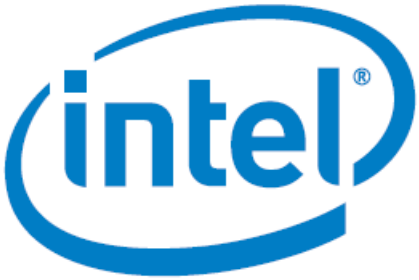


Intel® Arria® 10 GX Device Multi-Rate SDI II Pass-Through Using Video & Image Processing Pipeline Reference Design

Date: 1st July 2019

Revision: 1.0

Table of Contents

1 Overview

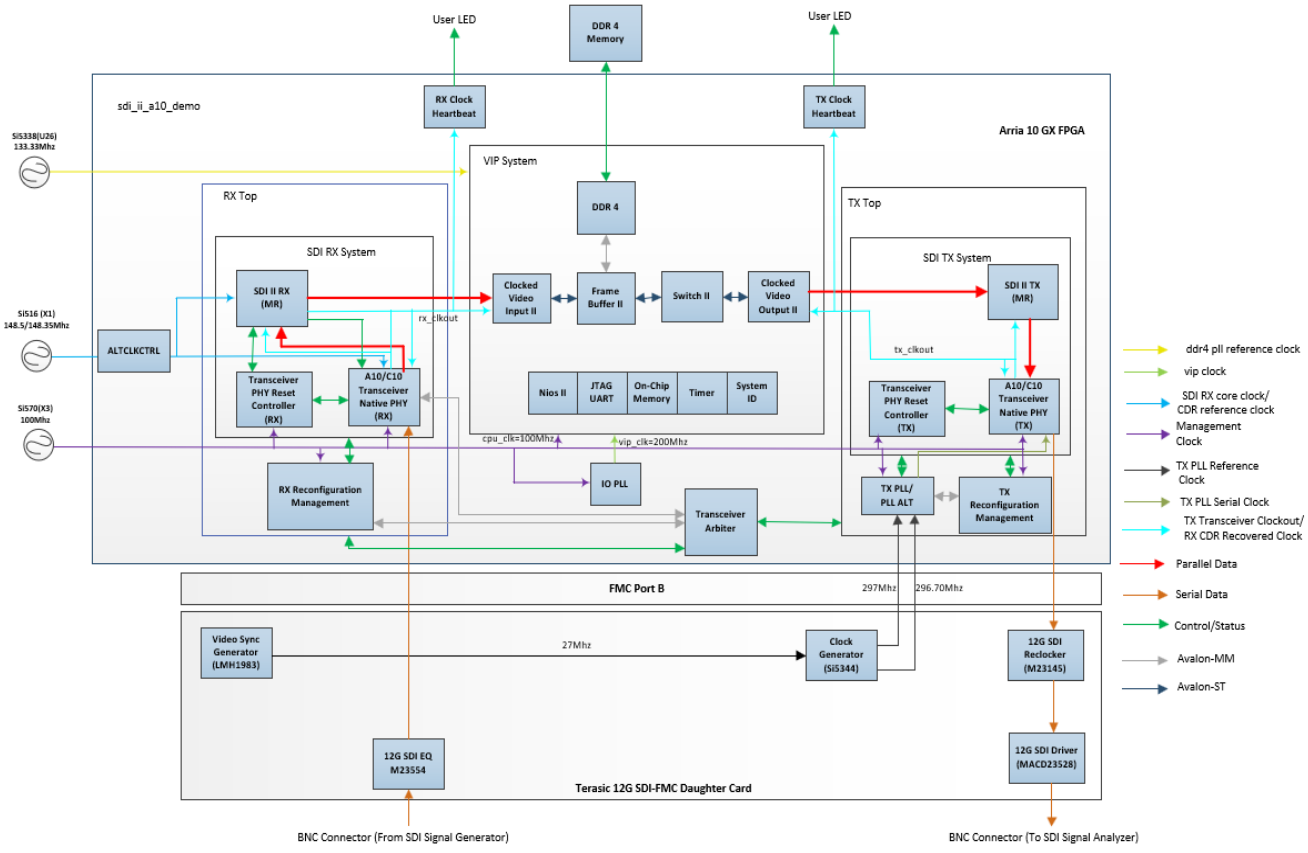
The Intel® Arria® 10 Serial Digital Interface II (SDI II) reference design demonstrates the multi-rate (up to 12G-SDI) pass-through video data with external voltage-controlled crystal oscillator (VCXO). The design uses key Video and Image Processing Suite (VIP) IP Cores, such as Clocked Video Input II (4k ready) Intel FPGA IP (CVI II), Clocked Video Output II (4k ready) Intel FPGA IP (CVO II), Frame Buffer II (4k ready) Intel FPGA IP (VFB II) and Switch II (4k ready) Intel FPGA IP for pass-through implementation. The SDI II pass-through implementation is typically seen in multichannel broadcast application such as switchers, routers and multiviewers.

2 Theory of Operation

2.1 Reference Design Block Diagram

The reference design consists of top file (sdi_ii_a10_demo) instantiates RX top (rx_top.v), TX top (tx_top.v), VIP system, Transceiver Arbiter and IOPLL. The RX top instantiates SDI RX system as well as RX reconfiguration management block. The TX top instantiates SDI TX system, TX PLL as well as TX reconfiguration management block. The IOPLL will generate 200Mhz clock to VIP system.

Figure 1: Reference design simplified block diagram



2.2 Reference Design Features

The reference design provides the following design features:

- Files for targeting Intel Arria 10 GX FPGA development kit
- Input:
    - SDI II receiver connectivity supports video source from 720x480 (SD-SDI) up to 4096x2160 (12G-SDI) resolution at any frame rate up to 60 fps.
    - Support 10-bit YCbCr 4:2:2 color formats only at the input. All supported and unsupported video formats on Clock Video Input II and Clock Video Output II IP cores can be found in Video and Image Processing Suite User Guide.
    - Software automatically detects the input format and sets up the VIP pipeline appropriately.
- Output:
    - SDI II transmitter connectivity supports from 720x480 (SD) up to 4096x2160 (12G) resolution at any frame rate up to 60 fps.
    - The video data at receiver is retransmitted through the VIP pipeline controlled by Nios II processor.

