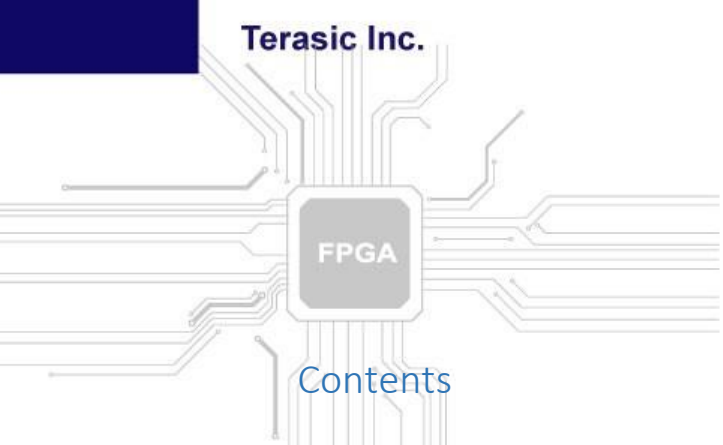
DE10-Agilex

for Intel® FPGA University Program

User Manual

terasIC
www.terasic.com



Contents

Chapter 1

Overview

This chapter provides an overview of the DE10-Agilex Development Board and installation guide.

1.1 General Description

Targeting the compute and acceleration needs from the edge to the core to the cloud, Terasic's DE10-Agilex accelerator is purpose-designed to meet the ever-increasing demands for acceleration, compute, and fast data movement.

The DE10-Agilex is based on the powerful Intel® Agilex™ FPGA to obtain speed and power breakthrough, with 40% higher performance, 40% lower power for equivalent performance. The accelerator includes PCI Express Gen 4.0 x16, two 200G QSFP-DD connectors and offers 32GB of DDR4 up to 680Gbps bandwidth to provide adaptable acceleration, maximum throughput and highly customizable processing of data for compute intensive applications.

The DE10-Agilex fully supports Intel® Open VINO™ toolkit, OpenCL™ BSP and Intel® oneAPI Toolkits to provide optimal Computer Vision and Deep Learning solutions. Our clients' systems can achieve highest computing performance and lowest cost for their Data Center and AI applications by leveraging the Agilex® FPGA on DE10-Agilex accelerator.

1.2 Key Features

The following hardware is implemented on the DE10-Agilex board:

Intel® Agilex™ F-Series

- AGFB014R24B2E2V or AGFB014R24B1E1V (Rev.C board)
  - 1,437K logic elements (LEs)
  - 139 Mbits embedded memory(M20K)
  - 9,020 18-bit x 19-bit multipliers
  - 4,510 Variable-precision DSP blocks
  - 1x PCIe Gen4 x16 Hard IP blocks
  - Transceivers
    - 16 x 28 Gbps NRZ transceivers (E-tile)
      - 8 of them can run 58 Gbps PAM4
    - 16 x 17.4 Gbps NRZ transceivers (P-tile)

JTAG Header and FPGA Configuration

- On-board USB Blaster II or JTAG header for use with the Quartus Prime Programmer
- MAX10 FPGA 10M04SCU169 System Controller and Avalon-ST x16 for configuration
- AS x4 configuration via EPCQ-L configuration device **(DNI)**

Memory devices

- 4 DDR4 SO-DIMM sockets, each supports up to 16GB ECC DDR4 SDRAM
- 128MB Flash(Connected to the System MAX10 FPGA)

General user I/O

- 4 user controllable LEDs
- 2 user push buttons
- 2 user dip switches

Clock interface

- 50MHz and 100Mhz Oscillators
- Programming PLL providing clock for QSFP-DD interface
- Dual clocks oscillators for DDR4 SDRAM SO-DIMM
- U.FL connector for external clock input
- One 2x5 GPIO timing expansion header

- **Communication Ports**
  - Two QSFP-DD Cages
    - Each can run up to 200Gbps
    - Compatible with QSFP and QSFP28
  - PCI Express x16 edge connector
    - Support for PCIe x16 Gen4
    - Edge connector for PC motherboard with x16 PCI Express slot
  - 4-Pin UART to USB (Integrated with USB-Blaster with USB Hub)

- **System Monitor and Control**
  - Dashboard System for System Management (Implement by System MAX10 FPGA)
  - Temperature sensor
  - Fan control
  - Power monitor

- **Power Source**
  - PCI Express 8-pin DC 12V power
  - PCI Express edge connector power

- **Mechanical Specification**
  - PCI Express full-height and 3/4-length

1.3. Block Diagram

Figure 1-1 shows the block diagram of the DE10-Agilex board. To provide maximum flexibility for the users, all key components are connected to the Agilex™ FPGA device. Thus, users can configure the FPGA to implement any system design.

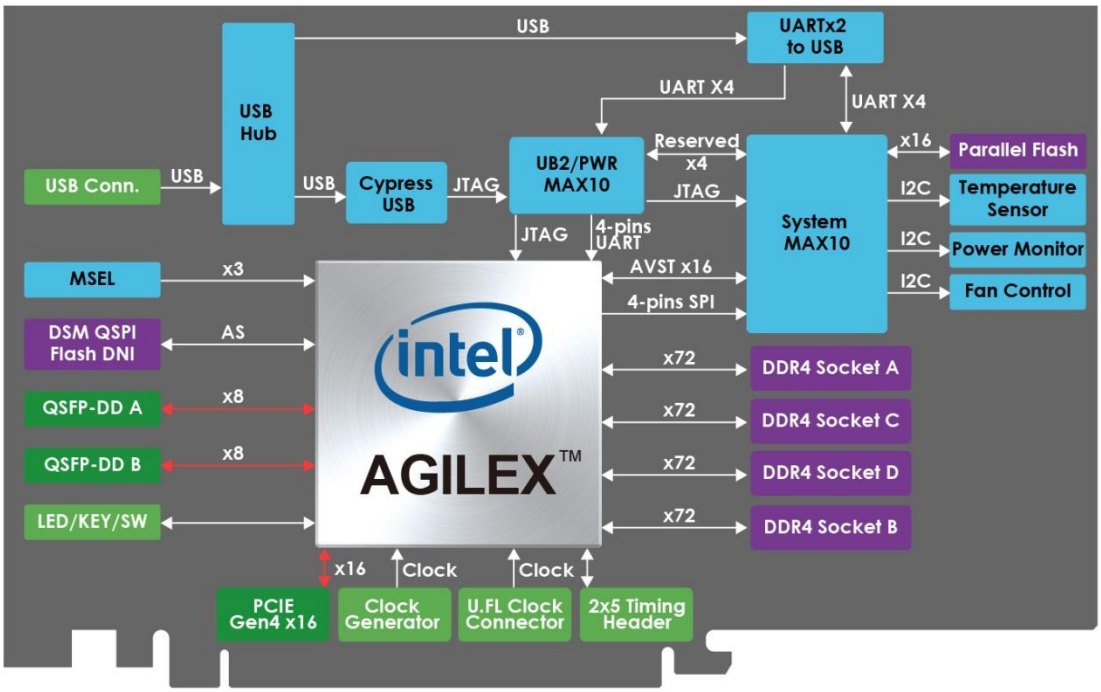


Figure 1-1 Block diagram of the DE10-Agilex board

1.4. Board Power On

The DE10-Agilex board can be used in stand-alone or be installed to the Host through PCIe slot. This section will introduce how to power on the board and the information that user should notice in these two modes.

■ Stand-alone Mode

When the DE10-Agilex board is used in stand-alone mode, users can use the 12V ATX power provided in the kit to connect to the 8-pin 12V ATX power connector (See **Figure 1-2**) of the DE10-Agilex board. To power up the board, user need to turn the power switch SW2 to “ON” position.

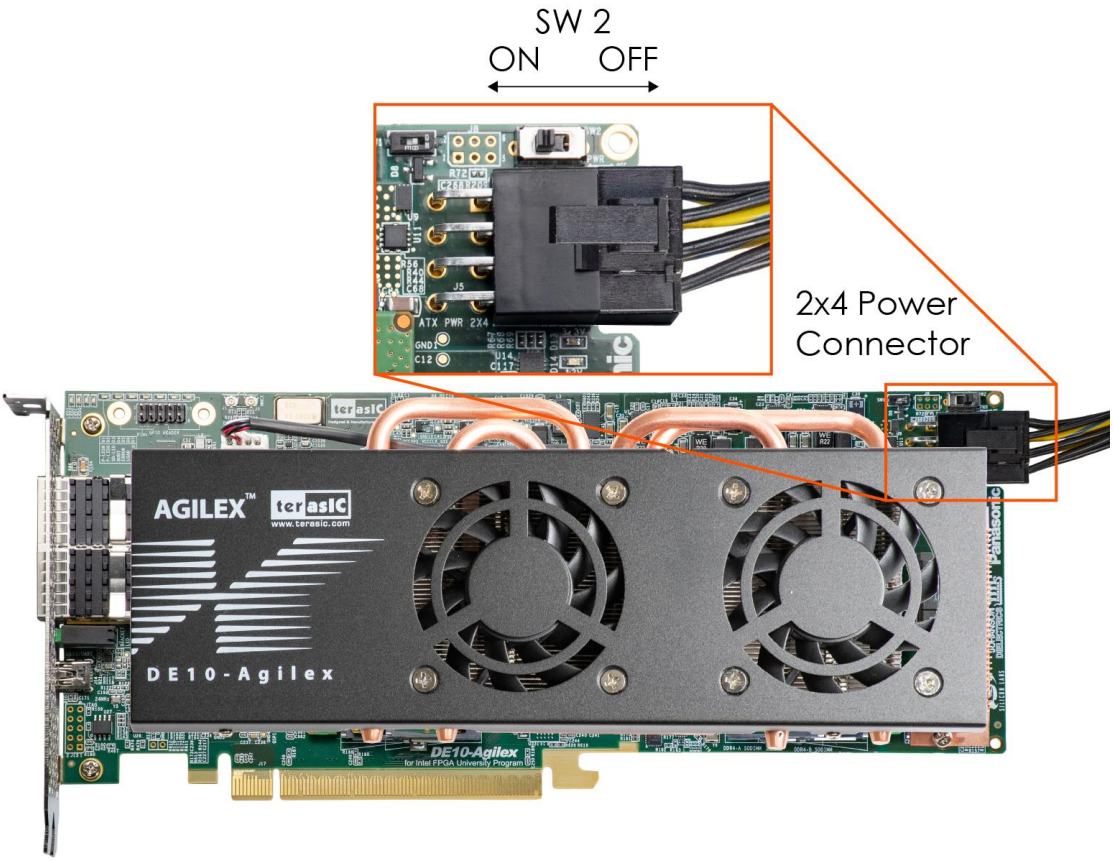


Figure 1-2 Board Power Control Switch

■ Install to Host

When the DE10-Agilex is installed on the Host via PCIe slot. Although the Host can provide power to DE10-Agilex board via PCIe slot, but Terasic strongly recommends that users connect an external power (through the 2x4 ATX power connector) to the board. This can prevent the power provided from Host unable to meet the power requirement of DE10-Agilex. If the power supply to the board is insufficient, it may cause some components to be abnormal.

In order to avoid insufficient power supply to the board, there is a force external power switch (**SW1**) on the board (See **Figure 1-4**). The **SW1** is default set as **ON**. When install the board on the PCIe slot in the PC, users must connect the 2x4 pin 12V DC external power connector to the board, otherwise the board will not be power on. This restriction is designed to avoid FPGA damage due to insufficient power. Users can set it as **OFF** if the FPGA utilization rate is low and PCIe edge power source is sufficient.

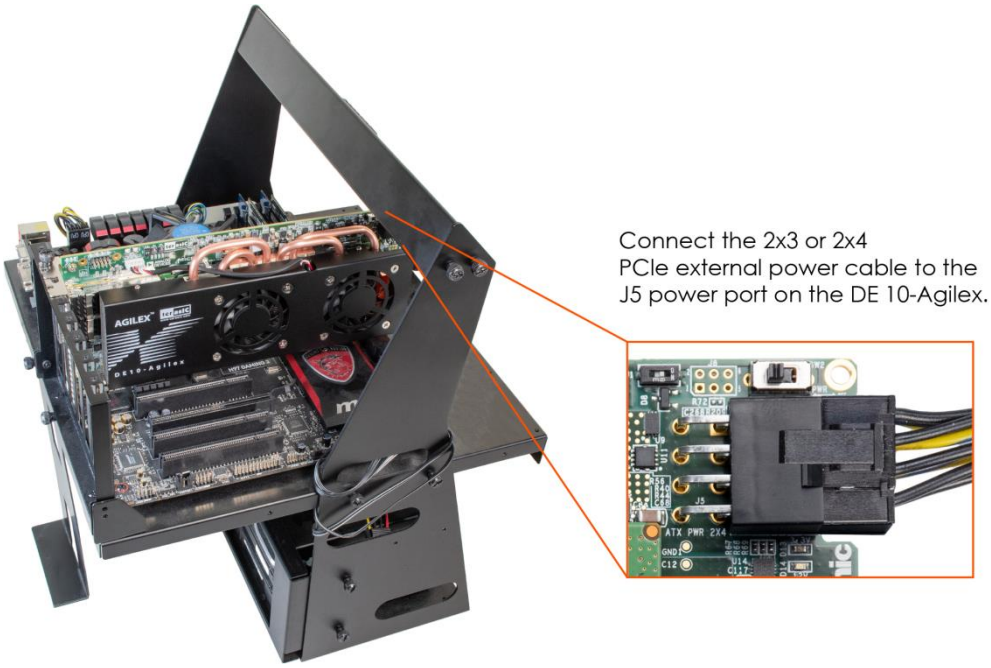


Figure 1-3 Plug external power on the board

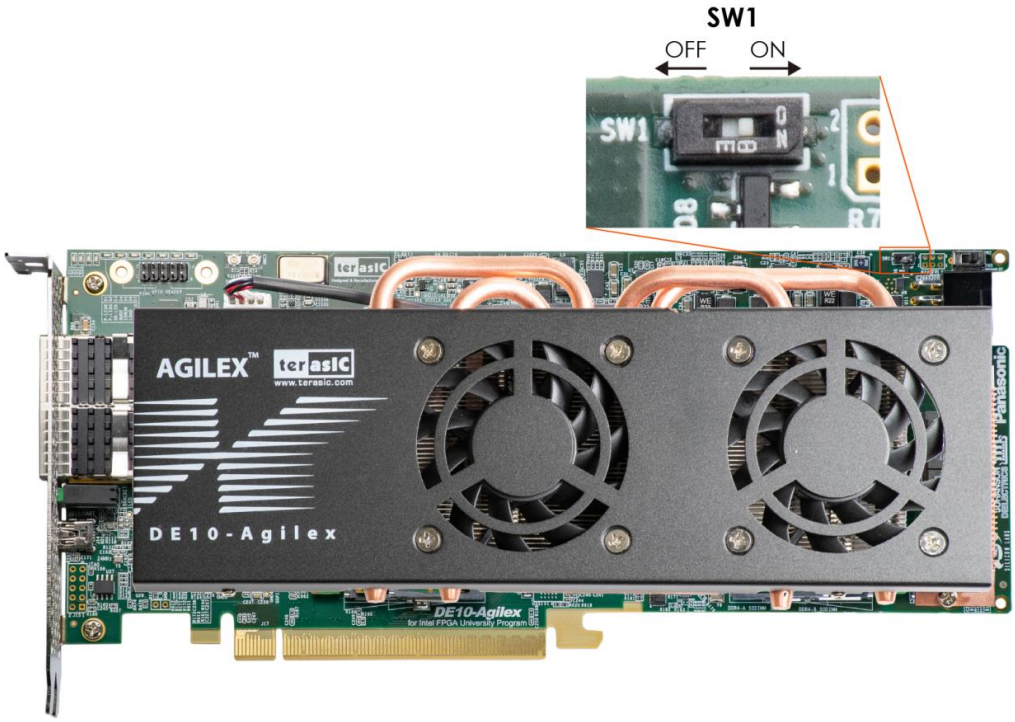


Figure 1-4 Force external power switch

1.5. Board Protection

The temperature of DE10-Agilex board will have a lot to do with the user's design code, chassis, and ambient temperature. When using DE10-Agilex board in the server. Customers should pay attention to whether the temperature of DE10-Agilex board is too high to avoid abnormal work for user’s design or even damage to the board.

The **Dashbaord_gui** software (see chapter 10 of this user manual) is provided in the system CD to allow users to monitor the temperature status of the board. A temperature monitor IP (see section 5.4 in the user manual) is also provided so that the user can directly monitor the temperature status in the Agilex FPGA.

If the board temperature is too high, it is recommended that customers can switch the PCIe slot position in the server chassis or increasing the fan strength in the chassis, or replace the chassis to a big space, or reduce the ambient temperature to improve cooling system.

In addition, the efficiency of the DE10-Agilex cooling system will decrease with the aging of dust and fans, so customers should re-evaluate the cooling efficiency regularly.

Chapter 2

Board Component

This chapter introduces all the important components on the DE10-Agilex.

2.1 Board Overview

Figure 2-1 and **Figure 2-2** is the top and bottom view of the DE10-Agilex development board. It depicts the layout of the board and indicates the location of the connectors and key components. Users can refer to this figure for relative location of the connectors and key components.

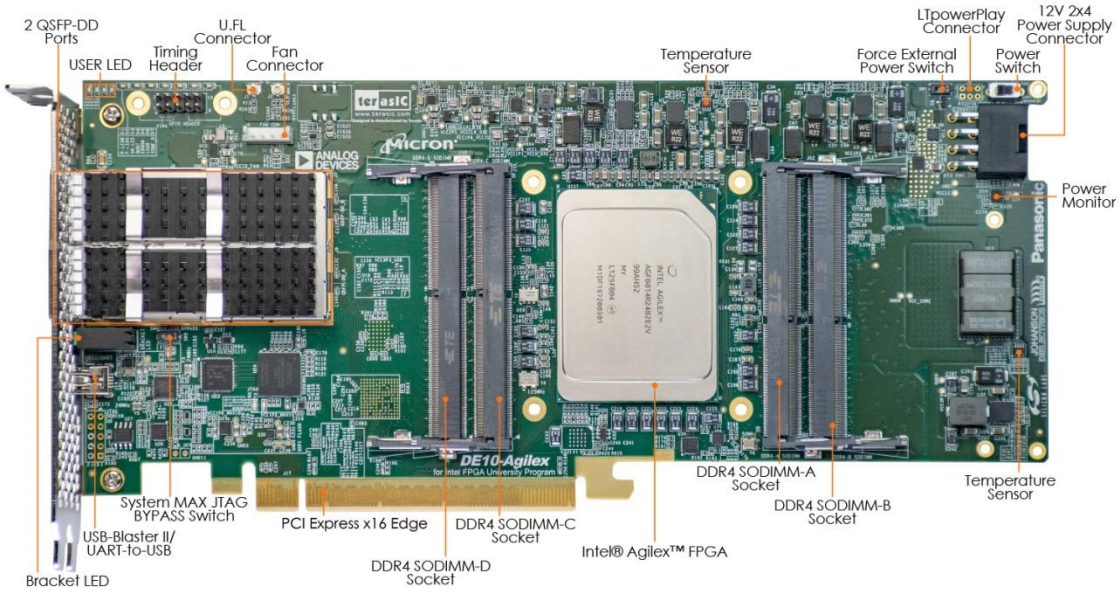


Figure 2-1 FPGA Board (Top)

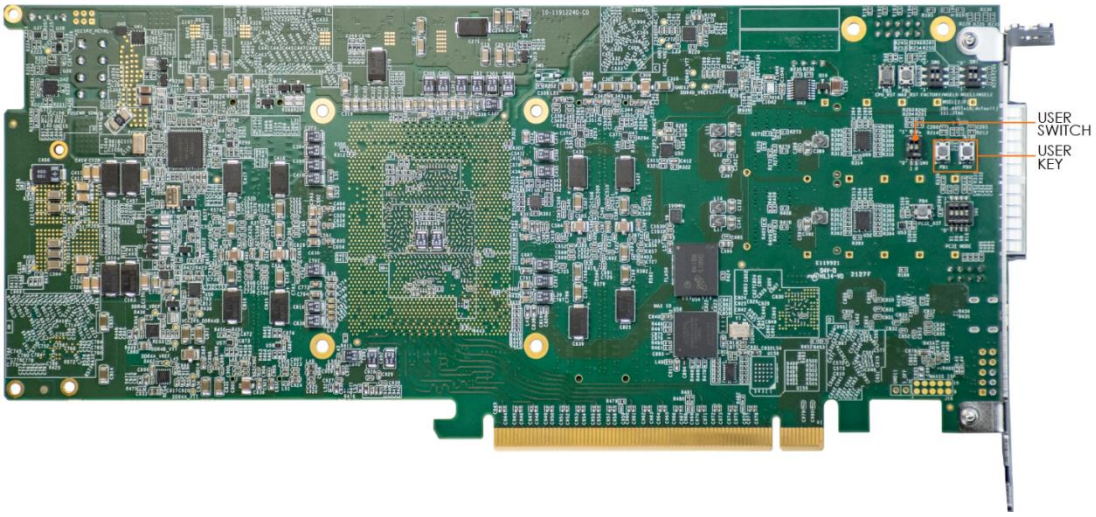


Figure 2-2 FPGA Board (Bottom)

2.2 Configuration

The FPGA board supports the following configuration methods for the Intel® Agilex™ FPGA:

- Avalon-ST x16
- JTAG
- Active Serial (AS) normal and fast modes (Need to Install Flash by user)

To switch these methods on the DE10-Agilex board, the user needs to switch the MSEL[2:0] pin of FPGA on **SW6** and **SW7** to change the configuration methods (See **Figure 2-3**). For details, please refer to **Setup Configure Mode** part of the section 2.3.

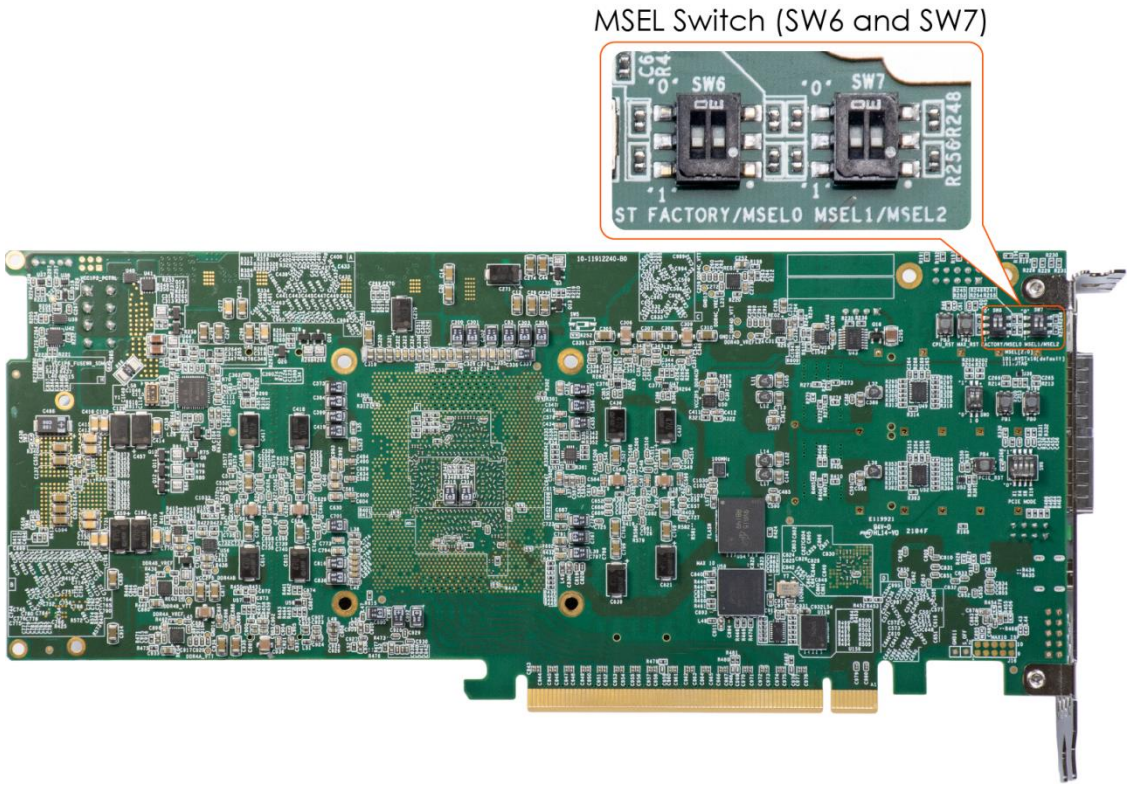


Figure 2-3 MSEL switch of the board

Detailed descriptions for each configuration methods are list in below.

■ Avalon-ST x16

Avalon-ST is the fastest configuration scheme for Intel® Agilex™ devices. When using this mode, while power up the DE10-Agilex board, the System MAX10 FPGA on the board (used as an external host) will read the configuration file in the Flash, and then programming the data into the FPGA through the Avalon-ST protocol.

On DE10-Agilex board, the data bus width of Avalon-ST mode is 16-bit (Avalon-ST x 16 mode). To set DE10-Agilex board to Avalon-ST mode, users need to set MSEL[2:0] to "101" (See **Setup Configure Mode** part of the section 2.3). For how to program the configuration file into the Flash, please refer to chapter 4.

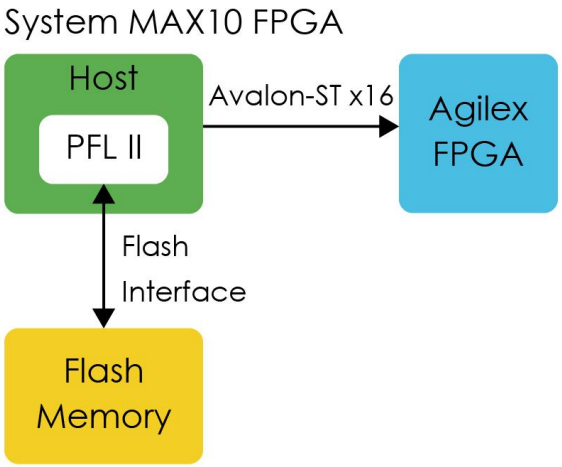


Figure 2-4 Block diagram of the Avalon-ST x 16 mode on the board

■ JTAG

JTAG-chain device programming is one of the most common and general methods. JTAG-chain device configuration uses the JTAG pins to configure the Intel® Agilex™ FPGA directly with the .sof file. Use this mode to configure FPGA, users do not need to modify MSEL specifically [2:0] pin, JTAG-chain device programming can be used under any setting. On DE10-Agilex board, users can use JTAG-chain device programming through the onboard USB blaster II circuit.

For JTAG programming by on-board USB-Blaster II, the following procedures show how to download a configuration bit stream into the Agilex FPGA:

- Make sure that power is provided to the FPGA board
- Connect your PC to the FPGA board using a Mini-USB cable and make sure the USB-Blaster II driver is installed on PC.
- Launch Quartus Prime programmer and make sure the USB-Blaster II is detected.
- In Quartus Prime Programmer, add the configuration bit stream file (.sof), check the associated “Program/Configure” item, and click “Start” to start FPGA programming.

■ Active Serial (AS)

A serial flash (QSPI interface) is used on the D10-Agilex board for AS mode boot. When the Agilex FPGA is running in AS mode, the SDM (Secure device manager) block in the FPGA will actively go to the serial flash to read the stored configuration file to boot the FPGA. To use AS mode, the MSEL[2:0] pins must be set to “**001**” (See

Setup Configure Mode part of the section 2.3 for detailed).User can the use Intel Quartus Prime Programmer tool to program the serial flash device through JTAG interface. The SDM in the FPGA will emulate AS programming. The manufacturer and part number of the serial flash are *Micron* and *MT25QU128ABA8E12-0SIT*.

.

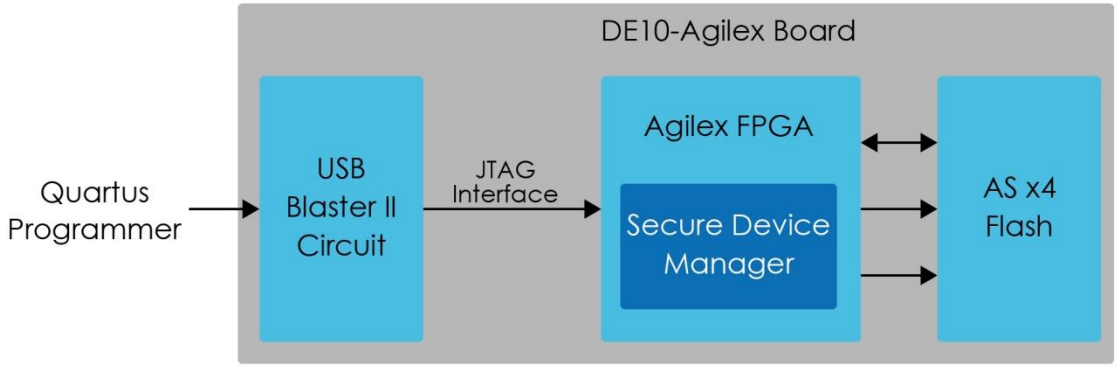


Figure 2-5 Block diagram of the Active serial mode on the board

2.3 Status and Setup Components

■ Status LED

The FPGA Board development board includes board-specific status LEDs to indicate board status. Please refer to **Figure 2-6** and **Table 2-1** for the description of the LED indicator.

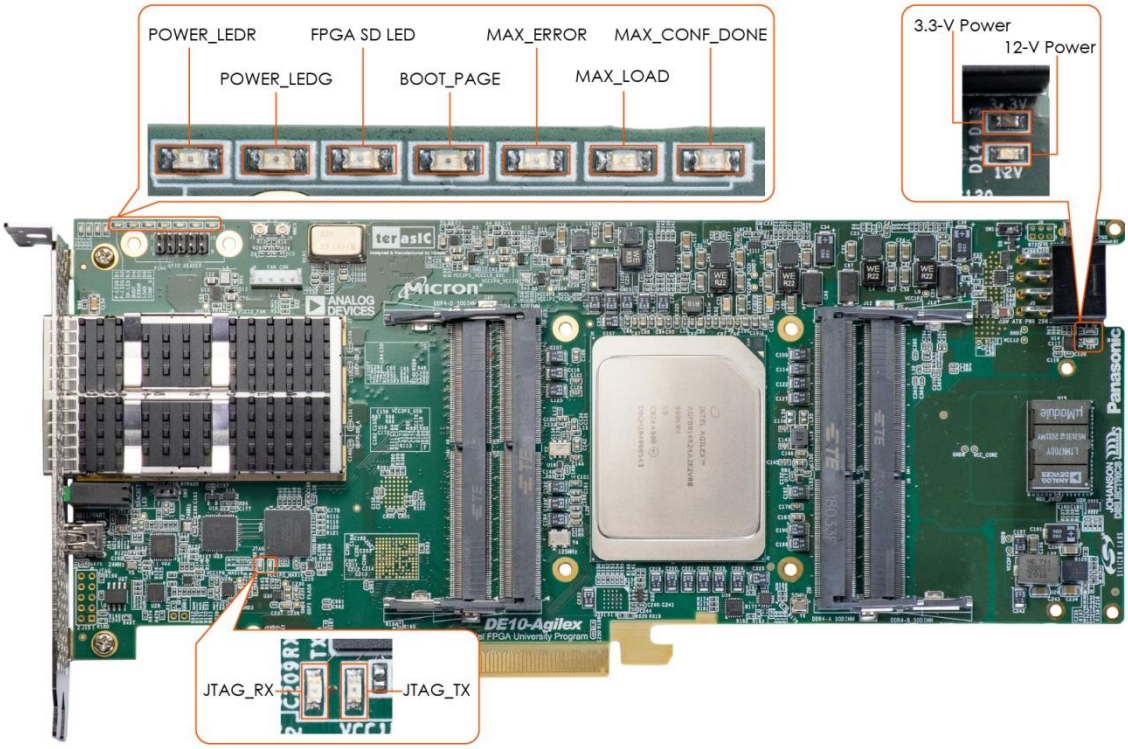


Figure 2-6 Position of the status LED

Table 2-1 Status LED

Board Reference	LED Name	Description
D14	12-V Power	Illuminates when 12-V power is active.
D13	3.3-V Power	Illuminates when 3.3-V power is active.
D11	FAN_ALERT_n	Illuminates when the fan is abnormal, such as when the fan speed is different from expected
D7	MAX_CONF_DONE	Illuminates when the FPGA is successfully configured.
D6	MAX_LOAD	Illuminates when the MAX 10 FPGA System Controller is actively configuring the FPGA.
D5	MAX_ERROR	Illuminates when the MAX 10 FPGA System Controller fails to configure the FPGA.
D4	BOOT_PAGE	Illuminates when FPGA is configured by the

		factory configuration bit stream.
D3	FPGA_SD_LED	Illuminates when the FPGA temperature or the board temperature exceeds 95 degrees or the power consumption exceeds 180W. Also, all the power of the FPGA will be cut off. (*1)
D2	POWER_LEDG	Illuminates when the 3.3V power good and power sequence process finished. (*1)
D1	POWER_LEDR	Illuminates when the 3.3V power abnormal or power sequence process failed. (*1)
D16	JTAG_RX	Illuminates when the USB Blaster II circuit is transmitting data
D17	JTAG_TX	Illuminates when the USB Blaster II circuit is receiving data

(*1) : ***This LED will light up for about 0.3 second when the board is powered up then turn off.***

■ Setup PCI Express Control DIP switch

The PCI Express Control DIP switch (SW8) is provided to enable or disable different configurations of the PCIe Connector. **Table 2-2** lists the switch controls and description.

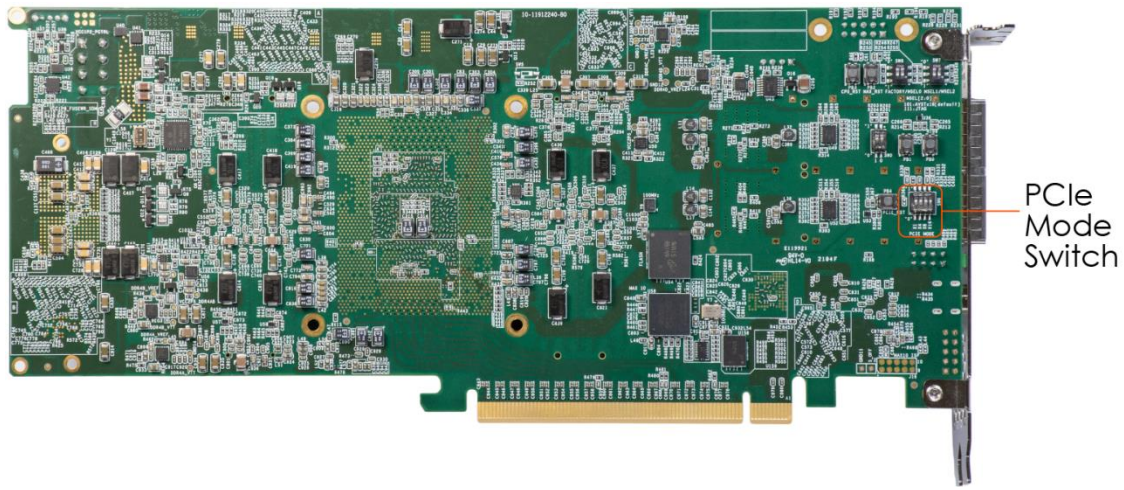


Figure 2-7 Position of the PCIe mode switch

Table 2-2 SW8 PCIe Control DIP Switch

Board Reference	Signal Name	Description	Default
SW8.1	PCIE_PRSNT2n_x1	On : Enable x1 presence detect Off: Disable x1 presence detect	Off
SW8.2	PCIE_PRSNT2n_x4	On : Enable x4 presence detect Off: Disable x4 presence detect	Off
SW8.3	PCIE_PRSNT2n_x8	On : Enable x8 presence detect Off: Disable x8 presence detect	Off
SW8.4	PCIE_PRSNT2n_x16	On : Enable x16 presence detect Off: Disable x16 presence detect	On

■ Setup Configure Mode

The SW7 and SW6 slide switches are used to specify the configuration mode of the FPGA. As currently only Avalon-ST x16 mode is supported (QSPI flash is not soldered on the board), please set MSEL[2:0] to 101 positions as shown in **Figure 2-8** and **Figure 2-9.**

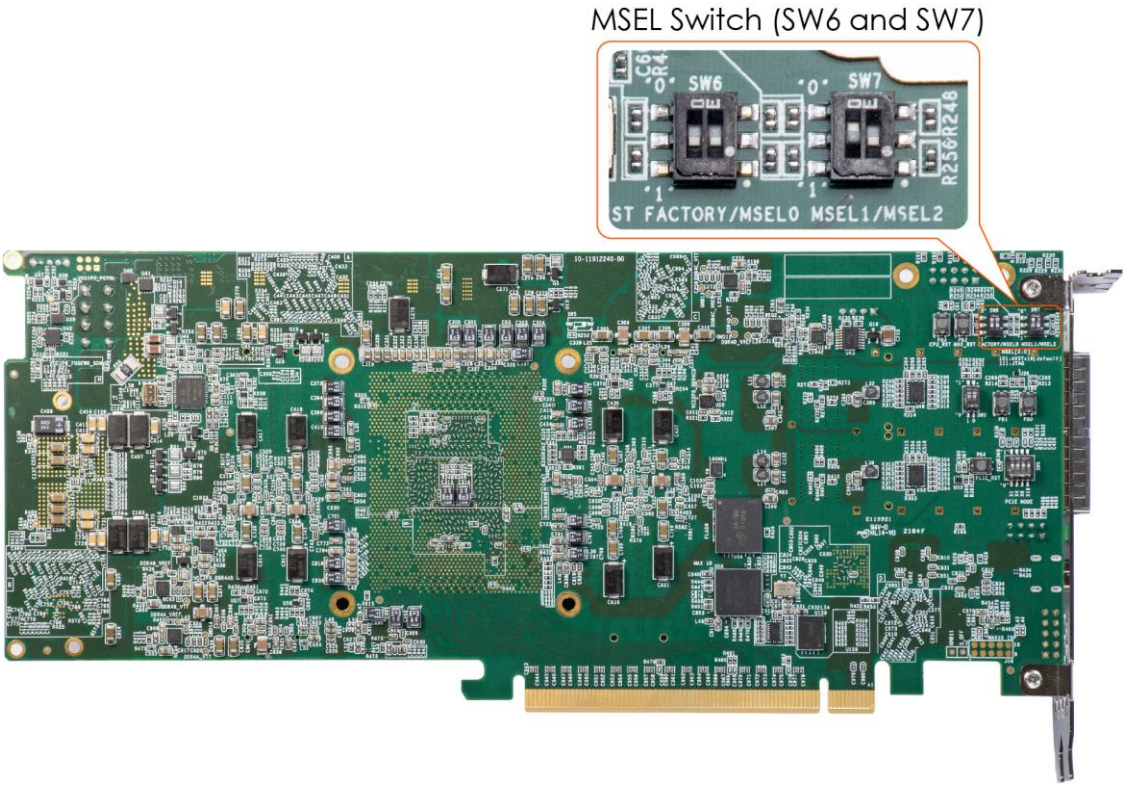


Figure 2-8 Position of slide switches SW6 and SW7 for Configuration Mode

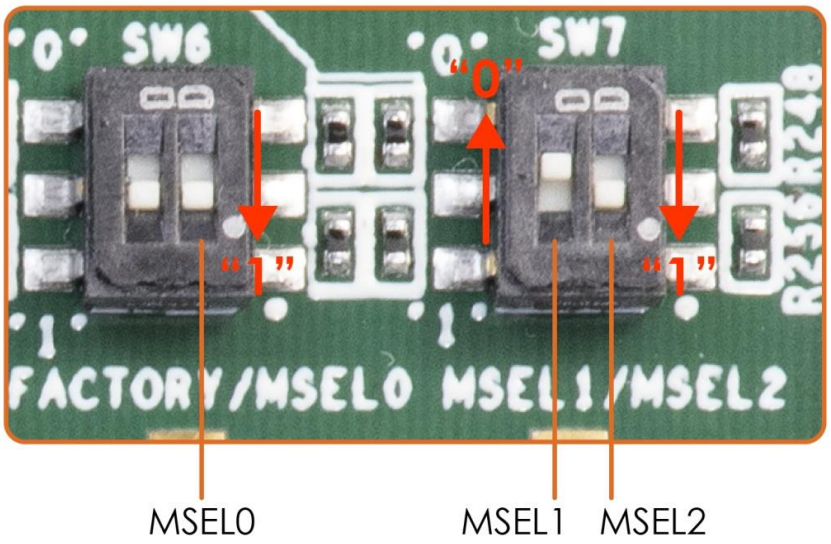


Figure 2-9 Position of slide switches SW6 and SW7 for Configuration Mode

Table 2-3 MSEL Settings for supported configuration Scheme of the board

FPGA Configuration Mode	MSEL2	MSEL1	MSEL0
Avalon-ST (x16)(Default)	1	0	1
AS (Fast mode – for CvP)(*2)	0	0	1

(*2) User need to solder a QSPI flash on the board by oneself.

■ Select Flash Image for Configuration

One of the position of slide switch SW6 is used to specify the image for configuration of the FPGA. Setting Position FACTORY of SW6 to “1” (down position) specifies the default factory image to be loaded, as shown in **Figure 2-10**. Setting Position FACTORY of SW6 to “0” (up position) specifies the DE10-Agilex to load a user-defined image.

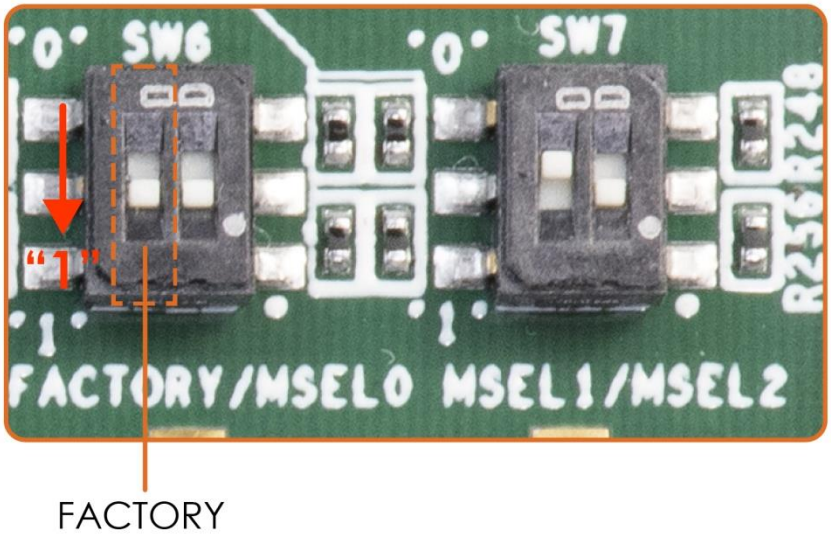


Figure 2-10 FACTORY position of slide switch SW6 for Image Select – Factory Image Load

■ System MAX JTAG Bypass Switch

The System MAX JTAG Bypass Switch(SW9) is a switch used to bypass the System MAX10 FPGA to the JTAG chain of the DE10-Agilex board. When this switch is switched to the "**OFF**" position, the system max10 FPGA will be in the JTAG chain. So users can use the JTAG interface to program the flash connected to the System MAX10 FPGA on the board. If the switch is switched to the "**ON**" position, there will only be Agilex FPGA on the JTAG chain on the entire board. This can speed up the scanning time for the FPGA device in the JTAG chain.

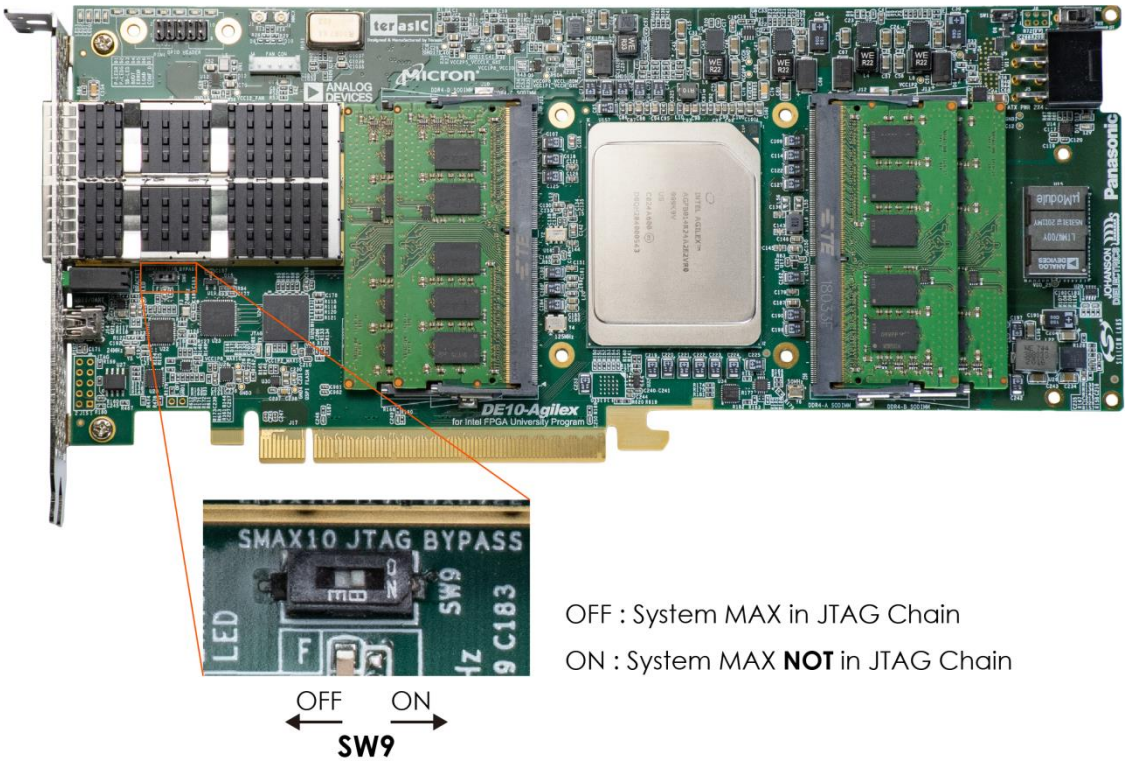


Figure 2-11 System MAX JTAG Bypass Switch

2.4 General User Input/Output

This section describes the user I/O interface of the FPGA. **Figure 2-12** shows the position of all these components and interface.

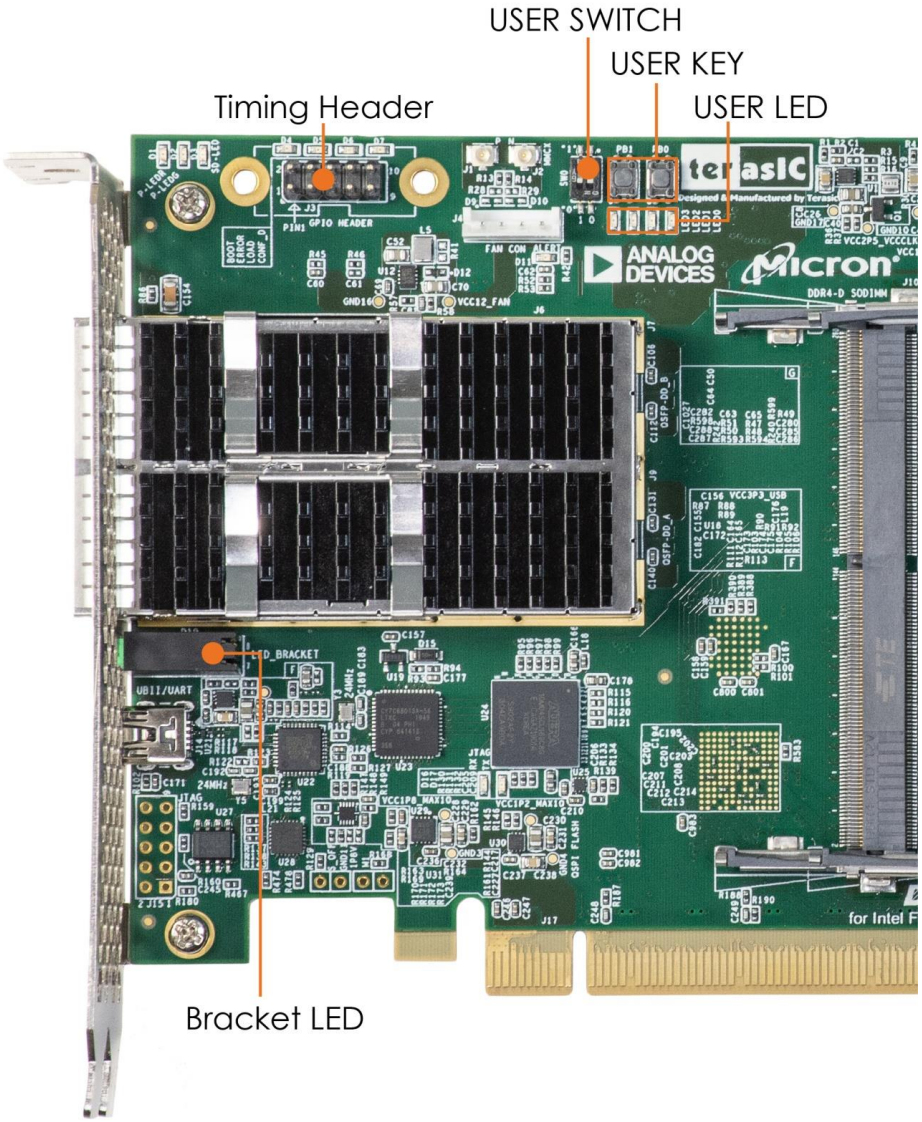


Figure 2-12 Position of all the general user components

■ User Defined Push-buttons

The FPGA board includes two user defined push-buttons that allow users to interact with the Agilex device. Each push-button provides a high logic level or a low logic level when it is not pressed or pressed, respectively. **Table 2-4** lists the board references, signal names and their corresponding Agilex device pin numbers.

Table 2-4 Push-button Pin Assignments, Schematic Signal Names, and Functions

Board Reference	Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
PB0	BUTTON0	High Logic Level when the button is not pressed	1.2V	PIN_J54
PB1	BUTTON1		1.2V	PIN_G54

There is also a reset button on the board connected to the Agilex FPGA and System MAX FPGA at the same time, allowing users to reset the logic in Agilex FPGA and System MAX 10 FPGA.

Table 2-5 CPU Reset Button Pin Assignments, Schematic Signal Names, and Functions

Board Reference	Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
PB3	CPU_RESET_n	High Logic Level when the button is not pressed	1.2V	PIN_G56

■ User-Defined Dip Switch

There are two positions dip switches (SW0 and SW1) on the FPGA board to provide additional FPGA input control. When a position of dip switch is in the DOWN position or the UPPER position, it provides a low logic level or a high logic level to the Agilex FPGA, respectively.

Table 2-6 lists the signal names and their corresponding Agilex device pin numbers.

Table 2-6 Dip Switch Pin Assignments, Schematic Signal Names, and Functions

Board Reference	Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
SW0	SW0	High logic level when SW in the	1.2V	PIN_H51

SW1	SW1	UPPER position.	1.2V	PIN_F51

■ User-Defined LEDs

The FPGA board consists of 4 user-controllable LEDs and a housing LED lamp with four LEDs to allow status and debugging signals to be driven to the LEDs from the designs loaded into the Agilex device. Each LED is driven directly by the Agilex FPGA. The LED is turned on or off when the associated pins are driven to a low or high logic level, respectively. A list of the pin names on the FPGA that are connected to the LEDs is given in **Table 2-7**.

Table 2-7 User LEDs Pin Assignments, Schematic Signal Names, and Functions

Board Reference	Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
LED0	LED0	Driving a logic 0 on the I/O port turns the LED ON. Driving a logic 1 on the I/O port turns the LED OFF.	1.2V	PIN_CR54
LED1	LED1		1.2V	PIN_DB57
LED2	LED2		1.2V	PIN_CY57
LED3	LED3		1.2V	PIN_CU52

Table 2-8 User LEDs Pin Assignments, Schematic Signal Names, and Functions

Board Reference	Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
D19.1	LED_BRACKET0	Driving a logic 0 on the I/O port turns the LED ON. Driving a logic 1 on the I/O port turns the LED OFF.	1.2V	PIN_CT57
D19.3	LED_BRACKET1		1.2V	PIN_CV55
D19.5	LED_BRACKET2		1.2V	PIN_CR56
D19.7	LED_BRACKET3		1.2V	PIN_CU56

■ 2x5 GPIO Header (Timing Expansion Header)

The FPGA board has one 2x5 GPIO header J5 for expansion function. The pin-out of J5 is shown in **Figure 2-13**. GPIO_P0 ~ GPIO_P3 are bi-direction 1.2V GPIO. GPIO_CLK0 and GPIO_CLK1 are connected to FPGA dedicated clock input and can be configured as two single-ended clock signals. **Table 2-9** shows the mapping of the FPGA pin assignments to the 2x5 GPIO header.

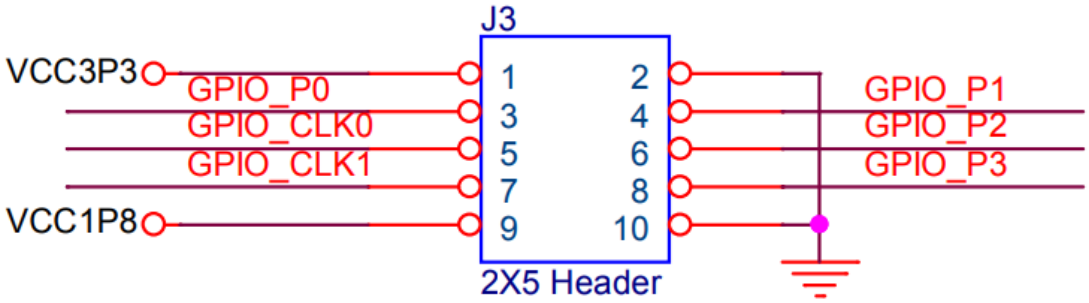


Figure 2-13 Pin-out of 2x5 expansion header

Table 2-9 2x5 GPIO Header Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
GPIO_P0	Bi-direction 1.2V GPIO	1.2V	PIN_CY21
GPIO_P1	Bi-direction 1.2V GPIO	1.2V	PIN_CT25
GPIO_P2	Bi-direction 1.2V GPIO	1.2V	PIN_CT17
GPIO_P3	Bi-direction 1.2V GPIO	1.2V	PIN_DC20
GPIO_CLK0	FPGA dedicated clock input or Bi-direction 1.2V GPIO	1.2V	PIN_DA22
GPIO_CLK1	FPGA dedicated clock input or Bi-direction 1.2V GPIO	1.2V	PIN_CR24

2.5 Clock Circuit

The development board includes several oscillator (50/100/125 MHz) and one programmable clock generators. **Figure 2-14** shows the default frequencies of on-board all external clocks going to the Agilex FPGA.

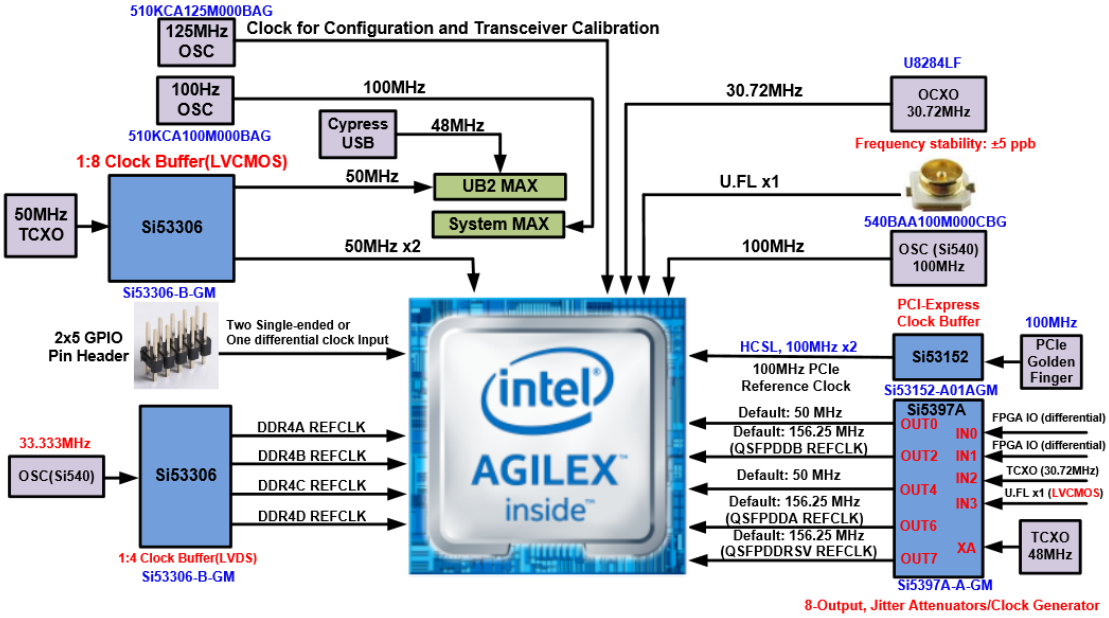


Figure 2-14 Clock circuit of the FPGA Board

A clock buffer (Si53306) is used to duplicate the 50 MHz TCXO output clock, so there are three 50MHz clocks fed into different Agilex FPGA banks and one clock for USB blaster II circuit. The programming clock generator (Si5397A) with low-jitter clock output which are used to provide special and high- quality clock signals for high-speed transceivers. Through I2C serial interface, the clock generator controllers in the Agilex FPGA can be used to program the Si5397As to generate many frequencies to each QSFP-DD port.

For memory interface, the board provide a 33.333Mhz clock and fan out it to four different clocks to the Agilex FPGA via clock buffer (Si53306). The four clocks are used for the reference clock of the four DDR4 SODIMMs.

Two UFL connectors provide two external single-ended clock inputs or one external differential clock inputs. One oscillator provides a 125 MHz clock used as configuration clock or used as the clock for transceiver calibration. Besides, there is one 100 MHz clock source to use as the FPGA input clock.

Finally, for PCIe application, the board uses the clock output on the PCIe edge connector as a clock source fed to the PCIe clock buffer, and outputs two reference clocks to Agilex FPGA.

Table 2-10 lists the clock source, signal names, default frequency and their corresponding Agilex device pin numbers.

