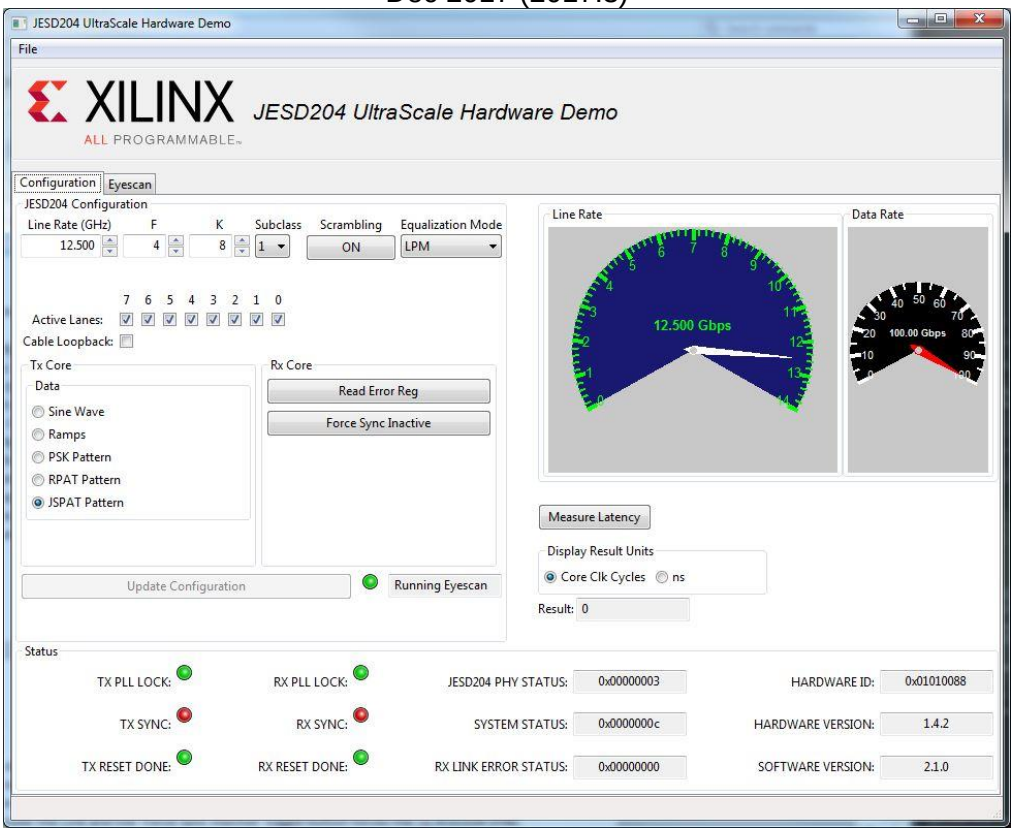


Figure 17 Configuration Tab When Running Eyescan

## Status LEDs

The user GPIO LEDs on the evaluation boards are used to provide additional visual status indications. The LEDS are assigned as shown in Table 1. The flashing LED's are used to indicate the status of individual clocks and are actually divided versions of the clock.

| GPIO LED | Function | Description |
|---|---|---|
| 0 | Rx SYNC | ON = SYNCB deasserted by Rx core |
| 1 | Tx SYNC | ON = SYNCB deasserted at Tx core |
| 2 | RX Reset done | ON = RX reset complete. |
| 3 | DRP CLK | Flashing = CLK Active. |
| 4 | AXI CLK | Flashing = CLK Active. |
| 5 | TX/RX CORECLK | Flashing = CLK Active. |
| 6 | GT REFCLK | Flashing = CLK Active. |
| 7 | TX EMPTY | ON = There is no data available for the TX core from the AXI DMA. |

Table 1 Status LEDs