2.3 Design Components

Table 1: Design components for SDI TX/RX systems

Design Component	Description
SDI II Intel FPGA IP	• TX – receives the video data from top level and encodes the necessary information (e.g. line number (LN), cyclic redundancy check (CRC), payload ID) into the data stream. The TX core oversamples the received data up to 11.88Gbps data rate for every video standard • RX – receives the parallel data from the Transceiver Native PHY Intel FPGA IP core and decodes the necessary information (e.g. descrambling, realigning data, and extracting the necessary information for user).
Transceiver Native PHY Intel Arria 10/Cylone 10 FPGA IP	• TX – hard transceiver block that receives parallel data from the SDI II Intel FPGA IP core and serializes the data before transmission. • RX – hard transceiver block that receives serial data from an external video source. *Note: you must connect the rx_analogreset_ack output signal from this block to the RX Reconfiguration Management module to indicate the transceiver is in reset.*
Transceiver PHY Reset Controller Intel FPGA IP	• TX: - The reset input of this controller is triggered from the top level. - The controller generates the corresponding analog and digital reset signal to the Transceiver Native PHY Intel Arria 10 FPGA IP block, according to the reset sequencing inside the block. - Use the tx_ready output signal from the block as a reset signal to the TX core to indicate the transceiver is up and running and ready to receive data from the core. • RX: - The reset input of this controller is triggered by the SDI II Intel FPGA IP core.

	- The controller generates the corresponding analog and digital reset signal to the Transceiver Native PHY Intel Arria 10 FPGA IP block according to the reset sequencing inside the block.
RX Reconfiguration Management	RX transceiver reconfiguration management block that reconfigures the Transceiver Native PHY Arria 10 FPGA IP block to receive different data rate from SD-SDI to 12G-SDI
TX Reconfiguration Management	TX PLL reconfiguration management block that reconfigures the TX PLL to change the TX clock dynamically for switching between integer and fractional frame rates.
TX PLL/TX PLL Alt	Transmitter PLL block that provides the serial fast clock to Transceiver Native PHY. For this design, TX PLL is configured to have reference clock 0 to generate integer frame rate and reference clock 1 to generate fractional frame rates.

Table 2: Design components for VIP system

Design Component	**Description**
Clocked Video Input II (4K ready) Intel FPGA IP (CVI II)	Convert the SDI II receiver parallel video data out to Avalon-ST video protocol. Strips incoming clocked video of horizontal and vertical blanking and capturing only active picture data
Frame Buffer II (4K ready) Intel FPGA IP (VFB II)	Buffers the progressive or interlaced video frames into external memory. Handles mismatch in receiver and transmitter video data rate through triple-buffering.
External Memory Interfaces Intel Arria 10 FPGA IP (DDR4)	This IP core allow to easily interface with today's higher speed memory devices such as DDR4. The Frame Buffer II IP core writes to the memory to store input pixels and reads from memory to retrieve video frames and then transmit them.
Switch II (4K ready) Intel FPGA IP	Enables the connection of input video streams to output video streams. This is to ensure video is clean during standard formats switching.
Clocked Video Output II (4K ready) Intel FPGA IP (CVO II)	Convert the Avalon-ST video protocol to parallel video data going into SDI II transmitter IP core.
Nios II Processor	Initialize the VIP Suite IP cores, enable/clear CVI II interrupt, monitor CVI II status registers, program CVO II mode banks, and switch set input to CVO II.
On-chip Memory (RAM or ROM) Intel FPGA IP	Provide both instruction and data memory space.
JTAG UART Intel FPGA IP	Serve as the main communication portal between the user and Nios II processor via the terminal console in Nios II Eclipse tool.
Interval Timer Intel FPGA IP	This is an optional component reserve for debugging. It can control to start, stop and reset the timer.
System ID Peripheral Intel FPGA IP	A simple read-only device that provides Qsys systems with a unique identifier. Nios II processor systems use the system ID core to verify that an executable program was compiled targeting the actual hardware image configured in the target FPGA.

Table 3: Common block

Component	**Description**
Transceiver Arbiter	This block prevents transceivers from recalibrating simultaneously when either RX or TX transceivers within the same physical channel require reconfiguration. The simultaneously recalibration impacts applications where RX and TX transceivers within the same channel are assigned to independent IP implementations.

	The transceiver arbiter is an extension to the resolution recommended for merging simplex TX and simplex RX into the same physical channel. It also assists in merging and arbitrating the Avalon-MM RX and TX reconfiguration requests targeting simplex RX and TX transceivers within a channel as the reconfiguration interface port of the transceivers can only be accessed sequentially. The transceiver arbiter identifies the requester of a reconfiguration through its Avalon-MM reconfiguration interfaces and ensures that the corresponding *tx_reconfig_cal_busy* or *rx_reconfig_cal_busy* is gated accordingly. The transceiver arbiter used in this reference design can be optional because RX and TX transceiver is not merged into a same channel.
ALTCLKCTRL Intel FPGA IP	Promote refclk_sdi_p to regional clock.
TX/RX Clock Heartbeat	A simple logic to generate a slow clock and display on the LEDs.

2.4 Top Level Signals

This section explains the top level interfaces for the reference design.

2.4.1 Clocking Scheme

The reference design requires several clock sources from Arria 10 FPGA development kit as well as Terasic 12G SDI-FMC daughter card for proper operation.