Table 2-10 Clock Source, Signal Name, Default Frequency, Pin Assignments and Functions

Source	Schematic Signal Name	Default Frequency	I/O Standard	Agilex Pin Number	Application
U35	CLK_50_B3A	50.0 MHz	1.2V	PIN_G52	User application
	CLK_50_B3C		1.2V	PIN_G26	User application
Y8	CLK_100_B2A_p	100.0MHz	LVDS	PIN_CU50	User application
Y4	OSC_CLK_1	125MHz	LVDS	PIN_CC60	User-supplied configuration clock
J1	UFL_CLKIN	User Defined	*(1)	PIN_DC52	External Clock Input
U13	CLK_from_SI5397A_p0	50.0 MHz	LVDS	PIN_DC22	User application
	CLK_from_SI5397A_p1	50.0 MHz	LVDS	PIN_A24	User application
	QSFPDDA_REFCLK_p	156.25 MHz	LVDS	PIN_AJ12	QSFP-DD A port
	QSFPDDB_REFCLK_p	156.25 MHz	LVDS	PIN_AT13	QSFP-DD B port
	QSFPDDRSV_REFCLK_p	156.25 MHz	LVDS	PIN_AR14	Reserved for QSFP-DD port
U161	CLK_30M72	30.72MHz	1.2V	PIN_CU24	Reserved
U51	DDR4A_REFCLK_p	33.333 MHz	LVDS	PIN_A10	DDR4 reference clock for A port
	DDR4B_REFCLK_p	33.333 MHz	LVDS	PIN_L40	DDR4 reference clock for B port
	DDR4C_REFCLK_p	33.333 MHz	LVDS	PIN_DC8	DDR4 reference clock for C port
	DDR4D_REFCLK_p	33.333 MHz	LVDS	PIN_CN38	DDR4 reference clock for D port

*(1): The I/O standard of the **UFL_CLKIN** clock input from J1 to the FPGA needs to be

1.2V. If the input clock I/O standard from the J1 exceeds 1.2V, it should be divided to 1.2V by using two series resistors (R28 and R13). **Figure 2-15** shows that in the case of different input clock voltage level, the corresponding resistance value settings for R13 and R28 are required.

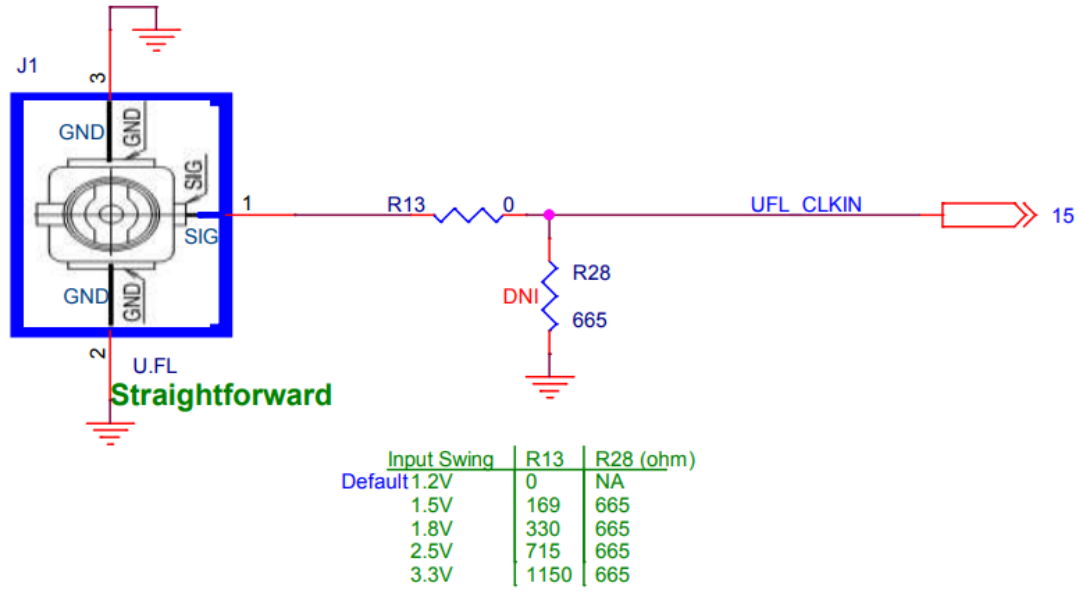


Figure 2-15 U.FL clock input signal level setting

Table 2-11 lists the programming clock generator (Si5397A and for QSFP-DD interface) control pin, signal names, I/O standard and their corresponding Agilex device pin numbers.

Table 2-11 Programmable clock generator control pin, Signal Name, I/O standard, Pin Assignments and Descriptions

Programmable clock generator	Schematic Signal Name	I/O Standard	Agilex Pin Number	Description
Si5397A (U13)	SI5397A_I2C_SCL	1.2V	PIN_CV27	I2C bus, connected with Si5397A
	SI5397A_I2C_SDA	1.2V	PIN_CR28	
	SI5397A_LOL	1.2V	PIN_CT19	Si5397A loss of lock, This pin indicates when the

				DSPLL™ is locked (high) or out-of-lock (low).
	SI5397A_LOS_XAXB	1.2V	PIN_CV19	Si5397A loss of XA/XB signal
	SI5397A_RST_n	1.2V	PIN_CU28	Si5397A reset signal
	SI5397A_OE_n	1.2V	PIN_CT29	Si5397A output enable signal

2.6 DDR4 SO-DIMM

The development board supports four independent banks of DDR4 SDRAM SO-DIMM. Each DDR4 SODIMM socket is wired to support a maximum capacity of 16GB (Dual rank) with ECC. Using differential DQS signaling for the DDR4 SDRAM interfaces, it is capable of running at up to 1333MHz(for AGFB014R24A2E2V FPGA device) or 1600Mhz(for AGFB014R24B1E1V FPGA device) memory clock for a maximum theoretical bandwidth up to 204.8Gbps (AGFB014R24B1E1V FPGA device). The memory clock of each DDR4 SO-DIMM SDRAM can be up to 1600 MHz (If plug single rank DDR4 SO-DIMM and use AGFB014R24B1E1V FPGA device). **Figure 2-16** shows the connections between the DDR4 SDRAM SO-DIMMs and Agilex FPGA..

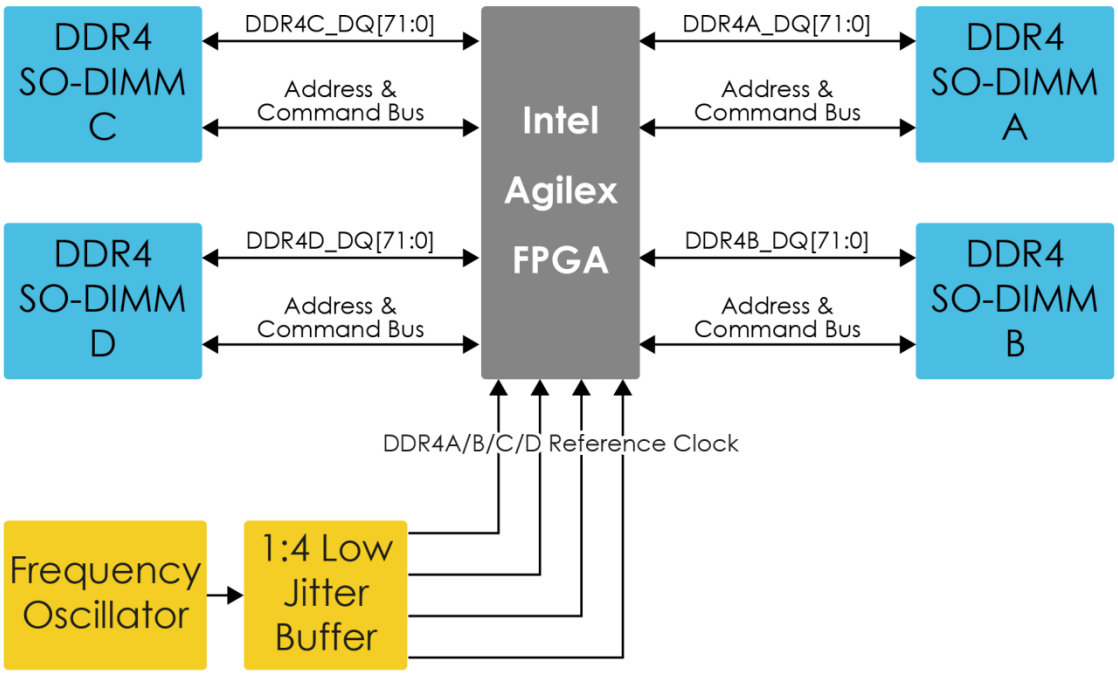


Figure 2-16 Connection between the DDR4 and Agilex FPGA

The pin assignments for DDR4 SDRAM SO-DIMM Bank-A, Bank-B, Bank-C and Bank-D are listed in **Table 2-12**, **Table 2-13**,**Table 2-14** and **Table 2-15**.

Table 2-12 DDR4-A Bank Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
DDR4A_DQ0	Data [0]	1.2V POD	PIN_W12
DDR4A_DQ1	Data [1]	1.2V POD	PIN_W16
DDR4A_DQ2	Data [2]	1.2V POD	PIN_U12
DDR4A_DQ3	Data [3]	1.2V POD	PIN_V13
DDR4A_DQ4	Data [4]	1.2V POD	PIN_T17
DDR4A_DQ5	Data [5]	1.2V POD	PIN_V17
DDR4A_DQ6	Data [6]	1.2V POD	PIN_T13
DDR4A_DQ7	Data [7]	1.2V POD	PIN_U16
DDR4A_DQ8	Data [8]	1.2V POD	PIN_N10
DDR4A_DQ9	Data [9]	1.2V POD	PIN_P5

DDR4A_DQ10	Data [10]	1.2V POD	PIN_L6
DDR4A_DQ11	Data [11]	1.2V POD	PIN_L10
DDR4A_DQ12	Data [12]	1.2V POD	PIN_N6
DDR4A_DQ13	Data [13]	1.2V POD	PIN_P9
DDR4A_DQ14	Data [14]	1.2V POD	PIN_M9
DDR4A_DQ15	Data [15]	1.2V POD	PIN_M5
DDR4A_DQ16	Data [16]	1.2V POD	PIN_U6
DDR4A_DQ17	Data [17]	1.2V POD	PIN_V9
DDR4A_DQ18	Data [18]	1.2V POD	PIN_W6
DDR4A_DQ19	Data [19]	1.2V POD	PIN_T9
DDR4A_DQ20	Data [20]	1.2V POD	PIN_W10
DDR4A_DQ21	Data [21]	1.2V POD	PIN_T5
DDR4A_DQ22	Data [22]	1.2V POD	PIN_V5
DDR4A_DQ23	Data [23]	1.2V POD	PIN_U10
DDR4A_DQ24	Data [24]	1.2V POD	PIN_N16
DDR4A_DQ25	Data [25]	1.2V POD	PIN_L12
DDR4A_DQ26	Data [26]	1.2V POD	PIN_M17
DDR4A_DQ27	Data [27]	1.2V POD	PIN_P17
DDR4A_DQ28	Data [28]	1.2V POD	PIN_P13
DDR4A_DQ29	Data [29]	1.2V POD	PIN_M13
DDR4A_DQ30	Data [30]	1.2V POD	PIN_L16
DDR4A_DQ31	Data [31]	1.2V POD	PIN_N12
DDR4A_DQ32	Data [32]	1.2V POD	PIN_B27
DDR4A_DQ33	Data [33]	1.2V POD	PIN_A30
DDR4A_DQ34	Data [34]	1.2V POD	PIN_B31
DDR4A_DQ35	Data [35]	1.2V POD	PIN_A26
DDR4A_DQ36	Data [36]	1.2V POD	PIN_C26
DDR4A_DQ37	Data [37]	1.2V POD	PIN_D27
DDR4A_DQ38	Data [38]	1.2V POD	PIN_D31
DDR4A_DQ39	Data [39]	1.2V POD	PIN_C30
DDR4A_DQ40	Data [40]	1.2V POD	PIN_M27
DDR4A_DQ41	Data [41]	1.2V POD	PIN_N26
DDR4A_DQ42	Data [42]	1.2V POD	PIN_N30
DDR4A_DQ43	Data [43]	1.2V POD	PIN_P31

DDR4A_DQ44	Data [44]	1.2V POD	PIN_P27
DDR4A_DQ45	Data [45]	1.2V POD	PIN_L26
DDR4A_DQ46	Data [46]	1.2V POD	PIN_L30
DDR4A_DQ47	Data [47]	1.2V POD	PIN_M31
DDR4A_DQ48	Data [48]	1.2V POD	PIN_T27
DDR4A_DQ49	Data [49]	1.2V POD	PIN_U30
DDR4A_DQ50	Data [50]	1.2V POD	PIN_W26
DDR4A_DQ51	Data [51]	1.2V POD	PIN_V31
DDR4A_DQ52	Data [52]	1.2V POD	PIN_V27
DDR4A_DQ53	Data [53]	1.2V POD	PIN_U26
DDR4A_DQ54	Data [54]	1.2V POD	PIN_T31
DDR4A_DQ55	Data [55]	1.2V POD	PIN_W30
DDR4A_DQ56	Data [56]	1.2V POD	PIN_V23
DDR4A_DQ57	Data [57]	1.2V POD	PIN_V19
DDR4A_DQ58	Data [58]	1.2V POD	PIN_U24
DDR4A_DQ59	Data [59]	1.2V POD	PIN_U20
DDR4A_DQ60	Data [60]	1.2V POD	PIN_W20
DDR4A_DQ61	Data [61]	1.2V POD	PIN_T23
DDR4A_DQ62	Data [62]	1.2V POD	PIN_W24
DDR4A_DQ63	Data [63]	1.2V POD	PIN_T19
DDR4A_DQ64	Data [64]	1.2V POD	PIN_M23
DDR4A_DQ65	Data [65]	1.2V POD	PIN_N20
DDR4A_DQ66	Data [66]	1.2V POD	PIN_L20
DDR4A_DQ67	Data [67]	1.2V POD	PIN_P23
DDR4A_DQ68	Data [68]	1.2V POD	PIN_P19
DDR4A_DQ69	Data [69]	1.2V POD	PIN_M19
DDR4A_DQ70	Data [70]	1.2V POD	PIN_L24
DDR4A_DQ71	Data [71]	1.2V POD	PIN_N24
DDR4A_DQS0	Data Strobe p[0]	DIFFERENTIAL 1.2V POD	PIN_T15
DDR4A_DQS_n0	Data Strobe n[0]	DIFFERENTIAL 1.2V POD	PIN_V15
DDR4A_DQS1	Data Strobe p[1]	DIFFERENTIAL 1.2V POD	PIN_L8

DDR4A_DQS_n1	Data Strobe n[1]	DIFFERENTIAL 1.2V POD	PIN_N8
DDR4A_DQS2	Data Strobe p[2]	DIFFERENTIAL 1.2V POD	PIN_U8
DDR4A_DQS_n2	Data Strobe n[2]	DIFFERENTIAL 1.2V POD	PIN_W8
DDR4A_DQS3	Data Strobe p[3]	DIFFERENTIAL 1.2V POD	PIN_M15
DDR4A_DQS_n3	Data Strobe n[3]	DIFFERENTIAL 1.2V POD	PIN_P15
DDR4A_DQS4	Data Strobe p[4]	DIFFERENTIAL 1.2V POD	PIN_B29
DDR4A_DQS_n4	Data Strobe n[4]	DIFFERENTIAL 1.2V POD	PIN_D29
DDR4A_DQS5	Data Strobe p[5]	DIFFERENTIAL 1.2V POD	PIN_M29
DDR4A_DQS_n5	Data Strobe n[5]	DIFFERENTIAL 1.2V POD	PIN_P29
DDR4A_DQS6	Data Strobe p[6]	DIFFERENTIAL 1.2V POD	PIN_T29
DDR4A_DQS_n6	Data Strobe n[6]	DIFFERENTIAL 1.2V POD	PIN_V29
DDR4A_DQS7	Data Strobe p[7]	DIFFERENTIAL 1.2V POD	PIN_U22
DDR4A_DQS_n7	Data Strobe n[7]	DIFFERENTIAL 1.2V POD	PIN_W22
DDR4A_DQS8	Data Strobe p[8]	DIFFERENTIAL 1.2V POD	PIN_L22
DDR4A_DQS_n8	Data Strobe n[8]	DIFFERENTIAL 1.2V POD	PIN_N22
DDR4A_DBI_n0	Data Bus Inversion [0]	1.2V POD	PIN_U14
DDR4A_DBI_n1	Data Bus Inversion [1]	1.2V POD	PIN_M7
DDR4A_DBI_n2	Data Bus Inversion	1.2V POD	PIN_T7

	[2]		
DDR4A_DBI_n3	Data Bus Inversion [3]	1.2V POD	PIN_L14
DDR4A_DBI_n4	Data Bus Inversion [4]	1.2V POD	PIN_A28
DDR4A_DBI_n5	Data Bus Inversion [5]	1.2V POD	PIN_L28
DDR4A_DBI_n6	Data Bus Inversion [6]	1.2V POD	PIN_U28
DDR4A_DBI_n7	Data Bus Inversion [7]	1.2V POD	PIN_T21
DDR4A_DBI_n8	Data Bus Inversion [8]	1.2V POD	PIN_M21
DDR4A_A0	Address [0]	SSTL-12	PIN_F17
DDR4A_A1	Address [1]	SSTL-12	PIN_H17
DDR4A_A2	Address [2]	SSTL-12	PIN_G16
DDR4A_A3	Address [3]	SSTL-12	PIN_J16
DDR4A_A4	Address [4]	SSTL-12	PIN_F15
DDR4A_A5	Address [5]	SSTL-12	PIN_H15
DDR4A_A6	Address [6]	SSTL-12	PIN_G14
DDR4A_A7	Address [7]	SSTL-12	PIN_J14
DDR4A_A8	Address [8]	SSTL-12	PIN_F13
DDR4A_A9	Address [9]	SSTL-12	PIN_H13
DDR4A_A10	Address [10]	SSTL-12	PIN_G12
DDR4A_A11	Address [11]	SSTL-12	PIN_J12
DDR4A_A12	Address [12]	SSTL-12	PIN_D9
DDR4A_A13	Address [13]	SSTL-12	PIN_A8
DDR4A_A14	Address [14]/ WE_n	SSTL-12	PIN_C8
DDR4A_A15	Address [15]/ CAS_n	SSTL-12	PIN_B7
DDR4A_A16	Address [16]/ RAS_n	SSTL-12	PIN_D7
DDR4A_BA0	Bank Select [0]	SSTL-12	PIN_C6

DDR4A_BA1	Bank Select [1]	SSTL-12	PIN_B5
DDR4A_BG0	Bank Group Select [0]	SSTL-12	PIN_D5
DDR4A_BG1	Bank Group Select [1]	SSTL-12	PIN_B17
DDR4A_CK0	Clock p	DIFFERENTIAL 1.2V SSTL	PIN_B13
DDR4A_CK_n0	Clock n	DIFFERENTIAL 1.2V SSTL	PIN_D13
DDR4A_CK1	Clock p	SSTL-12	PIN_F5
DDR4A_CK_n1	Clock n	SSTL-12	PIN_H5
DDR4A_CKE0	Clock Enable pin	SSTL-12	PIN_A14
DDR4A_CKE1	Clock Enable pin	SSTL-12	PIN_C14
DDR4A_ODT0	On Die Termination	SSTL-12	PIN_B15
DDR4A_ODT1	On Die Termination	SSTL-12	PIN_D15
DDR4A_CS_n0	Chip Select	SSTL-12	PIN_A16
DDR4A_CS_n1	Chip Select	SSTL-12	PIN_A12
DDR4A_PAR	Command and Address Parity Input	SSTL-12	PIN_C12
DDR4A_ALERT_n	Register ALERT_n output	1.2 V	PIN_A6
DDR4A_ACT_n	Activation Command Input	SSTL-12	PIN_C16
DDR4A_RESET_n	Chip Reset	1.2 V	PIN_D17
DDR4A_EVENT_n	Chip Temperature Event	1.2 V	PIN_F29
DDR4A_SDA	Chip I2C Serial Data Bus	1.2 V	PIN_J28
DDR4A_SCL	Chip I2C Serial Clock	1.2 V	PIN_G30
DDR4A_REFCLK_p	DDR4 A port Reference Clock p	True Differential Signaling	PIN_A10

DDR4A_REFCLK_n	DDR4 A port Reference Clock n	True Differential Signaling	PIN_C10
DDR4A_RZQ	Calibrated pins for OCT block	1.2 V	PIN_B9

Table 2-13 DDR4-B Bank Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
DDR4B_DQ0	Data [0]	1.2V POD	PIN_J44
DDR4B_DQ1	Data [1]	1.2V POD	PIN_J40
DDR4B_DQ2	Data [2]	1.2V POD	PIN_G44
DDR4B_DQ3	Data [3]	1.2V POD	PIN_F41
DDR4B_DQ4	Data [4]	1.2V POD	PIN_H45
DDR4B_DQ5	Data [5]	1.2V POD	PIN_F45
DDR4B_DQ6	Data [6]	1.2V POD	PIN_H41
DDR4B_DQ7	Data [7]	1.2V POD	PIN_G40
DDR4B_DQ8	Data [8]	1.2V POD	PIN_J34
DDR4B_DQ9	Data [9]	1.2V POD	PIN_H37
DDR4B_DQ10	Data [10]	1.2V POD	PIN_H33
DDR4B_DQ11	Data [11]	1.2V POD	PIN_F33
DDR4B_DQ12	Data [12]	1.2V POD	PIN_J38
DDR4B_DQ13	Data [13]	1.2V POD	PIN_G38
DDR4B_DQ14	Data [14]	1.2V POD	PIN_F37
DDR4B_DQ15	Data [15]	1.2V POD	PIN_G34
DDR4B_DQ16	Data [16]	1.2V POD	PIN_D41
DDR4B_DQ17	Data [17]	1.2V POD	PIN_A40
DDR4B_DQ18	Data [18]	1.2V POD	PIN_B45
DDR4B_DQ19	Data [19]	1.2V POD	PIN_C44
DDR4B_DQ20	Data [20]	1.2V POD	PIN_D45
DDR4B_DQ21	Data [21]	1.2V POD	PIN_C40
DDR4B_DQ22	Data [22]	1.2V POD	PIN_A44
DDR4B_DQ23	Data [23]	1.2V POD	PIN_B41
DDR4B_DQ24	Data [24]	1.2V POD	PIN_B33

DDR4B_DQ25	Data [25]	1.2V POD	PIN_A34
DDR4B_DQ26	Data [26]	1.2V POD	PIN_D37
DDR4B_DQ27	Data [27]	1.2V POD	PIN_A38
DDR4B_DQ28	Data [28]	1.2V POD	PIN_D33
DDR4B_DQ29	Data [29]	1.2V POD	PIN_C34
DDR4B_DQ30	Data [30]	1.2V POD	PIN_B37
DDR4B_DQ31	Data [31]	1.2V POD	PIN_C38
DDR4B_DQ32	Data [32]	1.2V POD	PIN_A54
DDR4B_DQ33	Data [33]	1.2V POD	PIN_D55
DDR4B_DQ34	Data [34]	1.2V POD	PIN_C58
DDR4B_DQ35	Data [35]	1.2V POD	PIN_F61
DDR4B_DQ36	Data [36]	1.2V POD	PIN_H61
DDR4B_DQ37	Data [37]	1.2V POD	PIN_C54
DDR4B_DQ38	Data [38]	1.2V POD	PIN_B55
DDR4B_DQ39	Data [39]	1.2V POD	PIN_D59
DDR4B_DQ40	Data [40]	1.2V POD	PIN_U48
DDR4B_DQ41	Data [41]	1.2V POD	PIN_V47
DDR4B_DQ42	Data [42]	1.2V POD	PIN_U52
DDR4B_DQ43	Data [43]	1.2V POD	PIN_W48
DDR4B_DQ44	Data [44]	1.2V POD	PIN_T51
DDR4B_DQ45	Data [45]	1.2V POD	PIN_T47
DDR4B_DQ46	Data [46]	1.2V POD	PIN_W52
DDR4B_DQ47	Data [47]	1.2V POD	PIN_V51
DDR4B_DQ48	Data [48]	1.2V POD	PIN_L58
DDR4B_DQ49	Data [49]	1.2V POD	PIN_L54
DDR4B_DQ50	Data [50]	1.2V POD	PIN_P55
DDR4B_DQ51	Data [51]	1.2V POD	PIN_P59
DDR4B_DQ52	Data [52]	1.2V POD	PIN_N54
DDR4B_DQ53	Data [53]	1.2V POD	PIN_M55
DDR4B_DQ54	Data [54]	1.2V POD	PIN_N58
DDR4B_DQ55	Data [55]	1.2V POD	PIN_M59
DDR4B_DQ56	Data [56]	1.2V POD	PIN_W58
DDR4B_DQ57	Data [57]	1.2V POD	PIN_V59
DDR4B_DQ58	Data [58]	1.2V POD	PIN_U58

DDR4B_DQ59	Data [59]	1.2V POD	PIN_V55
DDR4B_DQ60	Data [60]	1.2V POD	PIN_U54
DDR4B_DQ61	Data [61]	1.2V POD	PIN_T55
DDR4B_DQ62	Data [62]	1.2V POD	PIN_T59
DDR4B_DQ63	Data [63]	1.2V POD	PIN_W54
DDR4B_DQ64	Data [64]	1.2V POD	PIN_L48
DDR4B_DQ65	Data [65]	1.2V POD	PIN_P47
DDR4B_DQ66	Data [66]	1.2V POD	PIN_N48
DDR4B_DQ67	Data [67]	1.2V POD	PIN_M51
DDR4B_DQ68	Data [68]	1.2V POD	PIN_P51
DDR4B_DQ69	Data [69]	1.2V POD	PIN_M47
DDR4B_DQ70	Data [70]	1.2V POD	PIN_L52
DDR4B_DQ71	Data [71]	1.2V POD	PIN_N52
DDR4B_DQS0	Data Strobe p[0]	DIFFERENTIAL 1.2V POD	PIN_G42
DDR4B_DQS_n0	Data Strobe n[0]	DIFFERENTIAL 1.2V POD	PIN_J42
DDR4B_DQS1	Data Strobe p[1]	DIFFERENTIAL 1.2V POD	PIN_F35
DDR4B_DQS_n1	Data Strobe n[1]	DIFFERENTIAL 1.2V POD	PIN_H35
DDR4B_DQS2	Data Strobe p[2]	DIFFERENTIAL 1.2V POD	PIN_A42
DDR4B_DQS_n2	Data Strobe n[2]	DIFFERENTIAL 1.2V POD	PIN_C42
DDR4B_DQS3	Data Strobe p[3]	DIFFERENTIAL 1.2V POD	PIN_B35
DDR4B_DQS_n3	Data Strobe n[3]	DIFFERENTIAL 1.2V POD	PIN_D35
DDR4B_DQS4	Data Strobe p[4]	DIFFERENTIAL 1.2V POD	PIN_A56
DDR4B_DQS_n4	Data Strobe n[4]	DIFFERENTIAL 1.2V POD	PIN_C56
DDR4B_DQS5	Data Strobe p[5]	DIFFERENTIAL 1.2V	PIN_T49

		POD	
DDR4B_DQS_n5	Data Strobe n[5]	DIFFERENTIAL 1.2V POD	PIN_V49
DDR4B_DQS6	Data Strobe p[6]	DIFFERENTIAL 1.2V POD	PIN_L56
DDR4B_DQS_n6	Data Strobe n[6]	DIFFERENTIAL 1.2V POD	PIN_N56
DDR4B_DQS7	Data Strobe p[7]	DIFFERENTIAL 1.2V POD	PIN_U56
DDR4B_DQS_n7	Data Strobe n[7]	DIFFERENTIAL 1.2V POD	PIN_W56
DDR4B_DQS8	Data Strobe p[8]	DIFFERENTIAL 1.2V POD	PIN_M49
DDR4B_DQS_n8	Data Strobe n[8]	DIFFERENTIAL 1.2V POD	PIN_P49
DDR4B_DBI_n0	Data Bus Inversion [0]	1.2V POD	PIN_F43
DDR4B_DBI_n1	Data Bus Inversion [1]	1.2V POD	PIN_G36
DDR4B_DBI_n2	Data Bus Inversion [2]	1.2V POD	PIN_B43
DDR4B_DBI_n3	Data Bus Inversion [3]	1.2V POD	PIN_A36
DDR4B_DBI_n4	Data Bus Inversion [4]	1.2V POD	PIN_B57
DDR4B_DBI_n5	Data Bus Inversion [5]	1.2V POD	PIN_U50
DDR4B_DBI_n6	Data Bus Inversion [6]	1.2V POD	PIN_M57
DDR4B_DBI_n7	Data Bus Inversion [7]	1.2V POD	PIN_T57
DDR4B_DBI_n8	Data Bus Inversion [8]	1.2V POD	PIN_L50
DDR4B_A0	Address [0]	SSTL-12	PIN_T33
DDR4B_A1	Address [1]	SSTL-12	PIN_V33

DDR4B_A2	Address [2]	SSTL-12	PIN_U34
DDR4B_A3	Address [3]	SSTL-12	PIN_W34
DDR4B_A4	Address [4]	SSTL-12	PIN_T35
DDR4B_A5	Address [5]	SSTL-12	PIN_V35
DDR4B_A6	Address [6]	SSTL-12	PIN_U36
DDR4B_A7	Address [7]	SSTL-12	PIN_W36
DDR4B_A8	Address [8]	SSTL-12	PIN_T37
DDR4B_A9	Address [9]	SSTL-12	PIN_V37
DDR4B_A10	Address [10]	SSTL-12	PIN_U38
DDR4B_A11	Address [11]	SSTL-12	PIN_W38
DDR4B_A12	Address [12]	SSTL-12	PIN_P41
DDR4B_A13	Address [13]	SSTL-12	PIN_L42
DDR4B_A14	Address [14]/ WE_n	SSTL-12	PIN_N42
DDR4B_A15	Address [15]/ CAS_n	SSTL-12	PIN_M43
DDR4B_A16	Address [16]/ RAS_n	SSTL-12	PIN_P43
DDR4B_BA0	Bank Select [0]	SSTL-12	PIN_N44
DDR4B_BA1	Bank Select [1]	SSTL-12	PIN_M45
DDR4B_BG0	Bank Group Select [0]	SSTL-12	PIN_P45
DDR4B_BG1	Bank Group Select [1]	SSTL-12	PIN_M33
DDR4B_CK0	Clock p	DIFFERENTIAL 1.2V SSTL	PIN_M37
DDR4B_CK_n0	Clock n	DIFFERENTIAL 1.2V SSTL	PIN_P37
DDR4B_CK1	Clock p	SSTL-12	PIN_T45
DDR4B_CK_n1	Clock n	SSTL-12	PIN_V45
DDR4B_CKE0	Clock Enable pin	SSTL-12	PIN_L36
DDR4B_CKE1	Clock Enable pin	SSTL-12	PIN_N36
DDR4B_ODT0	On Die Termination	SSTL-12	PIN_M35

DDR4B_ODT1	On Die Termination	SSTL-12	PIN_P35
DDR4B_CS_n0	Chip Select	SSTL-12	PIN_L34
DDR4B_CS_n1	Chip Select	SSTL-12	PIN_L38
DDR4B_PAR	Command and Address Parity Input	SSTL-12	PIN_N38
DDR4B_ALERT_n	Register ALERT_n output	1.2 V	PIN_L44
DDR4B_ACT_n	Activation Command Input	SSTL-12	PIN_N34
DDR4B_RESET_n	Chip Reset	1.2 V	PIN_P33
DDR4B_EVENT_n	Chip Temperature Event	1.2 V	PIN_J56
DDR4B_SDA	Chip I2C Serial Data Bus	1.2 V	PIN_B19
DDR4B_SCL	Chip I2C Serial Clock	1.2 V	PIN_H55
DDR4B_REFCLK_p	DDR4 A port Reference Clock p	True Differential Signaling	PIN_L40
DDR4B_REFCLK_n	DDR4 A port Reference Clock n	True Differential Signaling	PIN_N40
DDR4B_RZQ	Calibrated pins for OCT block	1.2 V	PIN_M41

Table 2-14 DDR4-C Bank Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
DDR4C_DQ0	Data [0]	1.2V POD	PIN_CH25
DDR4C_DQ1	Data [1]	1.2V POD	PIN_CF29
DDR4C_DQ2	Data [2]	1.2V POD	PIN_CG24
DDR4C_DQ3	Data [3]	1.2V POD	PIN_CH29
DDR4C_DQ4	Data [4]	1.2V POD	PIN_CE28
DDR4C_DQ5	Data [5]	1.2V POD	PIN_CF25

DDR4C_DQ6	Data [6]	1.2V POD	PIN_CE24
DDR4C_DQ7	Data [7]	1.2V POD	PIN_CG28
DDR4C_DQ8	Data [8]	1.2V POD	PIN_CN28
DDR4C_DQ9	Data [9]	1.2V POD	PIN_CK25
DDR4C_DQ10	Data [10]	1.2V POD	PIN_CK29
DDR4C_DQ11	Data [11]	1.2V POD	PIN_CM25
DDR4C_DQ12	Data [12]	1.2V POD	PIN_CM29
DDR4C_DQ13	Data [13]	1.2V POD	PIN_CL24
DDR4C_DQ14	Data [14]	1.2V POD	PIN_CN24
DDR4C_DQ15	Data [15]	1.2V POD	PIN_CL28
DDR4C_DQ16	Data [16]	1.2V POD	PIN_CF21
DDR4C_DQ17	Data [17]	1.2V POD	PIN_CE22
DDR4C_DQ18	Data [18]	1.2V POD	PIN_CG18
DDR4C_DQ19	Data [19]	1.2V POD	PIN_CH17
DDR4C_DQ20	Data [20]	1.2V POD	PIN_CF17
DDR4C_DQ21	Data [21]	1.2V POD	PIN_CE18
DDR4C_DQ22	Data [22]	1.2V POD	PIN_CG22
DDR4C_DQ23	Data [23]	1.2V POD	PIN_CH21
DDR4C_DQ24	Data [24]	1.2V POD	PIN_CN18
DDR4C_DQ25	Data [25]	1.2V POD	PIN_CL22
DDR4C_DQ26	Data [26]	1.2V POD	PIN_CM21
DDR4C_DQ27	Data [27]	1.2V POD	PIN_CK17
DDR4C_DQ28	Data [28]	1.2V POD	PIN_CL18
DDR4C_DQ29	Data [29]	1.2V POD	PIN_CK21
DDR4C_DQ30	Data [30]	1.2V POD	PIN_CN22
DDR4C_DQ31	Data [31]	1.2V POD	PIN_CM17
DDR4C_DQ32	Data [32]	1.2V POD	PIN_CF11
DDR4C_DQ33	Data [33]	1.2V POD	PIN_CE14
DDR4C_DQ34	Data [34]	1.2V POD	PIN_CG14
DDR4C_DQ35	Data [35]	1.2V POD	PIN_CF15
DDR4C_DQ36	Data [36]	1.2V POD	PIN_CE10
DDR4C_DQ37	Data [37]	1.2V POD	PIN_CG10
DDR4C_DQ38	Data [38]	1.2V POD	PIN_CH15
DDR4C_DQ39	Data [39]	1.2V POD	PIN_CH11

DDR4C_DQ40	Data [40]	1.2V POD	PIN_CN14
DDR4C_DQ41	Data [41]	1.2V POD	PIN_CM15
DDR4C_DQ42	Data [42]	1.2V POD	PIN_CM11
DDR4C_DQ43	Data [43]	1.2V POD	PIN_CL10
DDR4C_DQ44	Data [44]	1.2V POD	PIN_CL14
DDR4C_DQ45	Data [45]	1.2V POD	PIN_CK15
DDR4C_DQ46	Data [46]	1.2V POD	PIN_CK11
DDR4C_DQ47	Data [47]	1.2V POD	PIN_CN10
DDR4C_DQ48	Data [48]	1.2V POD	PIN_CG4
DDR4C_DQ49	Data [49]	1.2V POD	PIN_CG8
DDR4C_DQ50	Data [50]	1.2V POD	PIN_CF7
DDR4C_DQ51	Data [51]	1.2V POD	PIN_CF3
DDR4C_DQ52	Data [52]	1.2V POD	PIN_CE8
DDR4C_DQ53	Data [53]	1.2V POD	PIN_CH7
DDR4C_DQ54	Data [54]	1.2V POD	PIN_CE4
DDR4C_DQ55	Data [55]	1.2V POD	PIN_CH3
DDR4C_DQ56	Data [56]	1.2V POD	PIN_CM7
DDR4C_DQ57	Data [57]	1.2V POD	PIN_CN8
DDR4C_DQ58	Data [58]	1.2V POD	PIN_CM3
DDR4C_DQ59	Data [59]	1.2V POD	PIN_CL4
DDR4C_DQ60	Data [60]	1.2V POD	PIN_CL8
DDR4C_DQ61	Data [61]	1.2V POD	PIN_CK7
DDR4C_DQ62	Data [62]	1.2V POD	PIN_CN4
DDR4C_DQ63	Data [63]	1.2V POD	PIN_CK3
DDR4C_DQ64	Data [64]	1.2V POD	PIN_DA28
DDR4C_DQ65	Data [65]	1.2V POD	PIN_DB29
DDR4C_DQ66	Data [66]	1.2V POD	PIN_CY25
DDR4C_DQ67	Data [67]	1.2V POD	PIN_DC24
DDR4C_DQ68	Data [68]	1.2V POD	PIN_DA24
DDR4C_DQ69	Data [69]	1.2V POD	PIN_CY29
DDR4C_DQ70	Data [70]	1.2V POD	PIN_DC28
DDR4C_DQ71	Data [71]	1.2V POD	PIN_DB25
DDR4C_DQS0	Data Strobe p[0]	DIFFERENTIAL 1.2V POD	PIN_CH27

DDR4C_DQS_n0	Data Strobe n[0]	DIFFERENTIAL 1.2V POD	PIN_CF27
DDR4C_DQS1	Data Strobe p[1]	DIFFERENTIAL 1.2V POD	PIN_CM27
DDR4C_DQS_n1	Data Strobe n[1]	DIFFERENTIAL 1.2V POD	PIN_CK27
DDR4C_DQS2	Data Strobe p[2]	DIFFERENTIAL 1.2V POD	PIN_CG20
DDR4C_DQS_n2	Data Strobe n[2]	DIFFERENTIAL 1.2V POD	PIN_CE20
DDR4C_DQS3	Data Strobe p[3]	DIFFERENTIAL 1.2V POD	PIN_CN20
DDR4C_DQS_n3	Data Strobe n[3]	DIFFERENTIAL 1.2V POD	PIN_CL20
DDR4C_DQS4	Data Strobe p[4]	DIFFERENTIAL 1.2V POD	PIN_CH13
DDR4C_DQS_n4	Data Strobe n[4]	DIFFERENTIAL 1.2V POD	PIN_CF13
DDR4C_DQS5	Data Strobe p[5]	DIFFERENTIAL 1.2V POD	PIN_CM13
DDR4C_DQS_n5	Data Strobe n[5]	DIFFERENTIAL 1.2V POD	PIN_CK13
DDR4C_DQS6	Data Strobe p[6]	DIFFERENTIAL 1.2V POD	PIN_CG6
DDR4C_DQS_n6	Data Strobe n[6]	DIFFERENTIAL 1.2V POD	PIN_CE6
DDR4C_DQS7	Data Strobe p[7]	DIFFERENTIAL 1.2V POD	PIN_CN6
DDR4C_DQS_n7	Data Strobe n[7]	DIFFERENTIAL 1.2V POD	PIN_CL6
DDR4C_DQS8	Data Strobe p[8]	DIFFERENTIAL 1.2V POD	PIN_DB27
DDR4C_DQS_n8	Data Strobe n[8]	DIFFERENTIAL 1.2V POD	PIN_CY27
DDR4C_DBI_n0	Data Bus Inversion	1.2V POD	PIN_CG26

	[0]		
DDR4C_DBI_n1	Data Bus Inversion [1]	1.2V POD	PIN_CN26
DDR4C_DBI_n2	Data Bus Inversion [2]	1.2V POD	PIN_CH19
DDR4C_DBI_n3	Data Bus Inversion [3]	1.2V POD	PIN_CM19
DDR4C_DBI_n4	Data Bus Inversion [4]	1.2V POD	PIN_CG12
DDR4C_DBI_n5	Data Bus Inversion [5]	1.2V POD	PIN_CN12
DDR4C_DBI_n6	Data Bus Inversion [6]	1.2V POD	PIN_CH5
DDR4C_DBI_n7	Data Bus Inversion [7]	1.2V POD	PIN_CM5
DDR4C_DBI_n8	Data Bus Inversion [8]	1.2V POD	PIN_DC26
DDR4C_A0	Address [0]	SSTL-12	PIN_CV15
DDR4C_A1	Address [1]	SSTL-12	PIN_CT15
DDR4C_A2	Address [2]	SSTL-12	PIN_CU14
DDR4C_A3	Address [3]	SSTL-12	PIN_CR14
DDR4C_A4	Address [4]	SSTL-12	PIN_CV13
DDR4C_A5	Address [5]	SSTL-12	PIN_CT13
DDR4C_A6	Address [6]	SSTL-12	PIN_CU12
DDR4C_A7	Address [7]	SSTL-12	PIN_CR12
DDR4C_A8	Address [8]	SSTL-12	PIN_CV11
DDR4C_A9	Address [9]	SSTL-12	PIN_CT11
DDR4C_A10	Address [10]	SSTL-12	PIN_CU10
DDR4C_A11	Address [11]	SSTL-12	PIN_CR10
DDR4C_A12	Address [12]	SSTL-12	PIN_CY7
DDR4C_A13	Address [13]	SSTL-12	PIN_DC6
DDR4C_A14	Address [14]/ WE_n	SSTL-12	PIN_DA6
DDR4C_A15	Address [15]/	SSTL-12	PIN_DB5

	CAS_n		
DDR4C_A16	Address [16]/ RAS_n	SSTL-12	PIN_CY5
DDR4C_BA0	Bank Select [0]	SSTL-12	PIN_CY3
DDR4C_BA1	Bank Select [1]	SSTL-12	PIN_CV1
DDR4C_BG0	Bank Group Select [0]	SSTL-12	PIN_CT1
DDR4C_BG1	Bank Group Select [1]	SSTL-12	PIN_DB15
DDR4C_CK0	Clock p	DIFFERENTIAL 1.2V SSTL	PIN_DB11
DDR4C_CK_n0	Clock n	DIFFERENTIAL 1.2V SSTL	PIN_CY11
DDR4C_CK1	Clock p	SSTL-12	PIN_CV3
DDR4C_CK_n1	Clock n	SSTL-12	PIN_CT3
DDR4C_CKE0	Clock Enable pin	SSTL-12	PIN_DC12
DDR4C_CKE1	Clock Enable pin	SSTL-12	PIN_DA12
DDR4C_ODT0	On Die Termination	SSTL-12	PIN_DB13
DDR4C_ODT1	On Die Termination	SSTL-12	PIN_CY13
DDR4C_CS_n0	Chip Select	SSTL-12	PIN_DC14
DDR4C_CS_n1	Chip Select	SSTL-12	PIN_DC10
DDR4C_PAR	Command and Address Parity Input	SSTL-12	PIN_DA10
DDR4C_ALERT_n	Register ALERT_n output	1.2 V	PIN_DA4
DDR4C_ACT_n	Activation Command Input	SSTL-12	PIN_DA14
DDR4C_RESET_n	Chip Reset	1.2 V	PIN_CY15
DDR4C_EVENT_n	Chip Temperature Event	1.2 V	PIN_CR20
DDR4C_SDA	Chip I2C Serial Data Bus	1.2 V	PIN_CU20

DDR4C_SCL	Chip I2C Serial Clock	1.2 V	PIN_CT21
DDR4C_REFCLK_p	DDR4 A port Reference Clock p	True Differential Signaling	PIN_DC8
DDR4C_REFCLK_n	DDR4 A port Reference Clock n	True Differential Signaling	PIN_DA8
DDR4C_RZQ	Calibrated pins for OCT block	1.2 V	PIN_DB7

Table 2-15 DDR4-D Bank Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
DDR4D_DQ0	Data [0]	1.2V POD	PIN_CF57
DDR4D_DQ1	Data [1]	1.2V POD	PIN_CG56
DDR4D_DQ2	Data [2]	1.2V POD	PIN_CG52
DDR4D_DQ3	Data [3]	1.2V POD	PIN_CH53
DDR4D_DQ4	Data [4]	1.2V POD	PIN_CF53
DDR4D_DQ5	Data [5]	1.2V POD	PIN_CE56
DDR4D_DQ6	Data [6]	1.2V POD	PIN_CE52
DDR4D_DQ7	Data [7]	1.2V POD	PIN_CH57
DDR4D_DQ8	Data [8]	1.2V POD	PIN_CM57
DDR4D_DQ9	Data [9]	1.2V POD	PIN_CN56
DDR4D_DQ10	Data [10]	1.2V POD	PIN_CM53
DDR4D_DQ11	Data [11]	1.2V POD	PIN_CL52
DDR4D_DQ12	Data [12]	1.2V POD	PIN_CK57
DDR4D_DQ13	Data [13]	1.2V POD	PIN_CL56
DDR4D_DQ14	Data [14]	1.2V POD	PIN_CN52
DDR4D_DQ15	Data [15]	1.2V POD	PIN_CK53
DDR4D_DQ16	Data [16]	1.2V POD	PIN_CH45
DDR4D_DQ17	Data [17]	1.2V POD	PIN_CE50
DDR4D_DQ18	Data [18]	1.2V POD	PIN_CF49
DDR4D_DQ19	Data [19]	1.2V POD	PIN_CH49
DDR4D_DQ20	Data [20]	1.2V POD	PIN_CF45

DDR4D_DQ21	Data [21]	1.2V POD	PIN_CG46
DDR4D_DQ22	Data [22]	1.2V POD	PIN_CE46
DDR4D_DQ23	Data [23]	1.2V POD	PIN_CG50
DDR4D_DQ24	Data [24]	1.2V POD	PIN_DA50
DDR4D_DQ25	Data [25]	1.2V POD	PIN_CY49
DDR4D_DQ26	Data [26]	1.2V POD	PIN_DC46
DDR4D_DQ27	Data [27]	1.2V POD	PIN_CY45
DDR4D_DQ28	Data [28]	1.2V POD	PIN_DC50
DDR4D_DQ29	Data [29]	1.2V POD	PIN_DB49
DDR4D_DQ30	Data [30]	1.2V POD	PIN_DA46
DDR4D_DQ31	Data [31]	1.2V POD	PIN_DB45
DDR4D_DQ32	Data [32]	1.2V POD	PIN_CY43
DDR4D_DQ33	Data [33]	1.2V POD	PIN_DA42
DDR4D_DQ34	Data [34]	1.2V POD	PIN_DC38
DDR4D_DQ35	Data [35]	1.2V POD	PIN_DA38
DDR4D_DQ36	Data [36]	1.2V POD	PIN_DB43
DDR4D_DQ37	Data [37]	1.2V POD	PIN_DC42
DDR4D_DQ38	Data [38]	1.2V POD	PIN_CY39
DDR4D_DQ39	Data [39]	1.2V POD	PIN_DB39
DDR4D_DQ40	Data [40]	1.2V POD	PIN_CM45
DDR4D_DQ41	Data [41]	1.2V POD	PIN_CN50
DDR4D_DQ42	Data [42]	1.2V POD	PIN_CL46
DDR4D_DQ43	Data [43]	1.2V POD	PIN_CK49
DDR4D_DQ44	Data [44]	1.2V POD	PIN_CK45
DDR4D_DQ45	Data [45]	1.2V POD	PIN_CM49
DDR4D_DQ46	Data [46]	1.2V POD	PIN_CN46
DDR4D_DQ47	Data [47]	1.2V POD	PIN_CL50
DDR4D_DQ48	Data [48]	1.2V POD	PIN_DA36
DDR4D_DQ49	Data [49]	1.2V POD	PIN_DC32
DDR4D_DQ50	Data [50]	1.2V POD	PIN_CY31
DDR4D_DQ51	Data [51]	1.2V POD	PIN_DB31
DDR4D_DQ52	Data [52]	1.2V POD	PIN_CY35
DDR4D_DQ53	Data [53]	1.2V POD	PIN_DC36
DDR4D_DQ54	Data [54]	1.2V POD	PIN_DB35

DDR4D_DQ55	Data [55]	1.2V POD	PIN_DA32
DDR4D_DQ56	Data [56]	1.2V POD	PIN_CT31
DDR4D_DQ57	Data [57]	1.2V POD	PIN_CR36
DDR4D_DQ58	Data [58]	1.2V POD	PIN_CV35
DDR4D_DQ59	Data [59]	1.2V POD	PIN_CT35
DDR4D_DQ60	Data [60]	1.2V POD	PIN_CU32
DDR4D_DQ61	Data [61]	1.2V POD	PIN_CV31
DDR4D_DQ62	Data [62]	1.2V POD	PIN_CR32
DDR4D_DQ63	Data [63]	1.2V POD	PIN_CU36
DDR4D_DQ64	Data [64]	1.2V POD	PIN_CV39
DDR4D_DQ65	Data [65]	1.2V POD	PIN_CR42
DDR4D_DQ66	Data [66]	1.2V POD	PIN_CT43
DDR4D_DQ67	Data [67]	1.2V POD	PIN_CV43
DDR4D_DQ68	Data [68]	1.2V POD	PIN_CR38
DDR4D_DQ69	Data [69]	1.2V POD	PIN_CT39
DDR4D_DQ70	Data [70]	1.2V POD	PIN_CU42
DDR4D_DQ71	Data [71]	1.2V POD	PIN_CU38
DDR4D_DQS0	Data Strobe p[0]	DIFFERENTIAL 1.2V POD	PIN_CG54
DDR4D_DQS_n0	Data Strobe n[0]	DIFFERENTIAL 1.2V POD	PIN_CE54
DDR4D_DQS1	Data Strobe p[1]	DIFFERENTIAL 1.2V POD	PIN_CN54
DDR4D_DQS_n1	Data Strobe n[1]	DIFFERENTIAL 1.2V POD	PIN_CL54
DDR4D_DQS2	Data Strobe p[2]	DIFFERENTIAL 1.2V POD	PIN_CH47
DDR4D_DQS_n2	Data Strobe n[2]	DIFFERENTIAL 1.2V POD	PIN_CF47
DDR4D_DQS3	Data Strobe p[3]	DIFFERENTIAL 1.2V POD	PIN_DB47
DDR4D_DQS_n3	Data Strobe n[3]	DIFFERENTIAL 1.2V POD	PIN_CY47
DDR4D_DQS4	Data Strobe p[4]	DIFFERENTIAL 1.2V	PIN_DC40

		POD	
DDR4D_DQS_n4	Data Strobe n[4]	DIFFERENTIAL 1.2V POD	PIN_DA40
DDR4D_DQS5	Data Strobe p[5]	DIFFERENTIAL 1.2V POD	PIN_CM47
DDR4D_DQS_n5	Data Strobe n[5]	DIFFERENTIAL 1.2V POD	PIN_CK47
DDR4D_DQS6	Data Strobe p[6]	DIFFERENTIAL 1.2V POD	PIN_DB33
DDR4D_DQS_n6	Data Strobe n[6]	DIFFERENTIAL 1.2V POD	PIN_CY33
DDR4D_DQS7	Data Strobe p[7]	DIFFERENTIAL 1.2V POD	PIN_CV33
DDR4D_DQS_n7	Data Strobe n[7]	DIFFERENTIAL 1.2V POD	PIN_CT33
DDR4D_DQS8	Data Strobe p[8]	DIFFERENTIAL 1.2V POD	PIN_CU40
DDR4D_DQS_n8	Data Strobe n[8]	DIFFERENTIAL 1.2V POD	PIN_CR40
DDR4D_DBI_n0	Data Bus Inversion [0]	1.2V POD	PIN_CH55
DDR4D_DBI_n1	Data Bus Inversion [1]	1.2V POD	PIN_CM55
DDR4D_DBI_n2	Data Bus Inversion [2]	1.2V POD	PIN_CG48
DDR4D_DBI_n3	Data Bus Inversion [3]	1.2V POD	PIN_DC48
DDR4D_DBI_n4	Data Bus Inversion [4]	1.2V POD	PIN_DB41
DDR4D_DBI_n5	Data Bus Inversion [5]	1.2V POD	PIN_CN48
DDR4D_DBI_n6	Data Bus Inversion [6]	1.2V POD	PIN_DC34
DDR4D_DBI_n7	Data Bus Inversion [7]	1.2V POD	PIN_CU34

DDR4D_DBI_n8	Data Bus Inversion [8]	1.2V POD	PIN_CV41
DDR4D_A0	Address [0]	SSTL-12	PIN_CH31
DDR4D_A1	Address [1]	SSTL-12	PIN_CF31
DDR4D_A2	Address [2]	SSTL-12	PIN_CG32
DDR4D_A3	Address [3]	SSTL-12	PIN_CE32
DDR4D_A4	Address [4]	SSTL-12	PIN_CH33
DDR4D_A5	Address [5]	SSTL-12	PIN_CF33
DDR4D_A6	Address [6]	SSTL-12	PIN_CG34
DDR4D_A7	Address [7]	SSTL-12	PIN_CE34
DDR4D_A8	Address [8]	SSTL-12	PIN_CH35
DDR4D_A9	Address [9]	SSTL-12	PIN_CF35
DDR4D_A10	Address [10]	SSTL-12	PIN_CG36
DDR4D_A11	Address [11]	SSTL-12	PIN_CE36
DDR4D_A12	Address [12]	SSTL-12	PIN_CK39
DDR4D_A13	Address [13]	SSTL-12	PIN_CN40
DDR4D_A14	Address [14]/ WE_n	SSTL-12	PIN_CL40
DDR4D_A15	Address [15]/ CAS_n	SSTL-12	PIN_CM41
DDR4D_A16	Address [16]/ RAS_n	SSTL-12	PIN_CK41
DDR4D_BA0	Bank Select [0]	SSTL-12	PIN_CL42
DDR4D_BA1	Bank Select [1]	SSTL-12	PIN_CM43
DDR4D_BG0	Bank Group Select [0]	SSTL-12	PIN_CK43
DDR4D_BG1	Bank Group Select [1]	SSTL-12	PIN_CM31
DDR4D_CK0	Clock p	DIFFERENTIAL 1.2V SSTL	PIN_CM35
DDR4D_CK_n0	Clock n	DIFFERENTIAL 1.2V SSTL	PIN_CK35
DDR4D_CK1	Clock p	SSTL-12	PIN_CH43
DDR4D_CK_n1	Clock n	SSTL-12	PIN_CF43

DDR4D_CKE0	Clock Enable pin	SSTL-12	PIN_CN34
DDR4D_CKE1	Clock Enable pin	SSTL-12	PIN_CL34
DDR4D_ODT0	On Die Termination	SSTL-12	PIN_CM33
DDR4D_ODT1	On Die Termination	SSTL-12	PIN_CK33
DDR4D_CS_n0	Chip Select	SSTL-12	PIN_CN32
DDR4D_CS_n1	Chip Select	SSTL-12	PIN_CN36
DDR4D_PAR	Command and Address Parity Input	SSTL-12	PIN_CL36
DDR4D_ALERT_n	Register ALERT_n output	1.2 V	PIN_CN42
DDR4D_ACT_n	Activation Command Input	SSTL-12	PIN_CL32
DDR4D_RESET_n	Chip Reset	1.2 V	PIN_CK31
DDR4D_EVENT_n	Chip Temperature Event	1.2 V	PIN_CY53
DDR4D_SDA	Chip I2C Serial Data Bus	1.2 V	PIN_DB53
DDR4D_SCL	Chip I2C Serial Clock	1.2 V	PIN_DC54
DDR4D_REFCLK_p	DDR4 A port Reference Clock p	True Differential Signaling	PIN_CN38
DDR4D_REFCLK_n	DDR4 A port Reference Clock n	True Differential Signaling	PIN_CL38
DDR4D_RZQ	Calibrated pins for OCT block	1.2 V	PIN_CM39

2.7 QSPF-DD Ports

The DE10-Agilex board can support two standard QSFP-DD (Quad Small Form Factor Pluggable Double Density) optical modules. The QSFP-DD ports on the DE10-Agilex board will be able to support **8** pairs of **25 Gb/s NRZ** modulation or **4** pairs of **50 Gb/s PAM4** modulation with FPGA's high speed transceivers to achieve 200 Gbps bandwidth. Furthermore, the QSFP-DD modules also can backward compatible with

QSFP28 and QSFP+ optical transceivers. **Figure 2-17** shows the connections between the QSFP-DD and Agilex FPGA.

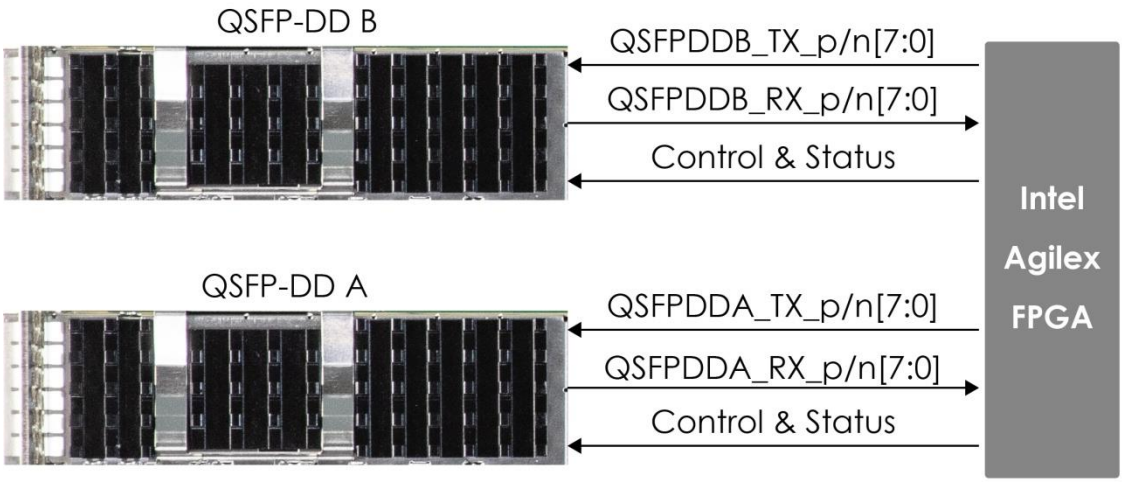


Figure 2-17 Connection between the QSFP28 and Agilex FPGA

Table 2-16, and **Table 2-17** list the QSFP-DD port A and B pin assignments and signal names relative to the Agilex device.

Table 2-16 QSFP-DD Port A Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
QSFPDDA_TX_p0	Transmitter non-inverted data of channel 0	High Speed Differential I/O	PIN_BF1
QSFPDDA_TX_n0	Transmitter inverted data of channel 0	High Speed Differential I/O	PIN_BE2
QSFPDDA_TX_p1	Transmitter non-inverted data of channel 1	High Speed Differential I/O	PIN_BK1
QSFPDDA_TX_n1	Transmitter inverted data of channel 1	High Speed Differential I/O	PIN_BJ2
QSFPDDA_TX_p2	Transmitter non-inverted data of channel 2	High Speed Differential I/O	PIN_BG4
QSFPDDA_TX_n2	Transmitter inverted data of channel 2	High Speed Differential I/O	PIN_BH5
QSFPDDA_TX_p3	Transmitter non-inverted	High Speed	PIN_BL4

	data of channel 3	Differential I/O	
QSFPDDA_TX_n3	Transmitter inverted data of channel 3	High Speed Differential I/O	PIN_BM5
QSFPDDA_TX_p4	Transmitter non-inverted data of channel 4	High Speed Differential I/O	PIN_BP1
QSFPDDA_TX_n4	Transmitter inverted data of channel 4	High Speed Differential I/O	PIN_BN2
QSFPDDA_TX_p5	Transmitter non-inverted data of channel 5	High Speed Differential I/O	PIN_BV1
QSFPDDA_TX_n5	Transmitter inverted data of channel 5	High Speed Differential I/O	PIN_BU2
QSFPDDA_TX_p6	Transmitter non-inverted data of channel 6	High Speed Differential I/O	PIN_BR4
QSFPDDA_TX_n6	Transmitter inverted data of channel 6	High Speed Differential I/O	PIN_BT5
QSFPDDA_TX_p7	Transmitter non-inverted data of channel 7	High Speed Differential I/O	PIN_BW4
QSFPDDA_TX_n7	Transmitter inverted data of channel 7	High Speed Differential I/O	PIN_BY5
QSFPDDA_RX_p0	Receiver non-inverted data of channel 0	High Speed Differential I/O	PIN_BF7
QSFPDDA_RX_n0	Receiver inverted data of channel 0	High Speed Differential I/O	PIN_BE8
QSFPDDA_RX_p1	Receiver non-inverted data of channel 1	High Speed Differential I/O	PIN_BK7
QSFPDDA_RX_n1	Receiver inverted data of channel 1	High Speed Differential I/O	PIN_BJ8
QSFPDDA_RX_p2	Receiver non-inverted data of channel 2	High Speed Differential I/O	PIN_BG10
QSFPDDA_RX_n2	Receiver inverted data of channel 2	High Speed Differential I/O	PIN_BH11
QSFPDDA_RX_p3	Receiver non-inverted data of channel 3	High Speed Differential I/O	PIN_BL10
QSFPDDA_RX_n3	Receiver inverted data of channel 3	High Speed Differential I/O	PIN_BM11

QSFPDDA_RX_p4	Receiver non-inverted data of channel 4	High Speed Differential I/O	PIN_BP7
QSFPDDA_RX_n4	Receiver inverted data of channel 4	High Speed Differential I/O	PIN_BN8
QSFPDDA_RX_p5	Receiver non-inverted data of channel 5	High Speed Differential I/O	PIN_BV7
QSFPDDA_RX_n5	Receiver inverted data of channel 5	High Speed Differential I/O	PIN_BU8
QSFPDDA_RX_p6	Receiver non-inverted data of channel 6	High Speed Differential I/O	PIN_BR10
QSFPDDA_RX_n6	Receiver inverted data of channel 6	High Speed Differential I/O	PIN_BT11
QSFPDDA_RX_p7	Receiver non-inverted data of channel 7	High Speed Differential I/O	PIN_BW10
QSFPDDA_RX_n7	Receiver inverted data of channel 7	High Speed Differential I/O	PIN_BY11
QSFPDDA_REFCLK_p	QSFP-DD port A transceiver reference clock p	LVDS	PIN_AJ12
QSFPDDA_REFCLK_n	QSFP-DD port transceiver reference clock n	LVDS	PIN_AH11
QSFPDDA_INITMODE	Initialization mode	1.2V	PIN_DB21
QSFPDDA_INTERRUPT_n	Interrupt	1.2V	PIN_CV25
QSFPDDA_MOD_PRS_n	Module Present	1.2V	PIN_CR26
QSFPDDA_MOD_SEL_n	Module Select	1.2V	PIN_DB19
QSFPDDA_RST_n	Module Reset	1.2V	PIN_DA20
QSFPDDA_SCL	2-wire serial interface clock	1.2V	PIN_H21
QSFPDDA_SDA	2-wire serial interface data	1.2V	PIN_H23

Table 2-17 QSFP-DD Port B Pin Assignments, Schematic Signal Names, and Functions

Schematic	Description	I/O Standard	Agilex Pin

Signal Name			Number
QSFPDDB_TX_p0	Transmitter non-inverted data of channel 0	High Speed Differential I/O	PIN_AK1
QSFPDDB_TX_n0	Transmitter inverted data of channel 0	High Speed Differential I/O	PIN_AJ2
QSFPDDB_TX_p1	Transmitter non-inverted data of channel 1	High Speed Differential I/O	PIN_AP1
QSFPDDB_TX_n1	Transmitter inverted data of channel 1	High Speed Differential I/O	PIN_AN2
QSFPDDB_TX_p2	Transmitter non-inverted data of channel 2	High Speed Differential I/O	PIN_AL4
QSFPDDB_TX_n2	Transmitter inverted data of channel 2	High Speed Differential I/O	PIN_AM5
QSFPDDB_TX_p3	Transmitter non-inverted data of channel 3	High Speed Differential I/O	PIN_AR4
QSFPDDB_TX_n3	Transmitter inverted data of channel 3	High Speed Differential I/O	PIN_AT5
QSFPDDB_TX_p4	Transmitter non-inverted data of channel 4	High Speed Differential I/O	PIN_AV1
QSFPDDB_TX_n4	Transmitter inverted data of channel 4	High Speed Differential I/O	PIN_AU2
QSFPDDB_TX_p5	Transmitter non-inverted data of channel 5	High Speed Differential I/O	PIN_BB1
QSFPDDB_TX_n5	Transmitter inverted data of channel 5	High Speed Differential I/O	PIN_BA2
QSFPDDB_TX_p6	Transmitter non-inverted data of channel 6	High Speed Differential I/O	PIN_AW4
QSFPDDB_TX_n6	Transmitter inverted data of channel 6	High Speed Differential I/O	PIN_AY5
QSFPDDB_TX_p7	Transmitter non-inverted data of channel 7	High Speed Differential I/O	PIN_BC4
QSFPDDB_TX_n7	Transmitter inverted data of channel 7	High Speed Differential I/O	PIN_BD5
QSFPDDB_RX_p0	Receiver non-inverted data of channel 0	High Speed Differential I/O	PIN_AK7

QSFPDDB_RX_n0	Receiver inverted data of channel 0	High Speed Differential I/O	PIN_AJ8
QSFPDDB_RX_p1	Receiver non-inverted data of channel 1	High Speed Differential I/O	PIN_AP7
QSFPDDB_RX_n1	Receiver inverted data of channel 1	High Speed Differential I/O	PIN_AN8
QSFPDDB_RX_p2	Receiver non-inverted data of channel 2	High Speed Differential I/O	PIN_AL10
QSFPDDB_RX_n2	Receiver inverted data of channel 2	High Speed Differential I/O	PIN_AM11
QSFPDDB_RX_p3	Receiver non-inverted data of channel 3	High Speed Differential I/O	PIN_AR10
QSFPDDB_RX_n3	Receiver inverted data of channel 3	High Speed Differential I/O	PIN_AT11
QSFPDDB_RX_p4	Receiver non-inverted data of channel 4	High Speed Differential I/O	PIN_AV7
QSFPDDB_RX_n4	Receiver inverted data of channel 4	High Speed Differential I/O	PIN_AU8
QSFPDDB_RX_p5	Receiver non-inverted data of channel 5	High Speed Differential I/O	PIN_BB7
QSFPDDB_RX_n5	Receiver inverted data of channel 5	High Speed Differential I/O	PIN_BA8
QSFPDDB_RX_p6	Receiver non-inverted data of channel 6	High Speed Differential I/O	PIN_AW10
QSFPDDB_RX_n6	Receiver inverted data of channel 6	High Speed Differential I/O	PIN_AY11
QSFPDDB_RX_p7	Receiver non-inverted data of channel 7	High Speed Differential I/O	PIN_BC10
QSFPDDB_RX_n7	Receiver inverted data of channel 7	High Speed Differential I/O	PIN_BD11
QSFPDDB_REFCLK_p	QSFP-DD port A transceiver reference clock p	LVDS	PIN_AT13
QSFPDDB_REFCLK_n	QSFP-DD port transceiver reference clock n	LVDS	PIN_AP13

QSFPDDB_INITMODE	Initialization mode	1.2V	PIN_DA18
QSFPDDB_INTERRUPT_n	Interrupt	1.2V	PIN_DC18
QSFPDDB_MOD_PRS_n	Module Present	1.2V	PIN_CY19
QSFPDDB_MOD_SEL_n	Module Select	1.2V	PIN_CR22
QSFPDDB_RST_n	Module Reset	1.2V	PIN_CU22
QSFPDDB_SCL	2-wire serial interface clock	1.2V	PIN_CY17
QSFPDDB_SDA	2-wire serial interface data	1.2V	PIN_H19

2.8 PCI Express

The FPGA development board is designed to fit entirely into a PC motherboard with x16 PCI Express slot. Utilizing built-in transceivers on an Agilex device, it is able to provide a fully integrated PCI Express-compliant solution for multi-lane (x1, x4, x8 and x16) applications. With the PCI Express hard IP block incorporated in the Agilex device, it will allow users to implement simple and fast protocol, as well as saving logic resources for logic application. **Figure 2-18** presents the pin connection established between the Agilex FPGA and PCI Express.

The PCI Express interface supports complete PCI Express Gen1 at 2.5Gbps/lane, Gen2 at 5.0Gbps/lane, Gen3 at 8.0Gbps/lane and Gen4 at 16.0Gbps/lane protocol stack solution compliant to PCI Express base specification 4.0 that includes PHY-MAC, Data Link, and transaction layer circuitry embedded in PCI Express hard IP blocks.

Please note that it is a requirement that you connect the PCIe external power connector 8-pin 12V DC power connector in the FPGA to avoid FPGA damage due to insufficient power. The PCIE_REFCLK_p signal is a differential input that is driven from the PC motherboard on this board through the PCIe edge connector. A DIP switch (SW6) is connected to the PCI Express to allow different configurations to enable x1, x4, x8 or x16 PCIe lane.

Table 2-18 summarizes the PCI Express pin assignments of the signal names relative to the Agilex FPGA.

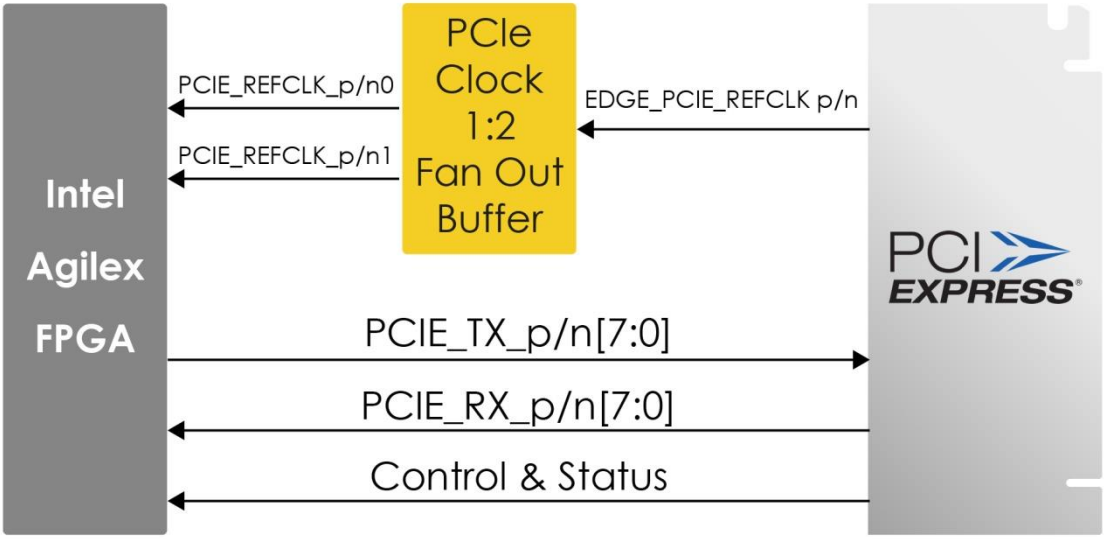


Figure 2-18 PCI Express pin connection

Table 2-18 PCI Express Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
PCIE_TX_p0	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BP55
PCIE_TX_n0	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BR56
PCIE_TX_p1	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BN52
PCIE_TX_n1	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BM53
PCIE_TX_p2	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BK55
PCIE_TX_n2	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BL56
PCIE_TX_p3	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BJ52
PCIE_TX_n3	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BH53
PCIE_TX_p4	Add-in card transmit bus	HIGH Speed	PIN_BF55

		Differential I/O	
PCIE_TX_n4	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BG56
PCIE_TX_p5	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BE52
PCIE_TX_n5	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BD53
PCIE_TX_p6	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BB55
PCIE_TX_n6	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BC56
PCIE_TX_p7	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_BA52
PCIE_TX_n7	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AY53
PCIE_TX_p8	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AV55
PCIE_TX_n8	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AW56
PCIE_TX_p9	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AU52
PCIE_TX_n9	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AT53
PCIE_TX_p10	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AP55
PCIE_TX_n10	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AR56
PCIE_TX_p11	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AN52
PCIE_TX_n11	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AM53
PCIE_TX_p12	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AK55
PCIE_TX_n12	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AL56

PCIE_TX_p13	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AJ52
PCIE_TX_n13	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AH53
PCIE_TX_p14	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AF55
PCIE_TX_n14	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AG56
PCIE_TX_p15	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AE52
PCIE_TX_n15	Add-in card transmit bus	HIGH Speed Differential I/O	PIN_AD53
PCIE_RX_p0	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BP61
PCIE_RX_n0	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BR62
PCIE_RX_p1	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BN58
PCIE_RX_n1	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BM59
PCIE_RX_p2	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BK61
PCIE_RX_n2	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BL62
PCIE_RX_p3	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BJ58
PCIE_RX_n3	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BH59
PCIE_RX_p4	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BF61
PCIE_RX_n4	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BG62
PCIE_RX_p5	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BE58
PCIE_RX_n5	Add-in card receive bus	HIGH Speed	PIN_BD59

		Differential I/O	
PCIE_RX_p6	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BB61
PCIE_RX_n6	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BC62
PCIE_RX_p7	Add-in card receive bus	HIGH Speed Differential I/O	PIN_BA58
PCIE_RX_n7	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AY59
PCIE_RX_p8	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AV61
PCIE_RX_n8	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AW62
PCIE_RX_p9	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AU58
PCIE_RX_n9	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AT59
PCIE_RX_p10	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AP61
PCIE_RX_n10	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AR62
PCIE_RX_p11	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AN58
PCIE_RX_n11	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AM59
PCIE_RX_p12	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AK61
PCIE_RX_n12	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AL62
PCIE_RX_p13	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AJ58
PCIE_RX_n13	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AH59
PCIE_RX_p14	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AF61

PCIE_RX_n14	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AG62
PCIE_RX_p15	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AE58
PCIE_RX_n15	Add-in card receive bus	HIGH Speed Differential I/O	PIN_AD59
PCIE_CLKREQ_n	Clock request signal	1.2V	PIN_F55
PCIE_REFCLK_p0	Motherboard reference clock	HCSL	PIN_AJ48
PCIE_REFCLK_n0	Motherboard reference clock	HCSL	PIN_AH49
PCIE_REFCLK_p1	Motherboard reference clock	HCSL	PIN_AE48
PCIE_REFCLK_n1	Motherboard reference clock	HCSL	PIN_AD49
PCIE_PERST_n	Reset	1.8V	PIN_BU58
PCIE_SMBCLK	SMB clock	1.2V	PIN_G50
PCIE_SMBDAT	SMB data	1.2V	PIN_J52
PCIE_WAKE_n	Wake signal	1.2V	PIN_J50
PCIE_PRSNT1n	Hot plug detect	-	-
PCIE_PRSNT2n_x1	Hot plug detect x1 PCIe slot enabled using SW6 dip switch	-	-
PCIE_PRSNT2n_x4	Hot plug detect x4 PCIe slot enabled using SW6 dip switch	-	-
PCIE_PRSNT2n_x8	Hot plug detect x8 PCIe slot enabled using SW6 dip switch	-	-
PCIE_PRSNT2n_x16	Hot plug detect x16 PCIe slot enabled using SW6 dip switch	-	-

2.9 System Status Interface

As shown in **Figure 2-19**, the DE10-Agilex board provides several sensors to monitor the status of the board, such as FPGA temperature, board power monitor, and fan speed status. These interfaces are connected to the System MAX10 FPGA on the board. The board management logic (Dashboard) in the System MAX10 FPGA will monitor those status and perform corresponding control according to the status. For example, when the temperature of the FPGA increases, the system will automatically increase the fan speed to reduce the temperature. When the temperature of the FPGA continues to exceed the working range (such as a fan failure condition), the FPGA

power will be cut to protect the board.

The board also provides USB to UART interface to connect with the System MAX10 FPGA, so that users can monitor the status of the board from the host through the UART interface. See chapter 8 for details.

Finally, the board status also can be read on the Agilex FPGA side via the SPI interface connected to the System MAX10 FPGA. Terasic had provided a “board information IP” that allow user can place it in the FPGA to read these board status. Please refer to the **section 5.4** for detailed. **Table 2-19** shows the pin assignments of the SPI interface on the Agilex FPGA.

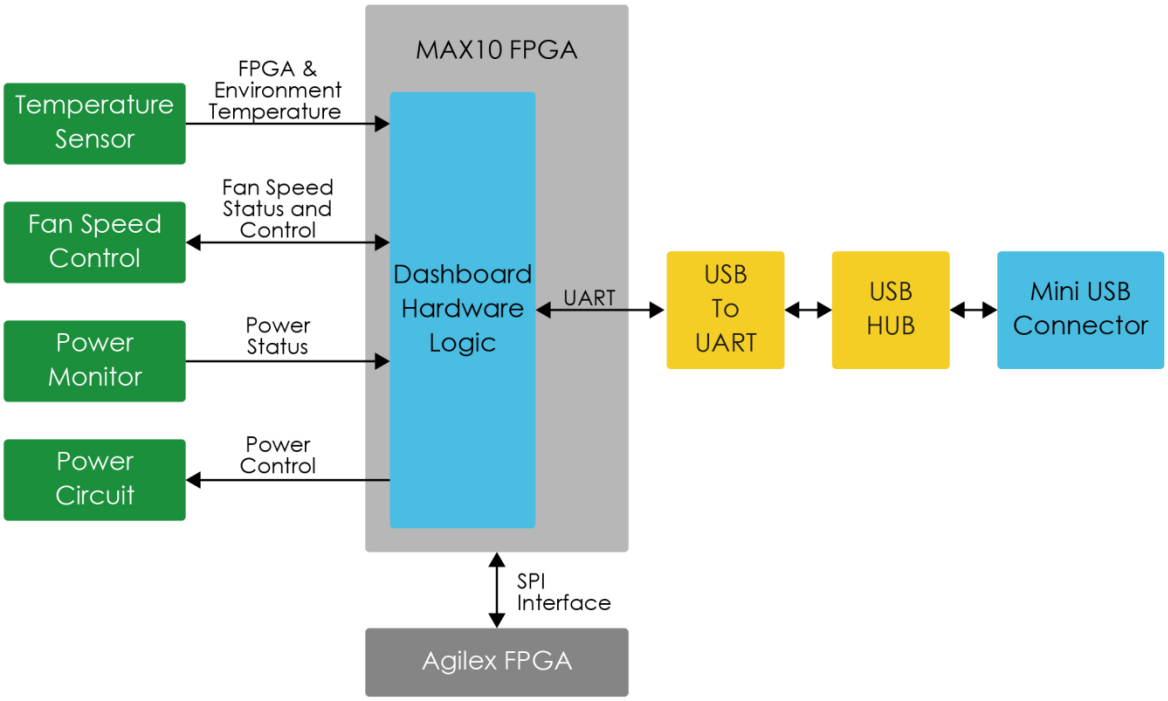


Figure 2-19 Block diagram of the system status interface

Table 2-19 Pin Assignments, Schematic Signal Names, and Functions for SPI interface of the board status

Schematic Signal Name	Description	I/O Standard	Agilex Pin Number
INFO_SPI_SCLK	Serial Clock, SPI master output to salve.	1.2V	PIN_CU18
INFO_SPI_MISO	Master input.	1.2V	PIN_CU26

INFO_SPI_MOSI	Master output.	1.2V	PIN_CV17
INFO_SPI_CS_n	Slave Select, Master output.	1.2V	PIN_CR18

Chapter 3

System Builder

This chapter describes how users can create a custom design project for the FPGA board from a software tool named System Builder.

3.1 Introduction

The System Builder is a Windows based software utility. It is designed to help users create a Quartus Prime project for the FPGA board within minutes. The Quartus Prime project files generated include:

- Quartus Prime Project File (.qpf)
- Quartus Prime Setting File (.qsf)
- Top-Level Design File (.v)
- External PLL Controller (.v)
- Synopsis Design Constraints file (.sdc)
- Pin Assignment Document (.htm)

The System Builder not only can generate the files above, but can also provide error-checking rules to handle situation that are prone to errors. The common mistakes that users encounter are the following:

- Board damaged for wrong pin/bank voltage assignment.
- Board malfunction caused by wrong device connections or missing pin counts for connected ends.
- Performance dropped because of improper pin assignments

3.2 General Design Flow

This section will introduce the general design flow to build a project for the FPGA board via the System Builder. The general design flow is illustrated in **Figure 3-1**.

Users should launch System Builder and create a new project according to their design requirements. When users complete the settings, the System Builder will generate two major files which include top-level design file (.v) and the Quartus Prime setting file (.qsf).

The top-level design file contains top-level Verilog wrapper for users to add their own design/logic. The Quartus Prime setting file contains information such as FPGA device type, top-level pin assignment, and I/O standard for each user-defined I/O pin.

Finally, the Quartus Prime programmer must be used to download SOF file to the FPGA board using JTAG interface.

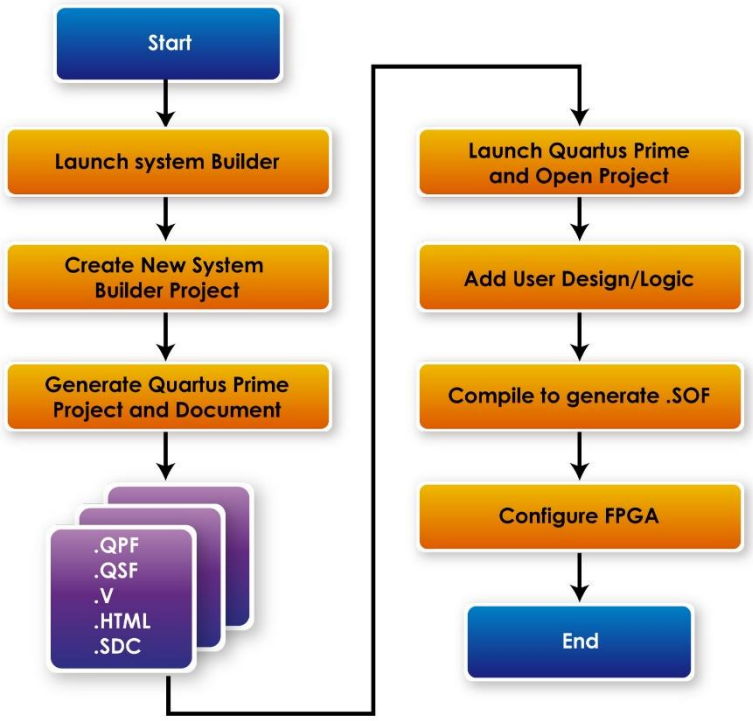


Figure 3-1 the general design flow of building a project

3.3 Using System Builder

This section provides detailed procedures on how the System Builder is used.

■ Install and Launch the System Builder

The System Builder is located under the directory: **"Tools\SystemBuilder"** in the System CD. Users can copy the entire folder to the host computer without installing the utility. Please execute the ***SystemBuilder.exe*** on the host computer, as shown in **Figure 3-2**.

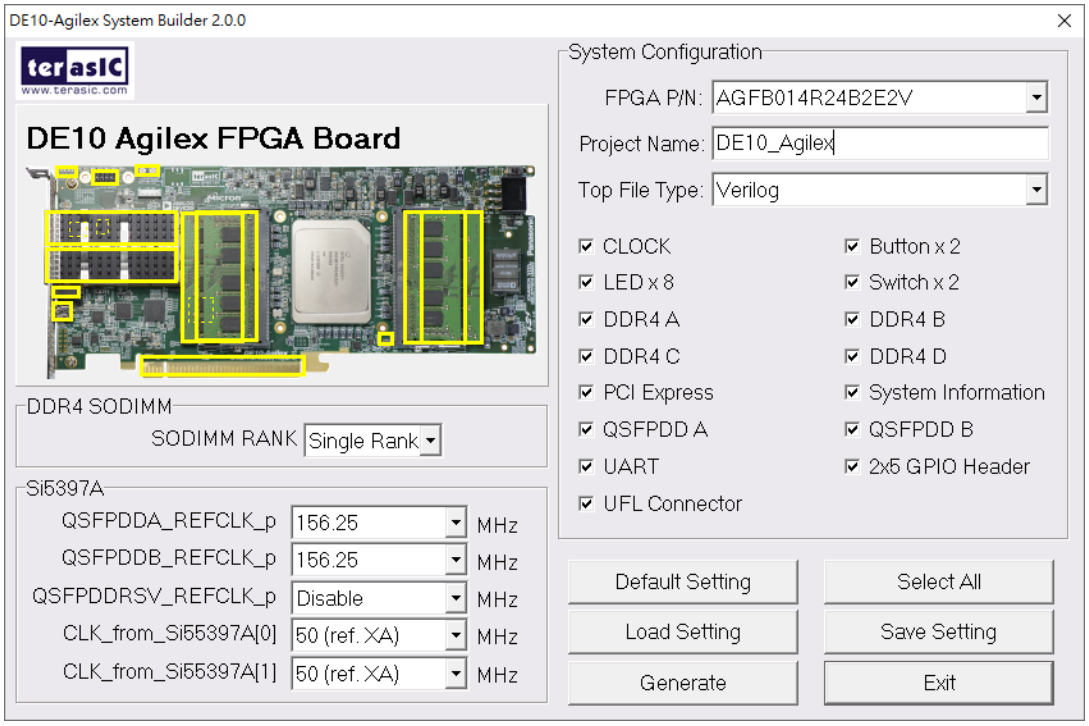


Figure 3-2 The System Builder window

■ Select FPGA

This function is used for if there are more FPGA device supported on the DE10-Agilex board in the feature, users can select FPGA P/N in System Builder as shown in **Figure 3-3**

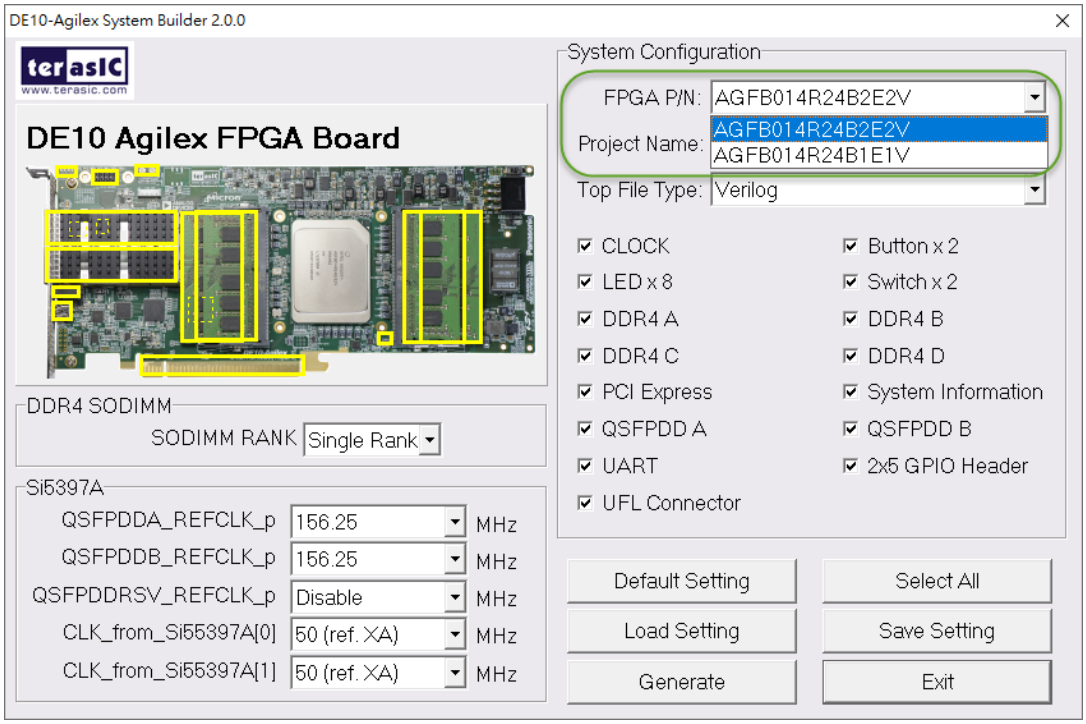


Figure 3-3 Select FPGA

■ Enter Project Name

The project name entered in the circled area as shown in **Figure 3-4**, will be assigned automatically as the name of the top-level design entry.

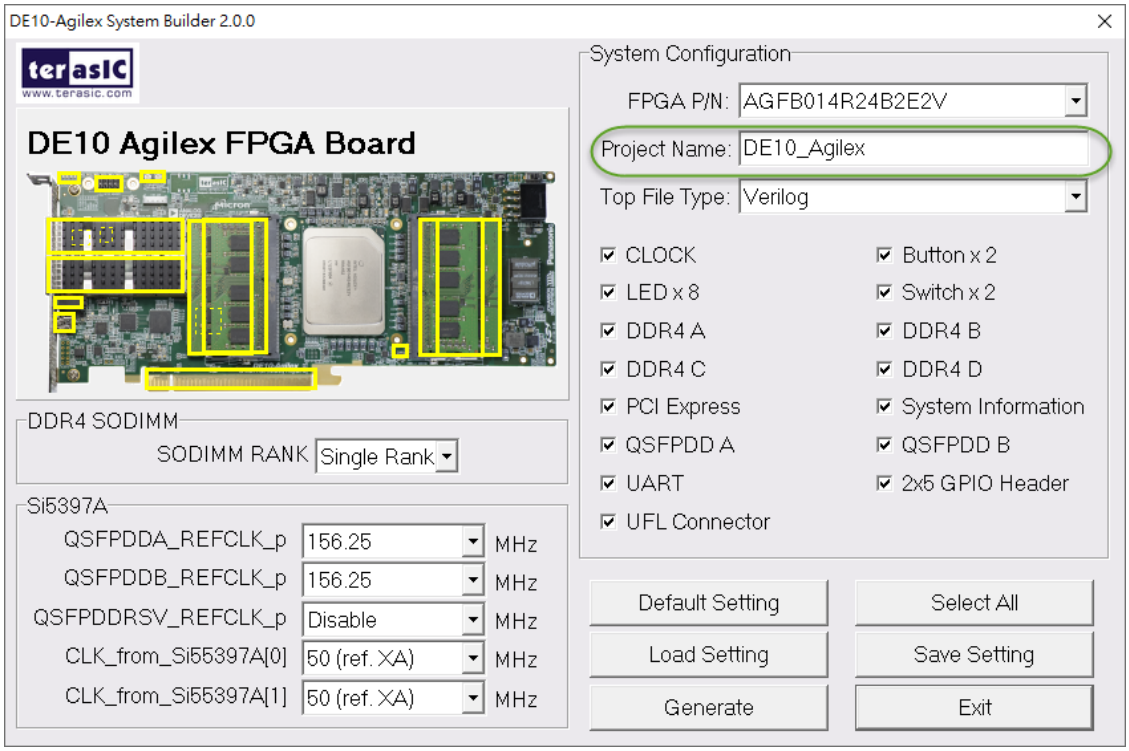


Figure 3-4 Project Name in the System Builder window

■ Select Top File Type

The system builder can generate Verilog or VHDL Quartus top file according to the users' requirements. Users can select their desired file type in the Top File Type list-box shown in **Figure 3-5**.

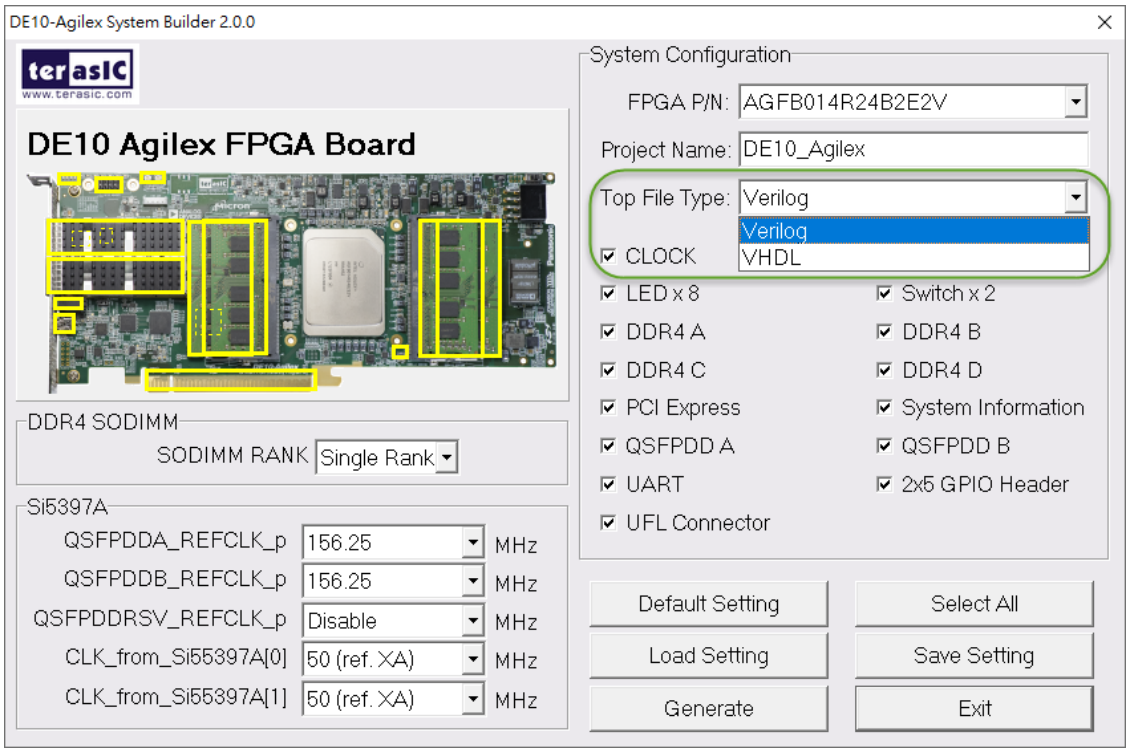


Figure 3-5 Top File Type in the System Builder window

■ System Configuration

Users are given the flexibility of enabling their choices of components connected to the FPGA under System Configuration, as shown in **Figure 3-6**. Each component of the FPGA board is listed to be enabled or disabled according to users' needs. If a component is enabled, the System Builder will automatically generate the associated pin assignments including its pin name, pin location, pin direction, and I/O standards.

Note: The pin assignments for some components (e.g. DDR4, PCIe and QSFP-DD) require associated controller codes in the Quartus project or it would result in compilation error. Hence please do not select them if they are not needed in the design. To use the DDR4 controller, please refer to the DDR4 SDRAM demonstration in

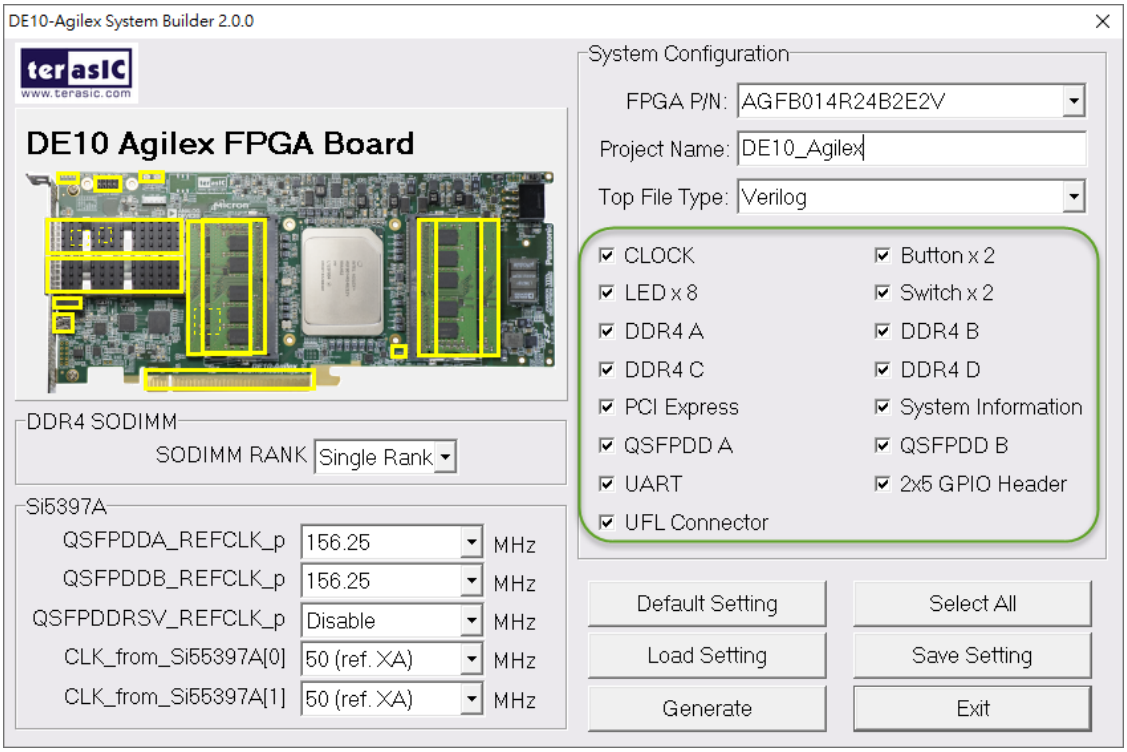


Figure 3-6 System Configuration Group

■ SODIMM RANK

The DDR4 SO-DIMM socket on DE10-Agilex can support single or dual rank DDR4 SO-DIMM modules. The FPGA control pins of these two DDR4 SO-DIMM modules have some different, such as clock enable pin or chip select pins (see **Figure 3-7**) So in system builder, users can choose the SO-DIMM type based on the actual DDR4 SO-DIMM module used as shown in **Figure 3-8**.

Dual Rank

```
////////////// Dual-Rank DDR4-2400 SODIMM with ECC,
output                      DDR4A_ACT_n,
input                       DDR4A_ALERT_n,
output          [16:0]      DDR4A_A,
output           [1:0]      DDR4A_BA,
output           [1:0]      DDR4A_BG,
output           [1:0]      DDR4A_CK,
output           [1:0]      DDR4A_CKE,
output           [1:0]      DDR4A_CK_n,
output           [1:0]      DDR4A_CS_n,
inout            [8:0]      DDR4A_DBI_n,
inout          [71:0]       DDR4A_DQ,
inout            [8:0]      DDR4A_DQS,
inout            [8:0]      DDR4A_DQS_n,
input                       DDR4A_EVENT_n,
output           [1:0]      DDR4A_ODT,
output                      DDR4A_PAR,
output                      DDR4A_RESET_n,
inout                       DDR4A_SCL,
inout                       DDR4A_SDA,
```

Single Rank

```
////////////// Single-Rank DDR4-2666 SODIMM with EC
output                      DDR4C_ACT_n,
output          [16:0]      DDR4C_A,
output           [3:0]      DDR4C_AC_R,
input                       DDR4C_ALERT_n,
output           [1:0]      DDR4C_BA,
output           [1:0]      DDR4C_BG,
output                      DDR4C_CK,
output                      DDR4C_CKE,
output           [1:0]      DDR4C_C,
output                      DDR4C_CK_n,
output                      DDR4C_CS_n,
inout            [8:0]      DDR4C_DBI_n,
inout          [71:0]       DDR4C_DQ,
inout            [8:0]      DDR4C_DQS,
inout            [8:0]      DDR4C_DQS_n,
input                       DDR4C_EVENT_n,
output                      DDR4C_ODT,
output                      DDR4C_PAR,
output                      DDR4C_RESET_n,
inout                       DDR4C_SCL,
inout                       DDR4C_SDA,
```

Figure 3-7 Different between dual and single rank DDR4 control pins

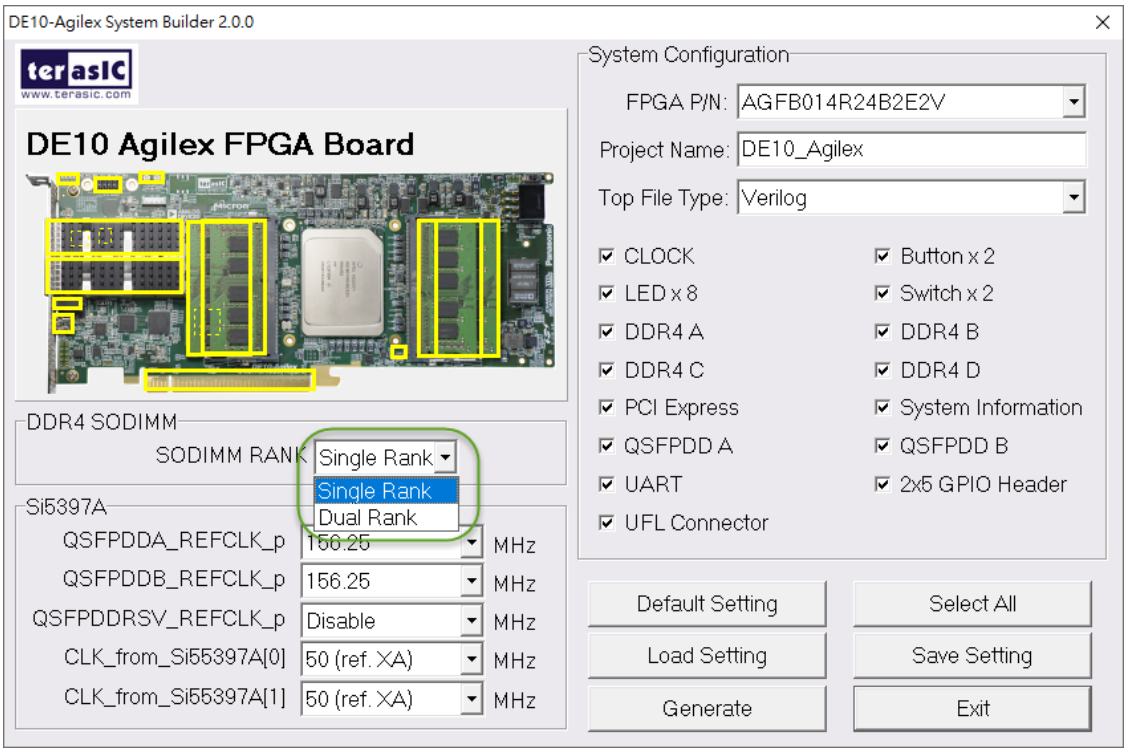


Figure 3-8 SODIMM RANK

■ Programmable Clock Generator

There is an external clock generator Si5397A on-board that provide reference clocks for the following signals:

- QSFPDDA_REFCLK

- QSFPDDB _REFCLK
- QSFPDDRSV_REFCLK
- CLK_from_Si55397A[0]
- CLK_from_Si55397A[1]

To use those clock, users can select the desired frequency on the Si5397A0 groups, as shown in **Figure 3-9**. QSFP-DD port must be checked before users can start to specify the desired frequency in the programmable oscillators.

As the Quartus project is created, System Builder automatically generates the associated controller according to users' desired frequency in Verilog which facilitates users' implementation as no additional control code is required to configure the programmable oscillator.

Note: If users need to dynamically change the frequency, they would need to modify the generated control code themselves.

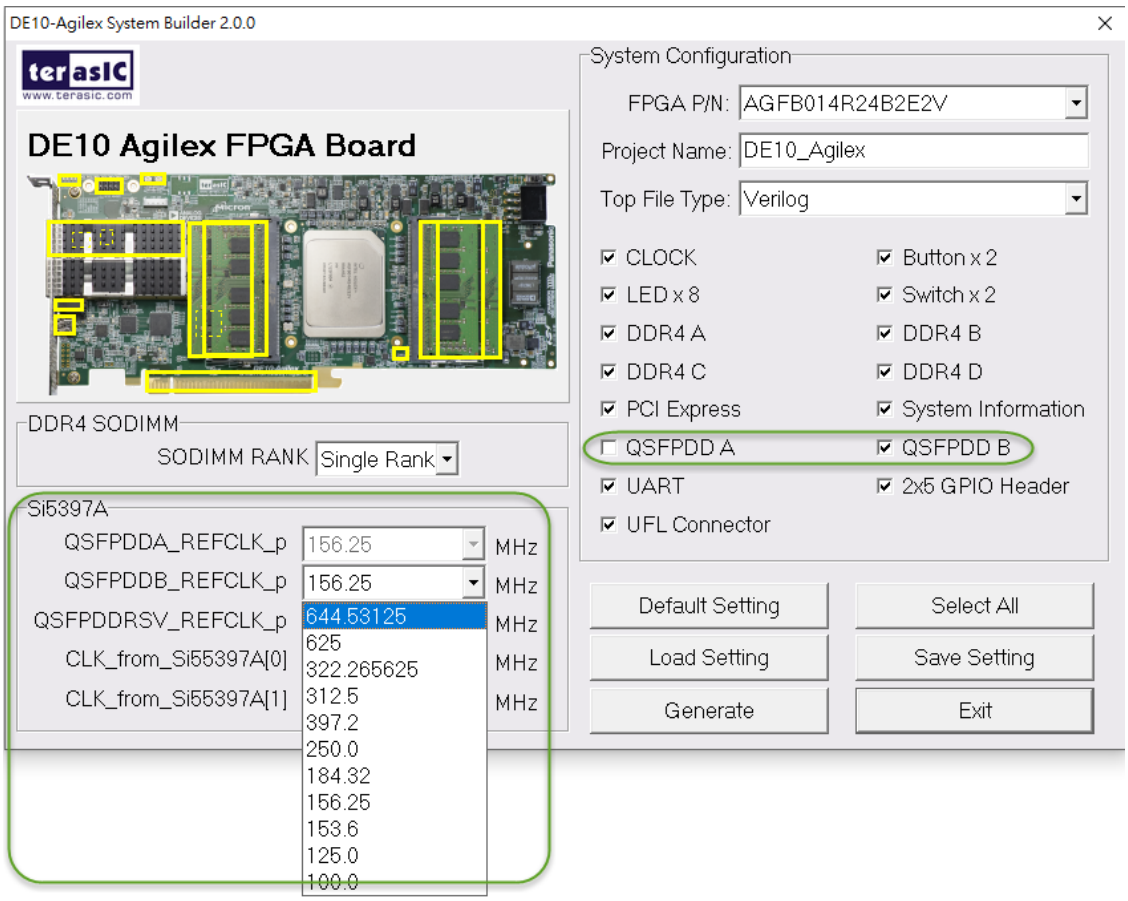


Figure 3-9 External programmable oscillators

The two out clocks (**CLK_from_SI5397A[0]** and **CLK_from_SI5397A[1]**) output from Si5397A to FPGA can have several frequencies to choose (see **Figure 3-10**).

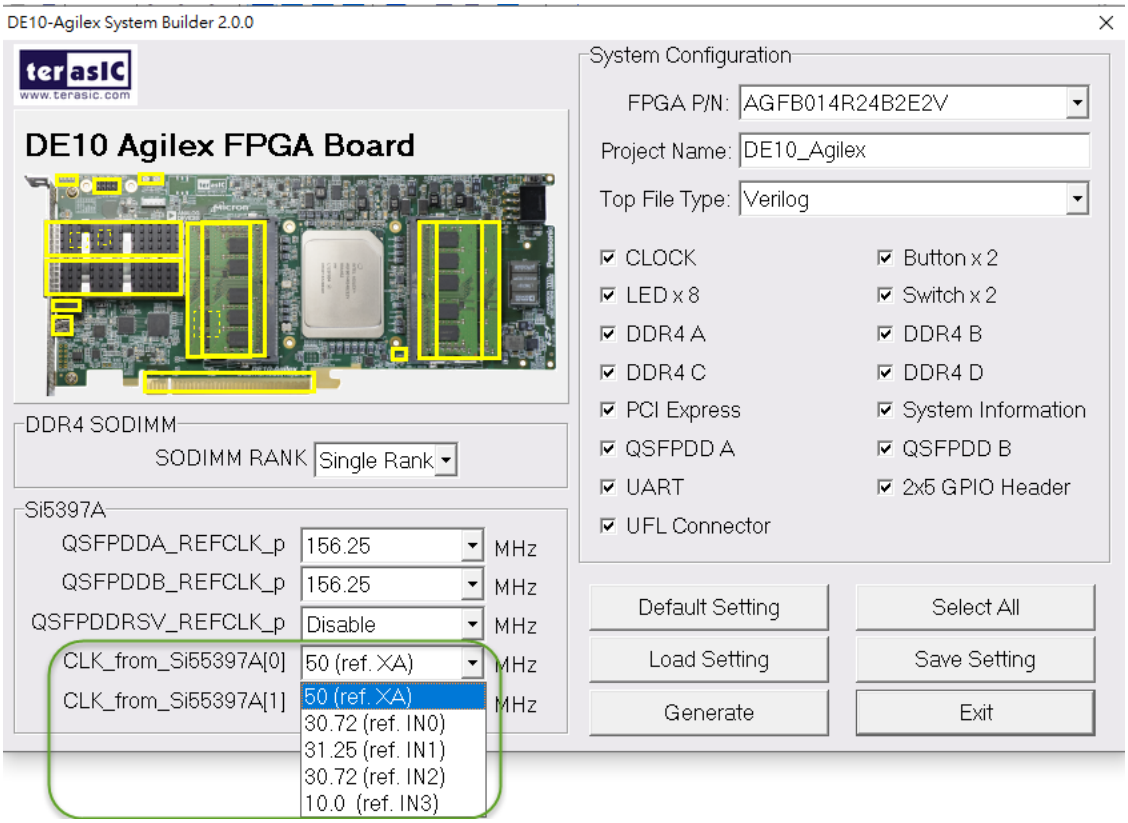


Figure 3-10 CLK_from_SI5397A[0]/[1] clock selection

For more detailed setting about this two clocks, please refer to the section 5,1.

■ Project Setting Management

The System Builder also provides functions to restore default DDR4/QDR-II+/QDR-IV setting, load a pre-saved setting, and save board configuration file, as shown in **Figure 3-11**. Users can save the current board configuration information into a .cfg file and load it into the System Builder later.

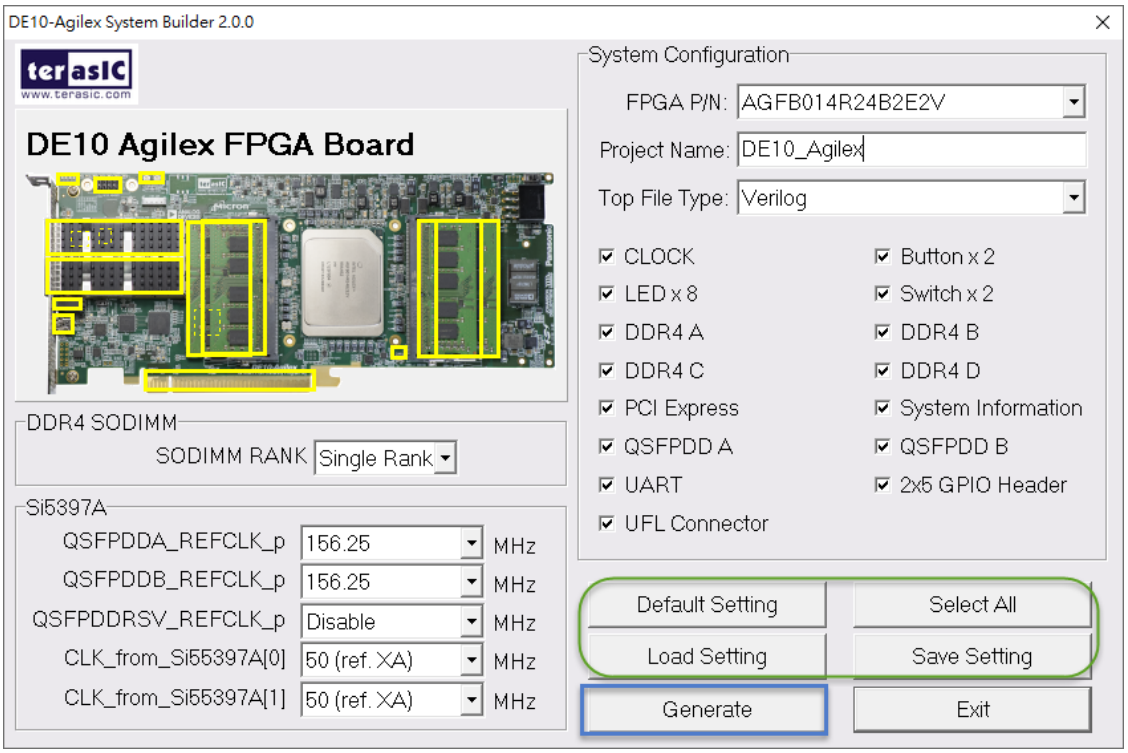


Figure 3-11 Project Settings

■ Project Generation

When users press the Generate button, the System Builder will generate the corresponding Quartus Prime files and documents as listed in the **Table 3-1** directory specified by the user.

Table 3-1 Files generated by the System Builder

No.	Filename	Description
1	<Project name>.v or <Project name>.vhdl	Top Level Verilog/VHDL File for Quartus Prime
2	si5397_controller (*)	Si5397A Clock Generator Controller IP
3	<Project name>.qpf	Quartus Prime Project File
4	<Project name>.qsf	Quartus Prime Setting File
5	<Project name>.sdc	Synopsis Design Constraints File for Quartus Prime
6	<Project name>.htm	Pin Assignment Document

(*) The si5397_controller is a folder which contains the Verilog files for the configuration of Si5397A clock generator chips.

Users can add custom logic into the project and compile the project in Quartus Prime to generate the SRAM Object File (.sof).

The Si5397A controllers will be instantiated in the Quartus Prime top-level file, as listed below:

- If user doesn't change any setting of the Si5397A area (i.e. using the default setting) in the system builder, the top-level file of the generated project will just lists the code of power on reset part. It won't configure the SI5397A.

```
//=======================================================
//  Enable SI5397A. Use default Setting.
//=======================================================
wire por_reset;
power_on_reset power_on_reset_inst(
    .clk(CLK_50_B3A),
    .reset(por_reset)
);

assign SI5397A_RST_n = CPU_RESET_n & ~por_reset;
assign SI5397A_OE_n = ~CPU_RESET_n;


endmodule
```

Figure 3-12 Project Settings

- If user has modified the parameter of the Si5397A area in the system builder, the the top-level file of the generated project will show the associated control IP as shown in below.

```
//=======================================================
//  Configure SI5397A
//=======================================================
wire por_reset;
power_on_reset power_on_reset_inst(
    .clk(CLK_50_B3A),
    .reset(por_reset)
);

assign SI5397A_RST_n = CPU_RESET_n & ~por_reset;
assign SI5397A_OE_n = ~CPU_RESET_n;

// configure Si5397A
`define XCVR_REF_644M53125     4'h0
`define XCVR_REF_625M          4'h1
`define XCVR_REF_322M265625    4'h2
`define XCVR_REF_312M5         4'h3
`define XCVR_REF_307M2         4'h4
`define XCVR_REF_250M          4'h5
`define XCVR_REF_184M32        4'h6
`define XCVR_REF_156M25        4'h7
`define XCVR_REF_153M6         4'h8
`define XCVR_REF_125M          4'h9
`define XCVR_REF_100M          4'hA

`define IO_50M                 4'h0
`define IO_IN0                 4'h1
`define IO_IN1                 4'h2
`define IO_IN2                 4'h3
`define IO_IN3                 4'h4

wire si5397a_config_done;
wire si5397a_controller_start;
assign si5397a_controller_start = ~BUTTON[0];

//  Reconfigure SI5397A
DE10AGILEX_SI5397A_CONFIG si5397a_controller(
    .iCLK(CLK_50_B3A),
    .iRST_n(CPU_RESET_n & ~por_reset),
    .iStart(si5397a_controller_start),
    .iXCVR_CLK_0(`XCVR_REF_184M32),//QSFPDDA_REFCLK_p
    .iXCVR_CLK_1(`XCVR_REF_156M25),//QSFPDDB_REFCLK_p
    .iXCVR_CLK_RSV(1'b0),//QSFPDDRSV_REFCLK_p. 1: output 156.25MHz, 0:
    .iIO_CLK_0(`IO_50M),//CLK_from_Si5397A[0]
    .iIO_CLK_1(`IO_50M),//CLK_from_Si5397A[1]
    .I2C_CLK(SI5397A_I2C_SCL),
    .I2C_DATA(SI5397A_I2C_SDA),
    .REG_CONFIG_DONE(si5397a_config_done)
);


endmodule
```

Figure 3-13 Project Settings

The following clock information also be automatically added in .sdc file.

```
#**************************************************************
# This .sdc file is created by Terasic Tool.
# Users are recommended to modify this file to match users logic.
#**************************************************************

#**************************************************************
# Create Clock
#**************************************************************
create_clock -period "100.000000 MHz" [get_ports CLK_100_B2A]
create_clock -period "30.720000 MHz" [get_ports CLK_30M72]
create_clock -period "50.000000 MHz" [get_ports CLK_50_B3A]
create_clock -period "50.000000 MHz" [get_ports CLK_50_B3C]
create_clock -period "50.000000 MHz" [get_ports CLK_from_SI5397A_p[0]]
create_clock -period "50.000000 MHz" [get_ports CLK_from_SI5397A_p[1]]
create_clock -period "50.000000 MHz" [get_ports UFL_CLKIN]
create_clock -period "100.000000 MHz" [get_ports PCIE_REFCLK_p[0]]
create_clock -period "100.000000 MHz" [get_ports PCIE_REFCLK_p[1]]
create_clock -period "184.320007 MHz" [get_ports QSFPDDA_REFCLK_p]
create_clock -period "156.250000 MHz" [get_ports QSFPDDB_REFCLK_p]
create_clock -period "33.333000 MHz" [get_ports DDR4A_REFCLK_p]
create_clock -period "33.333000 MHz" [get_ports DDR4B_REFCLK_p]
create_clock -period "33.333000 MHz" [get_ports DDR4C_REFCLK_p]
create_clock -period "33.333000 MHz" [get_ports DDR4D_REFCLK_p]

# for enhancing USB BlasterII to be reliable, 25MHz
create_clock -name {altera_reserved_tck} -period 40 {altera_reserved_t
set_input_delay -clock altera_reserved_tck -clock_fall 3 [get_ports alt
set_input_delay -clock altera_reserved_tck -clock_fall 3 [get_ports alt
set_output_delay -clock altera_reserved_tck 3 [get_ports altera_reserve

#**************************************************************
# Create Generated Clock
#**************************************************************
derive_pll_clocks


#**************************************************************
# Set Clock Latency
#**************************************************************
```

Figure 3-14 Project Settings

If the dynamic configurations for the Si5397A clock generators are required, users need to modify the code according to users' desired behavior.