Table 4: Clocking signals

Signal Name	Description	Usage	Pin Number	I/O Standard
clk_fpga_b2_p	100Mhz clock from X3 Si570 Programmable Oscillator (Default 100Mhz) on the Arria 10 FPGA development kit	• Video PLL (IO PLL) input reference clock • Used by TX & RX Avalon-MM reconfiguration and by the TX & RX transceiver PHY reset controller for transceiver reset sequence • Clock the Nios II Processor, on-chip memory, JTAG UART, interval timer and system IP	AR36/AR37	LVDS
refclk_sdi_p	148.5/148.35Mhz dedicated transceiver reference clock from X1 Si516 on the Arria 10 FPGA development kit	• SDI RX core reference clock • Transceiver clock data recovery (CDR) reference clock	L37/L38	LVDS
fmcb_gbtclk_m2c_p0	297Mhz dedicated transceiver reference clock from Arria 10 FPGA development kit FMC port B where this clock is provided by U4 Si5344 on Terasic 12G SDI-FMC daughter card	• TX PLL reference clock for integer frame rates	W8/W7	LVDS

fmcb_gbtclk_m2c_p1	296.7Mhz dedicated transceiver reference clock from Arria 10 FPGA development kit FMC port B where this clock is provided by U4 Si5344 on Terasic 12G SDI-FMC daughter card	• Secondary TX PLL reference clock for fractional frame rates	U8/U7	LVDS
ddr4_pll_ref_clk	133.33Mhz clock from U26 Si5338 programmable oscillator on the Arria 10 FPGA development kit	• DDR4 external memory interface input reference clock	F34/F35	LVDS

2.4.2 User Push Buttons and LEDs

The reference design uses the push buttons and LEDs on the Arria 10 FPGA development kit as functional controls and status indicators.

Table 5: Control & status signals

Signal Name	Description	Usage	Reference Designator	Pin Number/IO Standard
cpu_resetn	As a global reset for reference design	Reset	S4	BD27/1.8V
user_pb0	Push button to reset the Si5344 clock chip as well as SDI II transmitter IP Core	Reset	S3	T12/1.8V
user_led_g[7:0]	Green LEDs display. Details of LEDs indicator is explained in subsection 4.2.2	IPs status indicator	D3-D10	D18/1.8V, C18/1.8V, A19/1.8V, J24/1.8V, L25/1.8V, K25/1.8V, K26/1.8V, L28/1.8V
user_led_r[7:0]	Red LEDs display. Details of LEDs indicator is explained in subsection 4.2.2	IPs status indicator	D3-D10	M23/1.8V, D19/1.8V, C19/1.8V, B20/1.8V, L23/1.8V, K24/1.8V, J26/1.8V, L27/1.8V

2.4.3 Terasic 12G SDI-FMC Daughter Card Pins on FMC Port B

The reference design uses the Arria 10 FPGA development kit FMC port B to connect to Terasic 12G SDI-FMC daughter card.

Table 6: Terasic 12G SDI-FMC daughter card signals

Signal Name	Description	Pin Number	IO Standard
fmcb_dp_m2c_p8	SDI RX serial data from FMC port B	N3/N4	High Speed Differential I/0
fmcb_dp_c2m_p2	SDI TX serial data to FMC port B	V1/V2	High Speed Differential I/0
fmcb_la_rx_n3	RX cable equalizer lock status on Terasic 12G SDI-FMC daughter card	B18	1.8V
fmcb_la_rx_n13	Active low signal that perform power-on reset (POR) of the Si5344	M18	1.8V
fmcb_la_rx_p14	Si5344 output enable signal. This pin disable all outputs when held high	M21	1.8V
{fmcb_la_tx_p6, fmcb_la_rx_n14}	Input clock selection of Si5344 on Terasic 12G SDI-FMC daughter card. Select 00 to accept input clock of Si5344 to sync to LMH1983 clock	{B12,M20}	{1.8V,1.8V}
fmcb_la_rx_p1	Initialize LMH1983 on Terasic 12G SDI-FMC daughter card	G21	1.8V
fmcb_la_tx_n1	F sync signal LMH1983 on Terasic 12G SDI-FMC daughter card	B15	1.8V
fmcb_la_tx_n0	V sync signal LMH1983 on Terasic 12G SDI-FMC daughter card	B17	1.8V
fmcb_la_tx_p0	H sync signal LMH1983 on Terasic 12G SDI-FMC daughter card	A17	1.8V

2.5 Reference Design Files in RTL Folders

Table 7: All the relevant design files reside in project folder

Folders	Files
rtl/rx	/rx_top.v • /sdi_ii_rx_rcfg_a10.sv - /rcfg_sdi_cdr.sv • /sdi_rx_sys.qsys - */<qsys generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_reconfig_clk.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_reconfig_rst.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_cdr_refclk.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_coreclk.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_phy.ip

	▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_phy_reset.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_phy_rst_ctrl_clk.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_phy_rst_ctrl.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_rst.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_sdi.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>* - /ip/sdi_rx_sys/sdi_rx_sys_rx_sdi_clkout.ip ▪ /ip/sdi_rx_sys/*<ip generated folder>*
rtl/tx	/tx_top.v • /sdi_ii_tx_rcfg_a10.sv - /rcfg_refclk_sw.sv - /rcfg_pll_sw.sv • /tx_pll.ip - */<ip generated folder>* • /sdi_tx_sys.qsys - */<qsys generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_core_rst.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_phy.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_phy_reset.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_phy_reset_0.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_phy_rst_ctrl.ip ▪ /ip/sdi_tx_qsys/*<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_phy_rst_ctrl_clk.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_sdi.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>* - /ip/sdi_tx_qsys/sdi_tx_sys_tx_sdi_clkout.ip ▪ /ip/sdi_tx_qsys */<ip generated folder>*
rtl/vip_pipeline	/vip.qsys • */<qsys generated folder>* • /ip/vip/cpu.ip - /ip/vip */<ip generated folder>* • /ip/vip/cpu_clk.ip - /ip/vip */<ip generated folder>* • /ip/vip/cpu_ram.ip - /ip/vip */<ip generated folder>* • /ip/vip/cpu_reset.ip