Chapter 4

CFI-Flash Programming

In this chapter, we will introduce how to use the AVSTx16 configuration method to load their design file from the flash memory device to the FPGA after the board power on. As shown in **Figure 4-1**, the System MAX 10 FPGA is the core component of this configuration system. A Parallel Flash Loader II (PFL II) IP is implemented in the System MAX 10 FPGA, allowing users to send bit stream files of user's project form host to the System MAX 10 FPGA through the JTAG interface. Then, the bit stream files will be written into the CFI Flash connected to the System MAX 10 FPGA via the PFL II IP.

After the user's bit stream files are stored into the CFI Flash device, when the DE10-Agilex board is powered on, the PFL II IP in the System MAX 10 FPGA will automatically load the bit stream file from the CFI Flash first, and then configure the FPGA through the Avalon-ST x16 interface.

In this chapter, we will introduce how to correctly set the FPGA to work in AVSTx16 mode, how to program bit stream files into the CFI Flash, and how to switch the image file to be loaded.

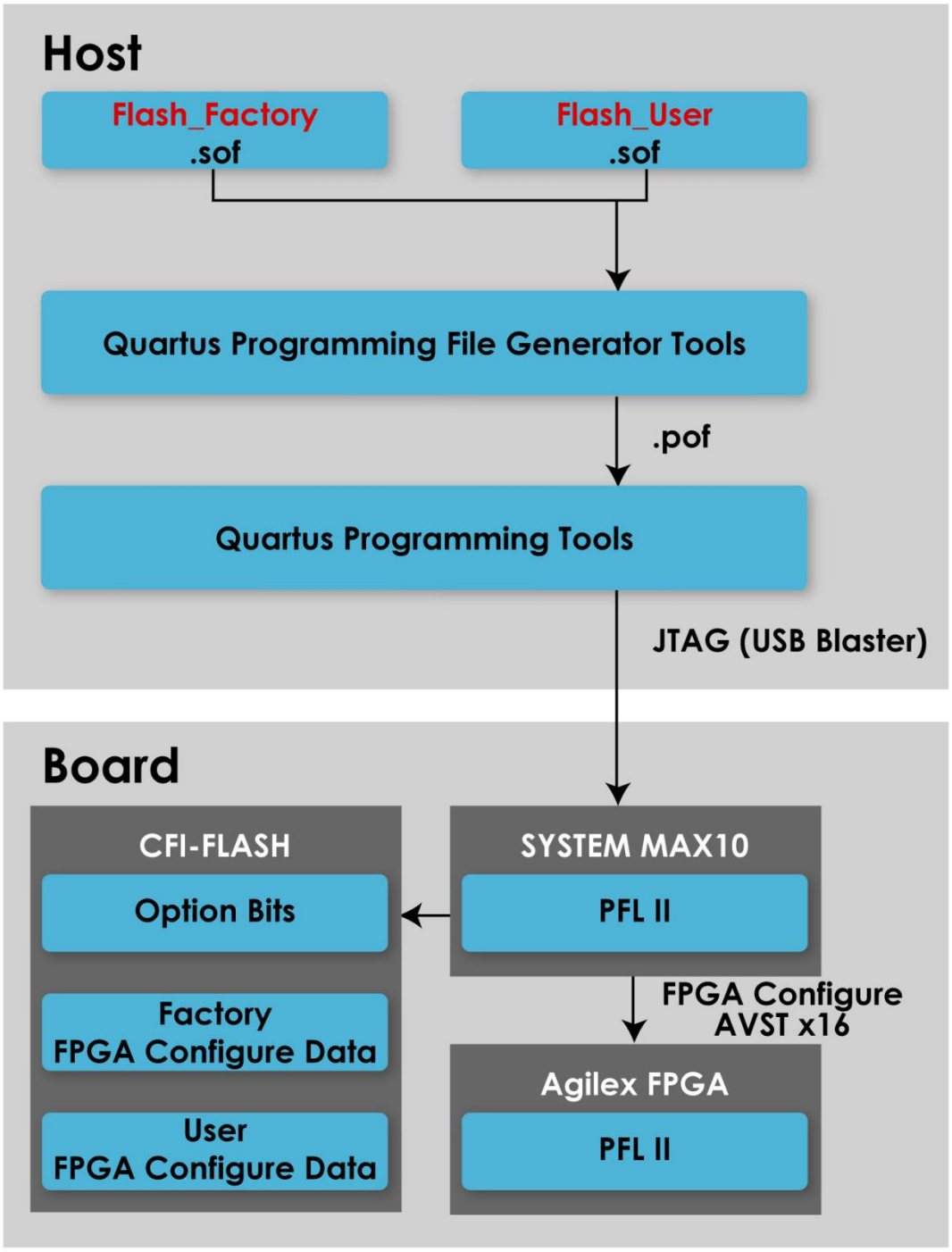


Figure 4-1 Block diagram of the Avalon-ST x 16 mode on the board

Note that the DE10-Agilex board ships with the CFI flash device preprogrammed with two FPGA configurations. The two configuration images are called: **factory** image and **user** image, respectively.

4.1 FPGA Configure Operation

This section will show the procedure for how to enable the FPGA configuration from CFI Flash (Set FPGA to AVSTx16 mode). Also, it will show how to select one boot image between factory image and user image.

1. Make sure the two default FPGA configurations data has been stored in the CFI flash (The board ships with two images in the CFI flash).
2. Set the FPGA configuration mode to AVSTx16 mode by setting **SW6** and **SW7** MSEL[2:0] as **101** as shown in **Figure 4-2** and Figure 4-3.
3. Specify the configuration of the FPGA using the default Factory Configuration Image or User Configuration Image by setting SW6 according to **Figure 4-4**. When the switch is in position “1”, the factory image is used when the system boots. When the switch is in position “0”, user image is used when the system boots.
4. Power on the FPGA board or press the MAX_RST button if board is already powered on,
5. When the configuration is completed, the green Configure Done LED will light. If there is an error, the red Configure Error LED will light (See **Figure 4-5**).

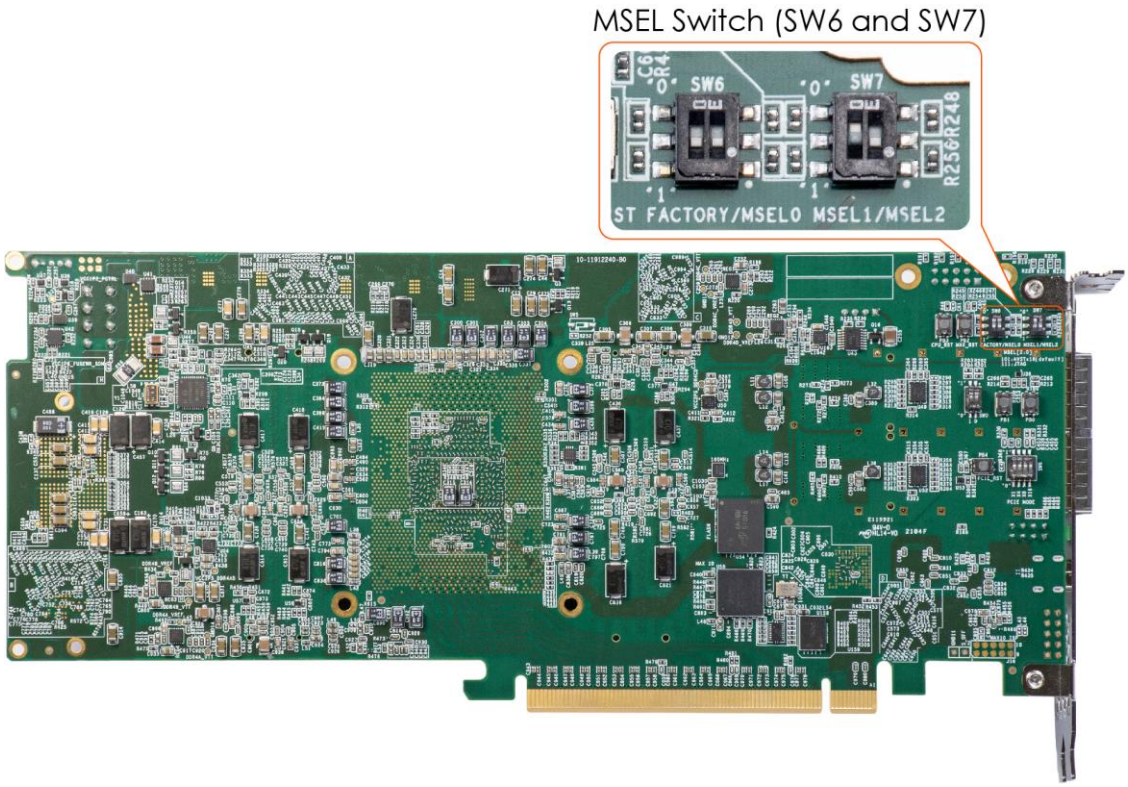


Figure 4-2 Position of the MSEL[2:0] switch

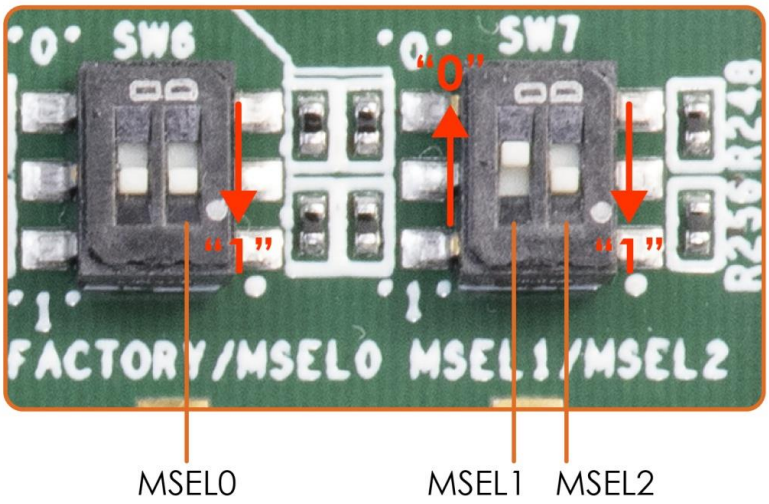


Figure 4-3 Set MSEL[2:0] to "110"

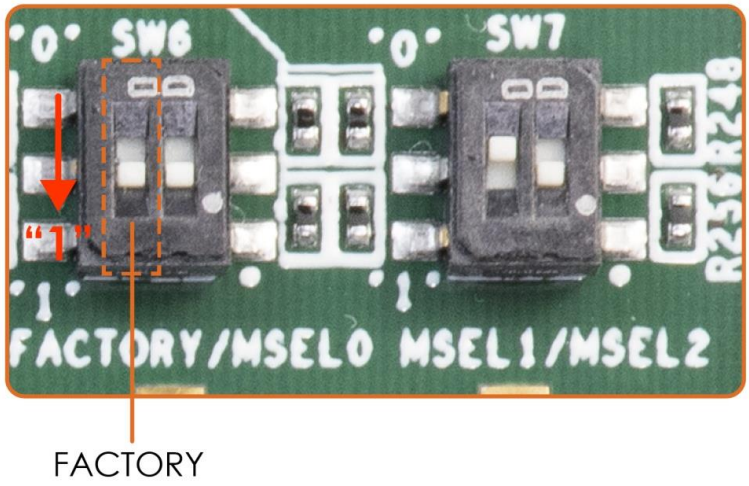


Figure 4-4 Configuration Image Selection

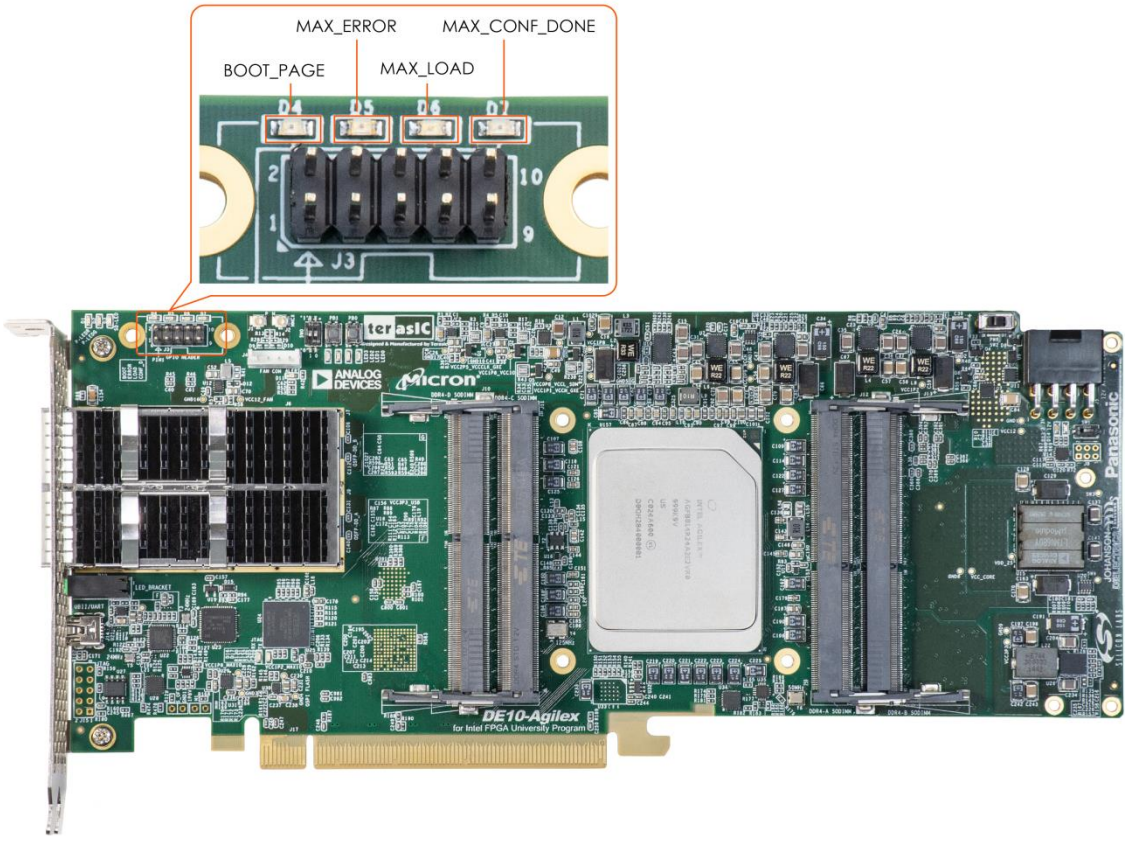


Figure 4-5 Position of the Configuration status LED

4.2 CFI Flash Memory Map

The DE10-Agilex has one 1-Gbit, 16-bit data width, CFI compatible synchronous flash

device for non-volatile storage of the FPGA configuration data. Only the System MAX10 FPGA can access this Flash device.

Table 4-1 shows the memory map for the on-board flash. This memory provides non-volatile storage space for two FPGA bit-streams, and FPL option bits for PFL II configuration bits and board information. For the factory default code to run correctly and update designs in the user memory, this memory map address must not be altered.

Table 4-1 Flash Memory Map (Byte Address)

Block Description	Size(KB)	Address Range
PFL option bits	64	0x00030000 – 0x0003FFFF
Factory image	57,856	0x00040000 – 0x038BFFFF
User image	57,856	0x038C0000 – 0x0713FFFF

The **PFL option bits** contain the image location of the **Factory image** and **User image**, so the PLF II IP in the System MAX10 FPGA can know where to find the FPGA configuration data. If developers erase all flash content, please ensure that the PFL option is reprogrammed with the FPGA configuration data.

For user's application, the **User image** must be stored with start address **0x038C0000**. Users also can overwrite the Factory hardware on their application. **Factory image** must be stored with start address **0x00040000**. We strongly recommend users to use the batch file in the **Flash_Restored** folder to write the hardware and software data into the CFI Flash.

4.3 Programming Bit Stream File Into CFI Flash

This section will introduce how to allow users to program the bit stream file generated by their own project into the CFI Flash on the DE10-Agilex board, so that the design file can be automatically loaded into the FPGA and executed after the board power on.

The CFI Flash of the DE10-Agilex board can only be programmed from the Host PC

through the JTAG interface .But because the CFI Flash is connected to the system MAX10 FPGA on the board. To program Flash, users need to make sure that the system MAX10 FPGA is in the JTAG chain on the board. So when the user wants to program Flash, please make sure the **System MAX JTAG Bypass Switch (SW9)** is switched to the "OFF" position. Please see section 2.3 “System MAX JTAG Bypass Switch” part for more.

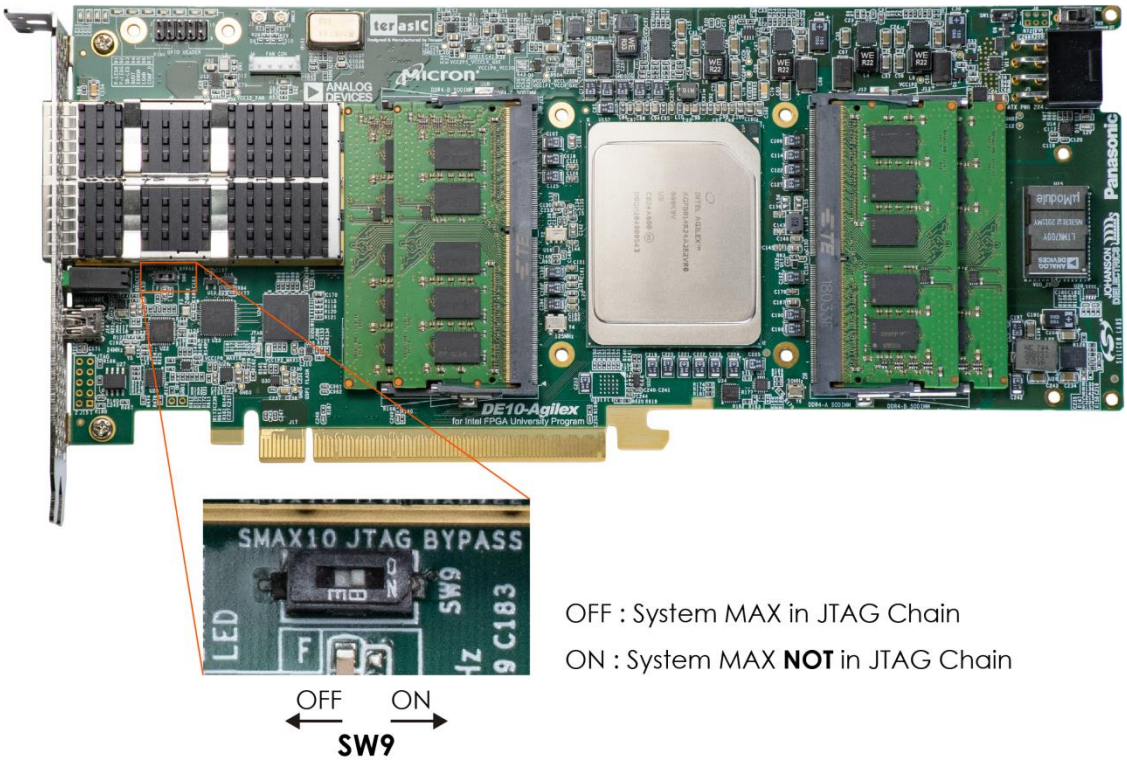


Figure 4-6 System MAX JTAG Bypass Switch

Figure 4-7 shows the standard CFI Flash programming steps. Users first need to convert the bit stream file (.sof) generated by their Quartus project into a Programmer Object File (.pof) file through the Quartus programmer tool. Then use the Quartus programmer tool to connect with the System MAX10 FPGA on the DE10-Agilex board through the JTAG interface and program the .pof file into CFI Flash. In order to help users to quickly program the CFI Flash on the board, Terasic provides some batch files so that users can quickly complete the operations for the CFI Flash such as erasing and programming. Users can find these batch files under the path: *\System CD\Demonstrations\Flash_Restore*.

Users can copy this "**Flash_Restore**" folder to Host for further use.

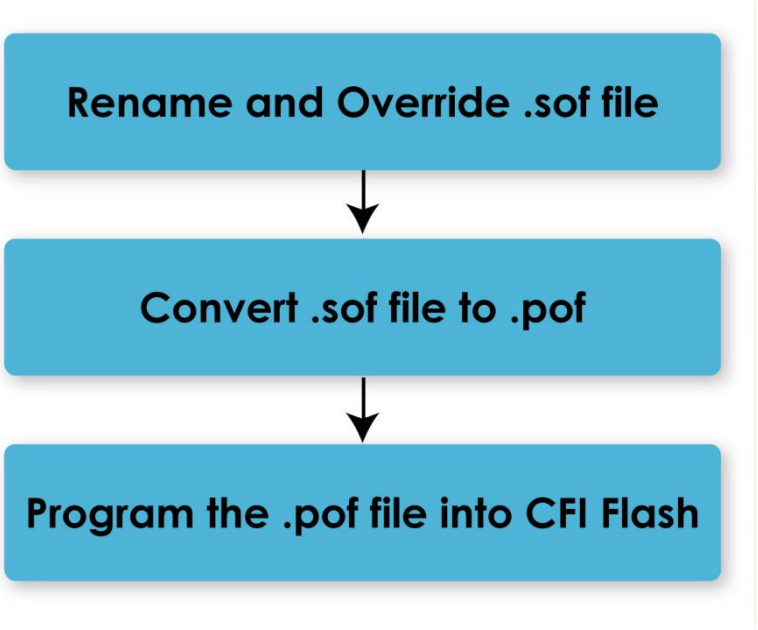


Figure 4-7 Program Flash flow

Table 4-2 is the file list and descriptions in the Flash_Restore folder.

Table 4-2 File list and descriptions of the batch file folder

Example Folder	Description
Factory_HW.sof	The bit stream file to be programmed into the factory image area of the CFI Flash.
User_HW.sof	The bit stream file to be programmed into the user image area of the CFI Flash.
DE10_agilex_flash.pof	This Programmer Object File (.pof) file will be programmed into the CFI Flash. It is converted from Factory_HW.sof and User_HW.sof by Convert.bat.
Convert.bat	Convert.bat will convert Factory_HW.sof and User_HW.sof files from .sof file to .pof file.
Program.bat	Program.bat will program the DE10_agilex_flash.pof into CFI Flash.
Erase.bat	This batch file can help to erase the CFI Flash.

DE10_agilex_flash_pof.map	A text file containing the byte addresses of pages and data stored in the memory of a configuration device for
DE10_Agilex_FLASH_Program.cdf	This Chain Description File (.cdf) will be used with Program.bat to program the .pof file into Flash.
DE10_Agilex_FLASH_Erace.cdf	This Chain Description File (.cdf) will be used with Erase.bat to erase the Flash.
DE10_AG_Config.pfg	This PFG setting file (.pfg) will record the conversion settings for “Programing File Generator” and can be use in the command line.

The detailed description about the usage flow of these batch files is as follows:

■ Override

After users complete their owned project design and generate the .sof file, if they want to use the batch file tools to program the .sof into the CFI Flash on the DE10-Agilex board. The user must first modify the file name of his own .sof file to **Factory_HW.sof** or **User_HW.sof**. Then overwrite the .sof file in the "Flash_Restore" folder. If users want to program their .sof file into the Factory image area, they need to rename their .sof file to **Factory_HW.sof** and copy it into the "**Flash_Restore**" folder (overwrite the original Factory_HW.sof). Similarly, if users want to program into the User image area, rename their .sof file to **User_HW.sof** and copy it into the "**Flash_Restore**" folder.

■ Convert

After the user’s own project’s .sof file has been overwritten into the "Flash_Restore" folder, the next step is to convert the .sof file to a .pof file. Please execute the **Convert.bat** in the **"Flash_Restore**" folder to automatically convert the **Factory_HW.sof** and **User_HW.sof** to **DE10_agilex_flash.pof**. The **DE10_agilex_flash.pof** is the file will be programmed into the CFI Flash.

■ Program

Executing the **Program.bat** will program the **DE10_agilex_flash.pof** file from Host to CFI Flash via System MAX10 FPGA on DE10-Agilex board (please make sure the

MSEL[2:0] on the board is set to 3'b101 and the USB connection is setup from Host to the USB blaster II port on the DE10-Agilex board). Before program the Flash, the system will erase Flash contents first. After the programming step is completed, the users only need to power cycle the board, and the user's design file will be automatically loaded into the FPGA for execution.

■ Erase

When the users want to clear the design in CFI Flash, they can execute **Erase.bat** to complete this action.

4.4 Restore Factory Settings

This section describes how to restore the original **Factory** image and **User** image into the flash memory device on the FPGA development board. A programming batch file located in the **Flash_Restored** folder is used to restore the flash content. Performing the following instructions can restore the flash content:

1. Make sure the Quartus II EDS (Quartus 21.2 Pro edition or later) and USB-Blaster II driver are installed.
2. Make sure the FPGA board and PC are connected with an USB Cable.
3. Power on the FPGA board.
4. Copy the “Demonstrations/Flash_Restored” folder under the CD to your PC’s local drive.
5. Execute the batch file program.bat to start flash programming.

After restoring the flash, perform the following procedures to test the restored boot code.

1. Power off the FPGA Board.
2. Set FPGA configuration mode as AVSTx16 Mode by setting SW6/7 MSEL[2:0] to **101**.
3. Set the System MAX JTAG Bypass Switch (**SW9**) to “**OFF**” position for allowing the MAX10 FPGA into the JTAG chain of the board. So the Flash connected to the system MAX10 FPGA can be programmed via JTAG interface, see section 2.3 “System MAX JTAG Bypass Switch part”.
4. Specify configuration of the FPGA to Factory Hardware by setting the FACTORY_LOAD dip in SW6 to the ‘1’ position.
5. Power on the FPGA Board, and the Configure Done LED should light up.

These batch file converts the **Factory** and **User** .sof and PFL option bit into a DE10_agilex_flash.pof file and use Quartus Programmer to program the CFI-Flash with the generated DE10_agilex_flash.pof. The Factory_HW.sof files generated by **Flash_Factory** project, and the User_HW.sof files generated by **Flash_User** project.

4.5 Flash_Factory Example

The **Flash_Factory** is a simple example for factory verifying purpose. The flash on each DE10-Agilex FPGA board had programmed with the binary file of this example. The binary file of this example is programmed to the "**Factory hardware**" address of the flash (See **Table 4-1**).It is used to allow the user to confirm that the System MAX10 FPGA on the board can read the binary file from the Flash when the DE10-Agilex is power up, and configure the Agilex FPGA through the AVSTx16 mode of the FPGA.

As shown in **Figure 4-8** Board Information, this example contains a NIOS system that can read some information from status sensor on the board, such as temperature, power consumption, and fan status.

It should be noted that, in order to make sure the binary file written in CFI Flash to be automatically loaded through AVSTx16 after DE10-Agilex is power on. User need to make related settings for the Quartus project of the example. Users need to go to Enabling Avalon-ST Device Configuration in the Quartus software of the example. Complete the following steps to specify an Avalon-ST interface for device configuration.

1. Open the Quartus project of this example
2. On the Assignments menu, click Device.
3. In the Device and Pin Options dialog box, select the Configuration category.
4. In the Configuration window, in the Configuration scheme dropdown list, select the appropriate Avalon-ST bus width. In the DE10-Agilex Project, please select AVST x 16 mode, as shown in **Figure 4-9**
5. Click OK to confirm and close the Device and Pin Options dialog box.

```
---------- Board Information ----------
==== Temperature ====
            FPGA: 35*C
            Board 1: 31*C
            Board 2: 33*C
            SDM: 35*C
            E-Tile: 34*C
            P-Tile: 36*C

==== Fan ====
            Fan 1 RPM: 2670
            Fan 2 RPM: 2670

==== Power (12V) Monitor ====
            Voltage       = 11.575 V
            Current       = 1.625 A
            Power         = 18.809 W
==== Core Power Monitor ====
            Voltage       = 0.819 V
            Current       = 3.029 A
            Power         = 2.481 W
```

Figure 4-8 Board Information

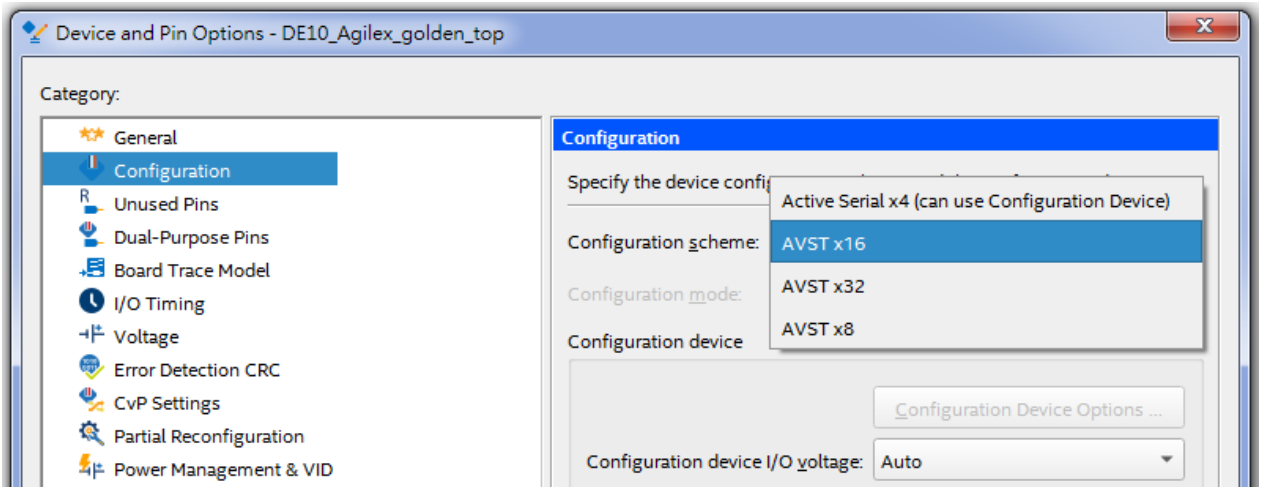


Figure 4-9 Configuration scheme dropdown list

4.6 Flash_User Example

The Flash_User project is similar with the above Flash_Factory example code. This project's FPGA configuration data and Nios II code are stored in the "**User hardware**" area (See **Table 4-1**) when the FPGA board is shipped.

The major difference between the Flash_User and Flash_Factory is the LED control code and the lack of access to the system status on the board.

Chapter 5

Peripheral Reference Design

This chapter introduces DE10-Agilex peripheral interface reference designs. It mainly introduces Si5397A chip which is a programmable clock generator. We provide two ways (Pure RTL IP and Nios II System) respectively to show how to control Si5397A to output desired frequencies, as well as how to control the fan speed. The source codes and tools of these examples are all available in the System CD.

5.1 Configure Si5397A in RTL

There is a Silicon Labs Si5397A clock generators on DE10-Agilex FPGA board can provide adjustable frequency reference clock (See **Figure 5-1** for QSFP-DD. The Si5397A clock generator can output three differential frequencies from 100MHz ~ 644.53125Mhz though I2C interface configuration. This section will show you how to use FPGA RTL IP to configure each Si5397A PLL and generate users desired output frequency to each peripheral

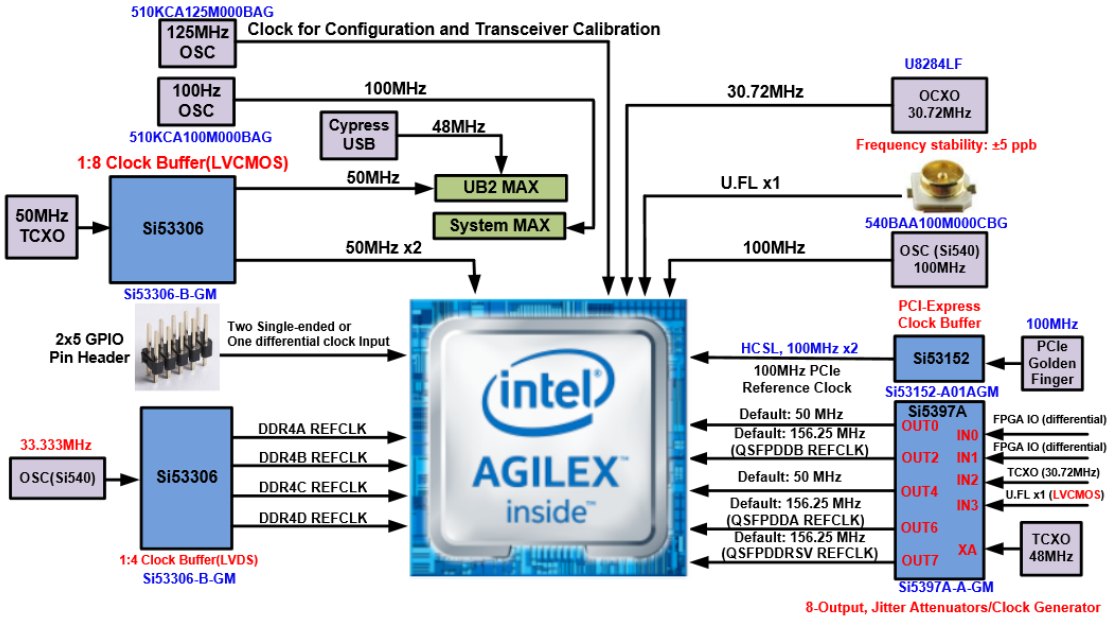


Figure 5-1 The clock tree of the DE10-Agilex

■ Creating Si5397A Control IP

The Si5397A control IP is located in the folder: "\Demonstrations\si5397_controller" in the System CD. Developers can use the IP directly in their Quartus top. Developers can refer to the example in Demonstrations/Clock_Controller folder. This example shows how to instantiate the IP in Quartus top project.

Also, System Builder tool (located in System CD) can be used to help developer to set Si5397A to output desired frequencies, and generate a Quartus project with control IP. In the System Builder window, users can select desired frequencies by selecting a desired output frequency in the pull down menu as shown in **Figure 5-2**. For details about the System Builder, please refer to *Chapter 3 : System Builder*.

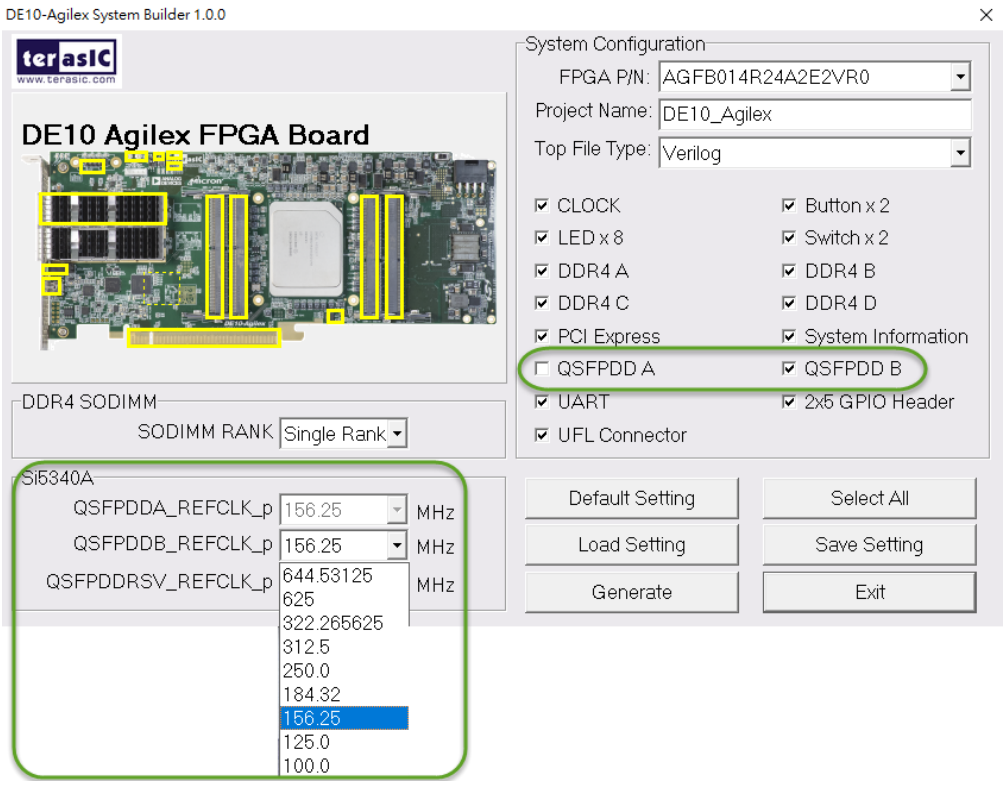


Figure 5-2 Select Desired Si5397A output frequency

■ Using Si5397 control IP

Table 5-1 lists the instruction ports of Si5397A Controller IP.

Table 5-1 Si5397A Controller Instruction Ports

Port	Direction	Description
iCLK	input	System Clock (50Mhz)
iRST_n	input	Synchronous Reset (0: Module Reset, 1: Normal)
iStart	input	Start to Configure（positive edge trigger）
iXCVR_REFCLK_0	input	Setting OUT2 Channel Frequency Value
iXCVR_REFCLK_1	input	Setting OUT6 Channel Frequency Value
iXCVR_REFCLK_RSV	input	Setting OUT7 output control. 1: enable out, 0: disable output
iIO_CLK_0	input	Choose whether the OUT0 pin of Si5397A

		is connected to IN0 or IN1 or IN2 or IN3 pin
iIO_CLK_1	input	Choose whether the OUT4 pin of Si5397A is connected to IN0 or IN1 or IN2 or IN3 pin
PLL_REG_CONFIG_DONE	output	Si5397A Configuration status (0: Configuration in Progress, 1: Configuration Complete)
I2C_DATA	inout	I2C Serial Data to/fromSi5397A
I2C_CLK	output	I2C Serial Clock to Si5397A

As shown in **Table 5-2** and **Table 5-3**, both two Si5397A control IPs have preset several output frequency parameters, if users want to change frequency, users can fill in the input ports " iXCVR_REFCLK_0", "iXCVR_REFCLK_1", and" iXCVR_REFCLK_RSV " with desired frequency values and recompile the project. For example, in the components Si5397A, change

. iXCVR_REFCLK_0 (`XCVR_REF_644M5312),
to.
iXCVR_REFCLK_0 (`XCVR_REF_100M),

Recompile project, the Si5397A OUT0 channel (for QSFPDDB_REFCLK_p) output frequency will change from 644.53125Mhz to 100Mhz.

Table 5-2 Si5397A Controller Reference Clock Frequency Setting for QSFP28

iXCVR0_REFCLK iXCVR1_REFCLK Input Setting	Si5397A Channel Clock Frequency(MHz)
4'h0	644.53125
4'h1	625
4'h2	322.265625
4'h3	312.5
4'h4	250
4'h5	184.32

4'h6	156.25
4'h7	125
4'h8	100
4'h9	125
4'hA	100

Table 5-3 Si5397A Controller Reference Clock Frequency Setting for Memory

XCVR_REFCLK_RSV Input Setting	Si5397A Channel Clock Frequency(MHz)
4'h0	OFF
4'h1	156.25

The two out clocks **iIO_CLK_0** and **iIO_CLK_1 (**CLK_from_SI5397A[0]/[1] **)** output from Si5397A to FPGA can have several frequencies to choose (see **Table 5-4** and **Figure 5-3**).

Table 5-4 Si5397A Controller Reference Clock Frequency Setting for Memory

iIO_CLK_0 iIO_CLK_1 Input Setting	Si5397A Channel Clock Frequency(MHz)
4'h0	50(free run)
4'h1	30.72(from IN0)
4'h2	31.25(from IN1)
4'h3	30.72(from IN2)
4'h4	10(from IN3)

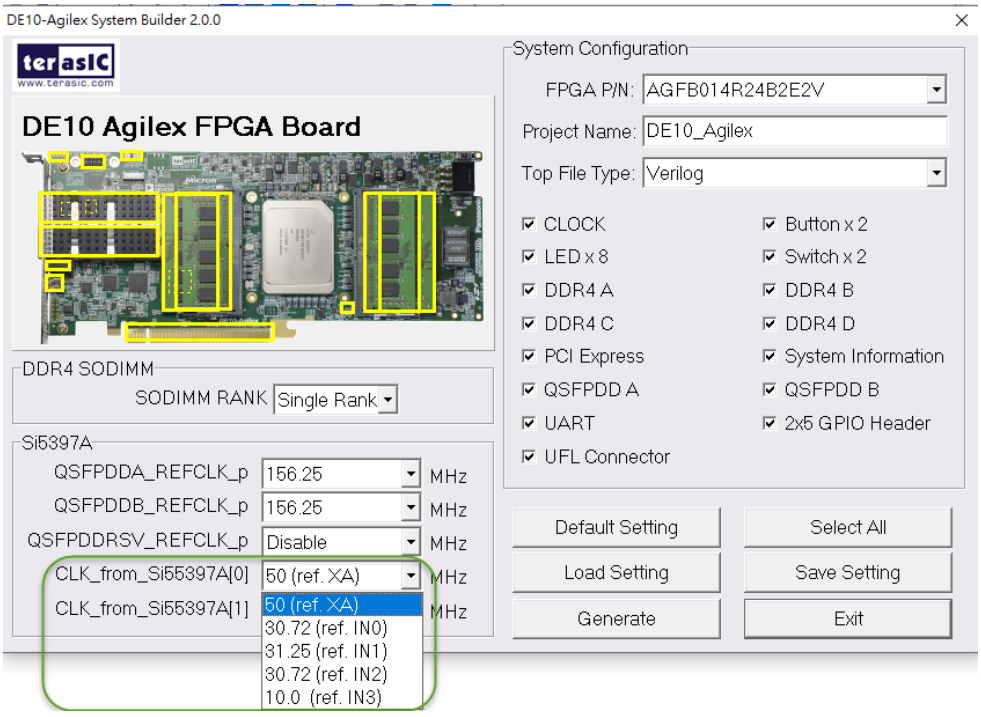


Figure 5-3 iIO_CLK_0/1 (CLK_from_SI5397A[0]/[1]) clock selection

Figure 5-4 shows the basic application status of these clocks. The output frequency of these two clocks are determined by several input clock sources pin (IN0~IN3 and XA pin) of Si5387A.

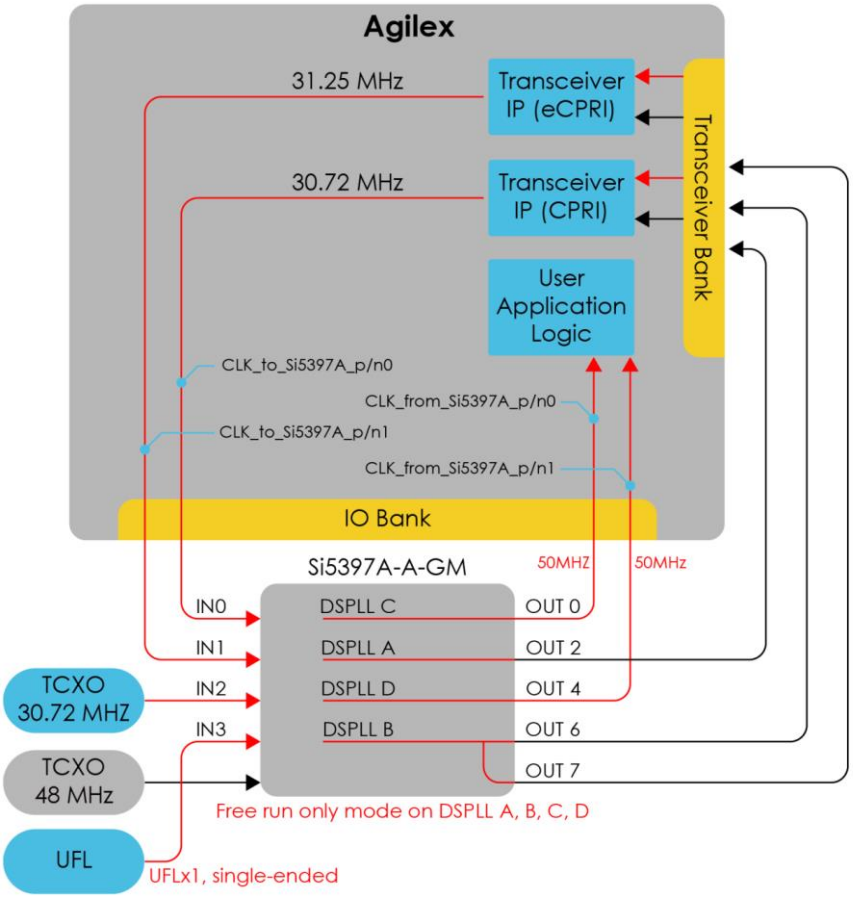


Figure 5-4 Block diagram of the iIO_CLK_0/1 (CLK_from_SI5397A[0]/[1])

The user can decide which input clock pin to use by the system builder or modify the IP parameters, then the corresponding Si5397A control code will be generated. When the FPGA boot up, the Si5397A control code will control the iIO_CLK_0(CLK_from_SI5397A[0]) and iIO_CLK_1(CLK_from_SI5397A[1]) to output the desired clock frequency from Si5397A. The following is the description of the input clock pin corresponding to each set of output clock frequency:

- **50(ref. XA)**: CLK_from_SI5397A[0]/[1] will output 50Mhz (This is the default setting), the input clock source of the SI5397A is an external 48Mhz crystal TCXO.
- **30.72(ref.IN0)**: CLK_from_SI5397A[0]/[1] will output 30.72Mhz, the input clock source of the SI5397A is the net **CLK_to_Si5397A_p/n0** (Si5397A IN0 pin) that FPGA outputs to Si5387A. Note that due to the Si5397A control IP setting, CLK_to_Si5397A_p/n0 must output 30.72Mhz clock.
- **31.25(ref.IN1)**: CLK_from_SI5397A[0]/[1] will output 31.25Mhz, the input clock source of the SI5397A is the net **CLK_to_Si5397A_p/n1** (Si5397A IN1 pin) that

FPGA outputs to Si5387A. Note that due to the Si5397A control IP setting, CLK_to_Si5397A_p/n0 must output 31.25Mhz clock.

- **30.72(ref.IN2**): CLK_from_SI5397A[0]/[1] will output 30.72Mhz, input clock source of the SI5397A is an external 30.72Mhz crystal TCXO (Si5397A IN2 pin).
- **10.0 (ref.IN3)**: CLK_from_SI5397A[0]/[1] will output 10.0Mhz, input clock source of the SI5397A is an U.FL connector (Si5397A IN3 pin), note that due to Si5397A control ip setting, U.FL input clock is required 10Mhz.

If users want to output other different clock frequency of the CLK_from_SI5397A[0]/[1] clock, user may need to manually modify the Si5397A register or contact Terasic for support.

Users can also dynamically modify the input parameters, and input a positive edge trigger for “iStart”, then, Si5397A output frequency can be modified.

After the manually modifying, please remember to modify the corresponding frequency value in SDC file.

■ Modify Clock Parameter for Your Own Frequency

If the Si5397A control IP built-in frequencies are not users’ desired, users can refer to the below steps to the modify control IP register parameter settings to modify the IP to output a desired frequency.

1. Firstly, download ClockBuilder Pro Software (See **Figure 5-3**), which is provided by Silicon Labs. This tool can help users to set the Si5397A’s output frequency of each channel through the GUI interface, and it will automatically calculate the Register parameters required for each frequency. The tool download link:
   http://url.terasic.com/clockuilder_ro_oftware

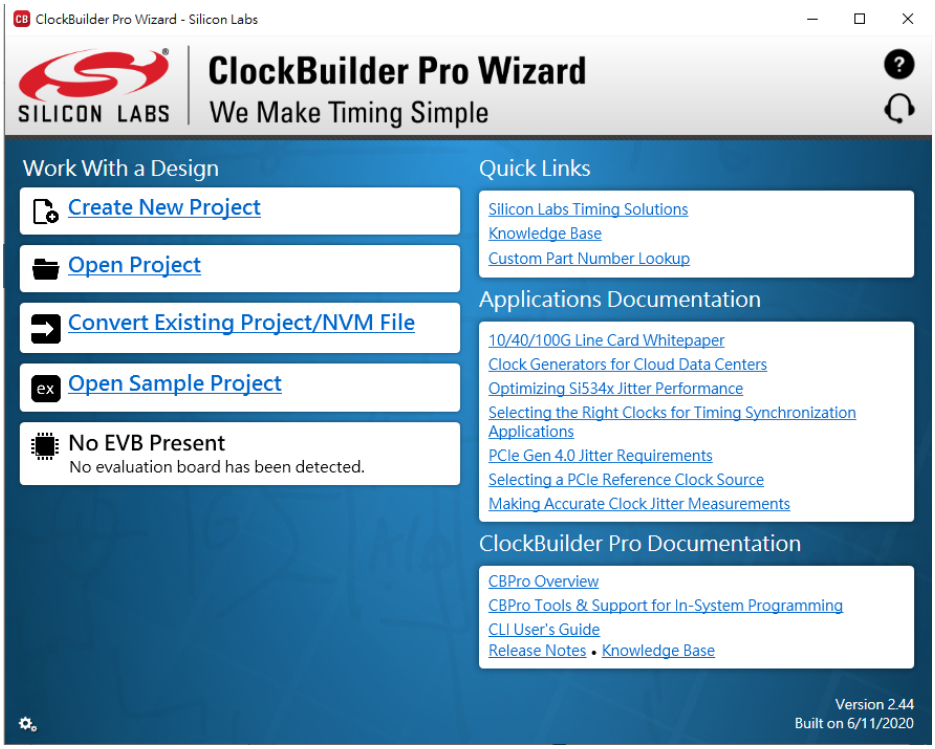


Figure 5-5 ClockBuilder Pro Wizard

2. After the installation, select Si5397, and configure the input frequency and output frequency as shown in **Figure 5-4**.

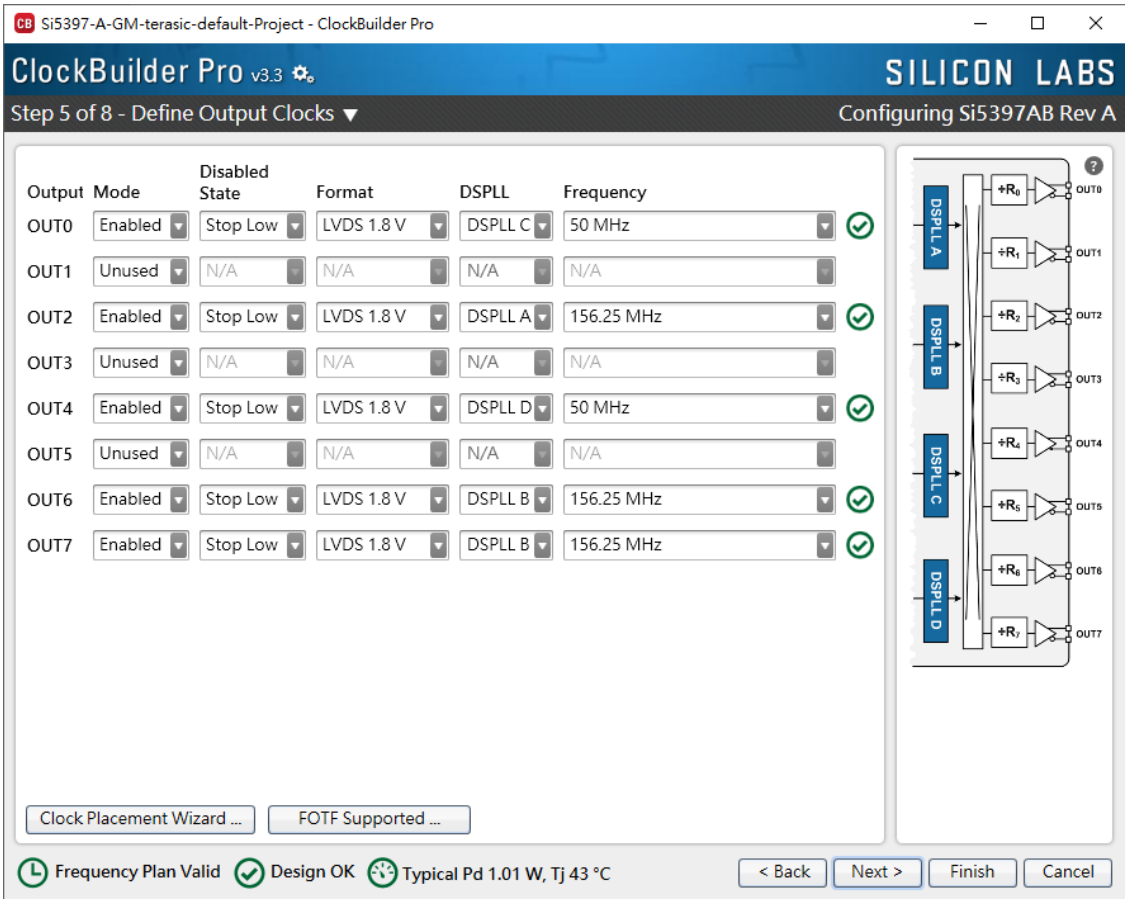


Figure 5-6 Define Output Clock Frequencies on ClockBuilder Pro Wizard

3. After the setting is completed, ClockBuilder Pro Wizard generates a Design Report(text), which contains users setting frequency corresponding register value (See **Figure 5-5**).

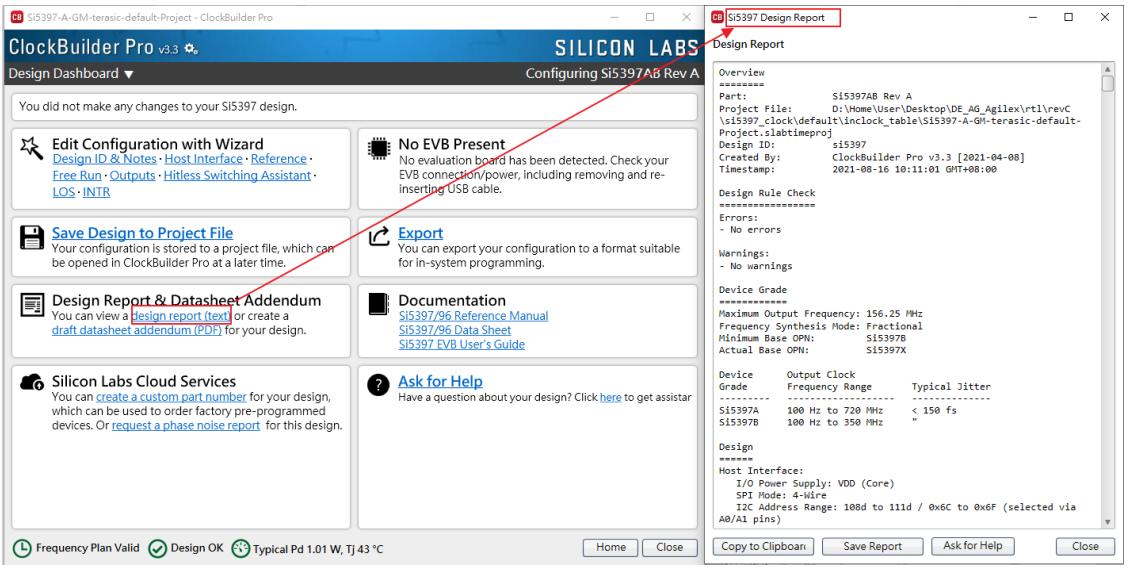


Figure 5-7 Open Design Report on ClockBuilder Pro Wizard

4. Open Si5397 control IP sub-module “si5397a_i2c_reg_controller.v “ as shown in **Figure 5-8**, refer to Design Report parameter to modify sub-module corresponding register value (See **Figure 5-9**).