	- /ip/vip */<ip generated folder>* • /ip/vip/jtag_uart.ip - /ip/vip */<ip generated folder>* • /ip/vip/wd_timer.ip - /ip/vip */<ip generated folder>* • /ip/vip/sysid.ip - /ip/vip /<ip generated folder> • /ip/vip/vip_clk.ip - /ip/vip */<ip generated folder>* • /ip/vip/vip_reset.ip - /ip/vip */<ip generated folder>* • /ip/vip/vip_sdi_cvi.ip - /ip/vip */<ip generated folder>* • /ip/vip/vip_sdi_vfb.ip - /ip/vip */<ip generated folder>* • /ip/vip/ddr4_buf.ip - /ip/vip */<ip generated folder>* • /ip/vip/vip_sdi_swi.ip - /ip/vip */<ip generated folder>* • /ip/vip/vip_sdi_cvo.ip - /ip/vip */<ip generated folder>* • /ip/vip/sdi_tx_clkout.ip - /ip/vip */<ip generated folder>* • /ip/vip/sdi_tx_clkout_reset_sync.ip - /ip/vip */<ip generated folder>*
rtl	/video_pll.ip • */<ip generated folder>*
rtl	/clk_ctrl.ip • /<ip generated folder>
rtl	/a10_reconfig_arbiter.sv
rtl	/clock_crossing.v
rtl	/clock_heartbeat.v
rtl	/edge_detector.sv
rtl	/sdi_ii_a10_demo.v
rtl	/sdi_ii_a10_demo.sdc

2.6 Software Operation

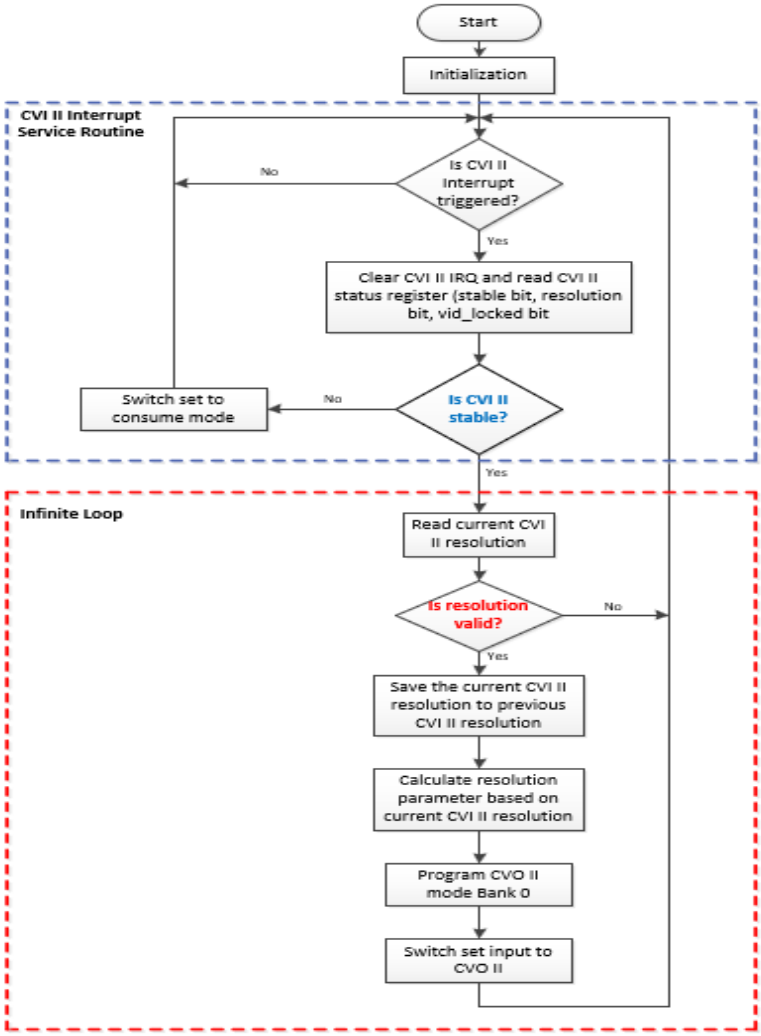
The reference design software determines the current inputs resolution and status automatically from Clocked Video Input II IP and then program the Clocked Video Output II mode bank 0.

The Nios II Processor perform the following tasks:

- Initialization process:
  - Registering an Interrupt Service Routine (ISR) with the hardware abstraction layer (HAL).
  - Set the *Go* bit to 1 to start the VFB II, CVI II, Switch II, and CVO II IP cores.

- Clear the CVI II interrupt status bit before checking the stability of video input availability. The CVI II is stable when stable bit (bit 8 of status register), resolution bit (bit 10 of status bit) and vid_locked bit (bit 11 of status register) are asserted.
- Once the CVI II is stable:
  - Read out all resolution from CVI II control registers.
  - Resolution valid is active sample <= 4096 and active line f0 <= 2160.
  - If resolution is valid and current CVI II resolution is not equal to previous CVI II resolution, then:
    - Save the current resolution to previous resolution of CVI II.
    - Calculate resolution parameters based on the current resolution.
    - Program CVO II Mode Bank 0.
    - Select din_0 video input stream to CVO II.

Figure 2: Software main C++ execution flow



2.6.1 Software Parameter

VERBOSE parameter defined in main.cpp controls whether to print debug messages. Set to 1 to print parameter value being written to CVO II registers for debugging during each time format switching. Else, set to 0. By default, set this parameter to 0 to disable print messages to avoid Interrupt Service Routine deadlock when Nios II software is not in terminal mode.

2.6.2 Software Function Description

This section describes the functions used in main.cpp file. Some of the functions are provided as debug subroutine only.

Table 8: Functions in main.cpp file

Function Prototype	Description
struct traffic_detail{};	Define a group of data elements under a name called traffic_detail.
traffic_detail read_cvi_resolution (unsigned int base)	Function to read all resolution from CVI II control registers.
void set_cvo_reso_prog(unsigned int base, traffic_detail info, unsigned int bank_addr)	Function to set resolution parameters in CVO II mode banks. bank_addr parameter to determine which of the mode banks the subsequent writes are committed to.
void program_cvo_mode_bank (unsigned int base, traffic_detail cvi_info)	Call set_cvo_reso_prog function and print the parameter value being programmed to CVO II mode bank 0 if VERBOSE parameter is set to 1.
*void cvi_isr (void *context)*	CVI II core ISR. Upon an interrupt event (IRQ asserted), the function clear the CVI II IRQ and read the CVI II status registers specifically stable bit, resolution bit and vid_locked bit and ensure these three bits are asserted.
unsigned int current_vip_status (unsigned int base)	Function to read current status register of video IP cores.
unsigned int current_vip_irq (unsigned int base)	Function to read current interrupt register of video IP cores.
void read_cvi_register(unsigned int base)	Function to read CVI II status & interrupt registers.
void read_cvo_register(unsigned int base)	Function to print the message for CVO II registers.

3 Requirements

3.1 Hardware Requirements

The reference design requires the following hardware tools:

- Arria 10 GX FPGA Development Kit (10AX115S2F45I1SG)
- Terasic 12G SDI-FMC Daughter Card

- Micro-USB cable and power adaptor. *Note the Micro-USB cable and power adaptor are part of Arria 10 GX FPGA Development Kit accessories.*

3.2 Software Requirements

The following software is required at your PC to run the reference design:

- Quartus Prime Pro 19.2 Edition
- Nios II 19.2 Eclipse
- Clock Control GUI
  Download page: Arria 10 GX FPGA Development Kit

4 Reference Design Walkthrough

4.1 Running the Reference Design

To run the reference design, follow these steps:

1. Compile the project.
2. Setup the hardware.
3. Configuring the FPGA.
4. Building and running the Nios II software.

4.1.1 Compiling the Project

To download the reference design from the Design Store, follow these steps:

1. To test the reference design targeted for the Intel Arria 10 GX device, download the reference design file to your local project directory.
2. Launch the Intel Quartus Prime Pro Edition software.
3. To prepare the design template in the Intel Quartus Prime Pro Edition software GUI, click **File** menu and select **Open** and change the file type to the Intel Quartus Prime Design Template File (***.par**). Browse to the **.par** file and click **OK**.
4. To compile the project, navigate to the **Processing** menu and select **Start Compilation**.

4.1.2 Setup the Hardware

Before programming with the .sof file, ensure that the connections and settings of the boards are correct. For the hardware setup to run the reference design, follow these steps:

1. Connect the Terasic 12G SDI-FMC daughter card to the FMC port B on Intel Arria 10 GX FPGA development kit. For more information, refer to figure 3.
2. Ensure all switches on the Intel Arria 10 GX FPGA development kit are in default position. For more information, refer to Intel Arria 10 FPGA Development Kit User Guide.
3. Connect the Terasic daughter card BNC RX connector (12G-SDI In 0) to the SDI Signal Generator and the Terasic daughter card BNC TX connector (12G-SDI Out 0) to the SDI Signal Analyzer.

4. Connect the USB cable to the Micro USB blaster connector on the development kit.
5. Connect the power adapter to the power supply jack.
6. Turn ON the power for the Intel Arria 10 GX FPGA development kit. The hardware system is now ready for programming.
7. Complete the following steps to configure the output frequency of the programmable clock generator (Si5338 (U26)) used in this reference design:
    a. Download and unzip the *arria10GX_10ax115sf45_fpga_v15.1.2p2.zip* design package from Intel Arria 10 GX FPGA development kit web page.
    b. Launch the Intel Quartus Prime Pro Edition software and then run the *ClockController.exe* application from the */arria10GX_10ax115sf45_fpga_v15.1.2/examples/board_test_system* directory.
    c. Set the *CLK3* frequency to **133.33Mhz**.
    d. Close the *Clock Controller* application.

Figure 3: Intel Arria 10 GX FPGA development kit and Terasic 12G SDI-FMC daughter card