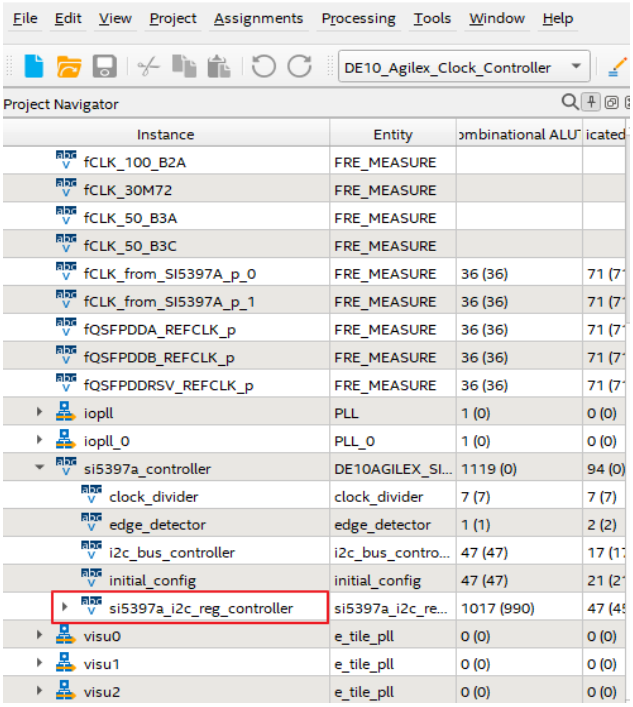


Figure 5-8 Sub-Module file "si5397a_i2c_reg_controller.v"

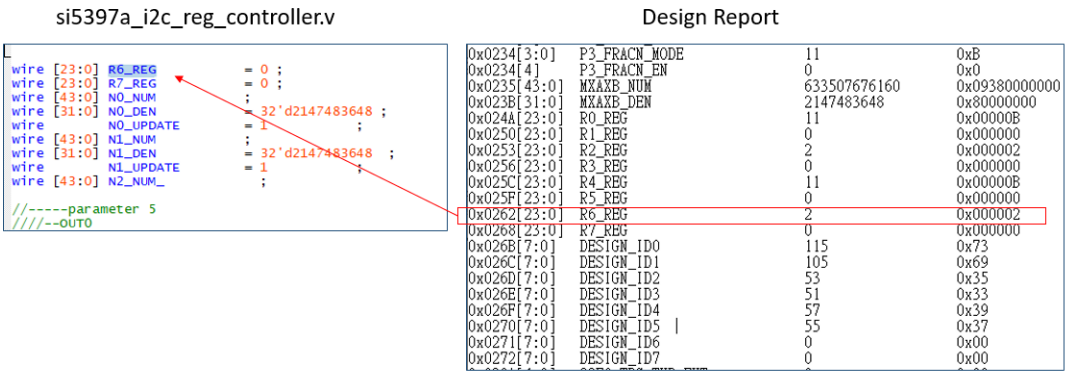


Figure 5-9 Modify Si5397 Control IP Base on Design Report

After modifying and compiling, Si5397A can output new frequencies according to the users' setting.

Note :

1. No need to modify all Design Report parameters in si5397a_i2c_reg_controller.v/si5397b_i2c_reg_controller.v, users can ignore parameters which have nothing to do with the frequency setting

2. After manually modifying, please remember to modify clock constrain setting in .SDC file

5.2 Clock_Controller demo for Si5397

This demonstration shows how to use SI5397A Control IP (written by Verilog) to make si5397 clock generator to generate the desired frequency, and using Quartus SignalTape II tool to verify whether the output frequency of SI5397A is correct.

■ System Block Diagram

Figure 5-10 shows the system block diagram of this demonstration. The Si5397A clock generator is controlled by si5397a_controller_ip through the I2C bus. In this demonstration, a si5397a_controller_ip can generate 9 kinds of frequency settings to control the si5397a clock controller. User can select the output frequency through button1. Each time button1 is pressed, the output frequency setting will be changed. After selecting the output frequency, press button0 to let si5397a_controller_ip output the selected frequency setting to the si5397a clock generator through I2C bus to

generate the desired frequency. Three clocks generated by the Si53040a clock generator will be output to the Agilex FPGA, The input frequency can be calculated by the three modules fQSFPDDA_REFCLK_p, fQSFPDDB_REFCLK_p and fQSFPDDRSV_REFCLK_p. The user can use the SignalTape II tool to observe the measurement registers in the module to verify whether the input frequency is correct

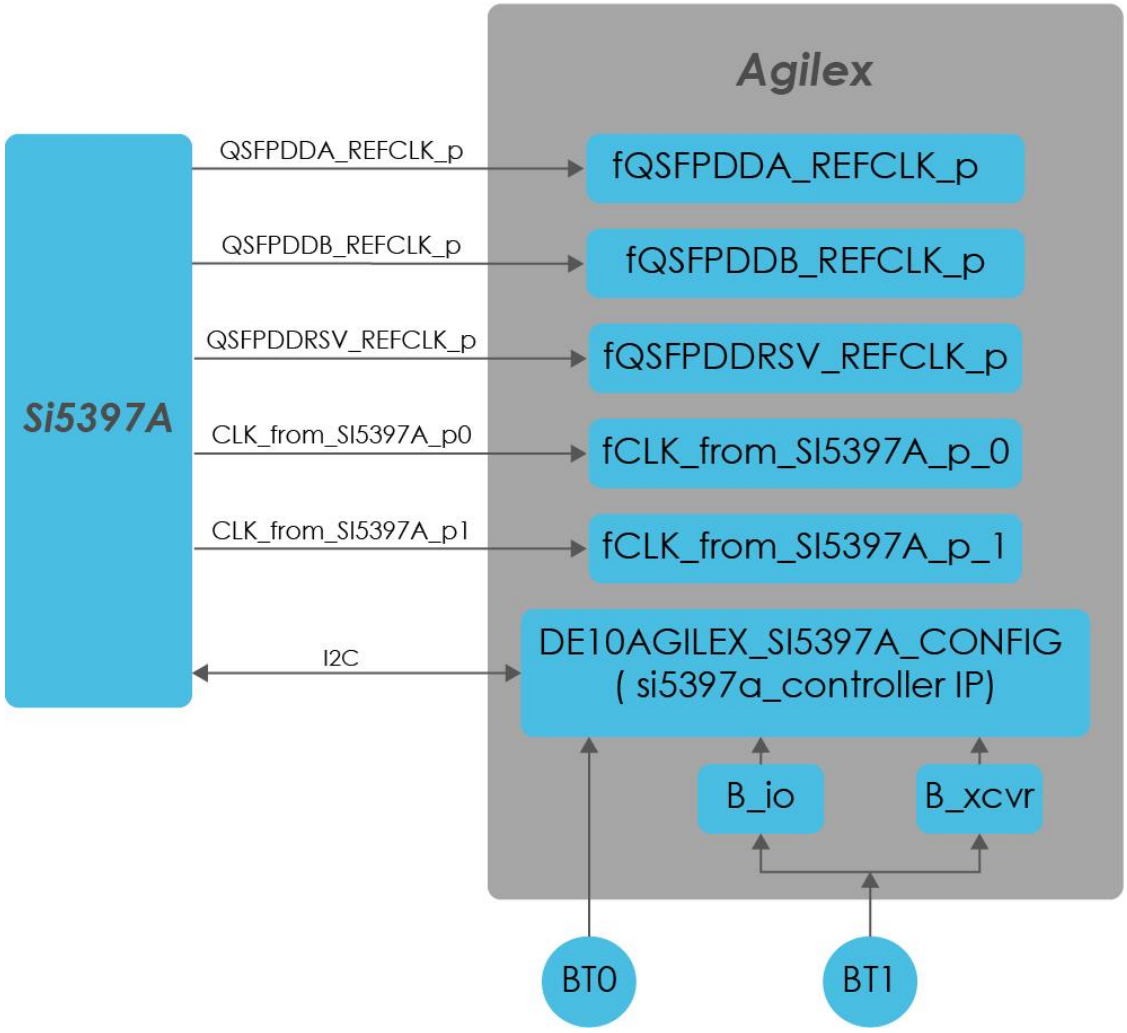


Figure 5-10 Block Diagram of the Clock_Controller Demonstration

■ Design Tools

- Quartus Prime 21.2 Pro Edition

■ Demonstration File Locations

- Hardware project directory: Clock_Controller
- Bitstream used: DE10_Agilex_Clock_Controller.sof
- Demo batch file: Clock_Controller\demo_batch\test.bat

■ Demonstration Setup and Instructions

- Make sure Quartus Pro is installed on the host PC.
- Power on the FPGA board.
- Use the USB Cable to connect your PC and the FPGA board and install USB Blaster II driver if necessary.
- Execute the demo batch file “test.bat” under the batch file folder: Clock_Controller \demo_batch
- Using Quartus II to open the SignalTabl II file “stp1.stp” under the batch file folder: Clock_Controller \demo_batch.
- Switch to the SignalTab II window to observe the clock registers and verify the frequency. IF the register number is “644531540”(See Figure 5-11) that present the input clock frequency is about 184.32Mhz.

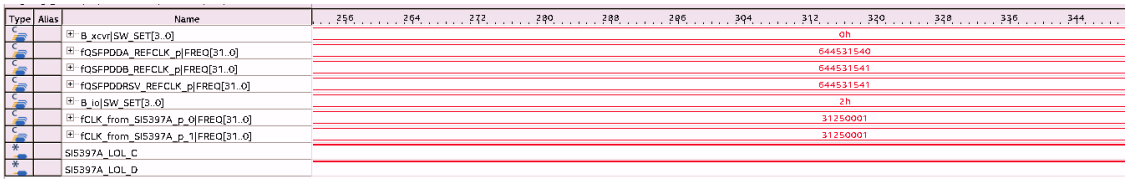


Figure 5-11 SignalTab II result

5.3 Nios II control for SI5397/ Temperature/ Power/Fan

This demonstration shows how to use the Nios II processor to program programmable clock generators (Si5397A) on the FPGA board, how to measure the power consumption based on the built-in power measure circuit. The demonstration also includes a function of monitoring system temperature with the on-board temperature sensor and monitoring fan rotation speed.

■ System Block Diagram

Figure 5-12 shows the system block diagram of this demonstration. The Si5397A clock generator is controlled through I2C controllers driven by Nios II program. The 12V input power monitor, temperature sensor and fan controller connected to the system MAX10 FPGA and controlled by internal logic circuits. All collected status data or control commands will be sent to the SPI slave block so that the Aiglex FPGA can read it through the SPI interface.

In the Agilex FPGA, an SPI master IP (implemented by HDL) will read these external sensor data from the System MAX10 FPGA through SPI interface. The Nios system will read these information or output the PLL control settings through PIO controllers.

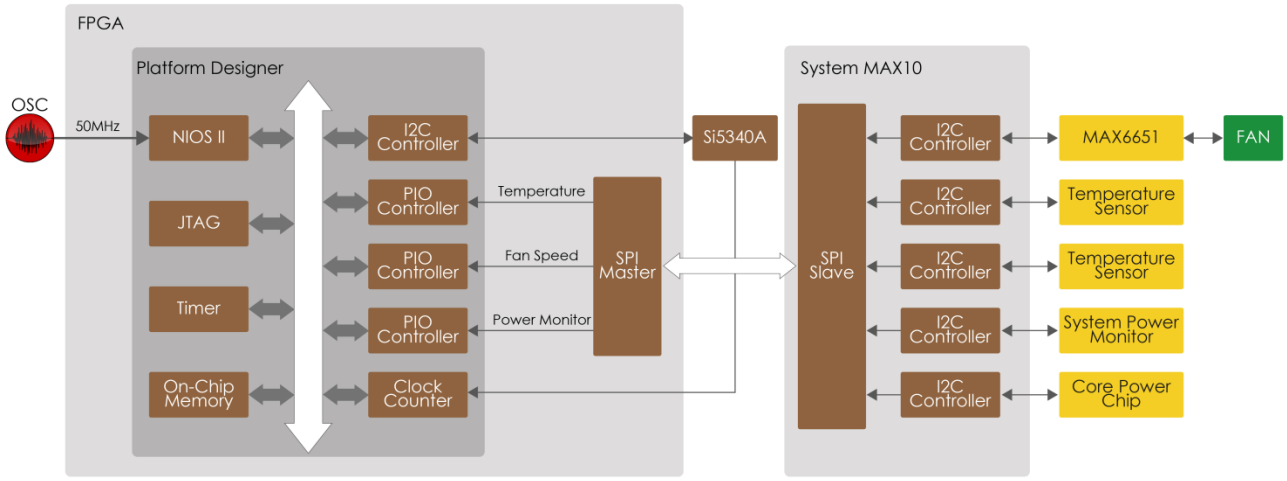


Figure 5-12 Block diagram of the Nios II Basic Demonstration

The system provides a menu in nios-terminal, as shown in **Figure 5-13** to provide an interactive interface. With the menu, users can perform the test for the external programmable PLL and board info sensor. Note, pressing 'ENTER' should be followed with the choice number.

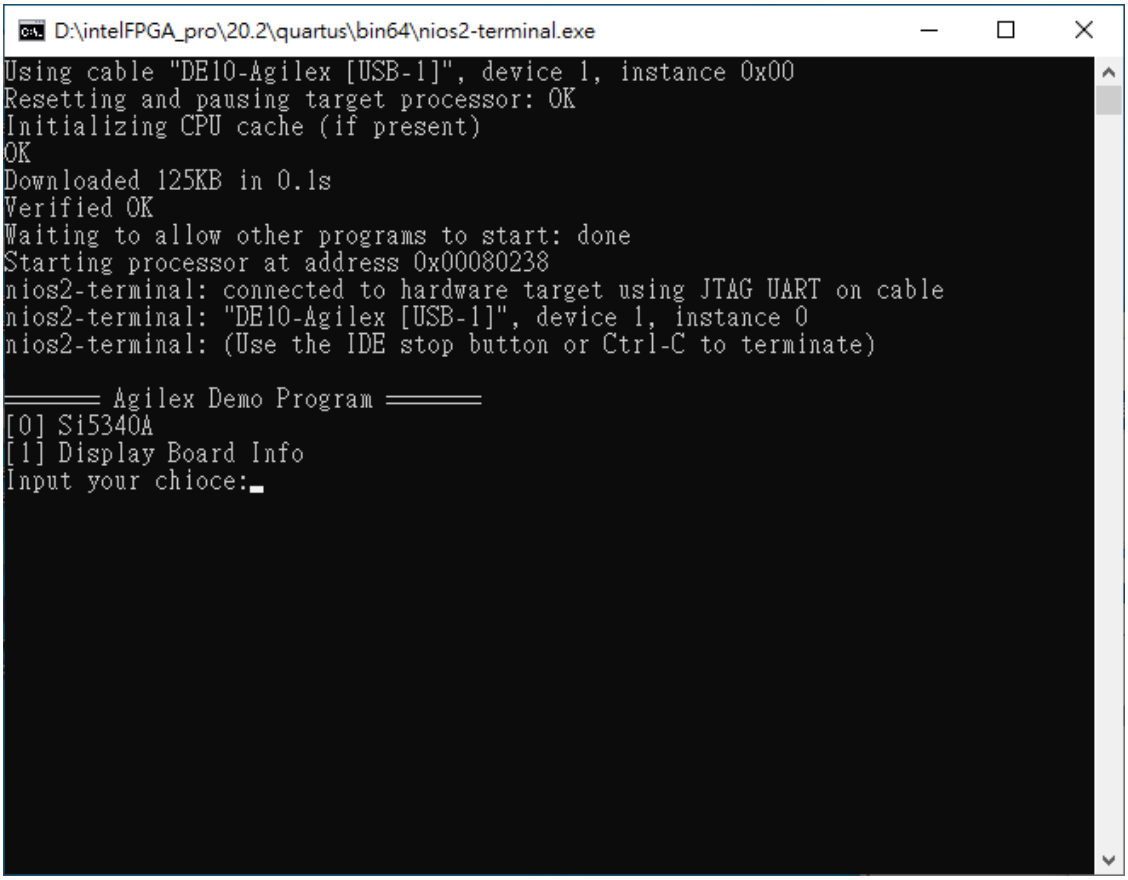


Figure 5-13 Menu of Demo Program

In board info test, the program will display remote temperature, local temperature, 12V input power monitor, core power and fan rotation speed. The remote temperature is the FPGA temperature, and the local temperature is the board temperature where the temperature sensor located. A power monitor IC (LTC2945) embedded on the board can monitor real-time current and power. This IC can work out current/power value as multiplier and divider are embedded in it. There is a sense resistor R4 (0.003 Ω) for LTC2945 in the circuit, when power on the Agilex-Pro, there will be a voltage drop (named ΔSENSE Voltage) on R4. Based on sense resistors, the program of power monitor can calculate the associated voltage, current and power consumption.

In the external PLL programming test, the program will program the PLL first, and subsequently use TERASIC custom Platform Designer CLOCK_COUNTER IP to count the clock count in a specified period to check whether the output frequency as changed as configured. For Si5397A programming, please note the device I2C address is 0xEE. The program can control the Si5397A to configure the output frequency of

QSFPDDA_REFCLK, QSFPDDB_REFCLK and QSFPDDRSV_REFCLK according to your choice.

Design Tools

- Quartus Prime 21.2 Pro Edition

Demonstration File Locations

- Hardware project directory: NIOS_BASIC_DEMO
- Bitstream used: DE10_Agilex.sof
- Software project directory: NIOS_BASIC_DEMO\software
- Demo batch file: NIOS_BASIC_DEMO\demo_batch\test.bat, test.sh

Demonstration Setup and Instructions

- Make sure Quartus Prime and Nios II are installed on your PC.
- Power on the FPGA board.
- Use the USB Cable to connect your PC and the FPGA board and install USB Blaster II driver if necessary.
- Execute the demo batch file “test.bat” under the batch file folder, NIOS_BASIC_DEMO\demo_batch.
- After the Nios II program is downloaded and executed successfully, a prompt message will be displayed in nios2-terminal.
- For the PLL Si5397A test, please input key ‘0’ and input the desired output frequency for two clock sources, as shown in **Figure 5-14**.

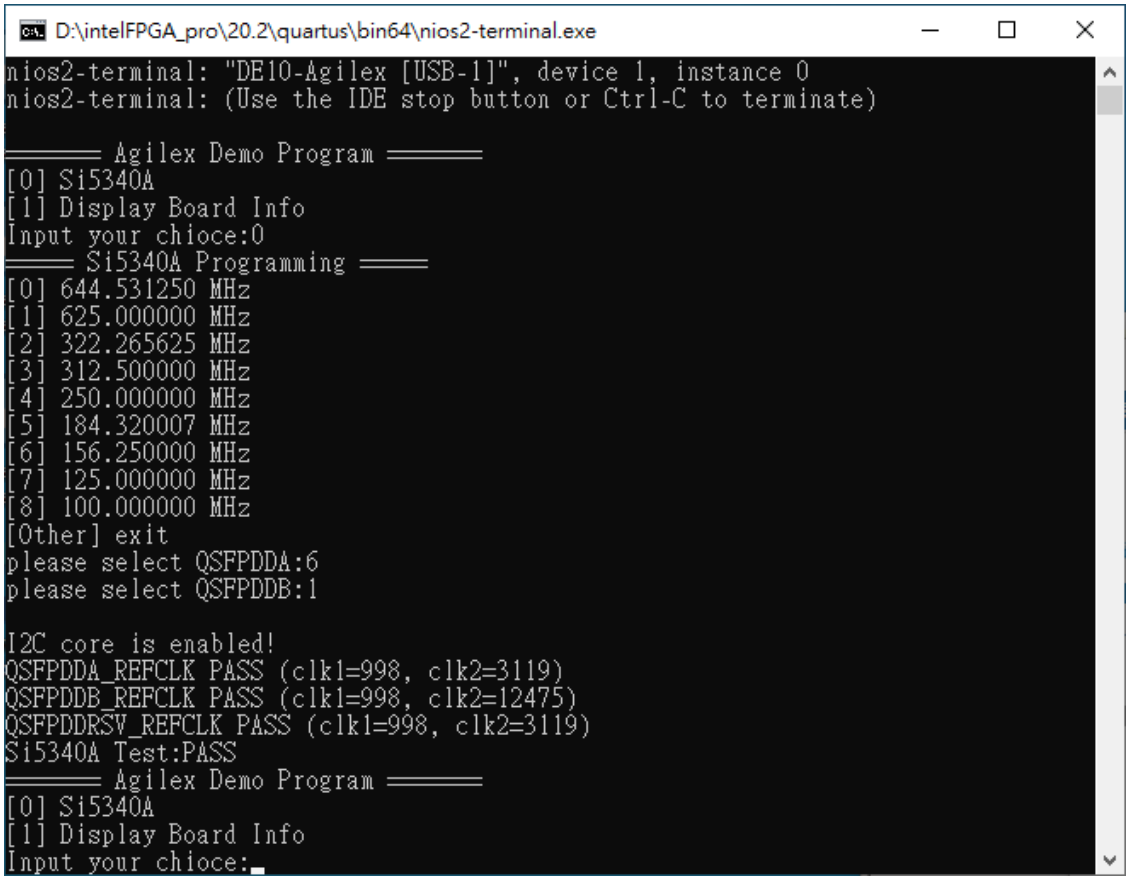


Figure 5-14 Si5397A Demo

- For temperature, power monitor and fan test, please input key '1' and press 'Enter' in the nios-terminal, as shown in **Figure 5-15**.

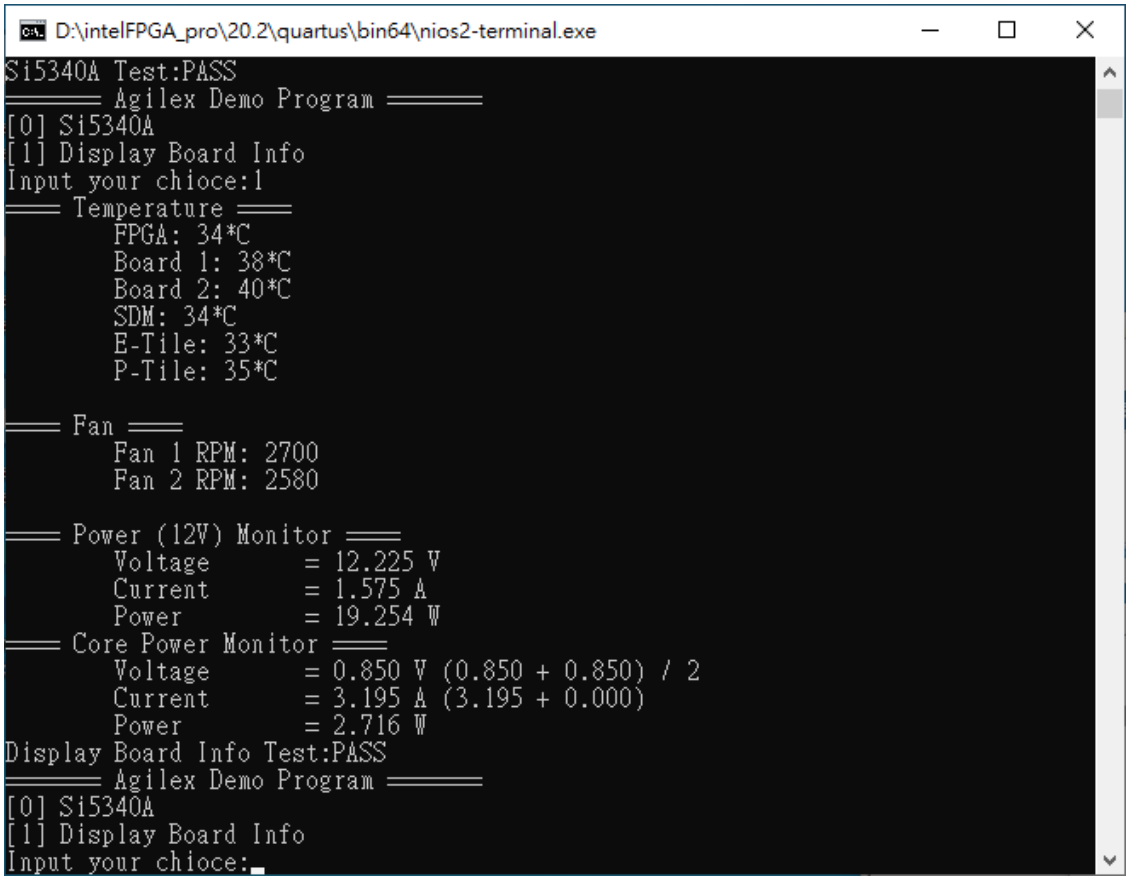


Figure 5-15 Board Info Demo

5.4 Board Information IP

This section will introduce an IP which can be placed in the Agilex FPGA and allows users to obtain board status information such as power, temperature status and fan speed on the DE10-Agilex board.

The DE10-Agilex board provides several sensors to monitor the status of the board, such as FPGA temperature, board power monitor, and fan speed status. These interfaces are connected to the system MAX FPGA on the board. The logic in the system MAX FPGA will automatically read the status values of these sensors and store them in the internal register. As shown in **Figure 5-16**, there is an SPI slave IP in the system MAX FPGA will read the value of the board status from these registers and it can be output to SPI master logic via SPI interface.

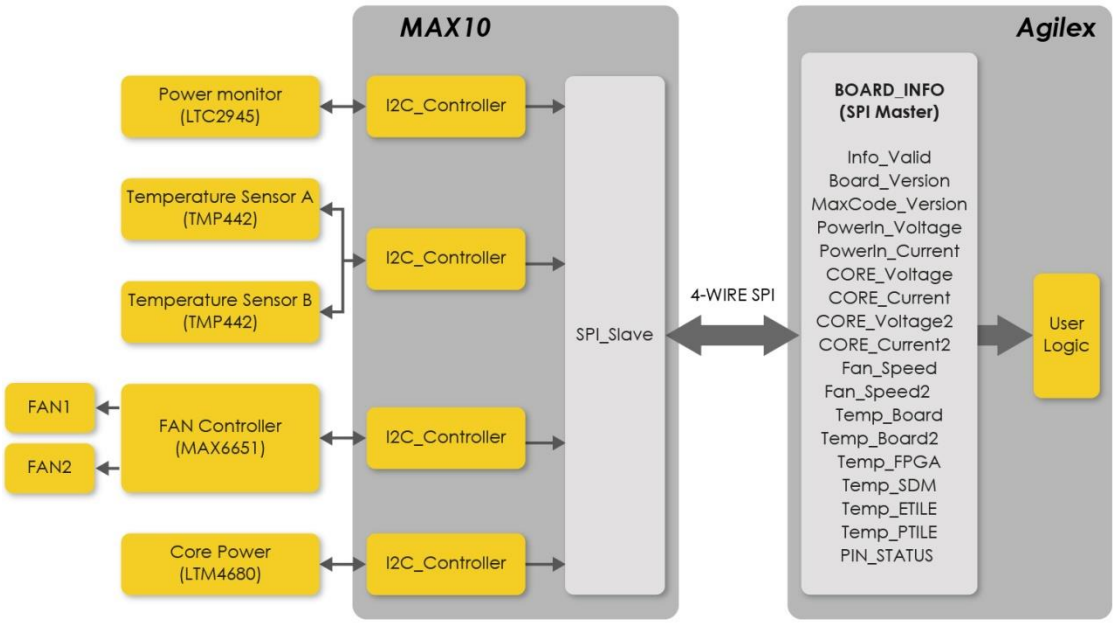


Figure 5-16 Block diagram of the fan speed control demonstration

User can placing a board information IP (BOARD_INFO.v ; SPI master) provided by Terasic in the Agilex FPGA, the board status can be obtained via SPI interface from the system MAX FPGA and output to user logic.

The board information IP can be obtained from the following path in the system CD: ***Demonstration/SPI_Master/SPI/BOARD_INFO.v***

Figure 5-17 shows the input and output pins of the board information IP. Detailed pin descriptions and functions can be obtained from **Table 5-5** Board information IP input and output ports. The user only needs to provide the IP 50Mhz clock and the reset control signal. The IP will automatically communicate with the system MAX FPGA to get the boar status value via the SPI interface. When the logic level of the Info_Valid signal is from low to high, it means that the board status has been updated and can be used.

Finally, **Figure 5-18** shows the status of the IP during execution.

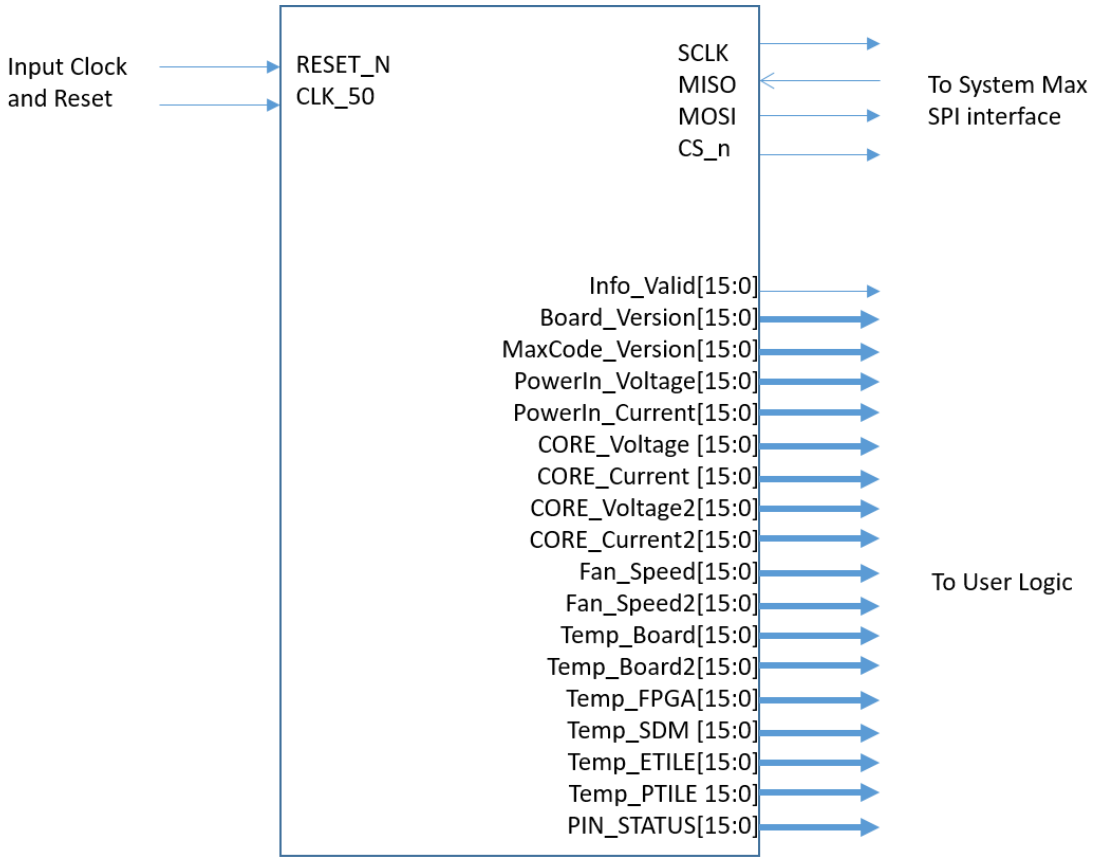


Figure 5-17 Pin out of the board information IP

Table 5-5 Board information IP input and output ports

Port Name	Direction	Width(Bit)	Description
CLK_50	Input	1	Clock input for IP, please input 50Mhz clock.
RESET_N	Input	1	Reset signal for IP, reset all logic.
MOSI	Output	1	Master data output. Please connect this signal to the **INFO_SPI_MOSI** pin.
MISO	Input	1	Master data input. Please connect this signal to the **INFO_SPI_MISO** pin.
CS_n	Output	1	Slave Select, Master output. Please connect this signal to the **INFO_SPI_CS_n** pin.
SCLK	Output	1	Serial Clock, SPI master output to salve. Please connect this signal to the **INFO_SPI_SCLK** pin.
Info_Valid	Output	1	Information valid, logic high indicates board status updated ready.

Board_Version	Output	16	This information indicates the version of the DE10-Agilex board. It will be started at 0x000A.
MaxCode_Version	Output	16	This information indicates the version of the System MAX 10 FPGA code. It will be started at 0x0001.
PowerIn_Voltage	Output	16	12V Voltage, the unit of the output value is mV. If the **PowerIn_Voltage** output value is “12050” that means 12.05V for 12V power
PowerIn_Current	Output	16	Current of the 12V power, the unit of the output value is mA. If the **PowerIn_Current**
CORE_Voltage	Output	16	Core voltage of the first power channel , Unit is mV
CORE_Current	Output	16	Current of the first power channel , Unit is mA
CORE_Voltage2	Output	16	Core voltage of the second power channel , Unit is mV
CORE_Current2	Output	16	Current of the second power channel , Unit is mA
Fan_Speed	Output	16	First fan speed of the board. The unit of the output value is RPM.
Fan_Speed2	Output	16	Second fan speed of the board. The unit of the output value is RPM.
Temp_Board	Output	16	First ambient temperature of the development board. The unit of the output value is Celsius.
Temp_Board2	Output	16	Second ambient temperature of the development board. The unit of the output value is Celsius.
Temp_FPGA	Output	16	Core FPGA temperature of the development board. The unit of the output value is Celsius.
Temp_SDM	Output	16	SDM FPGA temperature of the development board. The unit of the output value is Celsius
Temp_ETILE	Output	16	Temperature of the E-TILE transceiver in the FPGA. The unit of the output value is Celsius
Temp_PTILE	Output	16	Temperature of the P-TILE transceiver in the FPGA. The unit of the output value is Celsius

PIN_STATUS	Output	16	BIT8~15 : Reserved to 0. BIT7 :**FAN_ALERT_n**, When the fan speed is abnormal, this bit is 0. BIT6 : Reserved to 0. BIT5: When shutdown occurs, this bit is 0. BIT4: Reserved to 0 BIT3:**LED_BOOT_PAGE**, enables the access to the flash memory device,this bit is 0. BIT 2:**FPGA_CONF_DONE**,FPGA Configure success, this bit is 1. bit1:**LED_MAX_ERROR**, FPGA Configure failed, this bit is 0. bit0:**LED_MAX_LOAD**, When FPGA is during configuration, this bit is 0.

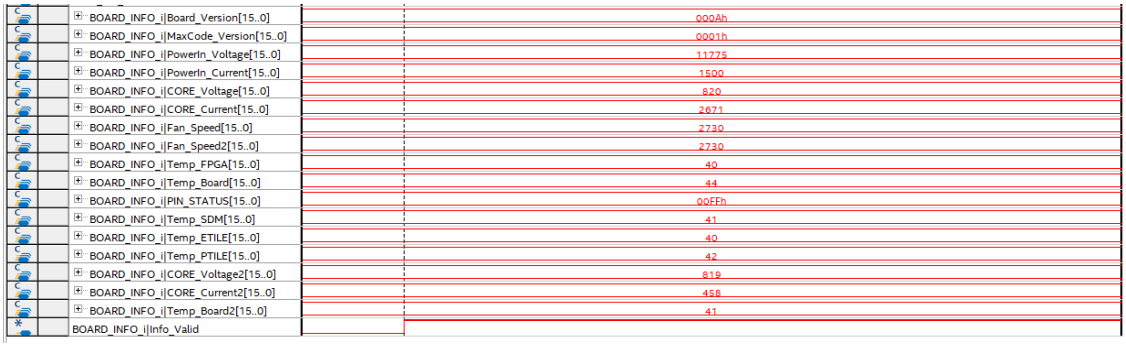


Figure 5-18 Waveform of the board status output

Chapter 6

Memory Reference Design

The FPGA development board includes four DDR4 SODIMM Sockets. This chapter will show three examples which use the memory controller **Agilex External Memory Interfaces (Agilex EMIF)** to perform memory test functions. The source codes of these examples are all available on the FPGA System CD. These three examples are:

- DDR4 SDRAM Test: Test four DDR4-2666 8GB ECC SODIMM Module.
- DDR4 SDRAM Test by Nios II: Test four DDR4-2666 8GB ECC SODIMM Module with Nios II program.

6.1 DDR4 SDRAM Test

This demonstration performs a memory test function on the four DDR4-2666 ECC SO-DIMM on the DE10-Agilex. The memory size of each DDR4 SDRAM SO-DIMM used in this test is 8 GB.

■ Function Block Diagram

Figure 6-1 shows the function block diagram of this demonstration. There are four DDR4 SDRAM controllers. For the Agilex **AGFB014R24A2E2V** FPGA device, the controller in this demo uses 33.33 MHz as a reference clock. It generates one 1333MHz clock (DDR4 2666) as memory clock from the FPGA to the memory. For the **AGFB014R24B1E1V** FPGA device, the controller in this demo uses 33.33 MHz as a reference clock. It generates one 1600MHz clock (DDR4 3200) as memory clock from the FPGA to the memory

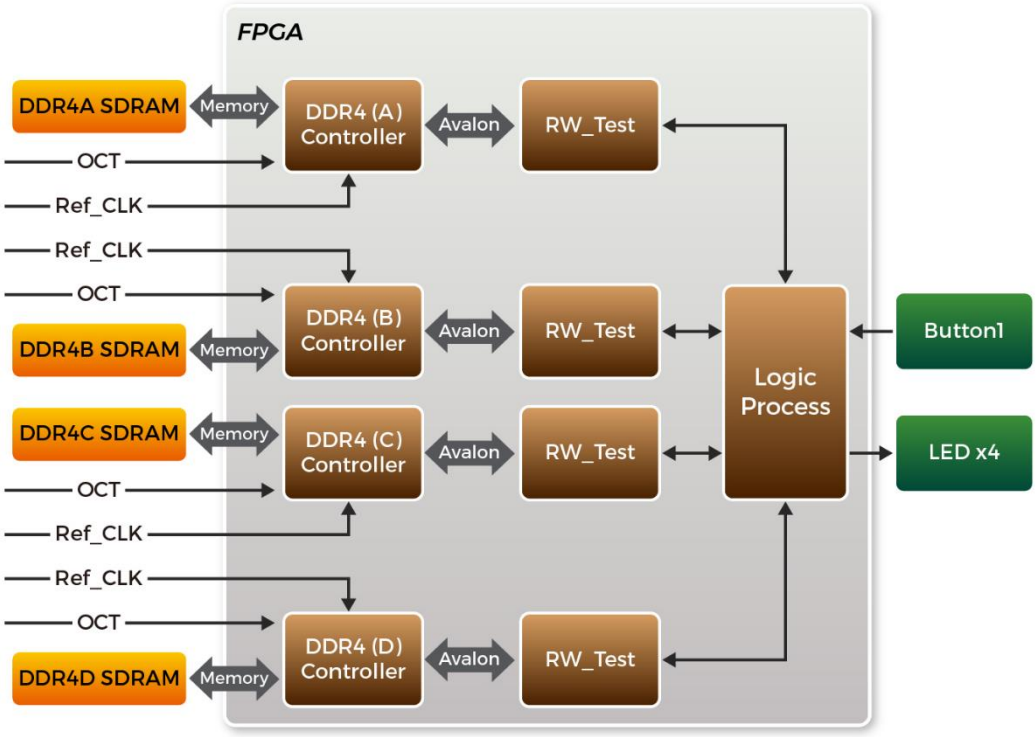


Figure 6-1 Block diagram of DDR4 SDRAM (8G) x4 demonstration

■ Agilex FPGA External Memory Interfaces

To use Agilex External Memory Interfaces controller for DDR4 SODIMM, please perform the two major steps below:

1. Create correct pin assignments for the DDR4 SODIMM.
2. Setup correct parameters in the dialog of the **Agilex FPGA External Memory Interfaces**.

■ Design Tools

- Quartus Prime 21.2 Pro Edition

■ Demonstration Source Code

- Project Directory: Demonstration\RTL_DDR4_x4/32GB
- Bit Stream: DE10_Agilex_golden_top.sof
- Demonstration Batch File

Demo Batch File Folder: RTL_DDR4_x4 \32GB\demo_batch

The demo batch file includes following files:

- Batch File: test.bat
- FPGA Configuration File: DE10_Agilex_golden_top.sof

■ Demonstration Setup

- Make sure Quartus Prime Pro Edition is installed on the Host.
- Connect the DE10-Agilex board to the Host via the USB cable. Install the USB-Blaster II driver if necessary.
- Power on the DE10-Agilex board.
- Execute the demo batch file "test.bat" under the batch file folder \RTL_DDR4_x4\32GB\demo_batch.
- After executing test.bat, it will take about 20 seconds before the test information be printed out in the Command Prompt window.
- Press **BUTTON1** to start DDR4 write & loopback verify process. It will take about one second to perform the test. While testing, the LED will blink. When LED stop blinking it means the test process is done. In this case, if the LED light, it means the test result is passed. If the LED is no light, it means the test result is failed. The LED0 represents the test result for the DDR4 on the SODIMM Socket A, the LED1 represents the test result for the DDR4 on the SODIMM Socket B, and so on.
- Press **BUTTON1** again to regenerate the test control signals for a repeat test.

6.2 DDR4 SDRAM Test by Nios II

Many applications use a high performance RAM, such as a DDR4 SDRAM, to provide temporary storage. In this demonstration hardware and software designs are provided to illustrate how to perform DDR4 memory access in the **Platform Designer** (formerly Qsys). We describe how the memory controller **Agilex FPGA External Memory Interfaces** is used to access the four DDR4 SO-DIMM's on the FPGA board, and how the Nios II processor is used to read and write the SDRAM for hardware verification. The DDR4 SDRAM controller handles the complex aspects of using the DDR4 SDRAM by initializing the memory devices, managing the SDRAM banks, and keeping the devices refreshed at the appropriate intervals.

■ System Block Diagram

Figure 6-2 shows the system block diagram of this demonstration. In the **Platform Designer** (formerly Qsys), one 50 MHz and four 33.333MHz clock source are used. The four 33.333MHz clock source is provided by a dual frequency OSC that can provide frequency to a 1:4 clock buffer, then fan out to four of differential clock pairs to the FPGA for four DDR4 SO-DIMMs. The 50MHz is used by the **Intel FPGA IOPLL** component to generate 200MHz for Nios Processor and On-Chip Memory. The four 33.333MHz clock are used as reference clocks for the DDR4 controllers. There are four DDR4 Controllers which are used in the demonstrations. Each controller is responsible for one DDR4-SODIMM. Each DDR4 controller is configured as a 4GB DDR4 controller (For AGFB014R24A2E2V FPGA device, it runs 1333 MHz; for AGFB014R24B1E1V device, it runs 1600 MHz). The Nios II processor is used to perform the memory test. The Nios II program is running in the On-Chip Memory. A PIO Controller is used to monitor buttons status which is used to trigger starting memory testing.

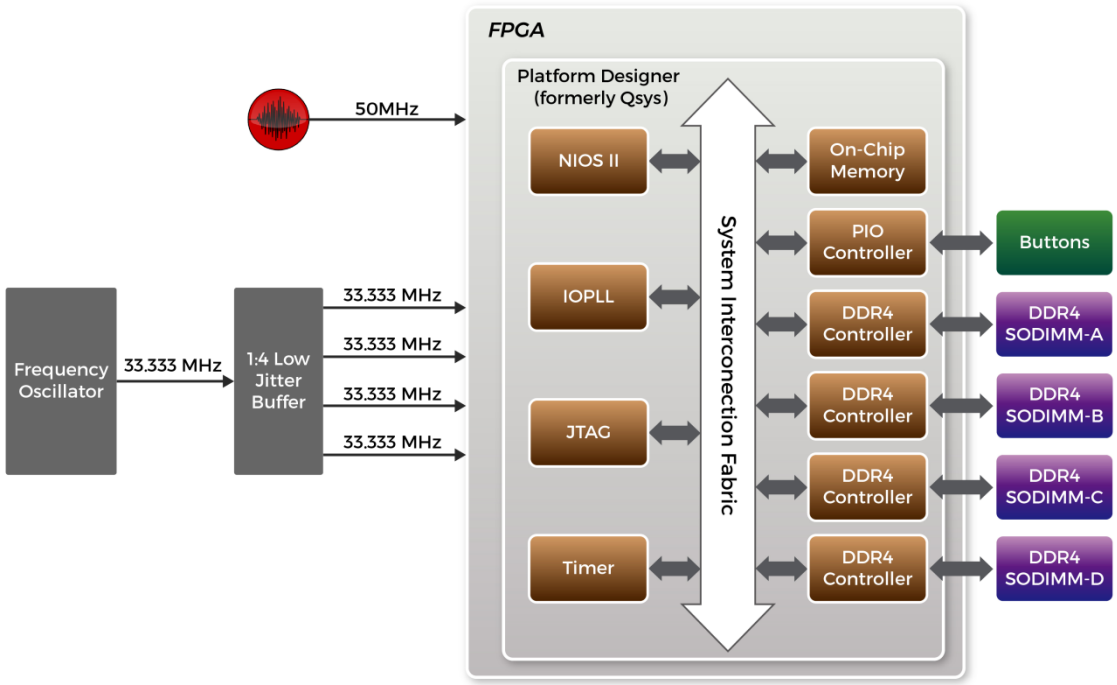


Figure 6-2 Block diagram of the DDR4 Basic Demonstration

The system flow is controlled by a Nios II program. First, the Nios II program writes test patterns into the whole 8GB of SDRAM. Then, it calls Nios II system function, alt_dache_flush_all(), to make sure all data has been written to SDRAM. Finally, it

reads data from SDRAM for data verification. Maybe the process takes a long time, and there is a quick test. The Nios II program writes a constant pattern into the address line and data line and reads it back for verification. The program will show progress in Nios II terminal when writing/reading data to/from the SDRAM. When verification process is completed, the result is displayed in the Nios II terminal.

■ Design Tools

- Quartus Prime 21.2 Pro Edition

■ Demonstration Source Code

- Quartus Project directory: Nios_DDR4_X4\32G
- Nios II Eclipse: NIOS_DDR4_X4\32G\software

■ Nios II Project Compilation

Before you attempt to compile the reference design under Nios II Eclipse, make sure the project is cleaned first by clicking 'Clean' from the 'Project' menu of Nios II Eclipse.

■ Demonstration Batch File

Demo Batch File Folder: NIOS_DDR4_X4\32G\demo_batch

The demo batch file includes following files:

- Batch File for USB-Blaster II: test.bat, test.sh
- FPGA Configure File: DE10_Agilex_golden_top.sof
- Nios II Program: MEM_TEST.elf

■ Demonstration Setup

Please follow below procedures to set up the demonstrations.

- Make sure Quartus Prime and Nios II are installed on your PC.
- Make sure four DDR4 SODIMMs are installed on the FPGA board.
- Power on the FPGA board.
- Use a USB Cable to connect the PC and the FPGA board and install USB Blaster II driver if necessary.
- Execute the demo batch file "test.bat" under the folder

"NIOS_DDR4_X4\32G\demo_batch".

- After the Nios II program is downloaded and executed successfully, a prompt message will be displayed in the nios2-terminal.
- For DDR4x4 test, please input key '0' and press 'Enter' in the nios2-terminal as shown in **Figure 6-3 and Figure 6-4** (DDR4A and DDR4B test). The program will display progressing and result information. Press Button0~Button1 of the FPGA board to start SDRAM verify process, and press Button0 for continued test.
- For DDR4x4 quick test, please input key '1' and press 'Enter' in the nios2-terminal as shown in **Figure 6-5**. The program will display progressing and result information. Press Button0~Button1 of the FPGA board to start SDRAM verify process, and press Button0 for continued test.

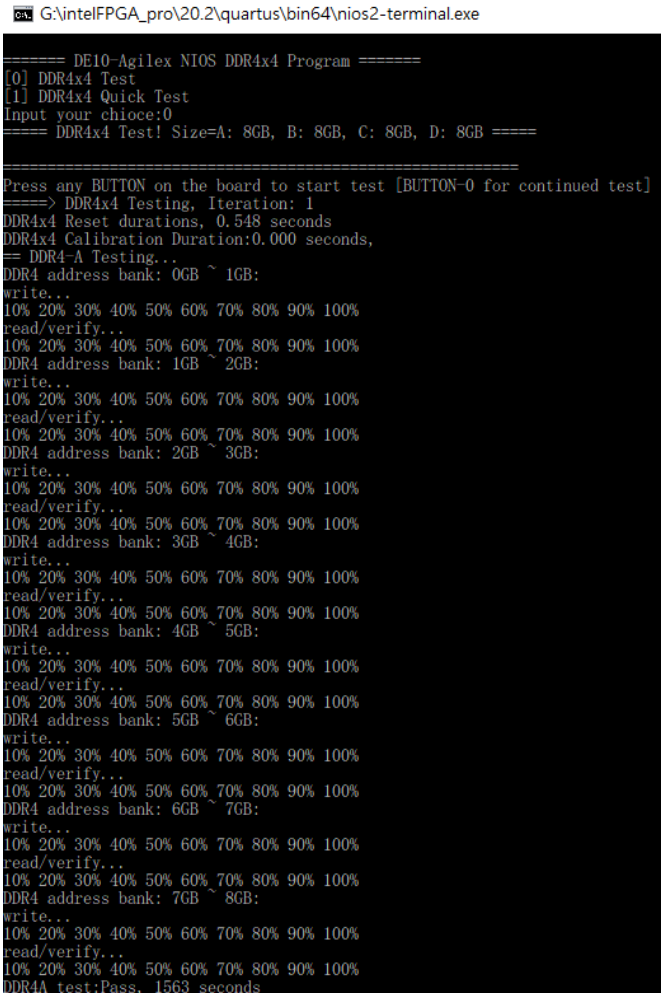


Figure 6-3 Progress and Result Information for the DDR4A Test

Figure 6-4 Progress and Result Information for the DDR4B Test

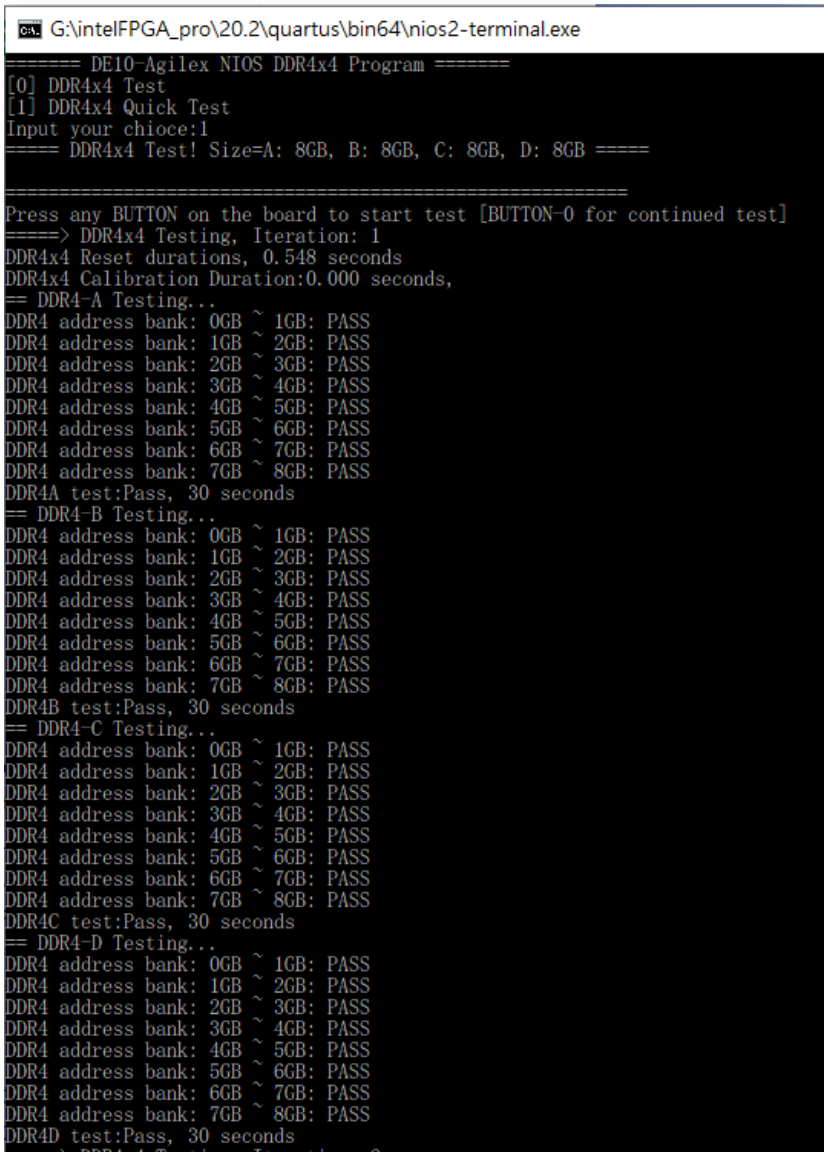


Figure 6-5 Progress and Result Information for the DDR4A~DDR4D quick test

Chapter 7

PCI Express Reference Design for Windows

PCI Express is commonly used in consumer, server, and industrial applications, to link motherboard-mounted peripherals. From this demonstration, it will show how the PC Windows and FPGA communicate with each other through the PCI Express interface. Agilex Hard IP for PCI Express with Avalon-MM DMA IP is used in this demonstration. For detail about this IP, please refer to Intel document ug_ptile_pcie_avmm.

7.1 PCI Express System Infrastructure

Figure 7-1 shows the infrastructure of the PCI Express System in this demonstration. It consists of two primary components: FPGA System and PC System. The FPGA System is developed based on Agilex Hard IP for PCI Express with Avalon-MM DMA. The application software on the PC side is developed by Terasic based on Intel's PCIe kernel mode driver.

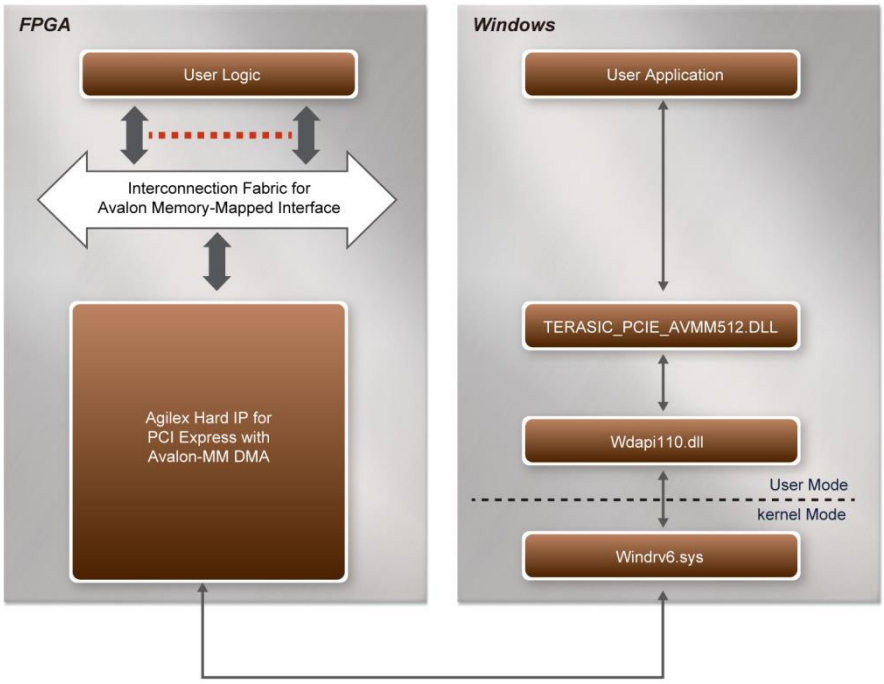


Figure 7-1 Infrastructure of PCI Express System

7.2 PC PCI Express Software SDK

The FPGA System CD contains a PC Windows based SDK to allow users to develop their 64-bit software application on 64-bits Windows 7 or Window XP. The SDK is located in the "CDROM\Demonstrations\PCIe_SW_KIT\Windows" folder which includes:

- PCI Express Driver
- PCI Express Library
- PCI Express Examples

The kernel mode driver assumes the PCIe vendor ID (VID) is 0x1172 and the device ID (DID) is 0x09C4. If different VID and DID are used in the design, users need to modify the PCIe vendor ID (VID) and device ID (DID) in the driver INF file accordingly.

The PCI Express Library is implemented as a single DLL named TERASIC_PCIE_AVMM512.DLL. This file is a 64-bit DLL. When the DLL is exported to the software API, users can easily communicate with the FPGA. The library provides the following functions:

- Basic data read and write
- Data read and write by DMA

For high performance data transmission, AVMM DMA is required as the read and write operations, which are specified under the hardware design on the FPGA.

7.3 PCI Express Software Stack

Figure 7-2 shows the software stack for the PCI Express application software on 64-bit Windows. The PCIe library module TERASIC_PCIE_AVMM512.dll provides DMA and direct I/O access allowing user application program to communicate with FPGA. Users can develop their applications based on this DLL. The altera_pcie_win_driver.sys kernel driver is provided by Altera.

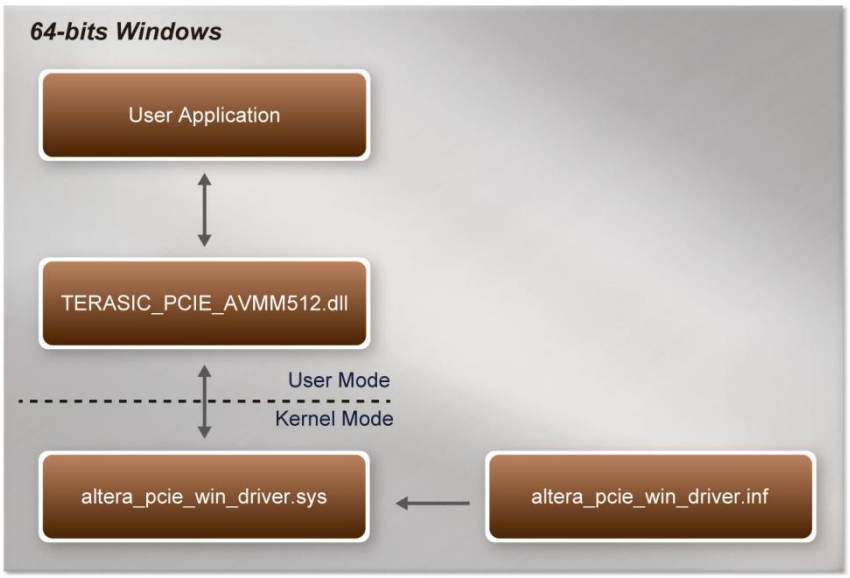


Figure 7-2 PCI Express Software Stack

■ Install PCI Express Driver on Windows

The PCIe driver is locate in the folder:

"CDROM\Demonstrations\PCIe_SW_KIT\Windows\PCIe_Driver"

The folder includes the following four files:

- Altera_pcie_win_driver.cat
- Altera_pcie_win_driver.inf
- Altera_pcie_win_driver.sys
- WdfCoinstaller01011.dll

To install the PCI Express driver, please execute the steps below:

1. Install the DE10-Agilex on the PCIe slot of the host PC
2. Make sure the Intel Programmer and USB-Blaster II driver are installed
3. Execute test.bat in "CDROM\Demonstrations\PCIe_Fundamental\demo_batch" to configure the FPGA
4. Restart windows operation system
5. Click the Control Panel menu from Windows Start menu. Click the Hardware and Sound item before clicking the Device Manager to launch the Device Manager dialog. There will be a PCI Device item in the dialog, as shown in **Figure 7-3**. Move the mouse cursor to the PCI Device item and right click it to select the Updated Driver Software... items.