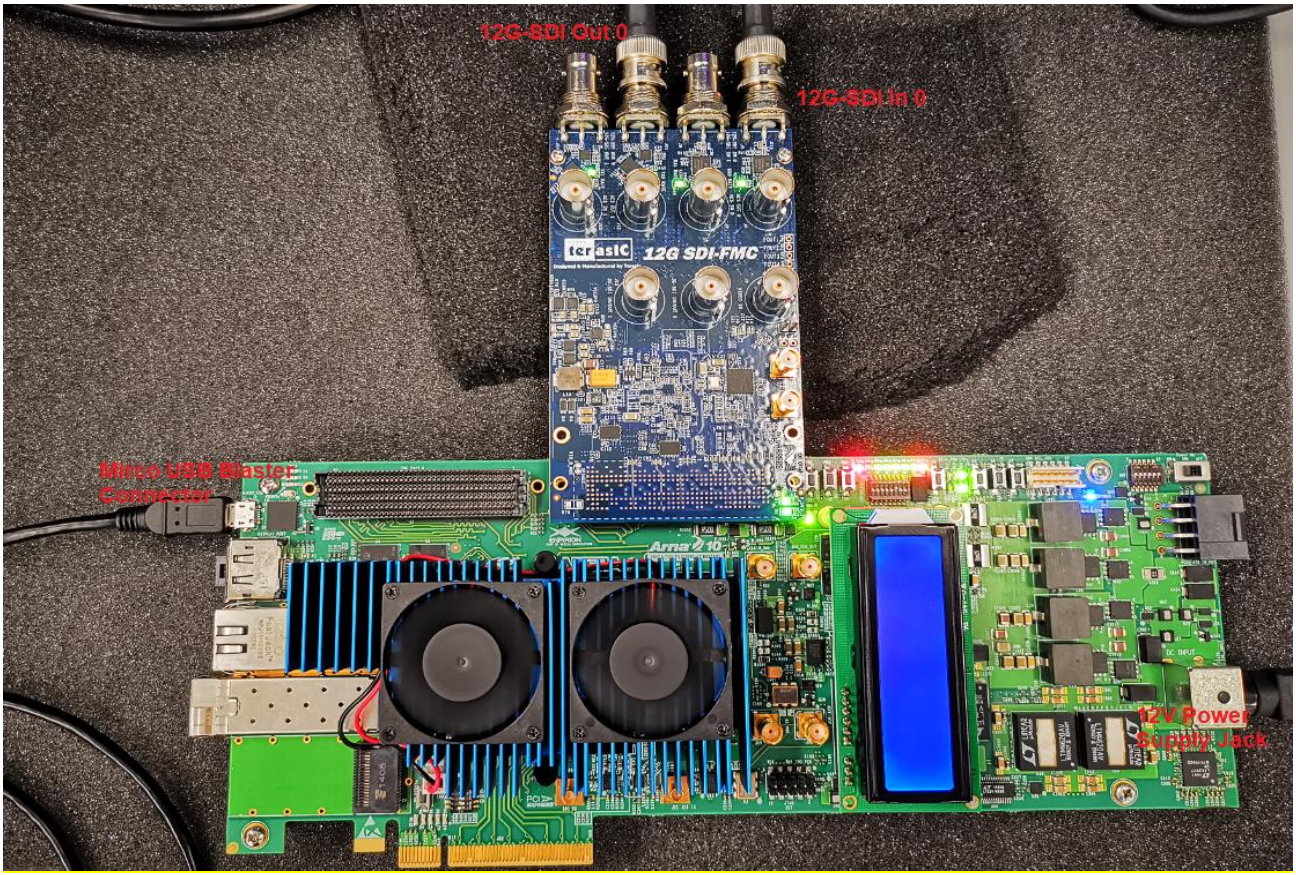
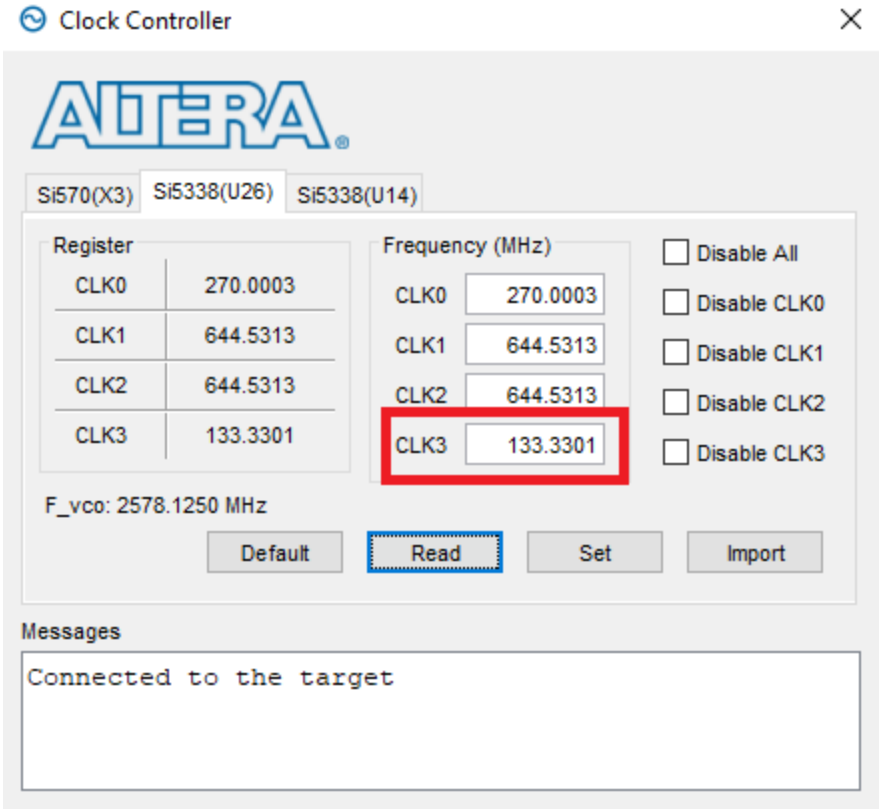


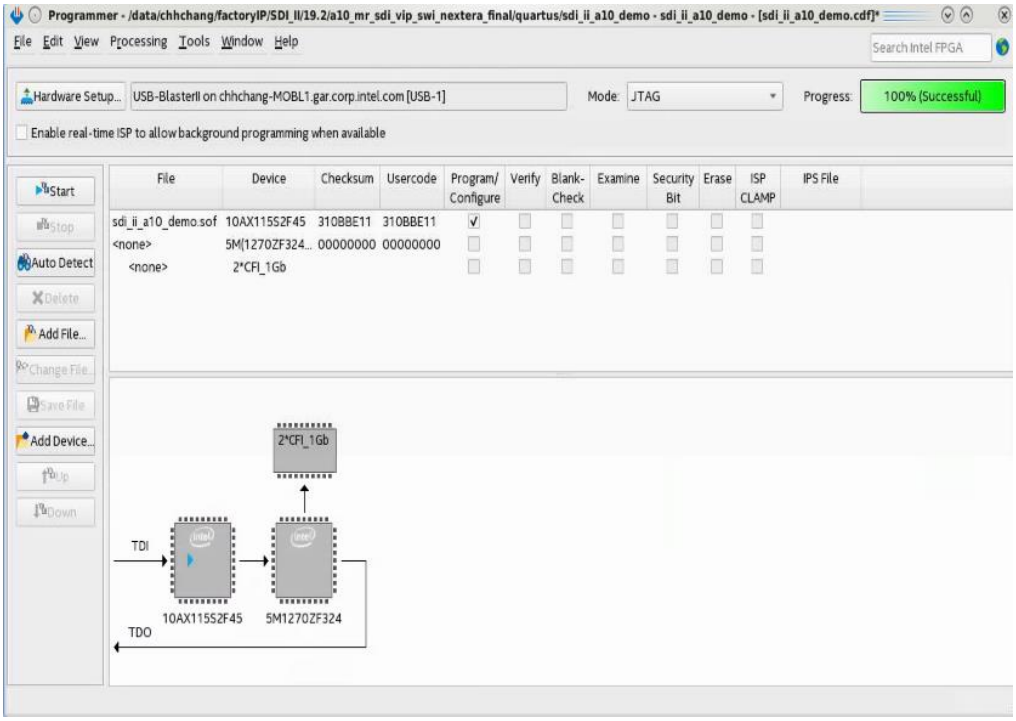
Figure 4: Clock controller – Si5338 (U26)



4.1.3 Configuring the FPGA

1. Before configure FPGA, ensure the following:
   a. The USB-blaster II driver is installed on host computer.
   b. The Arria 10 FPGA development kit is power-on.
   c. No other running application used the JTAG chain.
2. Click **Hardware Setup** to select the **Arria 10 GX Dev Kit [USB-1]**.
3. Click **Auto Detect** to display the devices in the JTAG chain and select device 10AX115S2.
4. Right click and select **Change File**. Next, choose the appropriate. sof file from the <project directory>/output_files directory and click **Open**.
5. Turn on the **Program/Configure** option for the. sof file.
6. Click **Start** to program the image into FPGA.

Figure 5: Quartus Prime Pro programmer



4.1.4 Building & Running Nios II Software

1. After device programming, open Nios II command shell and type “eclipse-nios2” command to launch **Nios II Software Build Tools for Eclipse**.
2. In the **Select a workspace** dialog box, navigate to the software workspace, **<project directory>/software** and click **OK**.
3. Create a new Nios II application and board support package (BSP) template. On the File menu, navigate to New and click **Nios II Application and BSP From Template**.
4. In the Nios II Application and BSP From Template window, enter the following information:
    - SOPC Information File Name: **<project directory>/rtl/vip_pipeline/vip/vip.sopcinfo**
    - Project name: **nios2_vip_ctrl** (You can choose any project name you like)
    - Project location: <project directory>/**software/ nios2_vip_ctrl**
    - Templates: Blank Project
5. Click **Next**. Verify that the default BSP name is **nios2_vip_ctrl _bsp**, then click **Finish**. The Nios II application project (**nios2_vip_ctrl**) and BSP (**nios2_vip_ctrl _bsp**) appears in the Project Explorer window.
   *Note: Whenever you regenerate the **vip.qsys**, you must regenerate the BSP files. In the Project Explorer window, right click the **nios2_vip_ctrl _bsp** project, navigate to Nios II and click **generate BSP**. This regenerates the BSP files based on your most current **vip.sopcinfo**.*

6. Import the design example source (*.cpp) into the application directory. In the Project Explorer window, right click on the **nios2_vip_ctrl** project and click **Import**.
7. In the Import window, select **General** > **File System** as the import source. Click **Next**.
8. Browse to the <project **directory>/software/source** directory. Check the **source** box on the left panel. This selects the source file in the **source** directory. Verify that the list of source file is as follows:
    - main.cpp

   Verify that the destination folder is **nios2_vip_ctrl**. Click **Finish**. The source file should be imported into the **nios2_vip_ctrl** project directory.

9. Expand the **nios2_vip_ctrl** application project in the Project Explorer window and verify that folder contains the required source file.
10. To compile the C++ code, navigate to the Project menu and select **Build all**. The compiler now compiles the C++ code into executable code.
11. To download the executable code to the development board, right click on the **nios2_vip_ctrl** in Project Explorer window and select **Run As** and then **Nios II Hardware**.
    Note: Press the push button (PB0) to trigger a device (Si5344) reset if you see no video image appear on SDI analyzer after board power up, FPGA programming and then Nios II hardware programming step completed. The reason is due to 297/296.7 MHz clock coming from Terasic 12G SDI-FMC daughter card is likely not stable during board power up.

4.2 Viewing the Results

4.2.1 Verify the Video Image and Jitter Reading at SDI Analyzer

The following figures shows examples of video image and jitter results captured at SDI analyzer.