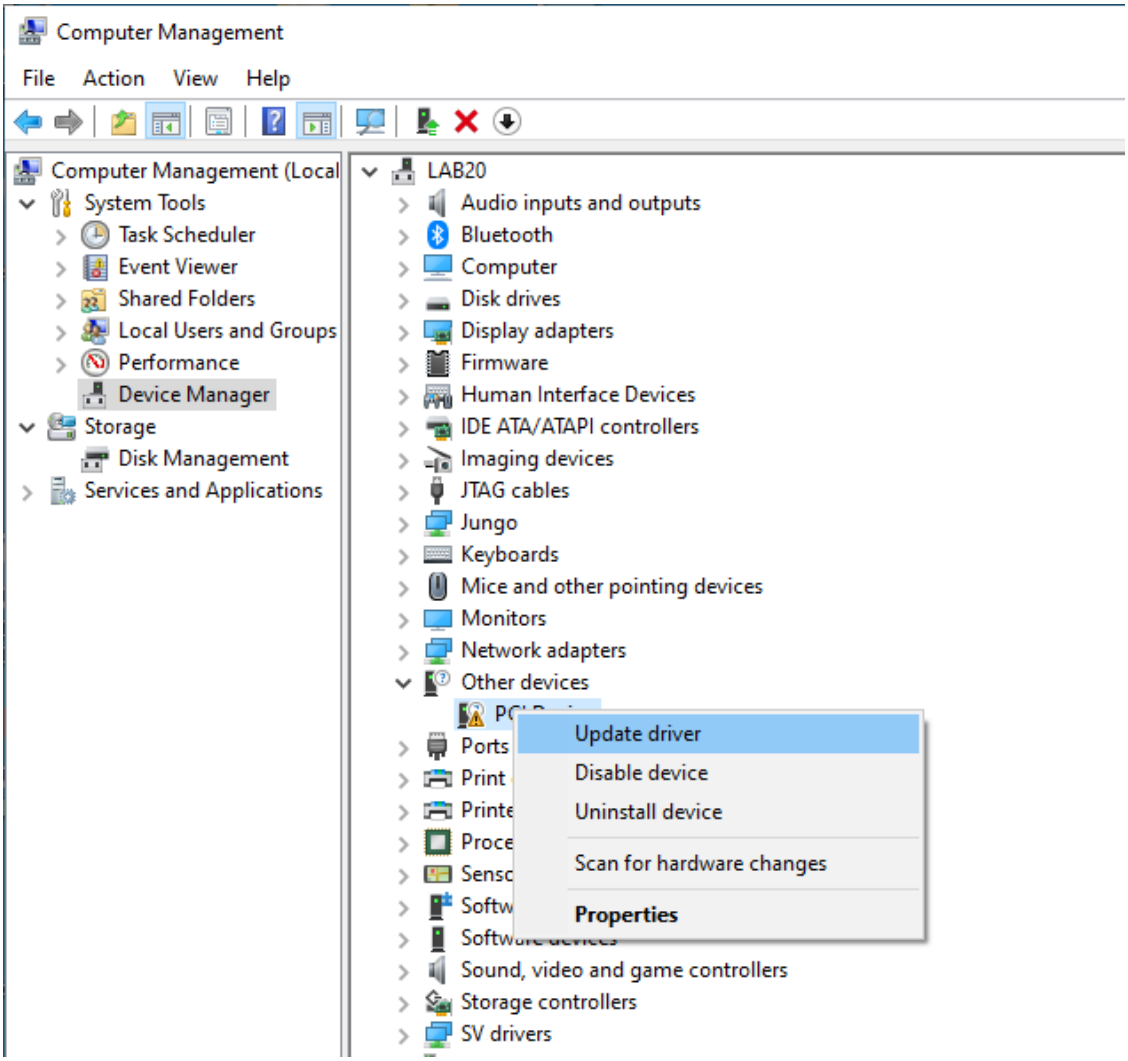


Figure 7-3 Screenshot of launching Update Driver Software… dialog

6. In the **How do you want to search for the driver software** dialog, click **Browse my computer for driver software** item, as shown in **Figure 7-4**

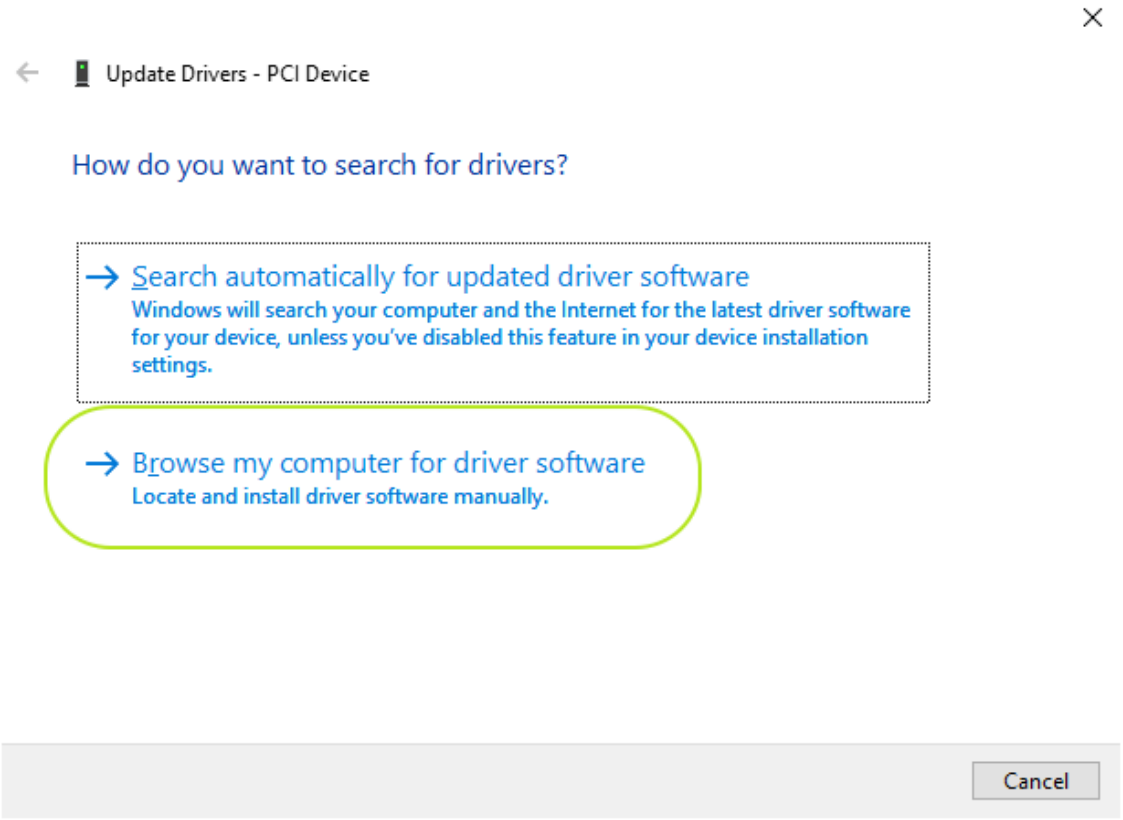


Figure 7-4 Dialog of Browse my computer for the driver software

7. In the **Browse for driver software on your computer** dialog, click the **Browse** button to specify the folder where altera_pcie_din_driver.inf is located, as shown in **Figure 7-5**. Click the **Next** button.

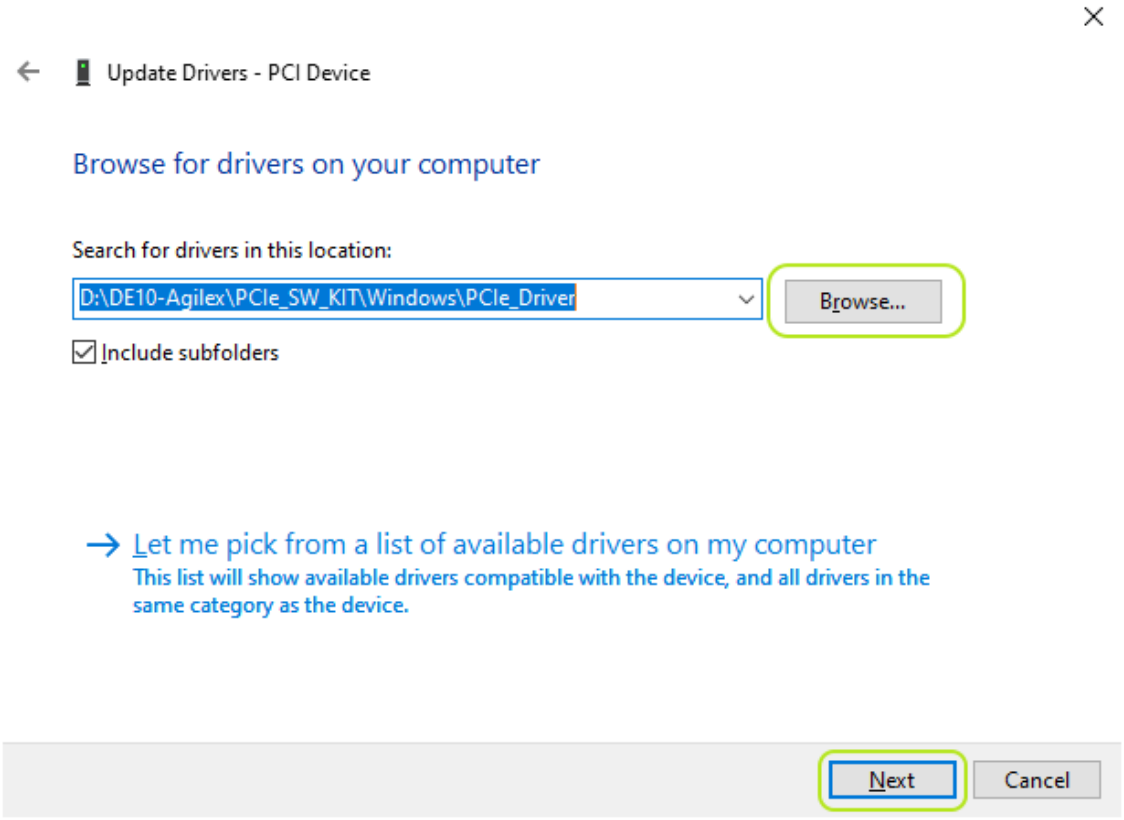


Figure 7-5 Browse for the driver software on your computer

8. When the **Windows Security** dialog appears, as shown **Figure 7-6**, click the **Install** button.

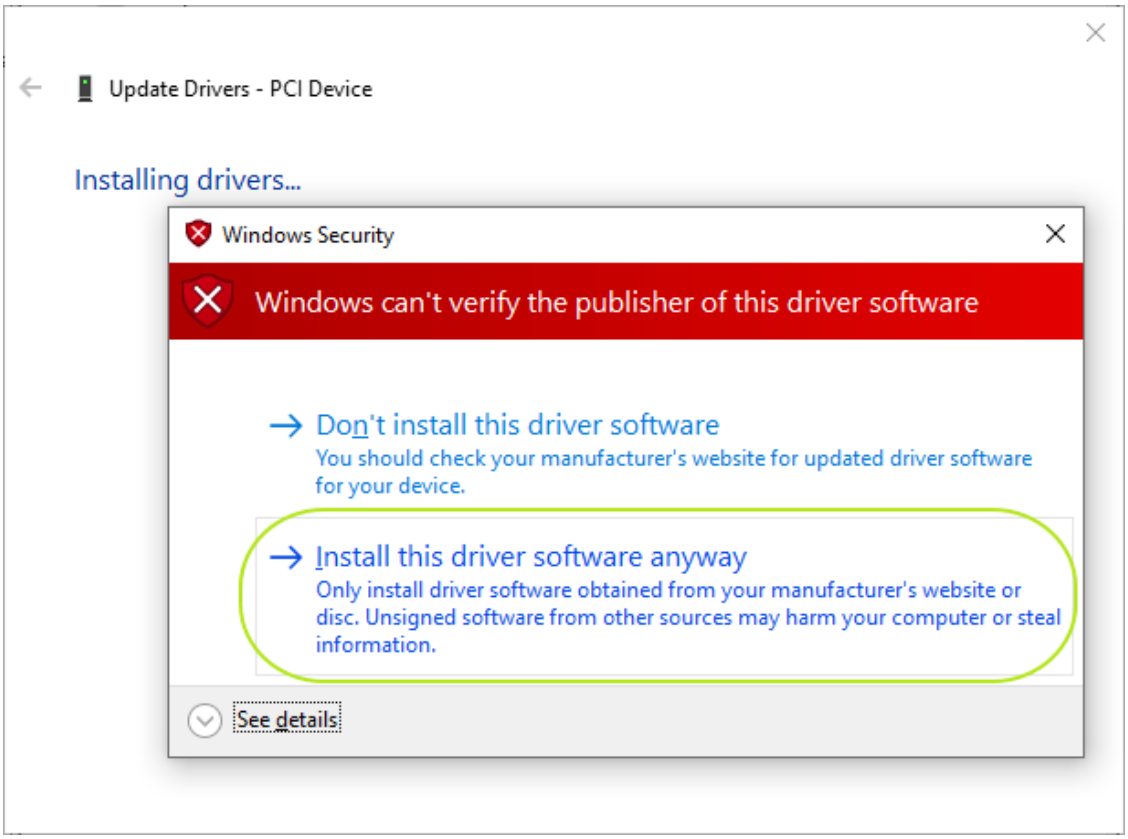


Figure 7-6 Click Install in the dialog of Windows Security

9. When the driver is installed successfully, the successfully dialog will appear, as shown in **Figure 7-7**. Click the **Close** button.

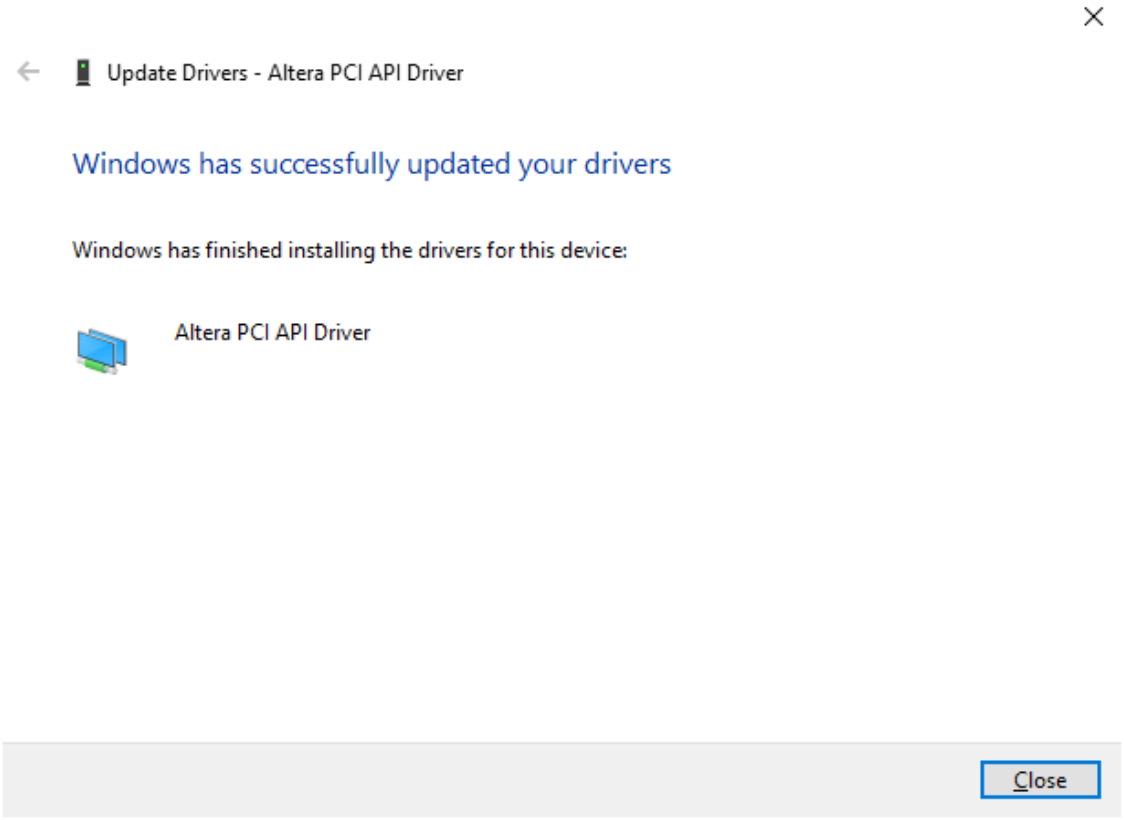


Figure 7-7 Click Close when the installation of the Altera PCI API Driver is complete

10. Once the driver is successfully installed, users can see the **Altera PCI API Driver** under the device manager window, as shown in **Figure 7-8**.

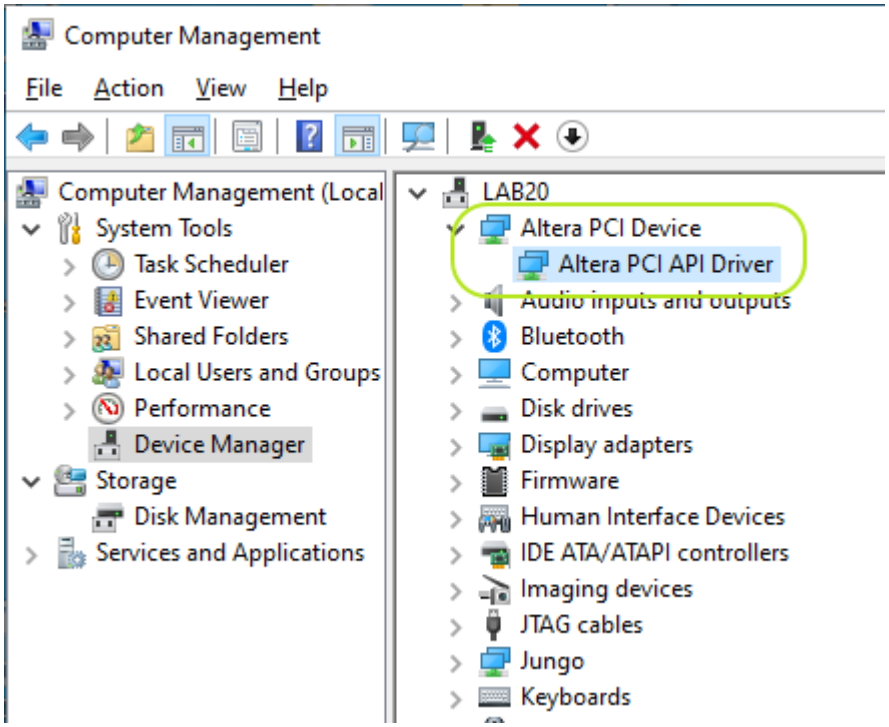


Figure 7-8 Altera PCI API Driver in Device Manager

■ Create a Software Application

All the files needed to create a PCIe software application are located in the directory CDROM\demonstration\PCIe_SW_KIT\Windows\PCIe_Library. It includes the following files:

- TERASIC_PCIE_AVMM512.h
- TERASIC_PCIE_AVMM512.DLL (64-bit DLL)

Below lists the procedures to use the SDK files in users' C/C++ project:

1. Create a 64-bit C/C++ project.
2. Include TERASIC_PCIE_AVMM512.h in the C/C++ project.
3. Copy TERASIC_PCIE_AVMM512.DLL to the folder where the project.exe is located.
4. Dynamically load TERASIC_PCIE_AVMM512.DLL in C/C++ program. To load the DLL, please refer to the PCIe fundamental example below.
5. Call the SDK API to implement the desired application.

Users can easily communicate with the FPGA through the PCIe bus through the

TERASIC_PCIE_AVMM512.DLL API. The details of API are described below:

7.4 PCI Express Library API

Below shows the exported API in the TERASIC_PCIE_AVMM512.DLL. The API prototype is defined in the TERASIC_PCIE_AVMM512.h.

Note: the Linux library terasic_pcie512_qsys.so also use the same API and header file.

■ PCIE_Open

Function:

Open a specified PCIe card with vendor ID, device ID, and matched card index.

Prototype:

```
PCIE_HANDLE PCIE_Open(
    uint8_t wVendorID,
    uint8_t wDeviceID,
    uint8_t wCardIndex);
```

Parameters:

wVendorID:

Specify the desired vendor ID. A zero value means to ignore the vendor ID.

wDeviceID:

Specify the desired device ID. A zero value means to ignore the device ID.

wCardIndex:

Specify the matched card index, a zero based index, based on the matched vendor ID and device ID.

Return Value:

Return a handle to presents specified PCIe card. A positive value is return if the PCIe card is opened successfully. A value zero means failed to connect the target PCIe card.

This handle value is used as a parameter for other functions, e.g. PCIE_Read32.

Users need to call PCIE_Close to release handle once the handle is no longer used.

■ PCIE_Close

Function:

Close a handle associated to the PCIe card.

Prototype:

void PCIE_Close(PCIE_HANDLE hPCIE);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function.
Return Value: None.

■ PCIE_Read32

Function: Read a 32-bit data from the FPGA board.
Prototype: bool PCIE_Read32(PCIE_HANDLE hPCIE, PCIE_BAR PcieBar, PCIE_ADDRESS PcieAddress, uint32_t *pdwData);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function. PcieBar: Specify the target BAR. PcieAddress: Specify the target address in FPGA. pdwData: A buffer to retrieve the 32-bit data.
Return Value: Return **true** if read data is successful; otherwise **false** is returned.

■ PCIE_Write32

Function: Write a 32-bit data to the FPGA Board.
Prototype: bool PCIE_Write32(PCIE_HANDLE hPCIE,

PCIE_BAR PcieBar, PCIE_ADDRESS PcieAddress, uint32_t dwData);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function. PcieBar: Specify the target BAR. PcieAddress: Specify the target address in FPGA. dwData: Specify a 32-bit data which will be written to FPGA board.
Return Value: Return **true** if write data is successful; otherwise **false** is returned.

■ PCIE_Read8

Function: Read an 8-bit data from the FPGA board.
Prototype: bool PCIE_Read8(PCIE_HANDLE hPCIE, PCIE_BAR PcieBar, PCIE_ADDRESS PcieAddress, uint8_t *pByte);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function. PcieBar: Specify the target BAR. PcieAddress: Specify the target address in FPGA. pByte: A buffer to retrieve the 8-bit data.
Return Value: Return **true** if read data is successful; otherwise **false** is returned.

■ PCIE_Write8

Function:

Write an 8-bit data to the FPGA Board.

Prototype:

```
bool PCIE_Write8(
    PCIE_HANDLE hPCIE,
    PCIE_BAR PcieBar,
    PCIE_ADDRESS PcieAddress,
    uint8_t Byte);
```

Parameters:

hPCIE:

A PCIe handle return by PCIE_Open function.

PcieBar:

Specify the target BAR.

PcieAddress:

Specify the target address in FPGA.

Byte:

Specify an 8-bit data which will be written to FPGA board.

Return Value:

Return **true** if write data is successful; otherwise **false** is returned.

■ PCIE_DmaRead

Function:

Read data from the memory-mapped memory of FPGA board in DMA.

Prototype:

```
bool PCIE_DmaRead(
    PCIE_HANDLE hPCIE,
    PCIE_LOCAL_ADDRESS LocalAddress,
    void *pBuffer,
    uint64_t dwBufSize64
    );
```

Parameters:

hPCIE:

A PCIe handle return by PCIE_Open function.

LocalAddress: Specify the target memory-mapped address in FPGA. pBuffer: A pointer to a memory buffer to retrieved the data from FPGA. The size of buffer should be equal or larger the dwBufSize. dwBufSize64: Specify the byte number of data retrieved from FPGA.
Return Value: Return **true** if read data is successful; otherwise **false** is returned.

■ PCIE_DmaWrite

Function: Write data to the memory-mapped memory of FPGA board in DMA.
Prototype: bool PCIE_DmaWrite(PCIE_HANDLE hPCIE, PCIE_LOCAL_ADDRESS LocalAddress, void *pData, uint64_t dwDataSize64);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function. LocalAddress: Specify the target memory mapped address in FPGA. pData: A pointer to a memory buffer to store the data which will be written to FPGA. dwDataSize64: Specify the byte number of data which will be written to FPGA.
Return Value: Return **true** if write data is successful; otherwise **false** is returned.

■ PCIE_ConfigRead32

Function: Read PCIe Configuration Table. Read a 32-bit data by given a byte offset.

Prototype: bool PCIE_ConfigRead32 (PCIE_HANDLE hPCIE, uint32_t Offset, uint32_t *pdwData);
Parameters: hPCIE: A PCIe handle return by PCIE_Open function. Offset: Specify the target byte of offset in PCIe configuration table. pdwData: A 4-bytes buffer to retrieve the 32-bit data.
Return Value: Return **true** if read data is successful; otherwise **false** is returned.

7.5 PCIe Reference Design - Fundamental

The application reference design shows how to implement fundamental control and data transfer in DMA. In the design, basic I/O is used to control the BUTTON and LED on the FPGA board. High-speed data transfer is performed by the DMA.

■ Demonstration Files Location

The demo file is located in the batch folder:

CDROM\Demonstrations\PCIe_Fundamental\demo_batch

The folder includes following files:

- FPGA Configuration File: DE10_Agilex.sof
- Download Batch file: test.bat
- Windows Application Software folder: windows_app, includes
  - ✧ PCIE_FUNDAMENTAL.exe
  - ✧ TERASIC_PCIE_AVMM512.dll

■ Demonstration Setup

1. Install the FPGA board on your PC as shown in **Figure 7-9**.

Figure 7-9 FPGA board installation on PC

2. Configure FPGA with DE10_Agilex.sof by executing the test.bat.
3. Install the PCIe driver if necessary. The driver is located in the folder:
   CDROM\Demonstration\PCIe_SW_KIT\Windows\PCIe_Driver.
4. Restart Windows
5. Make sure that Windows has detected the FPGA Board by checking the Windows Device Manager as shown in **Figure 7-10**.

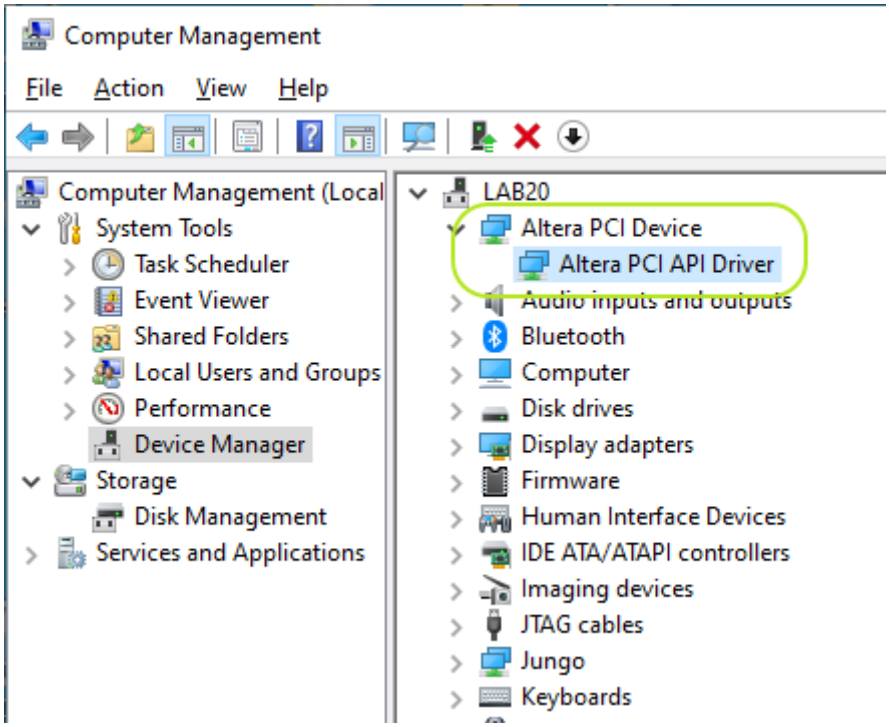


Figure 7-10 Screenshot for PCIe Driver

6. Go to windows_app folder, execute PCIE_FUNDMENTAL.exe. A menu will appear as shown in **Figure 7-11**.

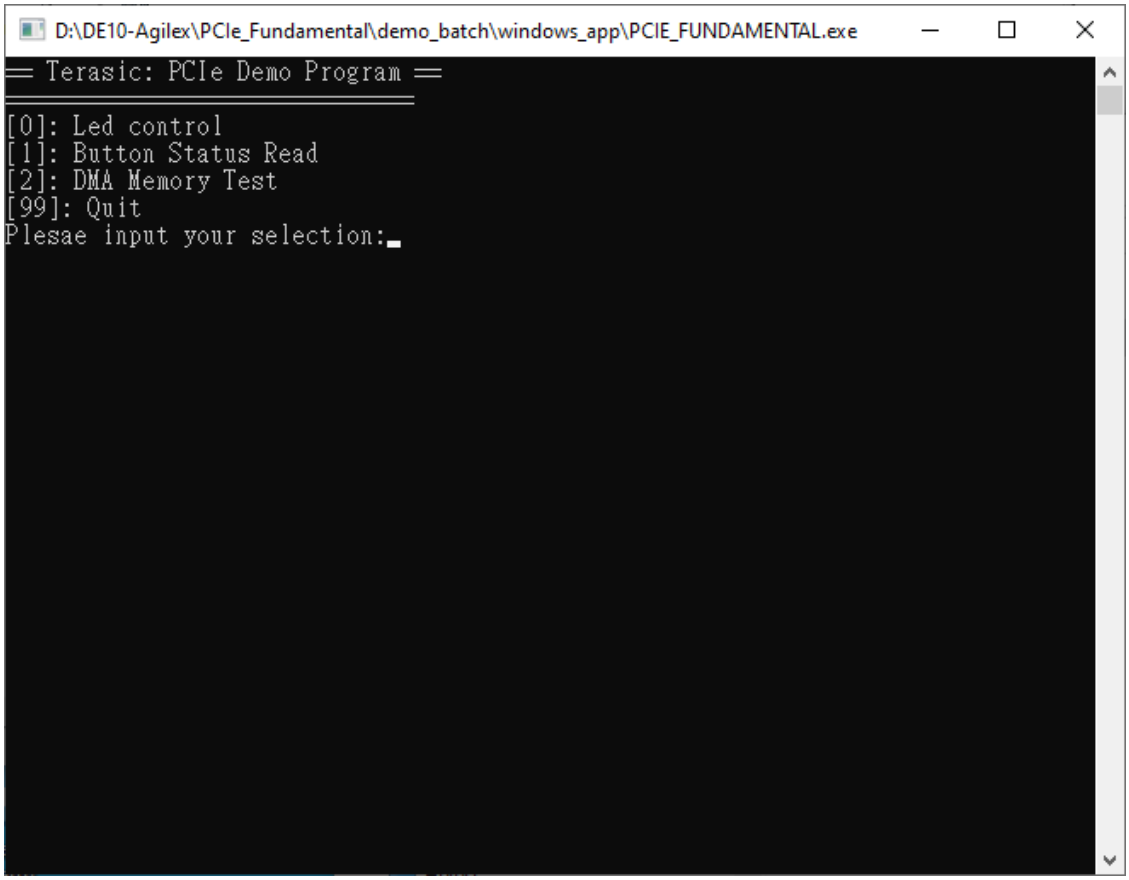


Figure 7-11 Screenshot of Program Menu

7. Type 0 followed by a ENTER key to select Led Control item, then input 15 (hex 0x0f) will make all LEDs on as shown in **Figure 7-12**. If input 0 (hex 0x00), all LEDs will be turned off.

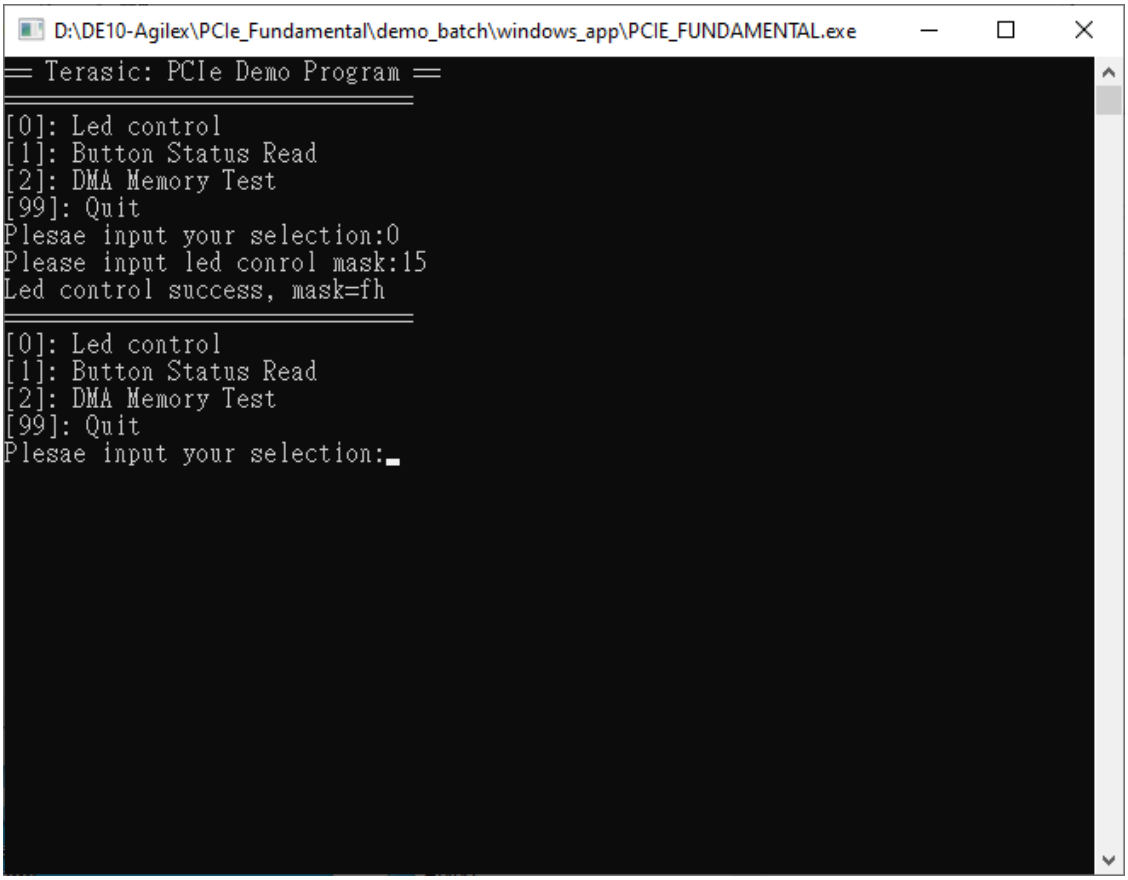


Figure 7-12 Screenshot of LED Control

8. Type 1 followed by an ENTER key to select Button Status Read item. The button status will be reported as shown in **Figure 7-13**.

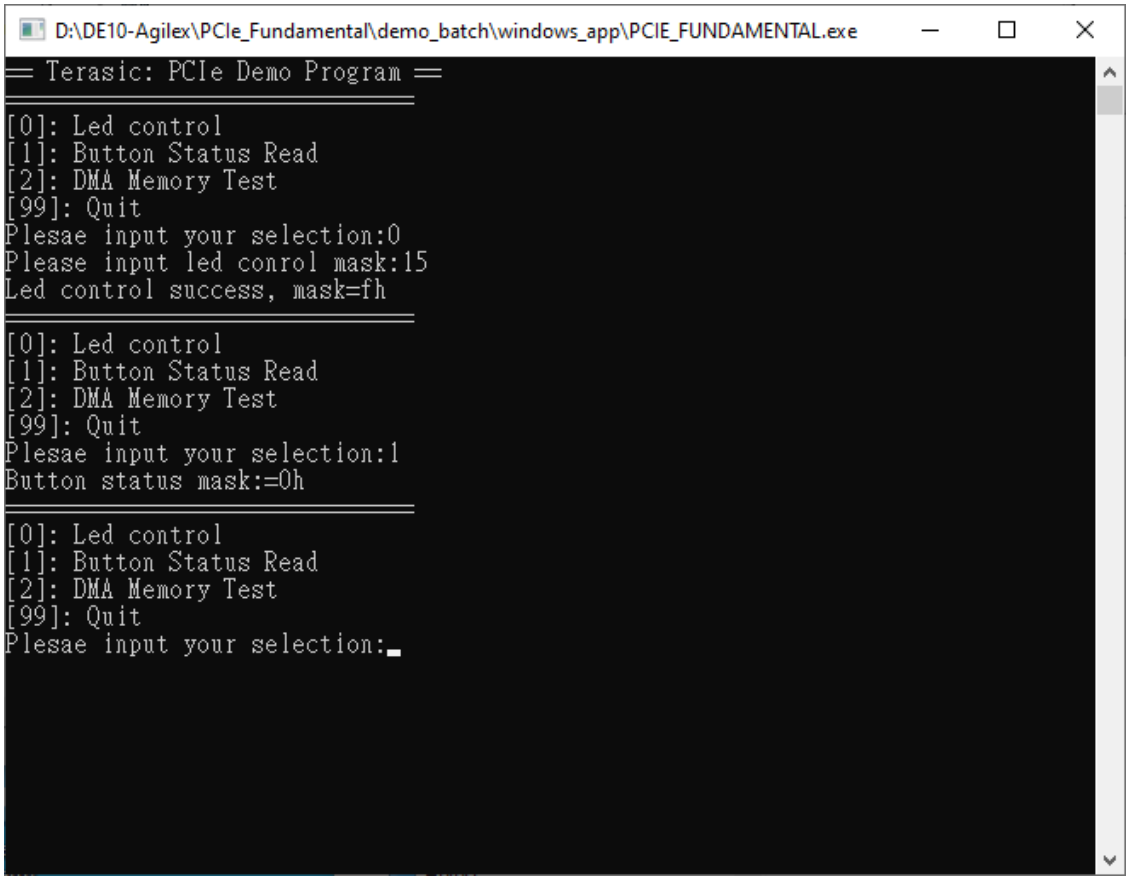


Figure 7-13 Screenshot of Button Status Report

9. Type 2 followed by an ENTER key to select the DMA Testing item. The DMA test result will be reported as shown in **Figure 7-14**.

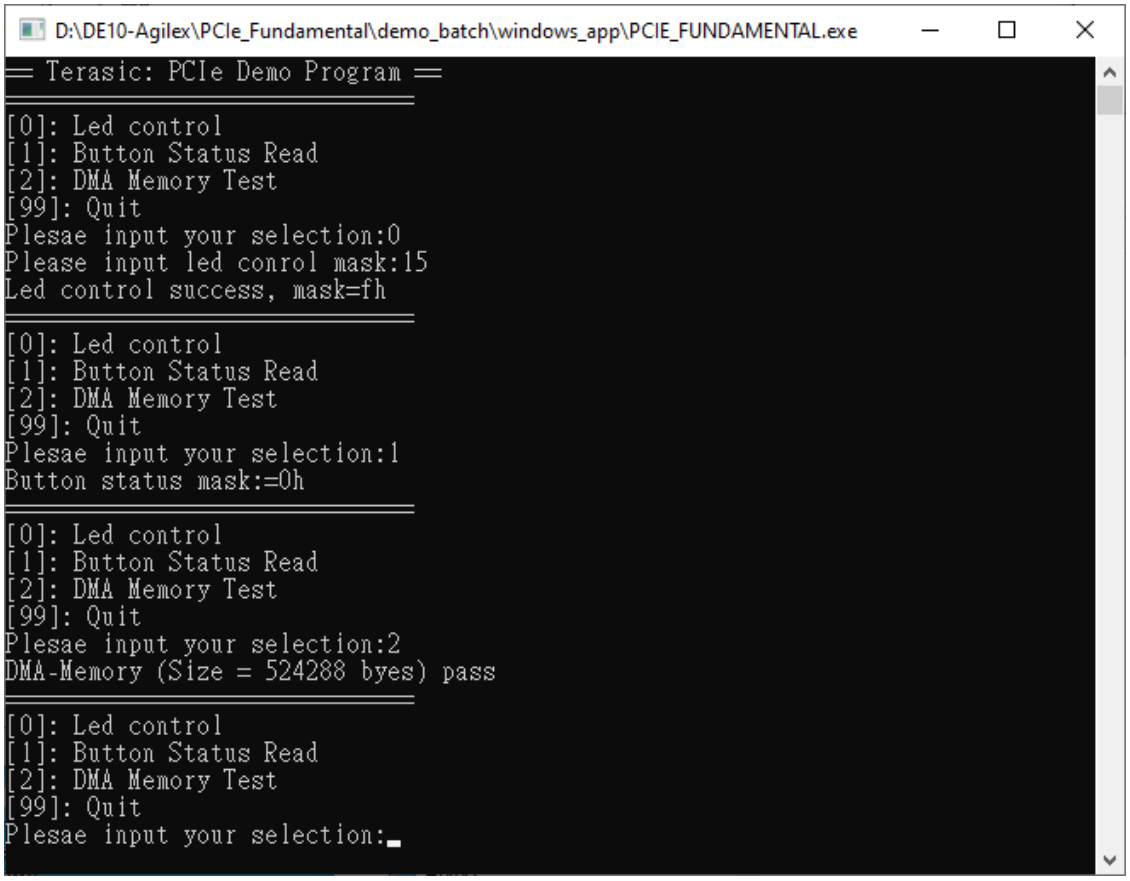


Figure 7-14 Screenshot of DMA Memory Test Result

10. Type 99 followed by an ENTER key to exit this test program

■ Development Tools

- Quartus Prime 21.2 Pro Edition
- Visual C++ 2019

■ Demonstration Source Code Location

- Quartus Project: Demonstrations\PCIe_Fundamental
- C++ Project: Demonstrations\PCIe_SW_KIT\Windows\PCIE_FUNDAMENTAL

■ FPGA Application Design

Figure 7-15 shows the system block diagram in the FPGA system. In the **Platform Designer** (formerly Qsys), the PIO controller is used to control the LED and monitor the Button Status, and the On-Chip memory is used for performing DMA testing. The PIO controllers and the On-Chip memory are connected to the PCI Express Hard IP

controller through the Memory-Mapped Interface.

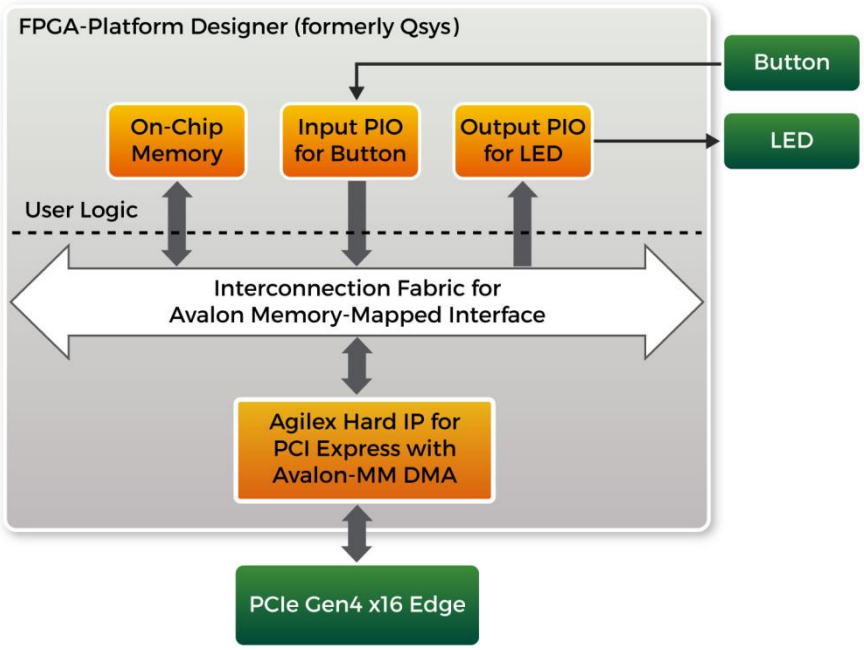


Figure 7-15 Hardware block diagram of the PCIe reference design

■ Windows Based Application Software Design

The application software project is built by Visual C++ 2019. The project includes the following major files:

Name	Description
PCIE_FUNDAMENTAL.cpp	Main program
PCIE.c	Implement dynamically load for TERAISC_PCIE_AVMM.DLL
PCIE.h	
TERASIC_PCIE_AVMM512.h	SDK library file, defines constant and data structure

The main program PCIE_FUNDAMENTAL.cpp includes the header file "PCIE.h" and defines the controller address according to the FPGA design.

```
#define DEMO_PCIE_USER_BAR          PCIE_BAR4
#define DEMO_PCIE_IO_LED_ADDR       0x800000
#define DEMO_PCIE_IO_BUTTON_ADDR    0x800040
#define DEMO_PCIE_MEM_ADDR          0x100000

#define MEM_SIZE                    (512*1024) //512KB
```

The base address of BUTTON and LED controllers are 0x800040 and 0x800000 based on the PCIE_BAR4, respectively. The on-chip memory base address is 0x100000 relative to the DMA controller.

Before accessing the FPGA through PCI Express, the application first calls the PCIE_Load to dynamically load the TERASIC_PCIE_AVMM512.DLL. Then, it calls PCIE_Open to open the PCI Express driver. The constant DEFAULT_PCIE_VID and DEFAULT_PCIE_DID used in the PCIE_Open are defined in TERASIC_PCIE_AVMM512.h. If developers change the Vendor ID and Device ID and PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM512.h. If the return value of PCIE_Open is zero, it means the driver cannot be accessed successfully. In this case, please make sure:

- The FPGA is configured with the associated bit-stream file and the host is rebooted.
- The PCI express driver is loaded successfully.

The LED control is implemented by calling **PCIE_Write32** API, as shown below:

```
bPass = PCIE_Write32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_LED_ADDR, (uint32_t) Mask);
```

The button status query is implemented by calling the **PCIE_Read32** API, as shown below:

```
bPass = PCIE_Read32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_BUTTON_ADDR, &Status);
```

The memory-mapped memory read and write test is implemented by **PCIE_DmaWrite** and **PCIE_DmaRead** API, as shown below:

```
bPass = PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);
bPass = PCIE_DmaRead(hPCIe, LocalAddr, pRead, nTestSize);
```

7.6 PCIe Reference Design - DDR4

The application reference design shows how to add the DDR4 Memory Controllers for the DDR4-A SODIMM, DDR4-B SODIMM, DDR4-C SODIMM and DDR4-D SODIMM into the PCIe Quartus project based on the PCIe_Fundamental Quartus project and perform 8GB data DMA for both SODIMM. Also, this demo shows how to call "PCIE_ConfigRead32" API to check PCIe link status.

■ Demonstration Files Location

The demo file is located in the batch folder:

CDROM\Demonstrations\PCIe_DDR4\demo_batch

The folder includes following files:

- FPGA Configuration File: DE10_Agilex.sof
- Download Batch file: test.bat
- Windows Application Software folder: windows_app, includes
  - ✧ PCIE_DDR4.exe
  - ✧ TERASIC_PCIE_AVMM512.dll

■ Demonstration Setup

1. Install four pieces of DDR4 2666 8GB SODIMM on the FPGA board.
2. Install the FPGA board on your PC.
3. Configure the FPGA with the DE10_Agilex sof by executing the test.bat.
4. Install the PCIe driver if necessary.
5. Restart Windows
6. Make sure that Windows has detected the FPGA Board by checking the Windows Control panel.
7. Go to windows_app folder, execute PCIE_DDR4.exe. A menu will appear as shown in **Figure 7-16**.

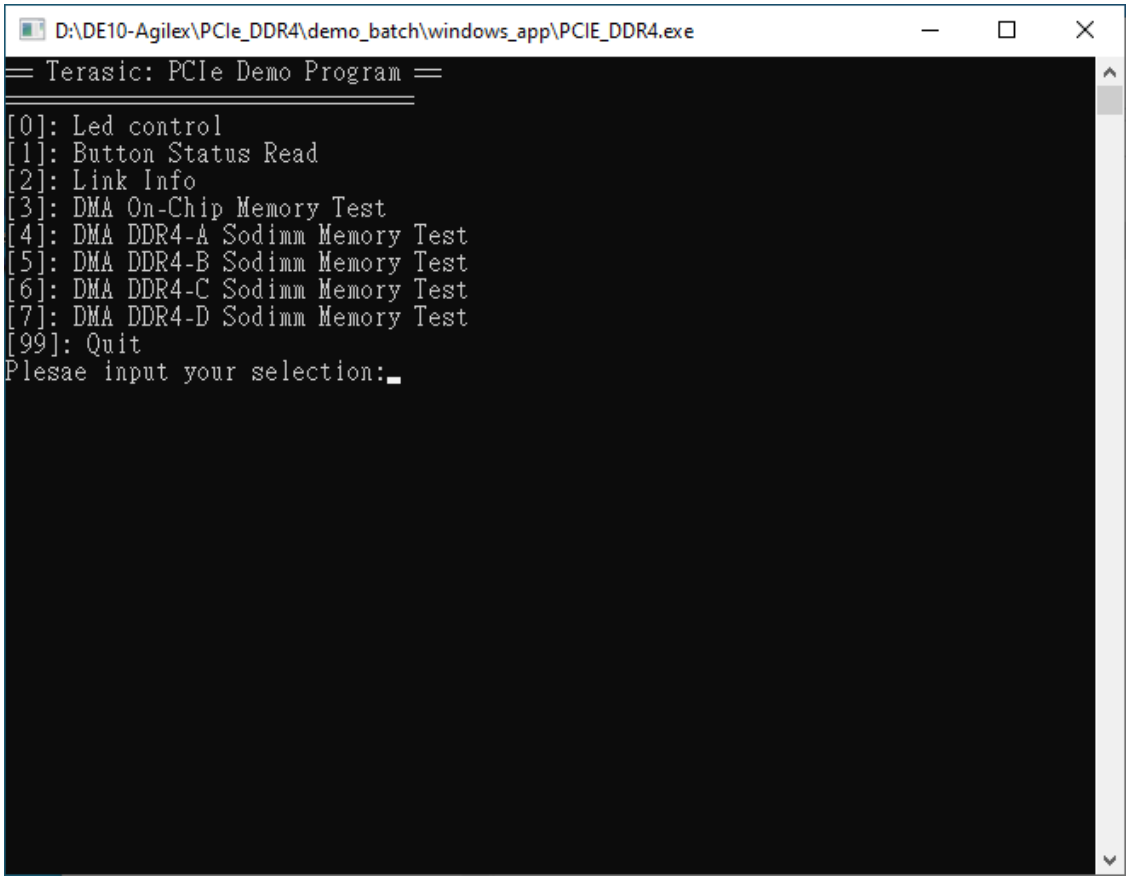


Figure 7-16 Screenshot of Program Menu

8. Type 2 followed by the ENTER key to select the Link Info item. The PCIe link information will be shown as in **Figure 7-17**. Gen3 link speed and x16 link width are expected.

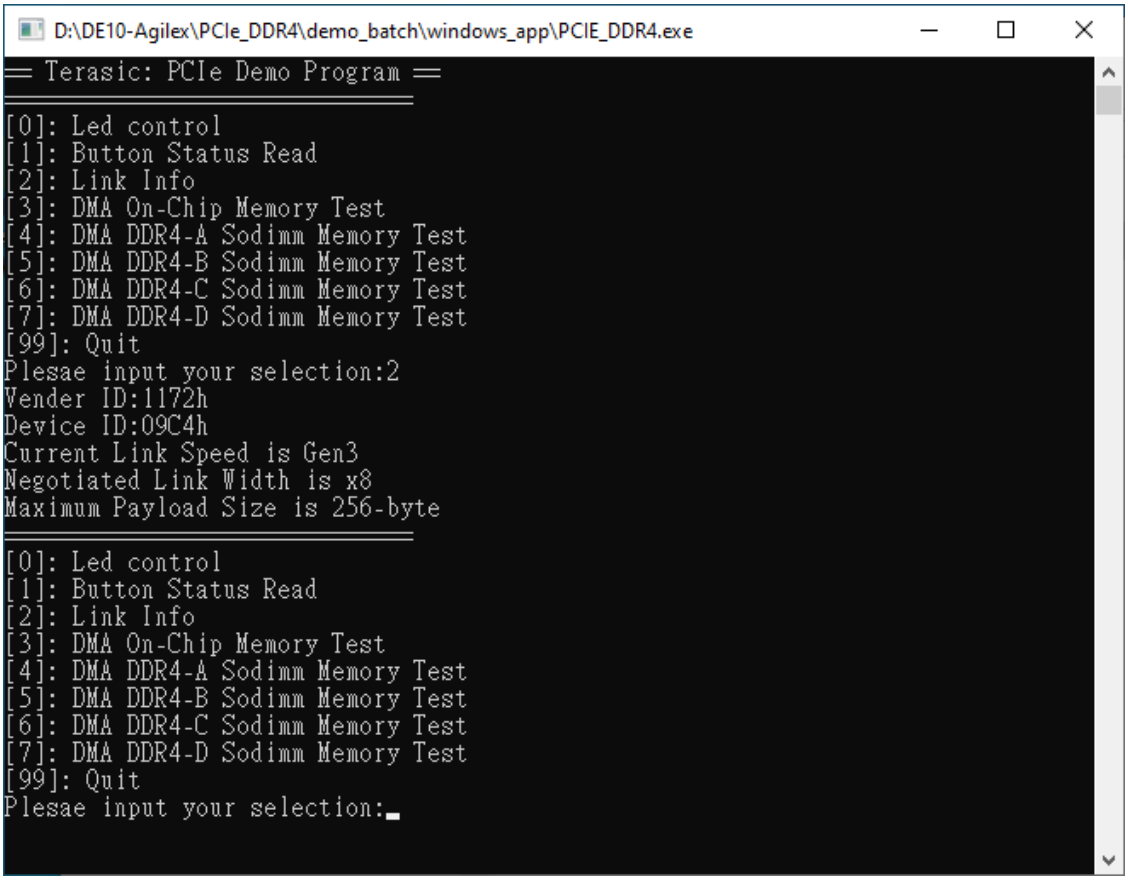


Figure 7-17 Screenshot of Link Info

9. Type 3 followed by the ENTER key to select DMA On-Chip Memory Test item. The DMA write and read test result will be reported as shown in **Figure 7-18**.

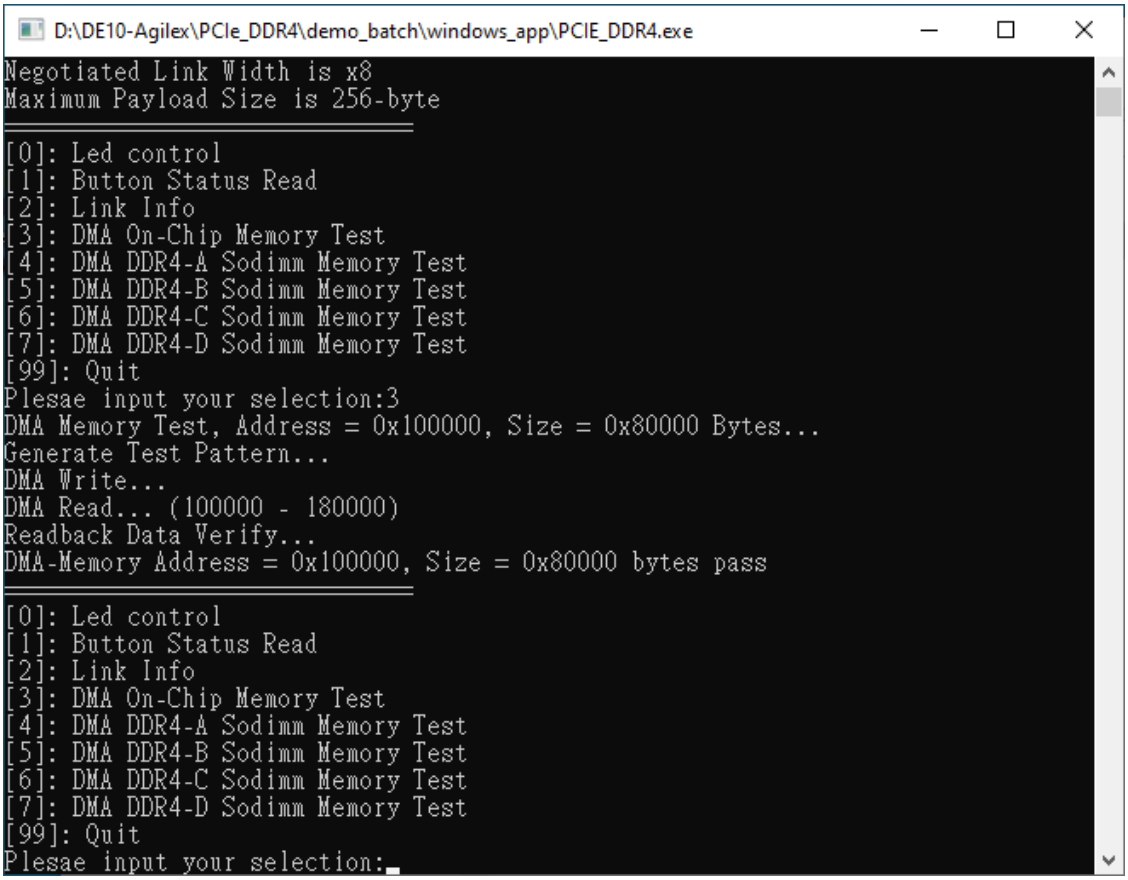


Figure 7-18 Screenshot of On-Chip Memory DMA Test Result

10. Type 4 followed by the ENTER key to select the DMA DDR4-A SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 7-19**.