Figure 6: Video image and jitter result for video format 3840x2160p59.94

Figure 7: Video image and jitter result for video format 3840x2160p60

Figure 8: Video image and jitter result for video format 1920x1080i59.94

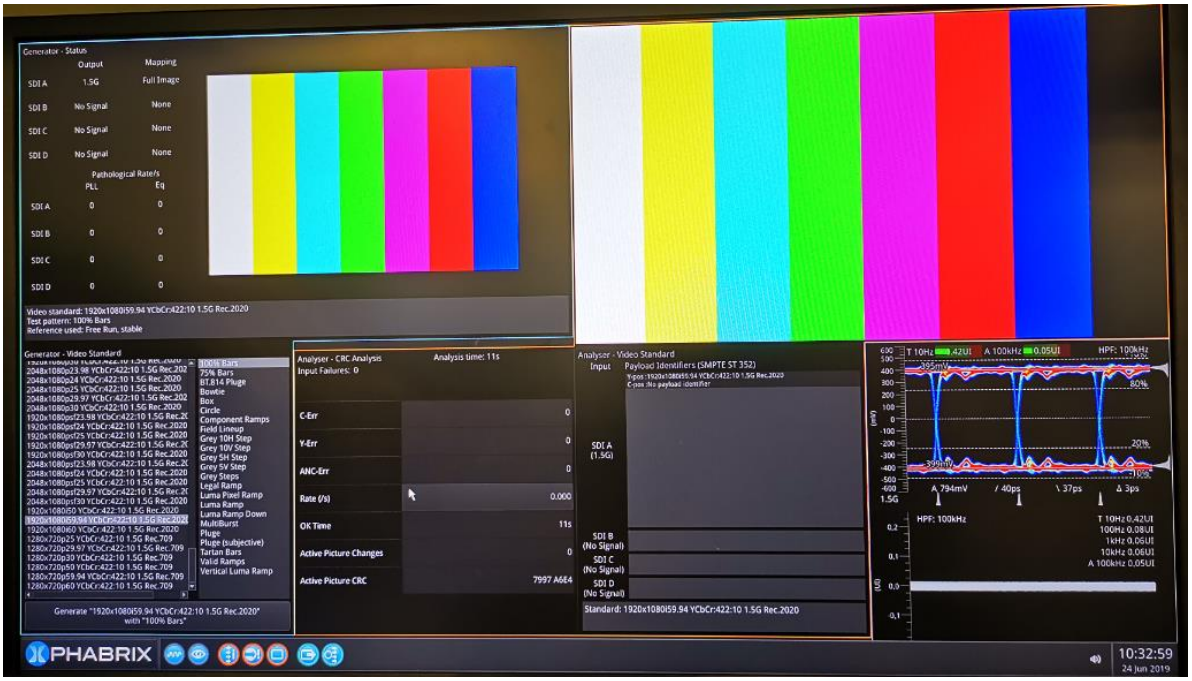


Figure 9: Video image and jitter result for video format 1920x1080i60

4.2.2 LEDs Indicator

The test uses the following LEDs to indicate the respective conditions. The user LEDs indicate the expected results. A logical 1 indicates that LED illuminates and a logical 0 indicates otherwise.

Figure 10: Example of illuminating user green LEDs when receiving 12G 8 SI format

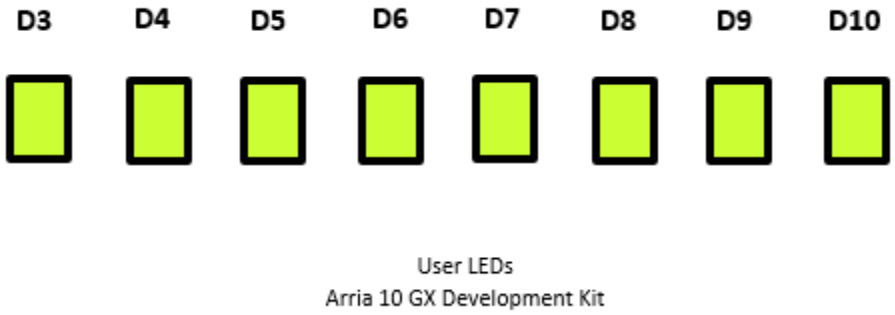
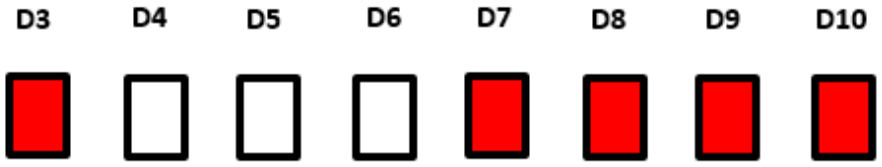


Table 9: Describe the functionality of user green LEDs

LED (Green)	Function
D3-D5	Indicate the SDI receiver video standards: - 000: SD-SDI - 001: HD-SDI - 010: 3G Level B (not supported) - 011: 3G Level A - 100: 6G 8 Stream Interleaved - 101: 6G 4 Stream Interleaved (not supported) - 110: 12G 16 Stream Interleaved (not supported) - 111: 12G 8 Stream Interleaved
D6	Shows slower version of the TX Transceiver parallel clock
D7	shows the slower version of the RX Transceiver parallel clock
D8	Illuminates when align_locked asserts
D9	Illuminates when trs_locked asserts
D10	Illuminates when frame_locked asserts

Figure 11: Example of illuminating user red LEDs when receiving 12G 8 SI format



User LEDs
Arria 10 GX Development Kit

Table 10: Describe the functionality of user red LEDs

LED (Red)	Function
D3	Illuminates when DDR4 status local calibration success
D4	Illuminates when DDR4 status local calibration failed
D5	Illuminates when CVI II overflow
D6	Illuminates when CVO II underflow
D7	Illuminates to indicates the data valid generated by the SDI II core
D8-D10	Indicate the SDI transmitter video standards output from CVO II video standards bus: - 000: SD-SDI - 001: HD-SDI - 010: 3G Level B (not supported) - 011: 3G Level A - 100: 6G 8 Stream Interleaved - 101: 6G 4 Stream Interleaved (not supported) - 110: 12G 16 Stream Interleaved (not supported) - 111: 12G 8 Stream Interleaved

Note: Refer to the video and Image Processing Suite User Guide regarding the supported and unsupported video formats by CVI II & CVO II IP cores.

4.3 Rebuilding Design

You need to regenerate the platform designer system if you modify the IP components settings and interfaces for any of system file, such as SDI RX, SDI TX, TX PLL or VIP system.

To regenerate the HDL files for any of the platform designer system file, follow these steps:

1. Open the system file (e.g. sdi_rx_sys.qsys, sdi_tx_sys.qsys, tx_pll.ip, vip.qsys)
2. Modify the parameter value according to your design specification
3. On **Generate** menu, select **Generate HDL** and then click **Generate** button.
4. Repeat the steps in subsection 4.1.1 to 4.1.4.

5 Document Revision History

Revision history for Intel Arria 10 GX Device Multi-Rate SDI II Pass-Through using Video and Image Processing Pipeline reference design user guide

Date	Version	Changes
June 2019	1.0	Initial release