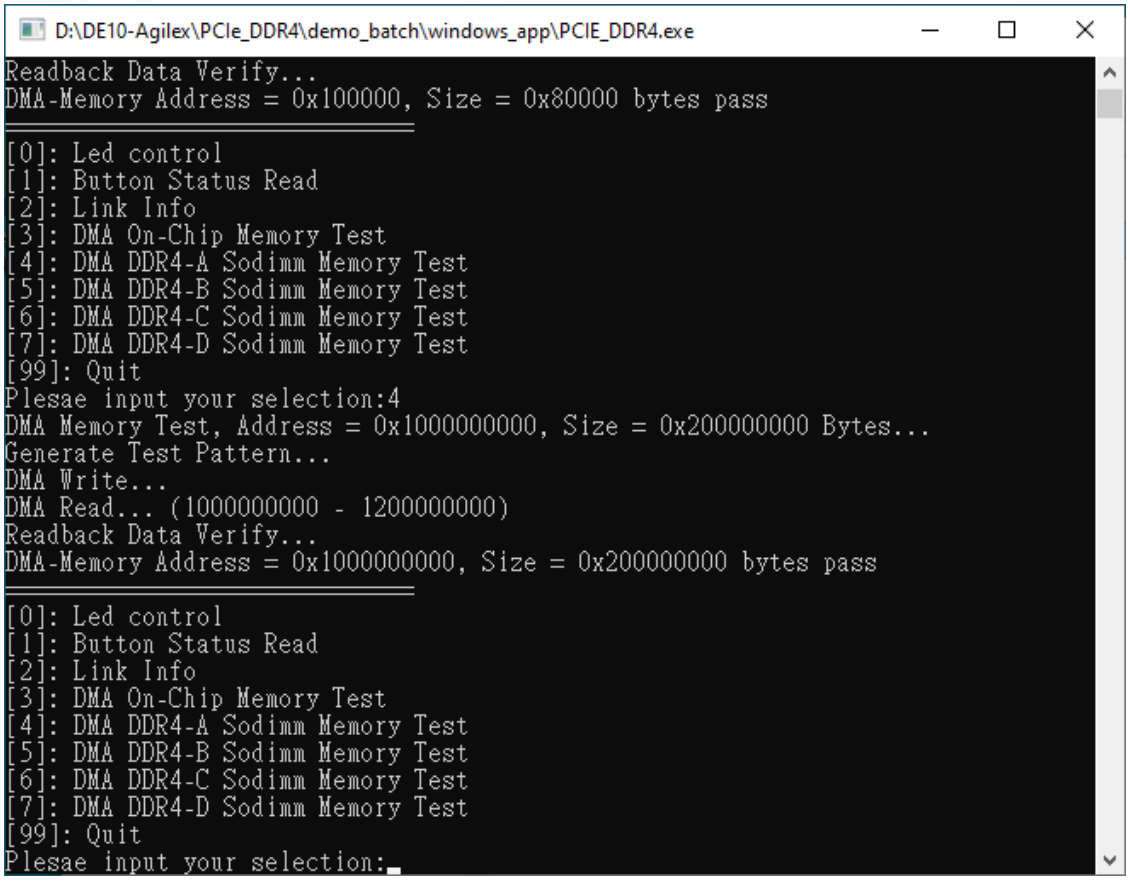


Figure 7-19 Screenshot of the DDR4-A SOSIMM Memory DMA Test Result

11. Type 5 followed by the ENTER key to select the DMA DDR4-B SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 7-20**.

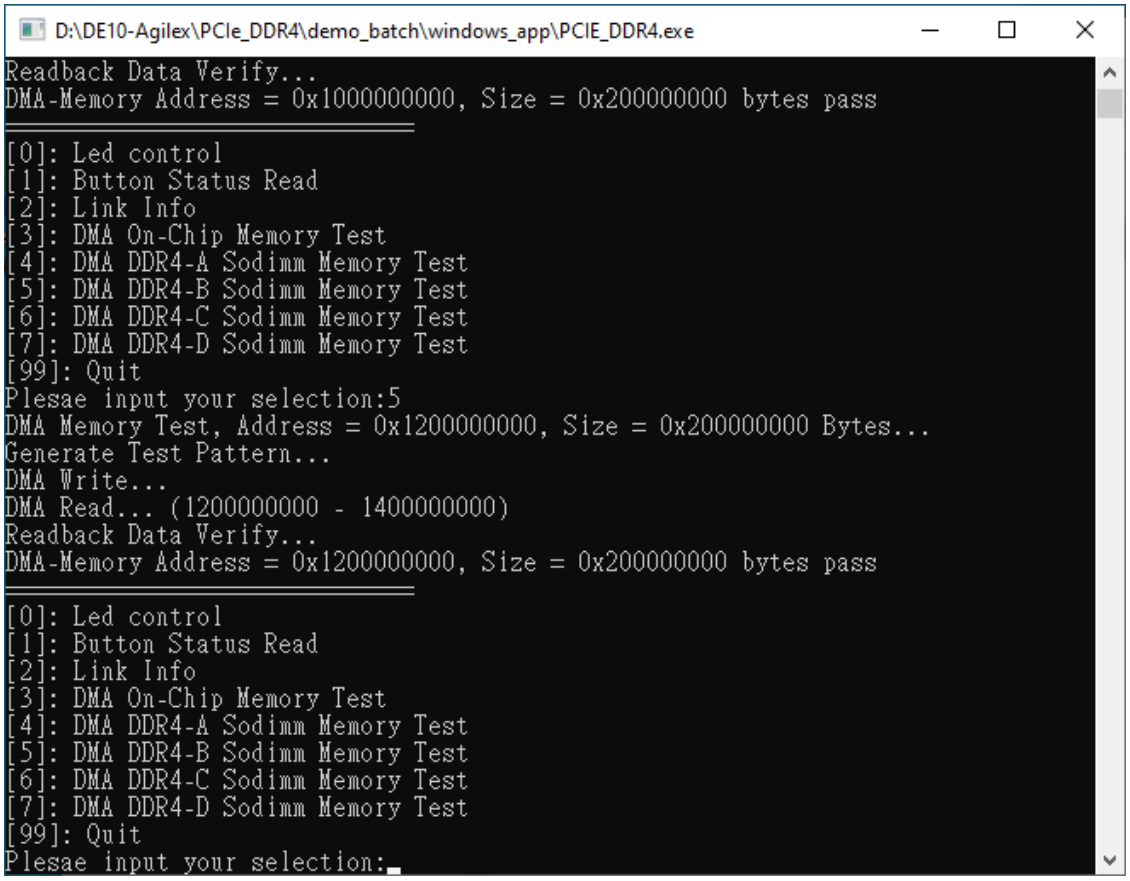


Figure 7-20 Screenshot of the DDR4-B SOSIMM Memory DMA Test Result

12. Type 6 followed by an ENTER key to select DMA DDR4-C SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 7-21**.

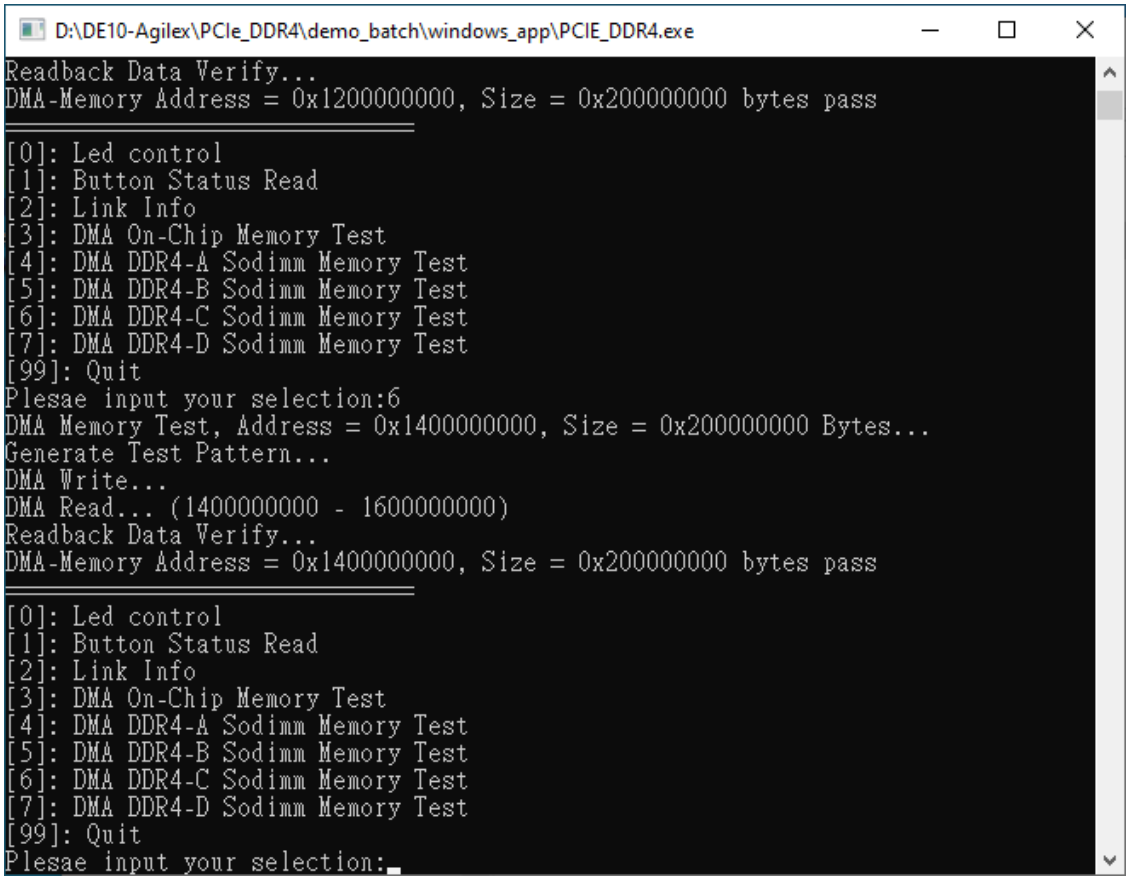


Figure 7-21 Screenshot of the DDR4-C SOSIMM Memory DMA Test Result

13. Type 7 followed by the ENTER key to select the DMA DDR4-D SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 7-22**.

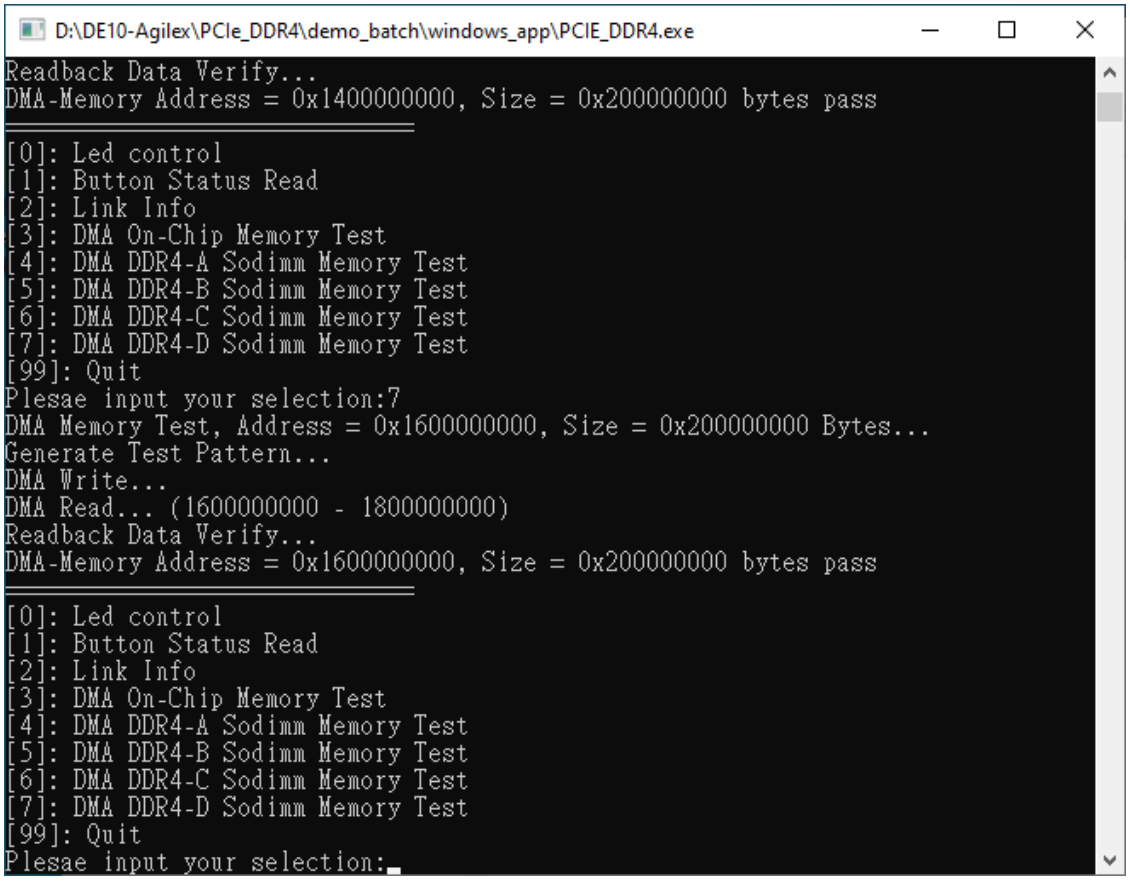


Figure 7-22 Screenshot of DDR4-D SOSIMM Memory DMA Test Result

14. Type 99 followed by the ENTER key to exit this test program.

■ Development Tools

- Quartus Prime 21.2 Pro Edition
- Visual C++ 2019

■ Demonstration Source Code Location

- Quartus Project: Demonstrations\PCIE_DDR4
- Visual C++ Project: Demonstrations\PCIe_SW_KIT\Windows\PCIe_DDR4

■ FPGA Application Design

Figure 7-23 shows the system block diagram in the FPGA system. In the **Platform Designer** (formerly Qsys), the PIO controller is used to control the LED and monitor the Button Status, and the On-Chip memory is used for performing DMA testing. The PIO controllers and the On-Chip memory are connected to the PCI Express Hard IP

controller through the Memory-Mapped Interface.

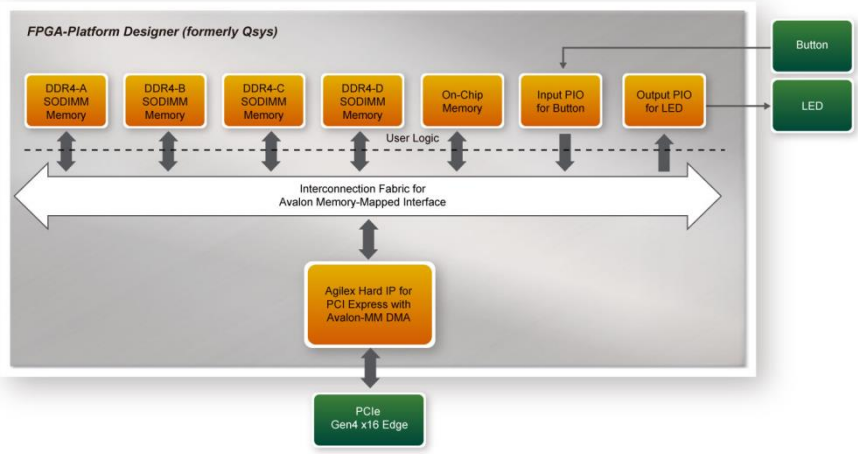


Figure 7-23 Hardware block diagram of the PCIe_DDR4 reference design

■ Windows Based Application Software Design

The application software project is built by Visual C++ 2019. The project includes the following major files:

Name	Description
PCIE_DDR4.cpp	Main program
PCIE.c	Implement dynamically load for TERAISC_PCIE_AVMM.DLL
PCIE.h	
TERASIC_PCIE_AVMM512.h	SDK library file, defines constant and data structure

The main program PCIE_DDR4.cpp includes the header file "PCIE.h" and defines the controller address according to the FPGA design.

```
#define DEMO_PCIE_USER_BAR          PCIE_BAR4
#define DEMO_PCIE_IO_LED_ADDR       0x800000
#define DEMO_PCIE_IO_BUTTON_ADDR    0x800040
#define DEMO_PCIE_ONCHIP_MEM_ADDR   0x100000
#define DEMO_PCIE_DDR4A_MEM_ADDR    0x1000000000
#define DEMO_PCIE_DDR4B_MEM_ADDR    0x1200000000
#define DEMO_PCIE_DDR4C_MEM_ADDR    0x1400000000
#define DEMO_PCIE_DDR4D_MEM_ADDR    0x1600000000

#define ONCHIP_MEM_TEST_SIZE        (512*1024) //512KB
#define DDR4A_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4B_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4C_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4D_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
```

The base address of BUTTON and LED controllers are 0x800040 and 0x800000 based on PCIE_BAR4, respectively. The on-chip memory base address is 0x100000 relative to the DMA controller. The above definitions are the same as those in the PCIe Fundamental demo.

Before accessing the FPGA through PCI Express, the application first calls PCIE_Load to dynamically load the TERASIC_PCIE_AVMM512.DLL. Then, it calls PCIE_Open to open the PCI Express driver. The constant DEFAULT_PCIE_VID and DEFAULT_PCIE_DID used in the PCIE_Open are defined in TERASIC_PCIE_AVMM512.h. If developers change the Vendor ID and Device ID and PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM512.h. If the return value of PCIE_Open is zero, it means the driver cannot be accessed successfully. In this case, please make sure:

- The FPGA is configured with the associated bit-stream file and the host is rebooted.
- The PCI express driver is loaded successfully.

The LED control is implemented by calling PCIE_Write32 API, as shown below:

```
bPass = PCIE_Write32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_LED_ADDR, (uint32_t) Mask);
```

The button status query is implemented by calling the **PCIE_Read32** API, as shown below:

```
bPass = PCIE_Read32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_BUTTON_ADDR, &Status);
```

The memory-mapped memory read and write test is implemented by **PCIE_DmaWrite** and **PCIE_DmaRead** API, as shown below:

```
bPass = PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);
bPass = PCIE_DmaRead(hPCIe, LocalAddr, pRead, nTestSize);
```

The PCIe link information is implemented by PCIE_ConfigRead32 API, as shown below:

```
// read config - link status
if (PCIE_ConfigRead32(hPCIe, 0x80, &Data32)) {
    switch ((Data32 >> 16) & 0x0F) {
    case 1:
        printf("Current Link Speed is Gen1\n");
        break;
    case 2:
        printf("Current Link Speed is Gen2\n");
        break;
    case 3:
        printf("Current Link Speed is Gen3\n");
        break;
    case 4:
        printf("Current Link Speed is Gen4\n");
        break;
    default:
        printf("Current Link Speed is Unknown\n");
        break;
    }
    switch ((Data32 >> 20) & 0x3F) {
    case 1:
        printf("Negotiated Link Width is x1\n");
        break;
    case 2:
        printf("Negotiated Link Width is x2\n");
        break;
    case 4:
        printf("Negotiated Link Width is x4\n");
        break;
    case 8:
        printf("Negotiated Link Width is x8\n");
        break;
    case 16:
        printf("Negotiated Link Width is x16\n");
        break;
    default:
        printf("Negotiated Link Width is Unknown\n");
        break;
    }
} else {
    bPass = false;
}
```

Chapter 8

PCI Express Reference Design for Linux

PCI Express is commonly used in consumer, server, and industrial applications, to link motherboard-mounted peripherals. From this demonstration, it will show how the PC Linux and FPGA communicate with each other through the PCI Express interface. Agilex Hard IP for PCI Express with Avalon-MM DMA IP is used in this demonstration. For detail about this IP, please refer to Intel document ug_ptile_pcie_avmm.

8.1 PCI Express System Infrastructure

Figure 8-1 shows the infrastructure of the PCI Express System in this demonstration. It consists of two primary components: FPGA System and PC System. The FPGA System is developed based on Agilex Hard IP for PCI Express with Avalon-MM DMA. The application software on the PC side is developed by Terasic based on Intel's PCIe kernel mode driver.

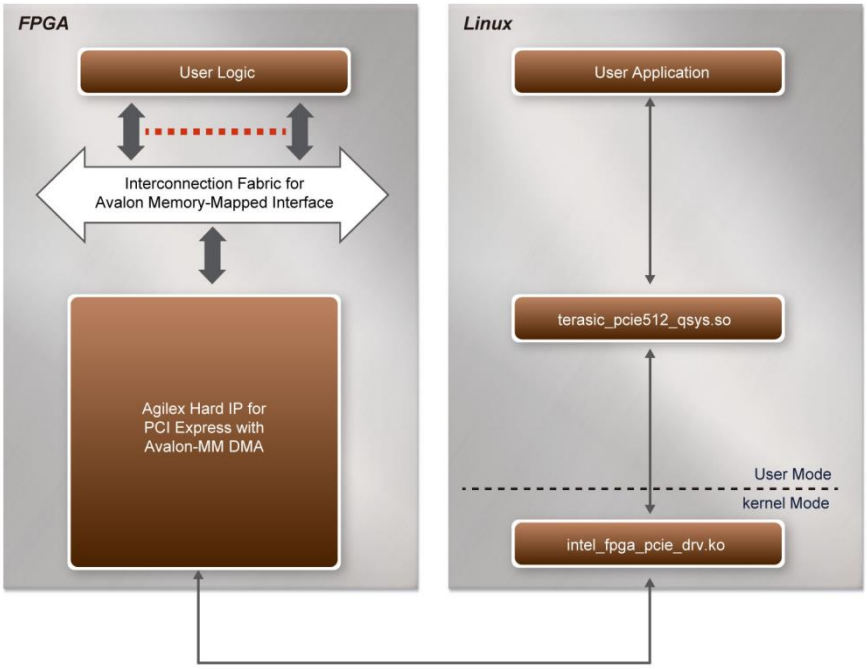


Figure 8-1 Infrastructure of PCI Express System

8.2 PC PCI Express Software SDK

The FPGA System CD contains a PC Linux based SDK to allow users to develop their 64-bit software application on 64-bits Linux. Ubuntu 20.04 is recommended. The SDK is located in the “CDROM/Demonstrations/PCIe_SW_KIT/Linux” folder which includes:

- PCI Express Driver
- PCI Express Library
- PCI Express Examples

The kernel mode driver assumes the PCIe vendor ID (VID) is 0x1172 and the device ID (DID) is 0x09C4. If different VID and DID are used in the design, users need to modify the PCIe vendor ID (VID) and device ID (DID) in the driver project and rebuild the driver. The ID is defined in the file PCIe_SW_KIT/Linux/PCIe_Driver/intel_fpga_pcie_setup.h.

The PCI Express Library is implemented as a single .so file named terasic_pcie512_qsys.so.
This file is a 64-bit library file. With the library exported software API, users can easily communicate with the FPGA. The library provides the following functions:

- Basic data read and write
- Data read and write by DMA

For high performance data transmission, AVMM DMA is required as the read and write operations are specified under the hardware design on the FPGA.

8.3 PCI Express Software Stack

Figure 8-2 shows the software stack for the PCI Express application software on 64-bit Linux. The PCIe library module terasic_pcie512_qsys.so provides DMA and direct I/O access for user application program to communicate with FPGA. Users can develop their applications based on this .so library file. The intel_fpga_pcie_drv.ko kernel driver is provided by Intel.

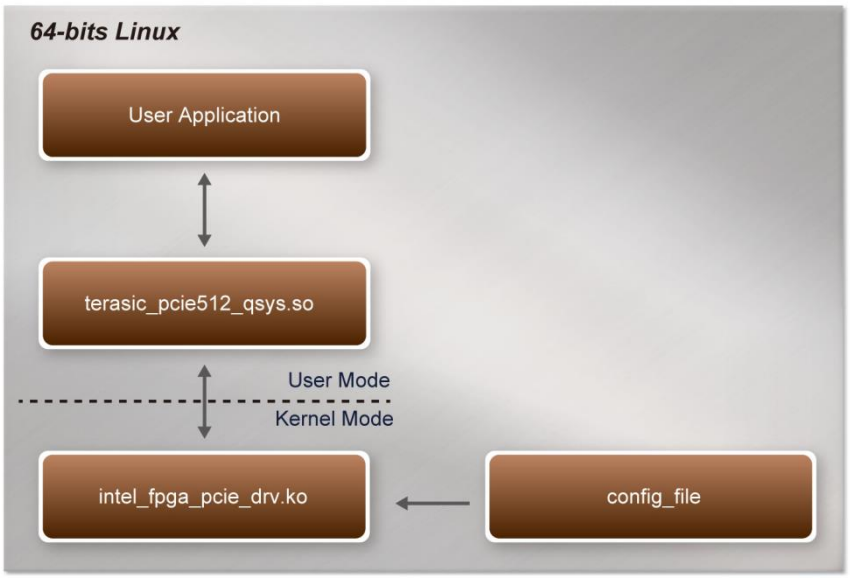


Figure 8-2 PCI Express Software Stack

■ Install PCI Express Driver on Linux

To make sure the PCIe driver can meet your kernel of Linux distribution, the driver intel_fpga_pcie_drv.ko should be recompiled before it is used. The PCIe driver project is locate in the folder:

"CDROM/Demonstrations/PCIe_SW_KIT/Linux/PCIe_Driver"

The folder includes the following files:

- intel_fpga_pcie_chr.c

- intel_fpga_pcie_chr.h
- intel_fpga_pcie_dma.c
- intel_fpga_pcie_dma.h
- intel_fpga_pcie_ioctl.c
- intel_fpga_pcie_ioctl.h
- intel_fpga_pcie_setup.c
- intel_fpga_pcie_setup.h
- intel_fpga_pcie.h
- intel_fpga_pcie_ip_params.h
- Makefile
- load_driver
- unload
- config_file

To compile and install the PCI Express driver, please execute the steps below:

1. Install the DE10-Agilex the PCIe slot of the host PC
2. Make sure Quartus Programmer and USB-Blaster II driver are installed
3. Open a terminal and use "cd" command to go to the folder "CDROM/Demonstrations/PCIe_Fundamental/demo_batch".
4. Set QUARTUS_ROOTDIR variable pointing to the Quartus installation path. Set QUARTUS_ROOTDIR variable by tying the following commands in terminal. Replace “/home/user/intelFPGA_pro/21.2/quartus” to your quartus installation path.

```
export QUARTUS_ROOTDIR=/home/user/intelFPGA_pro/21.2/quartus
```

5. Execute "sudo -E sh test.sh" command to configure the FPGA
6. Restart the Linux operation system. In Linux, open a terminal and use “cd” command to goto the PCIe_Driver folder
7. Type the following commands to compile and install the driver intel_fpga_pcie_drv.ko, and make sure driver is loaded successfully and FPGA is detected by the driver as shown in **Figure 8-3.**
    - make
    - sudo sh load_driver
    - dmesg | tail -n 15

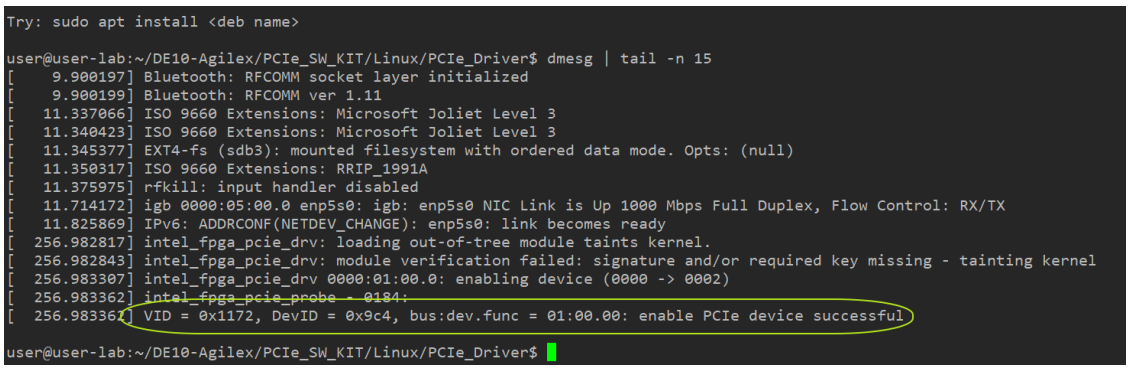


Figure 8-3 Screenshot of install PCIe driver

■ Create a Software Application

All the files needed to create a PCIe software application are located in the directory CDROM/Demonstrations/PCIe_SW_KIT/Linux/PCIe_Library. It includes the following files:

- TERASIC_PCIE_AVMM512.h
- terasic_pcie512_qsys.so (64-bit library)

Below lists the procedures to use the library in users' C/C++ project:

1. Create a 64-bit C/C++ project.
2. Include TERASIC_PCIE_AVMM512.h in the C/C++ project.
3. Copy terasic_pcie512_qsys.so to the folder where the project execution file is located.
4. Dynamically load terasic_pcie512_qsys.so in C/C++ program. To load the terasic_pcie512_qsys.so, please refer to the PCIe fundamental example below.
5. Call the library API to implement the desired application.

Users can easily communicate with the FPGA through the PCIe bus through the terasic_pcie512_qsys.so API. The details of API are described below sections.

8.4 PCI Express Library API

The API is the same as Windows Library. Please refer to the section **PCI Express Library API** in this document.

8.5 PCIe Reference Design -

Fundamental

The application reference design shows how to implement fundamental control and data transfer in the DMA. In the design, basic I/O is used to control the BUTTON and LED on the FPGA board. High-speed data transfer is performed by the DMA.

■ Demonstration Files Location

The demo file is located in the batch folder:

CDROM/Demonstrations/PCIe_Fundamental/demo_batch

The folder includes following files:

- FPGA Configuration File: DE10_Agilex.sof
- Download Batch file: test.sh
- Linux Application Software folder : linux_app, includes
  - ✧ PCIE_FUNDAMENTAL
  - ✧ terasic_pcie512_qsys.so

■ Demonstration Setup

1. Install the FPGA board on your PC as shown in **Figure 8-4**.

Figure 8-4 FPGA board installation on PC

2. Open a terminal and use "cd" command to goto "CDROM/Demonstrations/PCIe_Fundamental/demo_batch".
3. Set QUARTUS_ROOTDIR variable pointing to the Quartus installation path. Set QUARTUS_ROOTDIR variable by tying the following commands in terminal. Replace /home/user/intelFPGA_pro/21.2/quartus to your quartus installation path.

```
export QUARTUS_ROOTDIR=/home/user/intelFPGA_pro/21.2/quartus
```

4. Execute "sudo -E sh test.sh" command to configure the FPGA
5. Restart Linux
6. Install PCIe driver. The driver is located in the folder:
   CDROM/Demonstration/PCIe_SW_KIT/Linux/PCIe_Driver.
7. Type “lspci -nn | grep 1172:09c4” to make sure the Linux has detected the FPGA

Board as shown below.

```
user@user-lab:~/DE10-Agilex/PCIe_SW_KIT/Linux/PCIe_Driver$ lspci -nn | grep 1172:09c4
01:00.0 Non-VGA unclassified device [0000]: Altera Corporation Device [1172:09c4] (rev 01)
user@user-lab:~/DE10-Agilex/PCIe_SW_KIT/Linux/PCIe_Driver$
```

8. Goto linux_app folder, execute PCIE_FUNDAMENTAL. A menu will appear as shown in **Figure 8-5**.

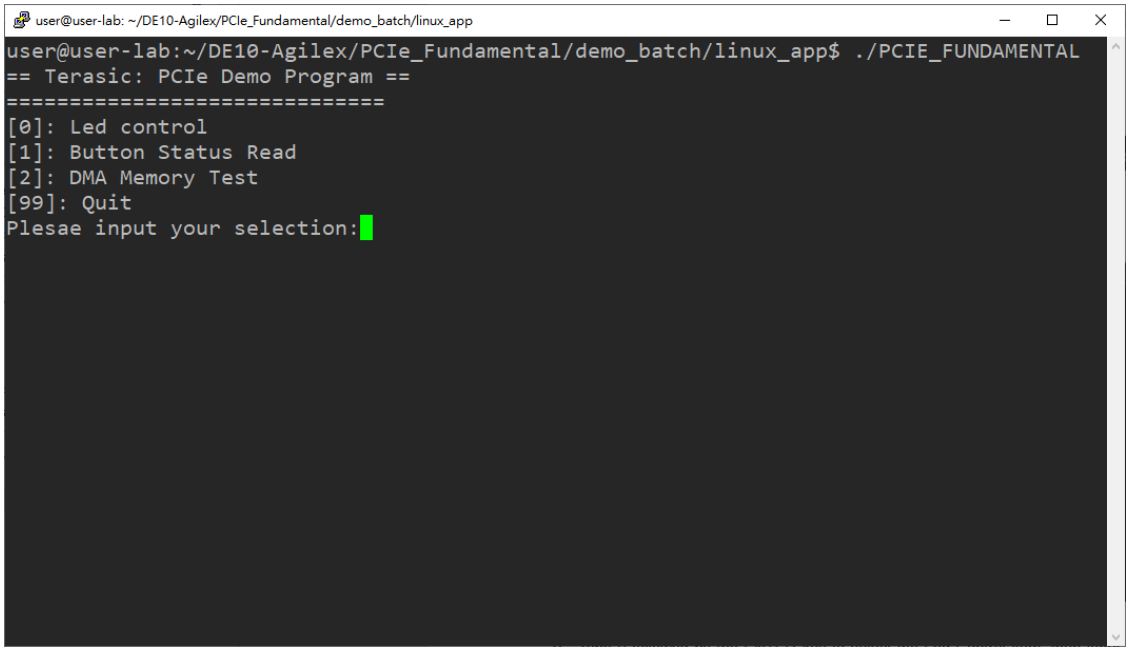


Figure 8-5 Screenshot of Program Menu

9. Type 0 followed by the ENTER key to select the Led Control item, then input 15 (hex 0x0f) will turn all leds on as shown in **Figure 8-6**. If input 0 (hex 0x00), all led will be turned off.

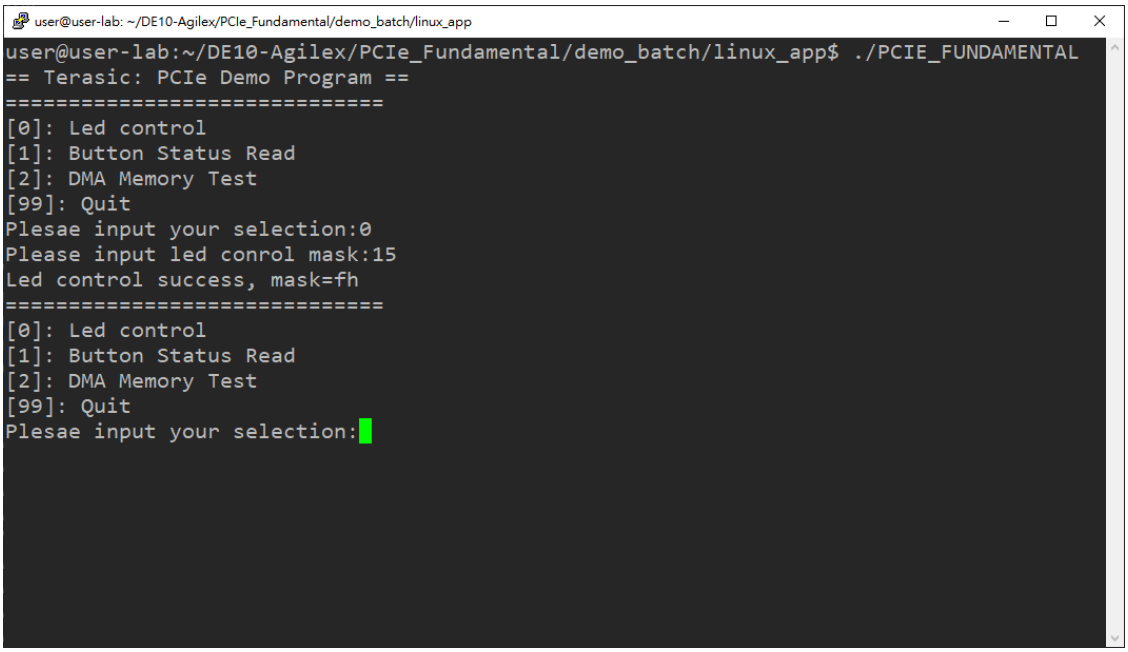


Figure 8-6 Screenshot of LED Control

10. Type 1 followed by the ENTER key to select the Button Status Read item. The button status will be reported as shown in **Figure 8-7**.

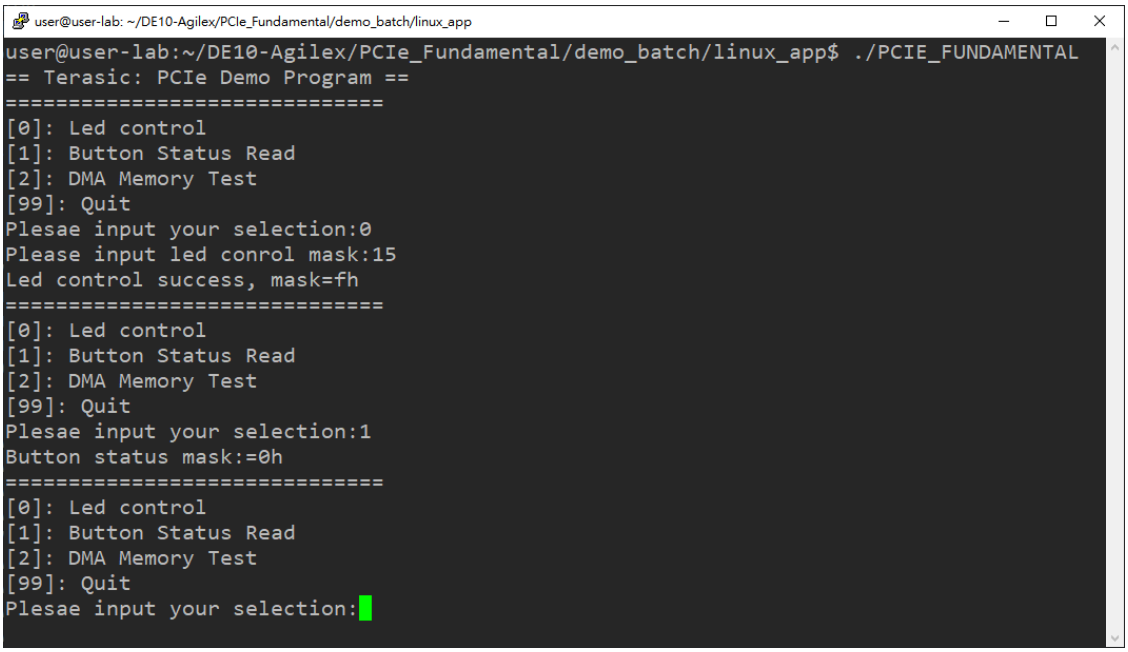


Figure 8-7 Screenshot of Button Status Report

11. Type 2 followed by the ENTER key to select the DMA Testing item. The DMA test result will be reported as shown in **Figure 8-8**.

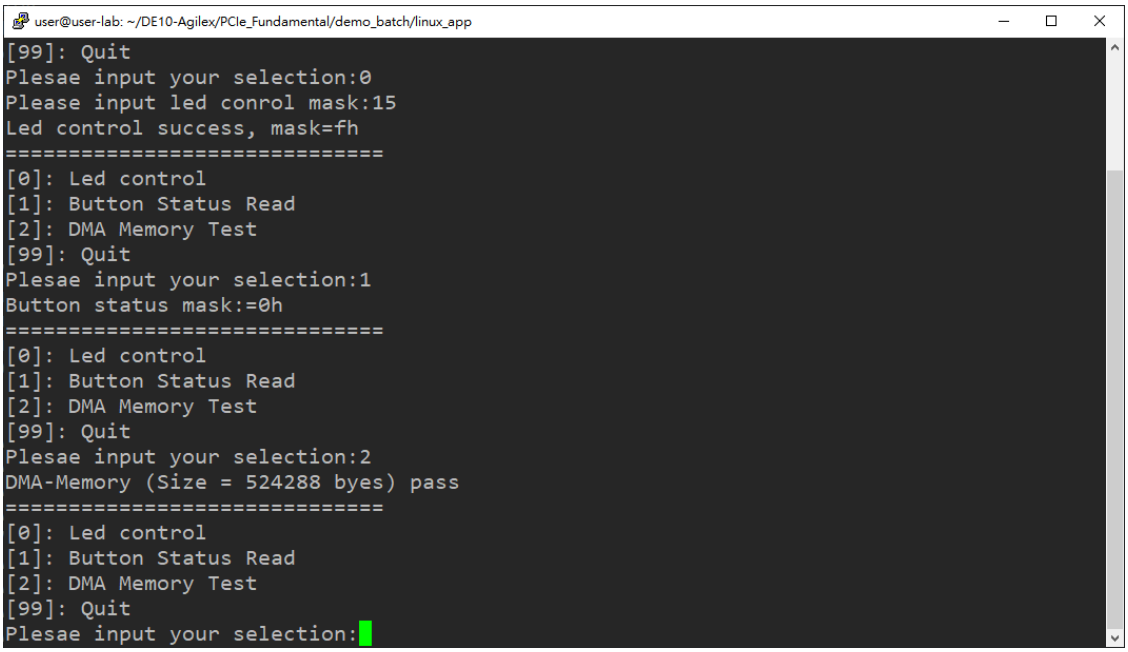


Figure 8-8 Screenshot of DMA Memory Test Result

12. Type 99 followed by the ENTER key to exit this test program

■ Development Tools

- Quartus Prime 21.2 Pro Edition
- GNU Compiler Collection, Version 9.3 is recommended

■ Demonstration Source Code Location

- Quartus Project: Demonstrations/PCIe_Fundamental
- C++ Project: Demonstrations/PCIe_SW_KIT/Linux/PCIE_FUNDAMENTAL

■ FPGA Application Design

Figure 8-9 shows the system block diagram in the FPGA system. In the **Platform Designer** (formerly Qsys), the PIO controller is used to control the LED and monitor the Button Status, and the On-Chip memory is used for performing DMA testing. The PIO controllers and the On-Chip memory are connected to the PCI Express Hard IP controller through the Memory-Mapped Interface.

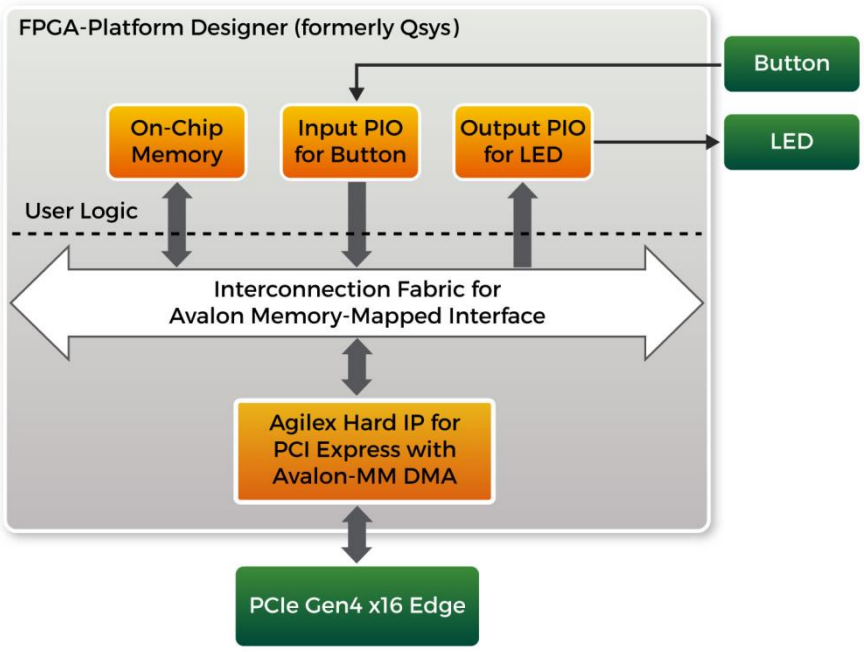


Figure 8-9 Hardware block diagram of the PCIe reference design

■ Linux Based Application Software Design

The application software project is built by GNU Toolchain. The project includes the following major files:

Name	Description
PCIE_FUNDAMENTAL.cpp	Main program
PCIE.c	Implement dynamically load for terasic_pcie512_qsys.so library file
PCIE.h	
TERASIC_PCIE_AVMM512.h	SDK library file, defines constant and data structure

The main program PCIE_FUNDAMENTAL.cpp includes the header file "PCIE.h" and defines the controller address according to the FPGA design.

```
#define DEMO_PCIE_USER_BAR          PCIE_BAR4
#define DEMO_PCIE_IO_LED_ADDR       0x800000
#define DEMO_PCIE_IO_BUTTON_ADDR    0x800040
#define DEMO_PCIE_MEM_ADDR          0x100000

#define MEM_SIZE                    (512*1024) //512KB
```

The base address of BUTTON and LED controllers are 0x800040 and 0x800000 based on PCIE_BAR4, respectively. The on-chip memory base address is 0x100000 relative to the DMA controller.

Before accessing the FPGA through PCI Express, the application first calls PCIE_Load to dynamically load the terasic_pcie512_qsys.so. Then, it call PCIE_Open to open the PCI Express driver. The constant DEFAULT_PCIE_VID and DEFAULT_PCIE_DID used in PCIE_Open are defined in TERASIC_PCIE_AVMM512.h. If developers change the Vendor ID and Device ID and PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM512.h. If the return value of PCIE_Open is zero, it means the driver cannot be accessed successfully. In this case, please make sure:

- The FPGA is configured with the associated bit-stream file and the host is rebooted.
- The PCI express driver is loaded successfully.

The LED control is implemented by calling PCIE_Write32 API, as shown below:

```
bPass = PCIE_Write32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_LED_ADDR, (uint32_t) Mask);
```

The button status query is implemented by calling the **PCIE_Read32** API, as shown below:

```
bPass = PCIE_Read32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_BUTTON_ADDR, &Status);
```

The memory-mapped memory read and write test is implemented by **PCIE_DmaWrite** and **PCIE_DmaRead** API, as shown below:

```
bPass = PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);
bPass = PCIE_DmaRead(hPCIe, LocalAddr, pRead, nTestSize);
```

8.6 PCIe Reference Design - DDR4

The application reference design shows how to add DDR4 Memory Controllers for

DDR4-A SODIMM,DDR4-B SODIMM, DDR4-C SODIMM and DDR4-D SODIMM into the PCIe Quartus project based on the PCIe_Fundamental Quartus project and perform 8GB data DMA for both SODIMM. Also, this demo shows how to call “PCIE_ConfigRead32” API to check PCIe link status.

■ Demonstration Files Location

The demo file is located in the batch folder:

CDROM\Demonstrations\PCIe_DDR4\demo_batch

The folder includes following files:

- FPGA Configuration File: DE10_Agilex.sof
- Download Batch file: test.sh
- Linux Application Software folder : linux_app, includes
  - ✧ PCIE_DDR4
  - ✧ terasic_pcie512_qsys.so

■ Demonstration Setup

1. Install four pieces of DDR4 2666 8GB SODIMM on the FPGA board.
2. Install the FPGA board on the PCIe Slot of your PC.
3. Open a terminal and use "cd" command to go to "CDROM/Demonstrations/PCIe_DDR4/demo_batch".
4. Set QUARTUS_ROOTDIR variable pointing to the Quartus installation path. Set QUARTUS_ROOTDIR variable by tying the following commands in the terminal. Replace /home/user/intelFPGA_pro/21.2/quartus to your Quartus installation path.

```
export QUARTUS_ROOTDIR=/home/user/intelFPGA_pro/21.2/quartus
```

5. Execute "sudo -E sh test.sh" command to configure the FPGA
6. Restart Linux
7. Install PCIe driver.
8. Make sure that Linux has detected the FPGA Board.
9. Go to the linux_app folder, execute PCIE_DDR4. A menu will appear as shown in **Figure 8-10**.

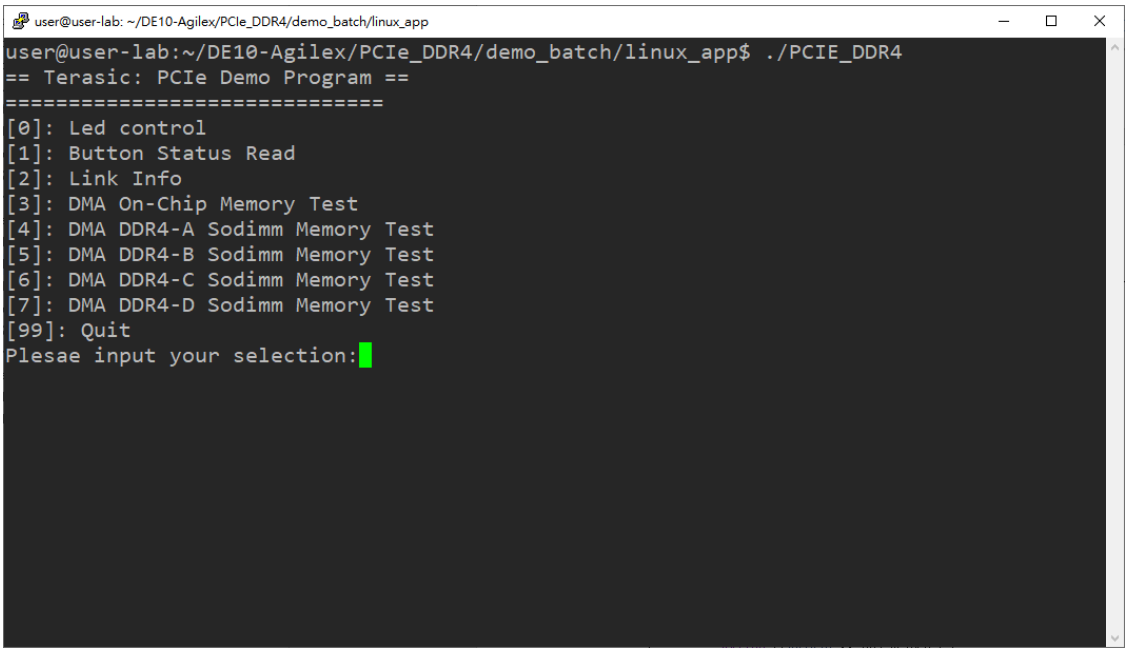


Figure 8-10 Screenshot of Program Menu

10. Type 2 followed by the ENTER key to select the Link Info item. The PCIe link information will be shown as in **Figure 8-11**. Gen3 link speed and x16 link width are expected.

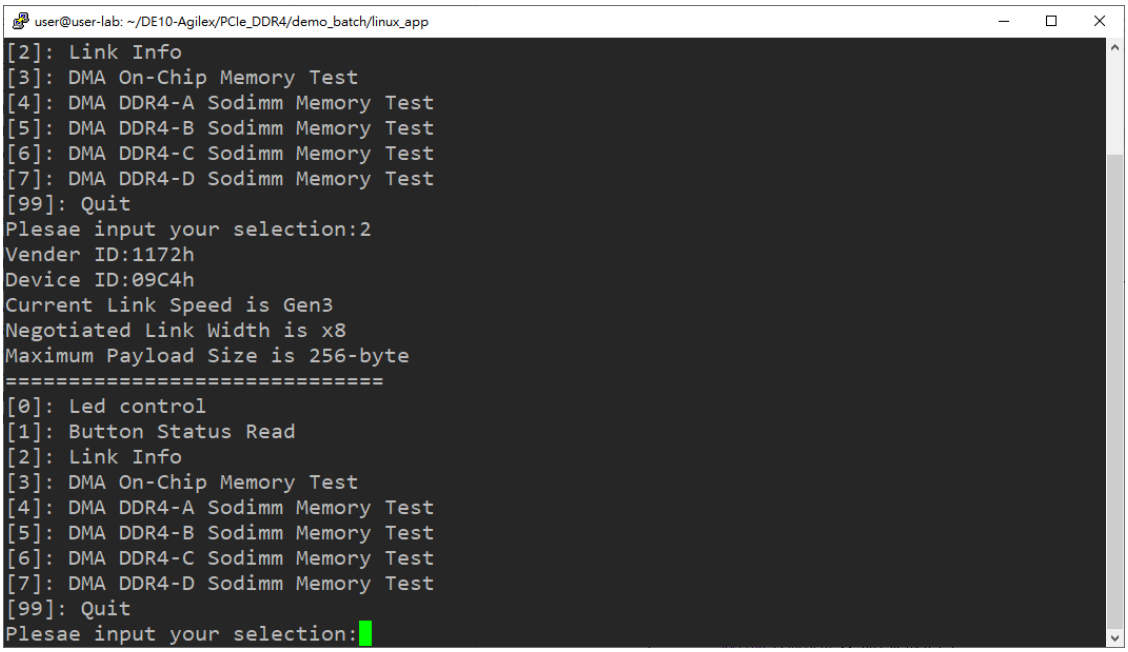


Figure 8-11 Screenshot of Link Info

11. Type 3 followed by the ENTER key to select DMA On-Chip Memory Test item. The DMA write and read test result will be report as shown in **Figure 8-12**.

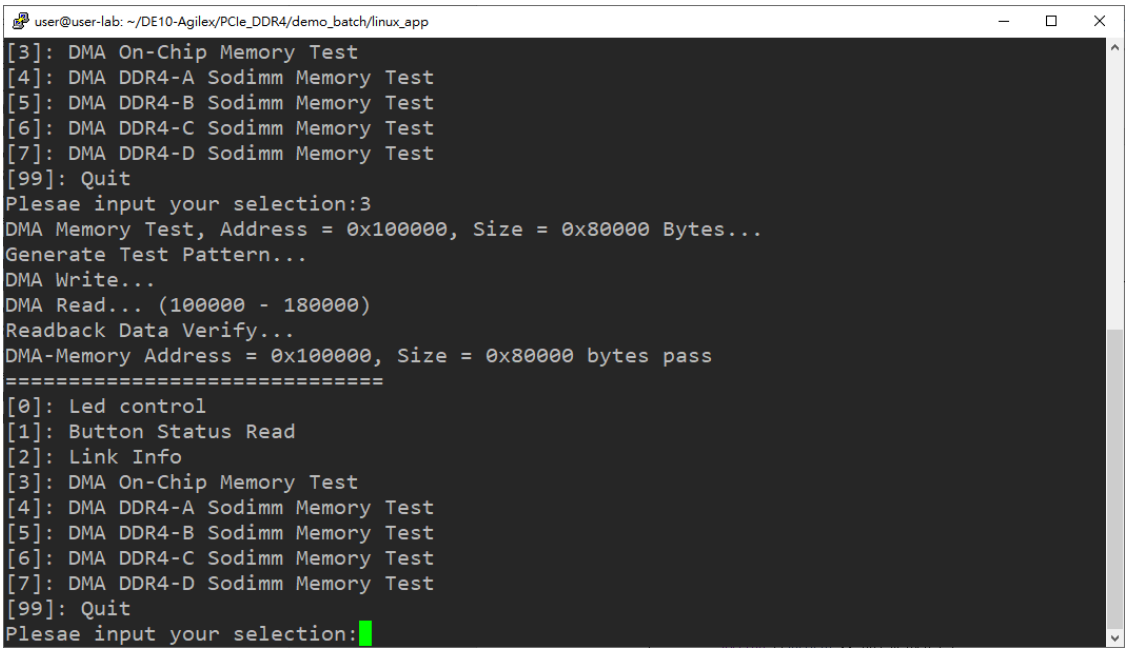


Figure 8-12 Screenshot of On-Chip Memory DMA Test Result

12. Type 4 followed by the ENTER key to select the DMA DDR4-A SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 8-13**.

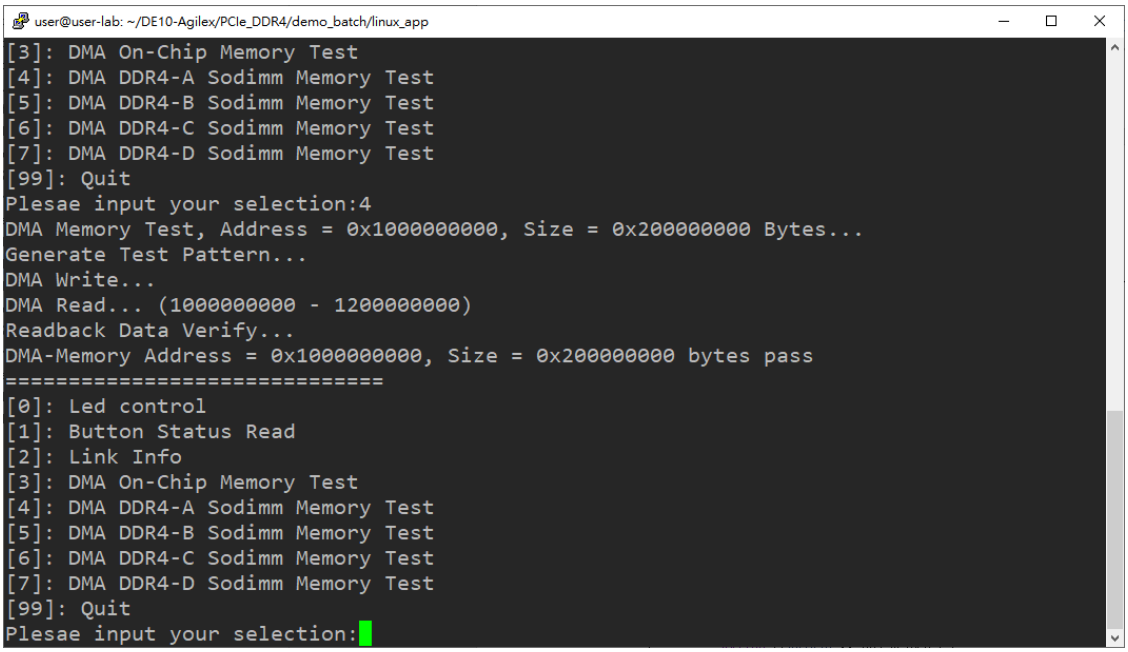


Figure 8-13 Screenshot of DDR4-A SOSIMM Memory DAM Test Result

13. Type 5 followed by the ENTER key to select the DMA DDR4-B SODIMM Memory

Test item. The DMA write and read test result will be reported as shown in **Figure 8-14**.

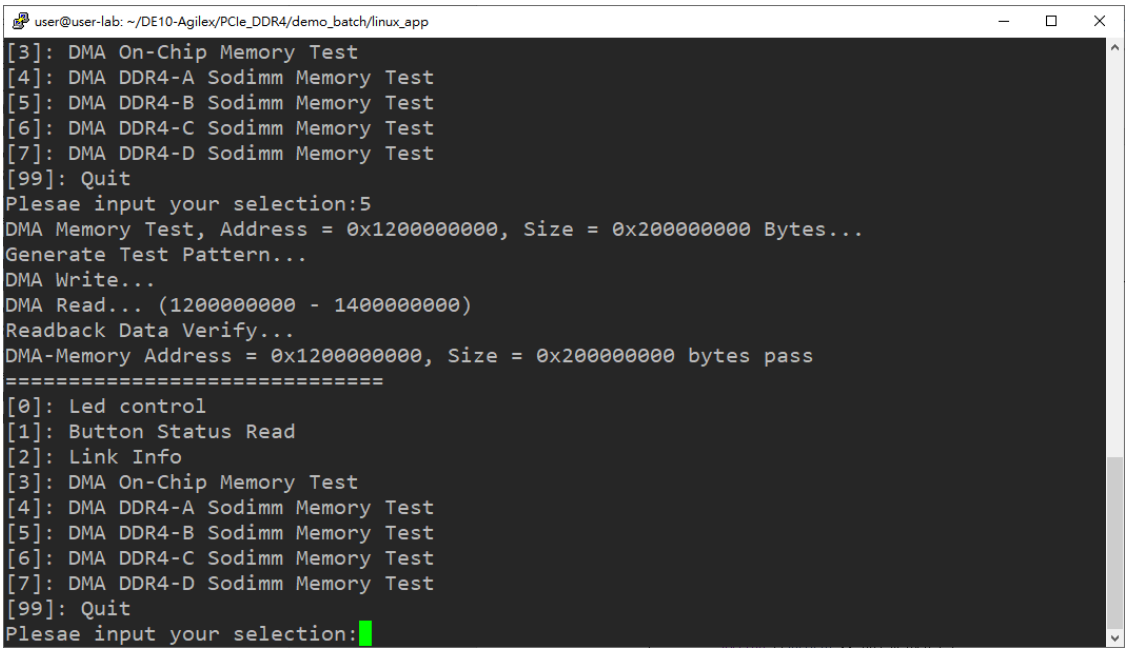


Figure 8-14 Screenshot of DDR4-B SOSIMM Memory DAM Test Result

14. Type 6 followed by the ENTER key to select the DMA DDR4-C SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 8-15**.

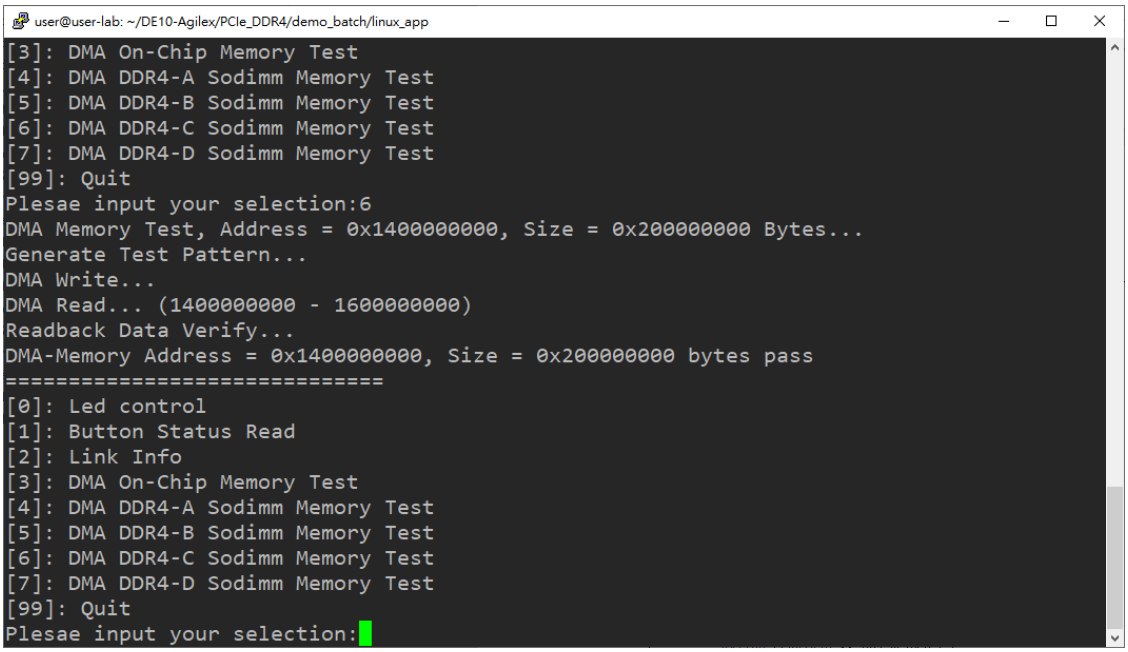


Figure 8-15 Screenshot of DDR4-C SOSIMM Memory DAM Test Result

15. Type 7 followed by the ENTER key to select the DMA DDR4-D SODIMM Memory Test item. The DMA write and read test result will be reported as shown in **Figure 8-16**.

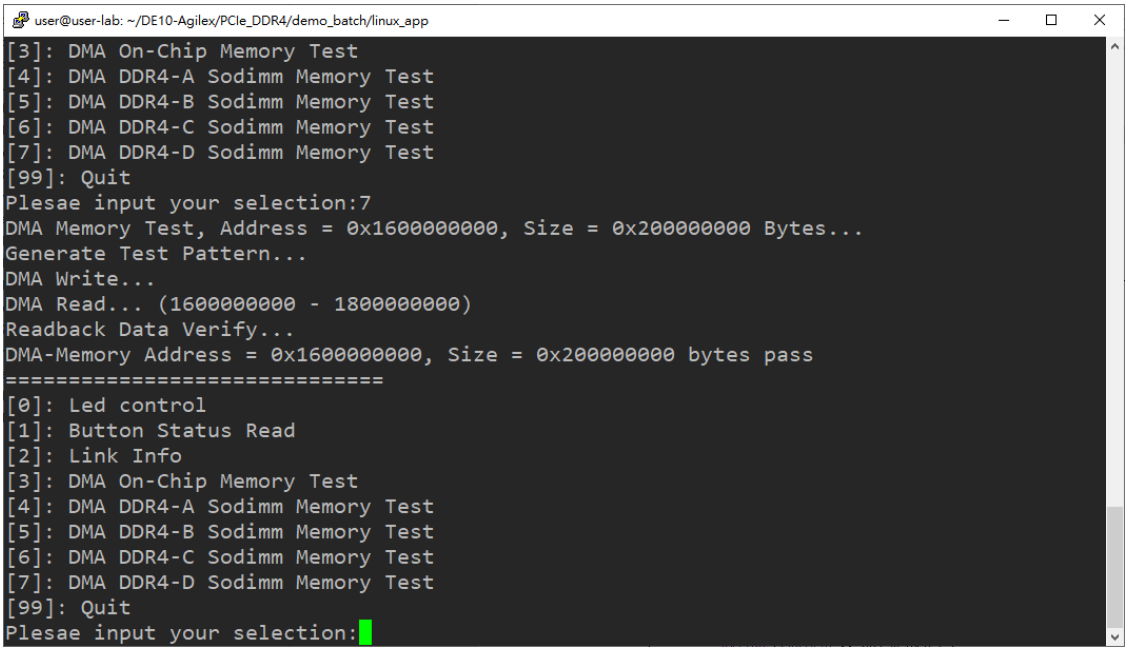


Figure 8-16 Screenshot of DDR4-D SOSIMM Memory DAM Test Result

16. Type 99 followed by the ENTER key to exit this test program.

■ Development Tools

- Quartus Prime 21.2 Pro Edition
- GNU Compiler Collection, Version 9.3 is recommended

■ Demonstration Source Code Location

- Quartus Project: Demonstrations/PCIE_DDR4
- C++ Project: Demonstrations/PCIe_SW_KIT/Linux/PCIe_DDR4

■ FPGA Application Design

Figure 8-17 shows the system block diagram in the FPGA system. In the **Platform Designer** (formerly Qsys), the PIO controller is used to control the LED and monitor the Button Status, and the On-Chip memory is used for performing DMA testing. The PIO controllers and the On-Chip memory are connected to the PCI Express Hard IP

controller through the Memory-Mapped Interface.

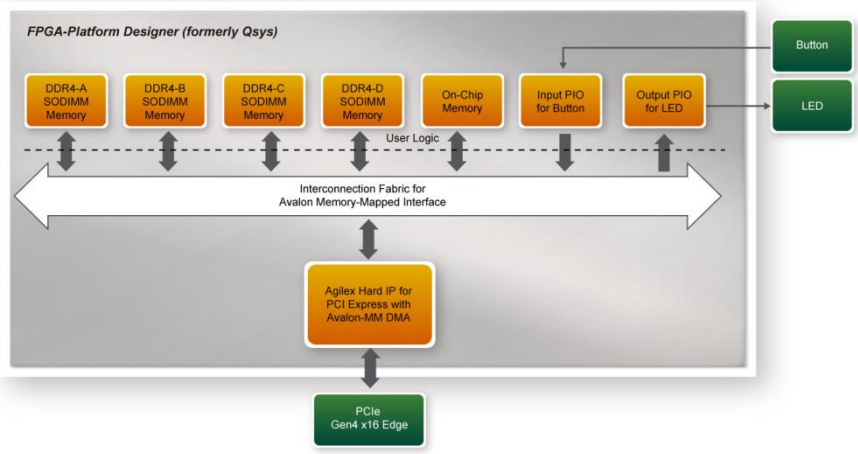


Figure 8-17 Hardware block diagram of the PCIe_DDR4 reference design

■ Linux Based Application Software Design

The application software project is built by GNU Toolchain. The project includes the following major files:

Name	Description
PCIE_DDR4.cpp	Main program
PCIE.c	Implement dynamically load for terasic_pcie512_qsys.so library file
PCIE.h	
TERASIC_PCIE_AVMM512.h	SDK library file, defines constant and data structure

The main program PCIE_DDR4.cpp includes the header file "PCIE.h" and defines the controller address according to the FPGA design.

```
#define DEMO_PCIE_USER_BAR          PCIE_BAR4
#define DEMO_PCIE_IO_LED_ADDR       0x800000
#define DEMO_PCIE_IO_BUTTON_ADDR    0x800040
#define DEMO_PCIE_ONCHIP_MEM_ADDR   0x100000
#define DEMO_PCIE_DDR4A_MEM_ADDR    0x1000000000
#define DEMO_PCIE_DDR4B_MEM_ADDR    0x1200000000
#define DEMO_PCIE_DDR4C_MEM_ADDR    0x1400000000
#define DEMO_PCIE_DDR4D_MEM_ADDR    0x1600000000

#define ONCHIP_MEM_TEST_SIZE        (512*1024) //512KB
#define DDR4A_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4B_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4C_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
#define DDR4D_MEM_TEST_SIZE         (8ull*1024*1024*1024) //8GB
```

The base address of BUTTON and LED controllers are 0x800040 and 0x800000 based on PCIE_BAR4, respectively. The on-chip memory base address is 0x100000 relative to the DMA controller. The above definition is the same as those in PCIe Fundamental demo.

Before accessing the FPGA through PCI Express, the application first calls the PCIE_Load to dynamically load the terasic_pcie512_qsys.so. Then, it calls the PCIE_Open to open the PCI Express driver. The constant DEFAULT_PCIE_VID and DEFAULT_PCIE_DID used in the PCIE_Open are defined in TERASIC_PCIE_AVMM512.h. If developers changes the Vendor ID and Device ID and PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM512.h. If the return value of the PCIE_Open is zero, it means the driver cannot be accessed successfully. In this case, please make sure:

- The FPGA is configured with the associated bit-stream file and the host is rebooted.
- The PCI express driver is loaded successfully.

The LED control is implemented by calling PCIE_Write32 API, as shown below:

```
bPass = PCIE_Write32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_LED_ADDR, (uint32_t) Mask);
```

The button status query is implemented by calling the **PCIE_Read32** API, as shown below:

```
bPass = PCIE_Read32(hPCIe, DEMO_PCIE_USER_BAR, DEMO_PCIE_IO_BUTTON_ADDR, &Status);
```

The memory-mapped memory read and write test is implemented via **PCIE_DmaWrite** and the **PCIE_DmaRead** API, as shown below:

```
bPass = PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);
bPass = PCIE_DmaRead(hPCIe, LocalAddr, pRead, nTestSize);
```

The PCIe link information is implemented by PCIE_ConfigRead32 API, as shown below:

```
// read config - link status
if (PCIE_ConfigRead32(hPCIe, 0x80, &Data32)) {
    switch ((Data32 >> 16) & 0x0F) {
    case 1:
        printf("Current Link Speed is Gen1\n");
        break;
    case 2:
        printf("Current Link Speed is Gen2\n");
        break;
    case 3:
        printf("Current Link Speed is Gen3\n");
        break;
    case 4:
        printf("Current Link Speed is Gen4\n");
        break;
    default:
        printf("Current Link Speed is Unknown\n");
        break;
    }
    switch ((Data32 >> 20) & 0x3F) {
    case 1:
        printf("Negotiated Link Width is x1\n");
        break;
    case 2:
        printf("Negotiated Link Width is x2\n");
        break;
    case 4:
        printf("Negotiated Link Width is x4\n");
        break;
    case 8:
        printf("Negotiated Link Width is x8\n");
        break;
    case 16:
        printf("Negotiated Link Width is x16\n");
        break;
    default:
        printf("Negotiated Link Width is Unknown\n");
        break;
    }
} else {
    bPass = false;
}
```

Chapter 9

Transceiver Verification

This chapter describes how to verify the FPGA transceivers via the QSFP-DD connector. There are two test codes available in the DE10-Agilex System CD. In addition to providing the loopback test code of the transceiver in the System CD, several Intel transceiver IP example design for Intel® Agilex® devices are also provided. For example, Serial Lite IV IP, CPRI PHY and 00G Ethernet Example are provides in the System CD.

9.1 Transceiver Test Code

The transceiver test code is used to verify the transceiver channels via the QSPF-DD ports through an external loopback method. The transceiver channels are verified with the data rates 25G bps with **NRZ** modulation (Total 200G bps for 8 channels).

9.2 Loopback Fixture

To enable an external loopback of the transceiver channels, QSPF-DD loopback fixtures, as shown in **Figure 9-1**, are required.

Figure 9-1 QSFP-DD Loopback Cable

The loopback fixture can be obtained from the DE10-Agilex product webpage (200G QSFP-DD Loopback Module). The link of the product webpage is list in below:

https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=142&No=1252&PartNo=6

Figure 9-2 the FPGA board with two QSPF-DD loopback fixtures installed.

Figure 9-2 QSPF-DD Transceiver Loopback Test in Progress

9.3 Testing by Transceiver Test Code

The transceiver test code is available in the folder System CD\Tool\Transceiver_Test, which has two QSFP-DD transceiver loopback test codes.

Figure 9-3 and **Figure 9-4** shows the Transceiver Native PHY IP settings in the test code. The data rate of each transceiver channel is set to 25781.25 Mbps and the PMA modulation type is **NRZ**. So the 200Gbps QSPF-DD loopback test code is implemented (8 channels in total).

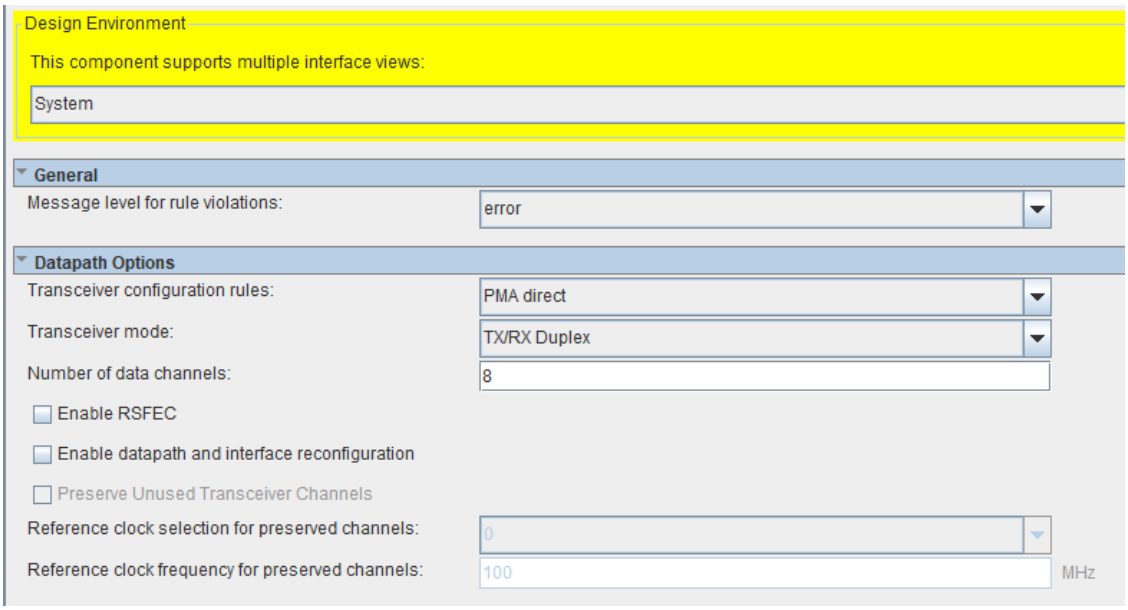


Figure 9-3 The Transceiver PHY setting

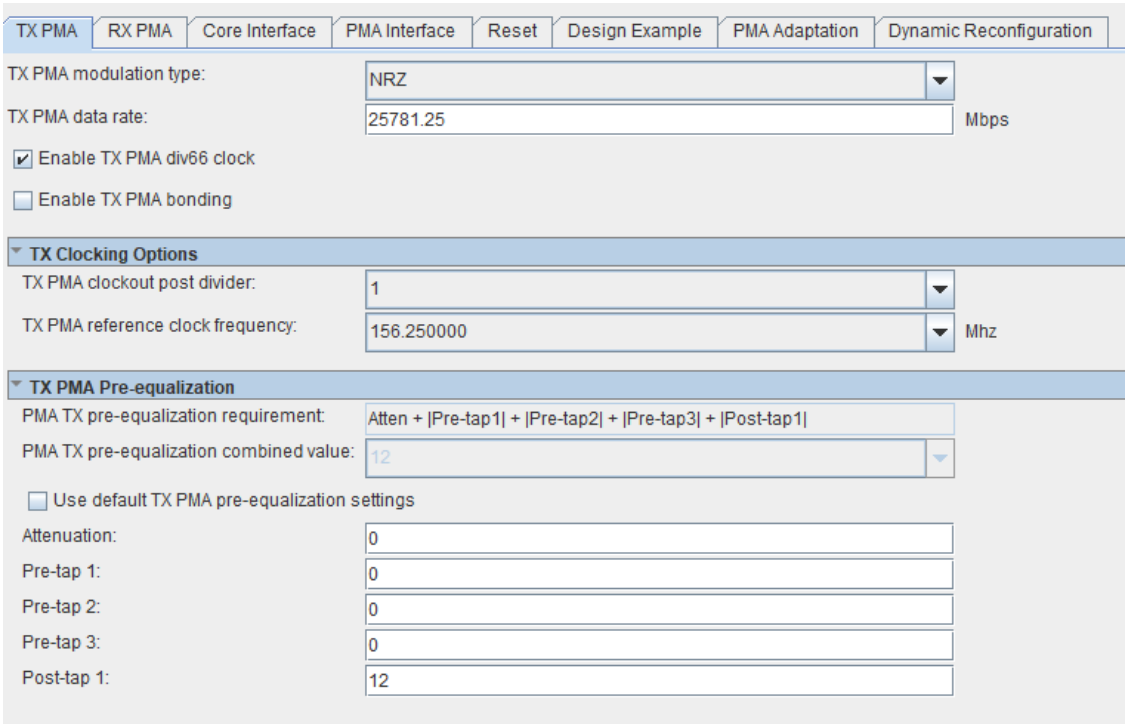


Figure 9-4 The Transceiver PHY setting

The FPGA transceiver PMA setting used are shown in the table below.

Direction	Item	Value
TX	VOD Control	0
	Pre-emphasis first post-tap	12
RX	Initial Adaption	

Here are the procedures to perform transceiver channel test:

1. Copy the Transceiver_Test folder to your local disk.
2. Ensure that the FPGA board is NOT powered on.
3. Plug-in the QSPF-DD loopback fixtures.
4. Connect your FPGA board to your PC with a mini USB cable.
5. Power on the FPGA board
6. Execute 'test.bat" in the Transceiver_Test folder under your local disk.
7. The batch file will download .sof and .elf files, and start the test immediately. The test result is shown in the Nios II Terminal, as shown in **Figure 9-5.**
8. To terminate the test, press one of the BUTTON0~1 buttons on the FPGA board. The loopback test will terminate as shown in **Figure 9-6**.

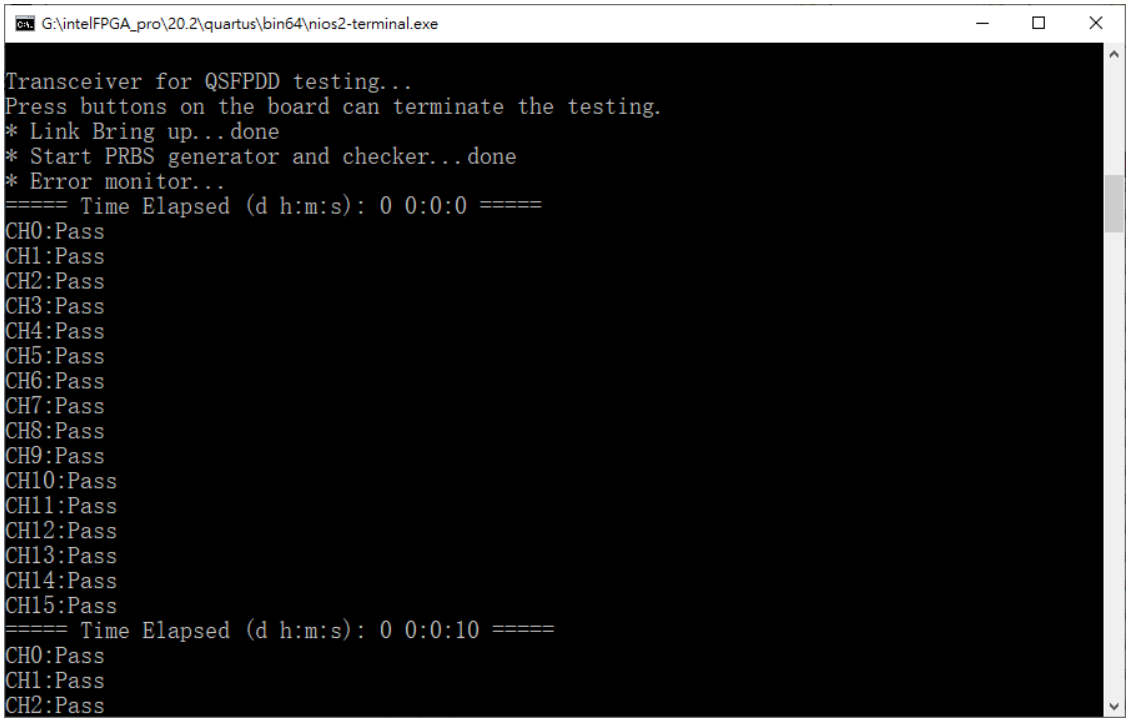


Figure 9-5 QSPF-DD Transceiver Loopback Test in Progress

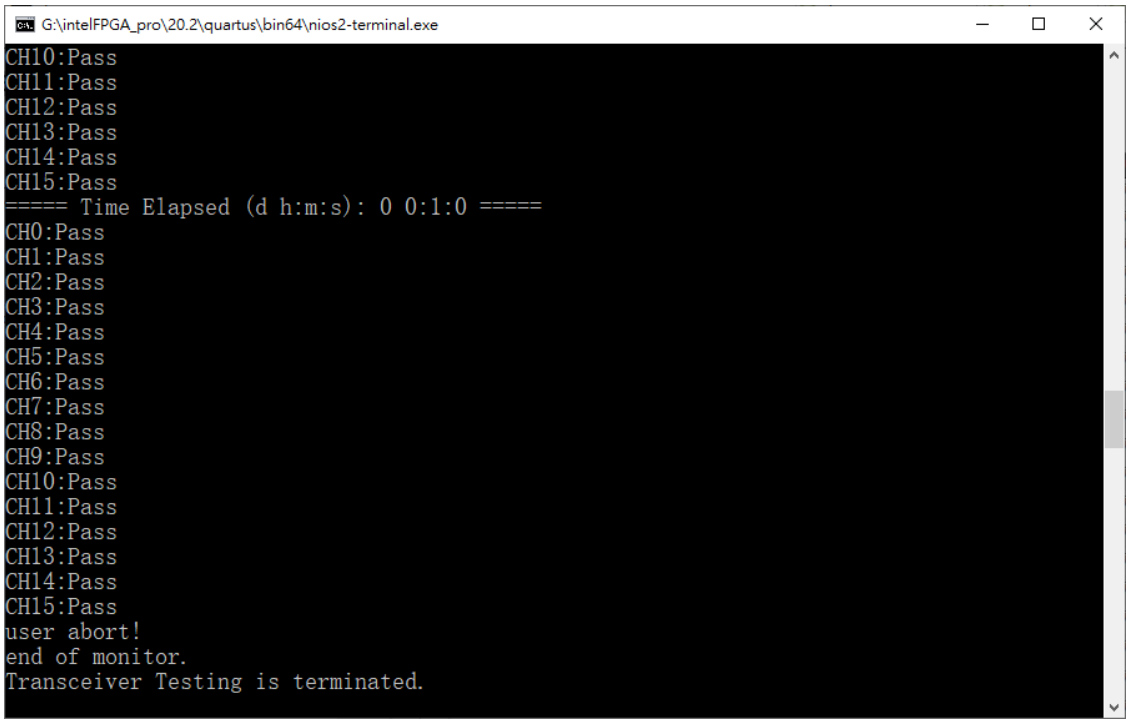


Figure 9-6 QSPF-DD Transceiver Loopback is terminated

9.4 100G Ethernet Example (E-Tile FPGA)

This 100G Ethernet example is generated according to the documents E-Tile Ethernet IPfor Intel Agilex FPGA Design Example. The E-Tile Ethernet IP is used in the example design. The IP is configured as 100GE MAC+PC with (528,514) RS-FEC. This example executes the internal and external loopback test through four-channel of one QSFPDD ports on the FPGA main board. For external loopback test, a QSFPDD or QSFP28 loopback fixture is required, otherwise only internal loopback test be available. **Figure 9-7** shows the block diagram of this demonstration.

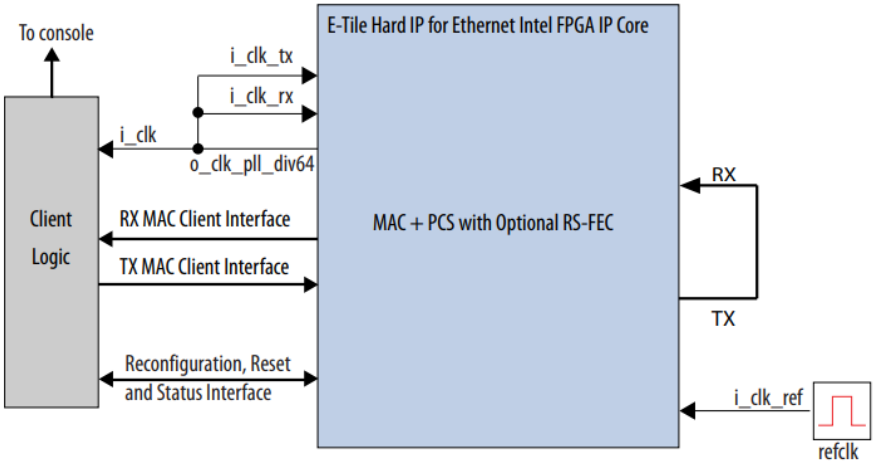


Figure 9-7 Block diagram of 100GbE demo

■ Project Information

The Quartus project is located in CD\Demonstration folder. Project information is shown in the table below.

Item	Description
Project Location	alt_ehipc3_fm_100GE
Quartus Project	alt_ehipc3_fm_100GE\hardware_test_design
FPGA Bit Stream	alt_ehipc3_fm_100GE\demo_batch
Test Scrip File	alt_ehipc3_fm_100GE\hardware_test_design\hwtest\main.tcl
Quartus Version	Quartus Prime 21.2 Pro Edition

Figure 9-8 shows the IP setup for the demonstration. **Single 100GE with optional RSFEC** Core Variant is selected and **Enable RSFEC** is checked.

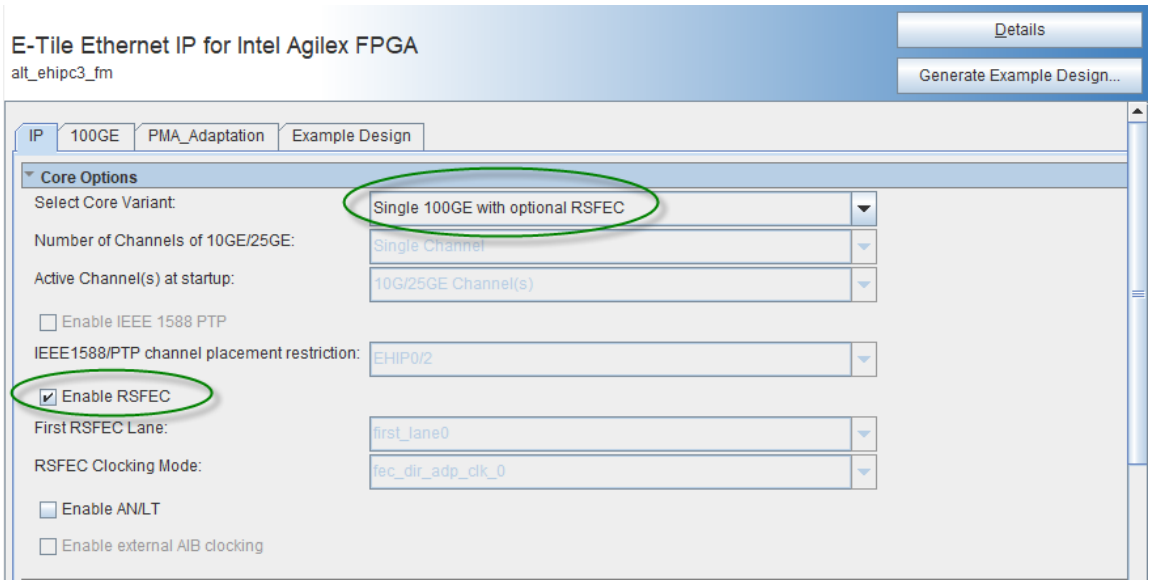


Figure 9-8 Core Variant Setup for Ethernet IP

Four transceiver channels are used in the Quartus Project. Developers can use three constants **ENABLE_QSFPDDA**, **ENABLE_QSFPDDB** and **USE_CH_4567** to selected desired four transceiver channels as table below. The pre-compiled .sof files for below four cases are available on the demo_batch folder.

Used Channel	ENABLE_QSFPDDA	ENABLE_QSFPDDA	USE_CH_4567
QSFPDDA Channel 0~3	Yes	No	No
QSFPDDA Channel 4~7	Yes	No	Yes
QSFPDDB Channel 0~3	No	Yes	No
QSFPDDB Channel 4~7	No	Yes	Yes

■ Demonstration Setup

Here is the procedure to setup the demonstration. A QSFP28 or QSFPDD loopback fixtures are required for external loopback. If you don't have the loopback fixture, please use **run_test** instead of **run_test_ex** in the following demonstration procedure. The **run_test** enables transceiver serial loopback for internal loopback.

1. Insert a QSFP28 or QSFPDD loopback fixture into the QSFPDD port on the DE10-Agilex board, as shown in **Figure 9-9**.
2. Connect the host PC to the FPGA board using a mini-USB cable. Please make sure the USB-Blaster II driver is installed on the host PC.

3. Goto demo_batch folder and execute test.bat to configure FPGA. There are four pre-compiled sof files are available.
4. Open alt_ehipc3_fm_100GE Quartus Project and launch the System Console by selecting the menu item **Tools ➔ System Debugging Tools ➔System Console** in Quartus.
5. In the System Console window, input the following commands to start the loopback test, as shown in **Figure 9-10**.

   %cd hwtest

   %source main.tcl

   %run_test_ex
6. The loopback test report will be displayed in the Tcl Console, as shown in **Figure 9-11** and **Figure 9-12.**

Figure 9-9 Setup QSFP28 or QSFPDD loopback fixture

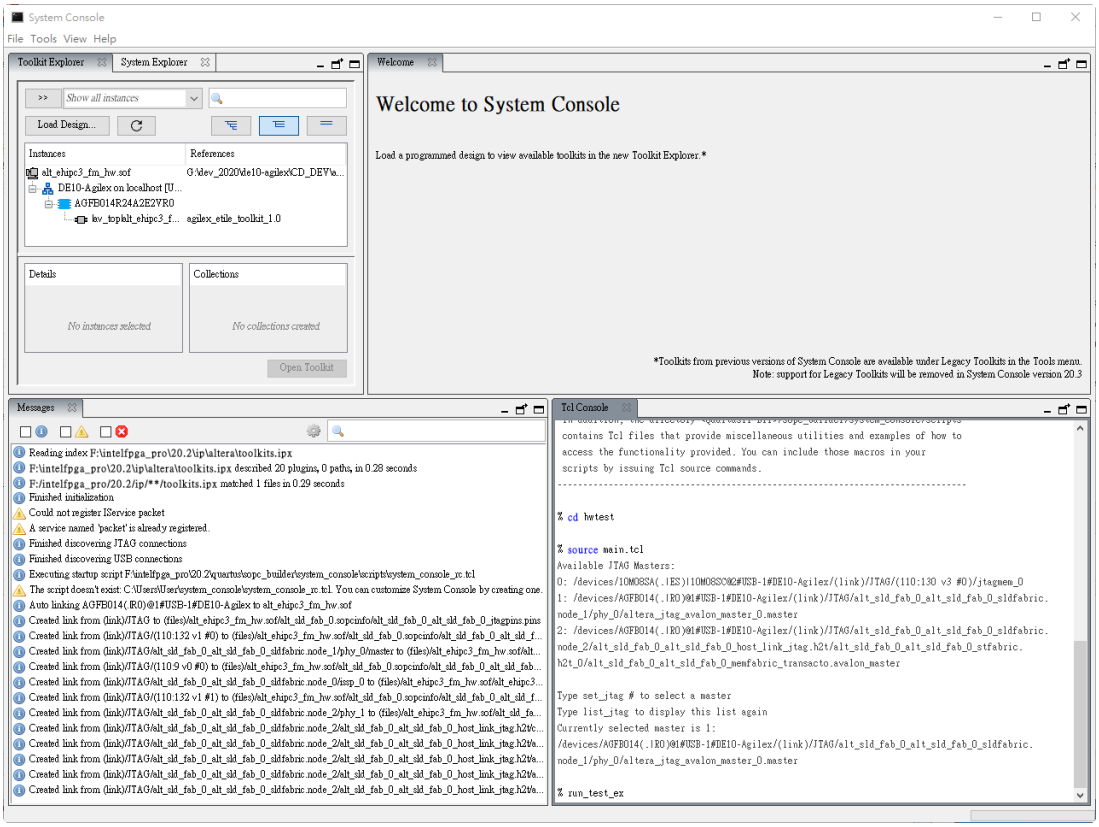


Figure 9-10 Launch the System Console for Ethernet 100G Demo

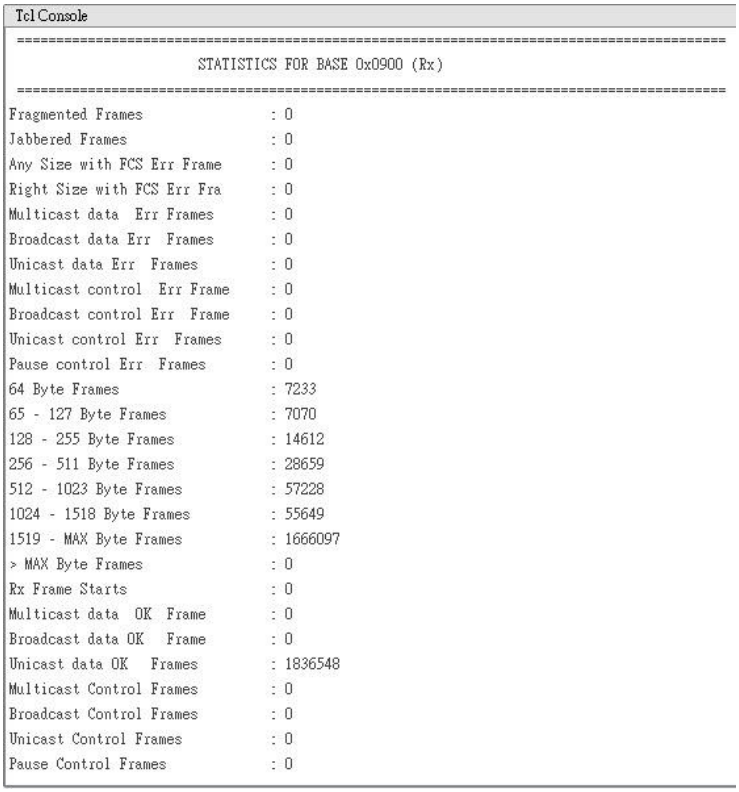


Figure 9-11 Ethernet 100G loopback test report for RX

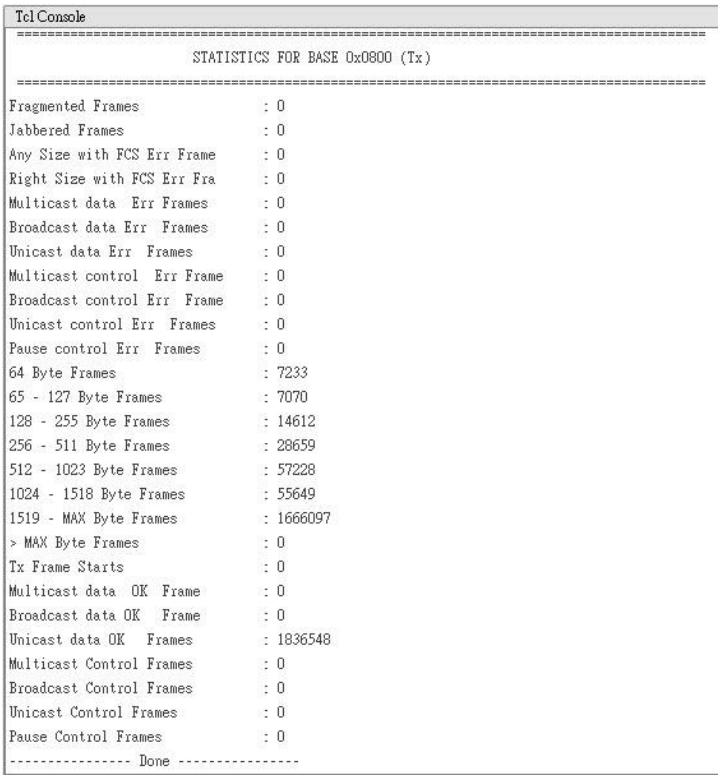


Figure 9-12 Ethernet 100G loopback test report for TX

Chapter 10

Dashboard GUI

The DE10-Agilex Dashboard GUI is a board status monitor system. This system is connected from the Host to the System MAX10 FPGA on the DE10-Agilex through the UART interface, and reads various status on the board (See section 2.9 for detailed). The reported status includes FPGA/Board temperature, fan speed, FPGA core power and 12V input power. **Figure 10-1** shows the block diagram of the DE10-Agilex Dashboard. Note that, the Dashboard GUI software only support windows OS.

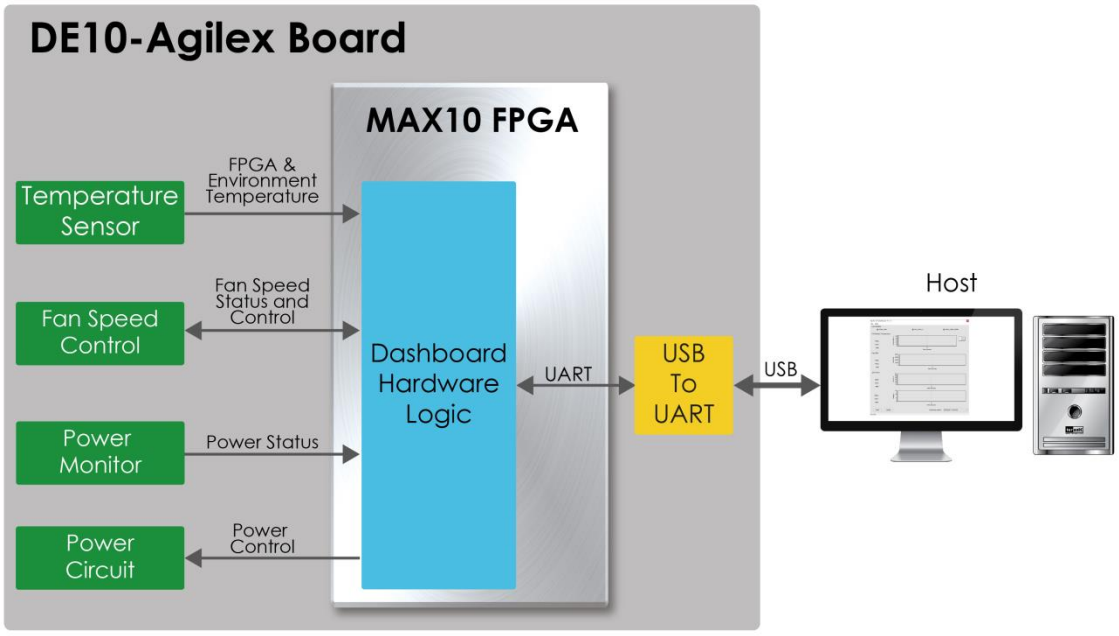


Figure 10-1 Block Diagram of the DE10-Agilex Dashboard

10.1 Driver Installed on Host

To use the dashboard system, users need to install the USB to UART driver on the host first, so that user can establish a connection with the DE10-Agilex board. This section will describe how to install USB to UART driver on the windows OS host.

■ USB to UART driver location

Users can find it from the path: Tool\dashboard_gui\Driver in the DE10-Agilex system CD and copy it to the Host.

■ Connection Setting

1. Connect the USB Mini USB connector of the DE10-Agilex board to the Host USB port through mini USB cable.
2. Connect power to the DE10-Agilex board.
3. Power on the DE10-Agilex board.

Figure 10-2 Connection setup for using dashboard system

■ Install Driver

When connect the DE10-Agilex board to the Host. As shown in **Figure 10-3**, two USB to UART Com Port device is shown in “Device Manager” of Host.

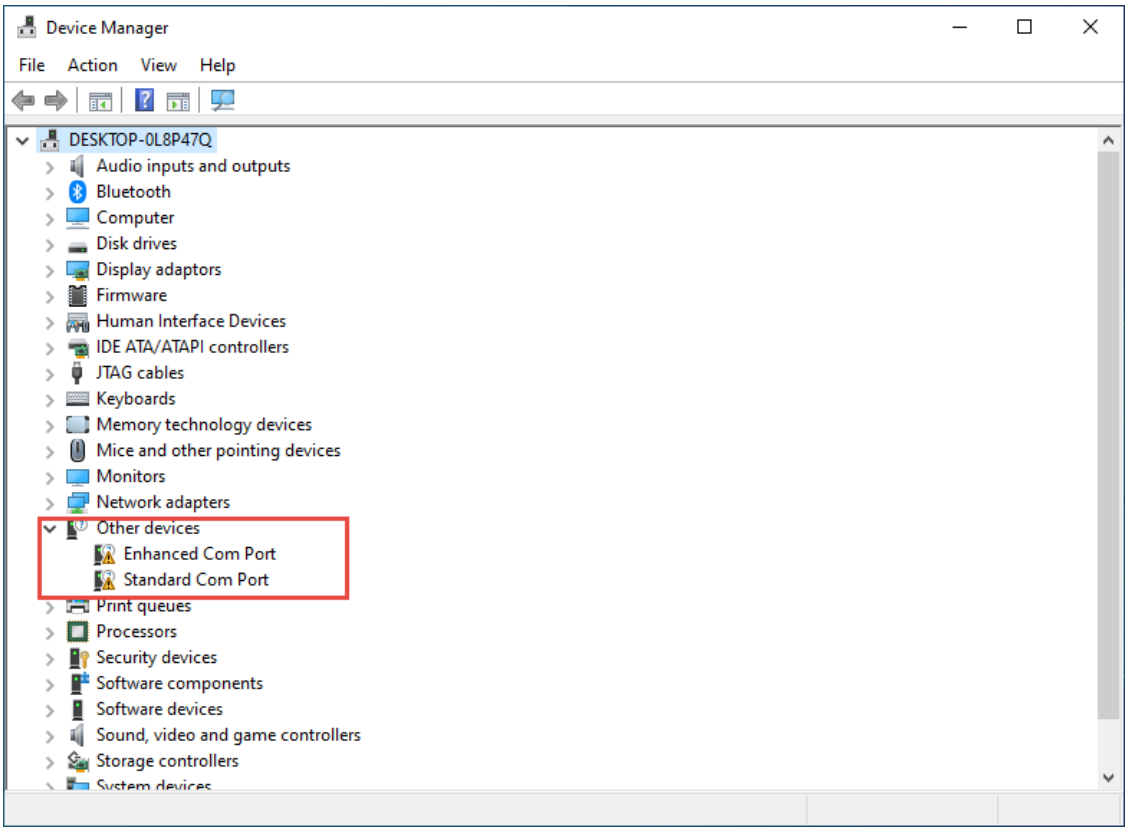


Figure 10-3 Uninstalled USB to UART device

Copy the device driver (System CD\Tool\dashboard_gui\Driver) to the Host and install it, as shown in **Figure 10-4.** Please note that the COM Port number is different in different Host.

Figure 10-4 Install USB to UART driver

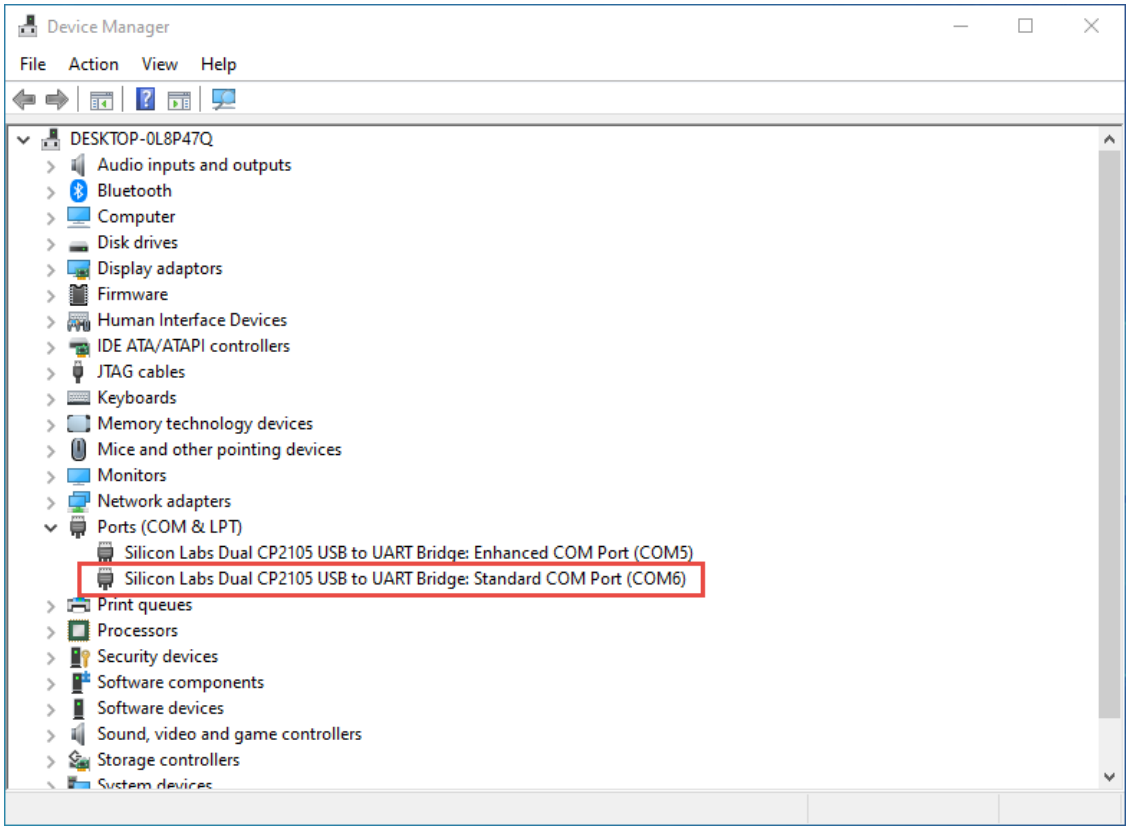


Figure 10-5 The USB to UART device after driver is installed successfully

10.2 Run Dashboard GUI

■ Dashboard GUI software location

Users can find it from the path: Tool\dashboard_gui\Dashboard.exe in the DE10-Agilex system CD and copy it to the Host.

Execute the Dashboard.exe, a window will show as **Figure 10-6**. It will describe the detail functions as below.

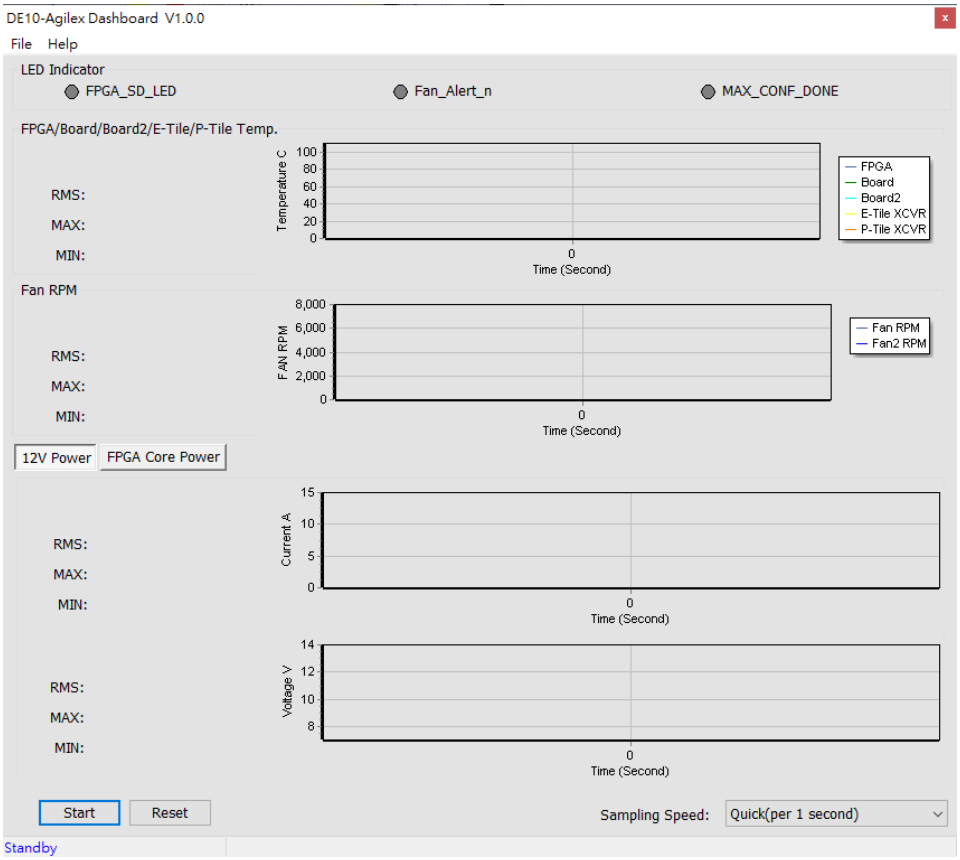


Figure 10-6 Dashboard GUI

■ Dashboard GUI function introduction

- **Start/Stop**: As shown in **Figure 10-7**, there is a Start button at the bottom-left of the GUI window. Click it to run the program (Start will change to Stop), it will show the DE10-Agilex board status. Users can press Stop button to stop the status data transmission and display.
- **Reset Button**: Press this button to clear the historical data shown in GUI, and record the data again.

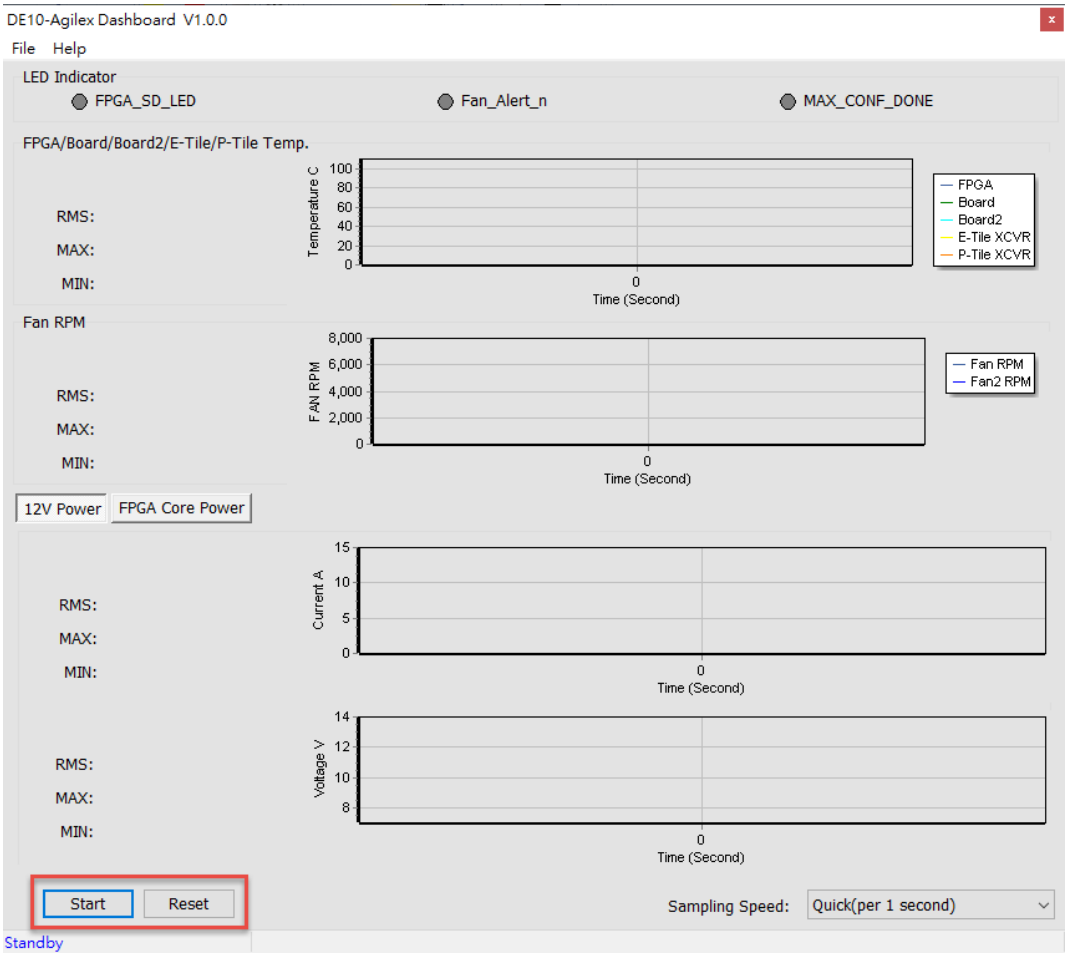


Figure 10-7 Start and Reset button

- **FPGA Status**: As shown in **Figure 10-8**, it will show the status LED number on the DE10-Agilex board. The definitions of these indicator LEDs are as follows:

  - **FPGA_SD_LED**

    When this status is shown in green on the GUI, it means that the FPGA temperature or the board temperature exceeds 95 degrees or the power consumption exceeds 180W. All the power of the FPGA will be cut off.

  - **FPGA_Altert_n**

    When this status is shown in green on the GUI, it means that the fan is abnormal, such as when the fan speed is different from expected

  - **MAX_CONF_DONE**

    Stands for FPGA configure done status. When this status is shown in green

on the GUI, it means that FPGA configuration has been completed.

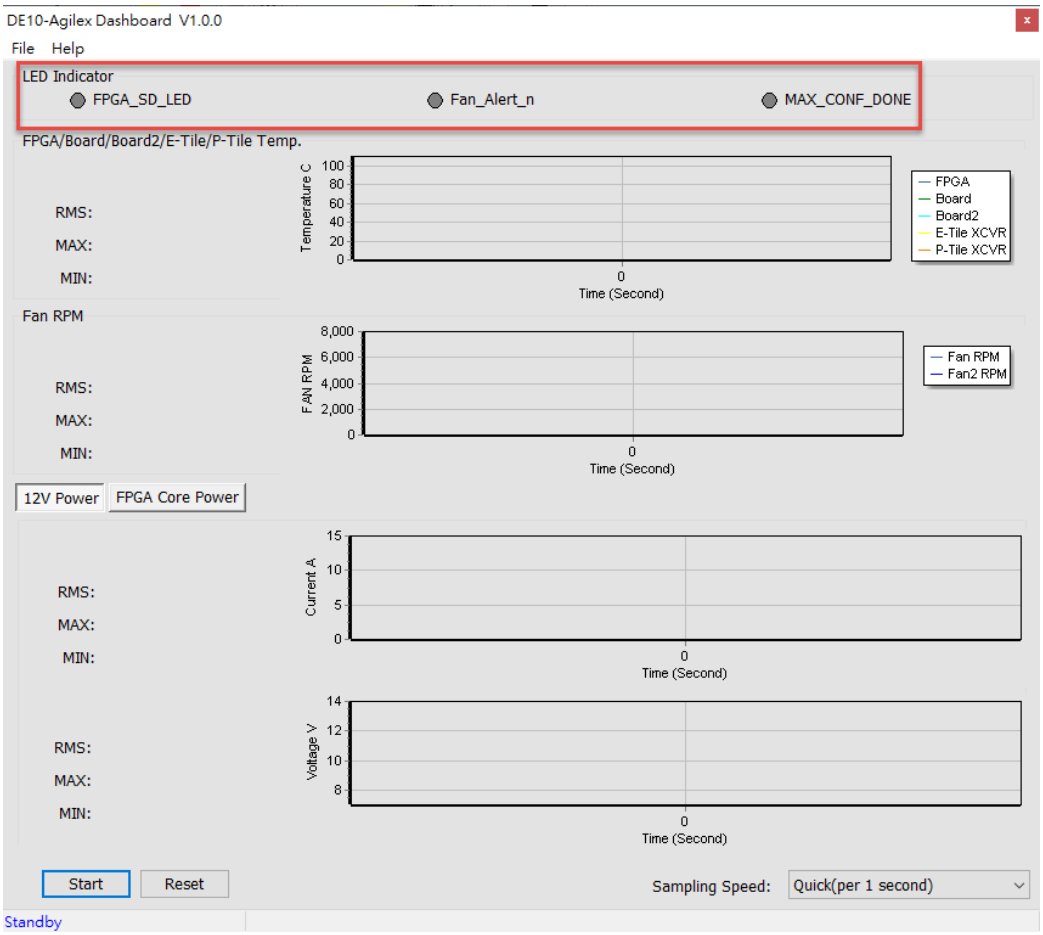


Figure 10-8 FPGA Status section

- **FPGA/Board/Transceiver Temperature**: The Dashboard GUI will real-time show the DE10-Agilex board's ambient temperature (**Board** and **Board2** data in the GUI) and FPGA and FPGA's transceiver (**E-Tile** and **P-Tile**) temperature. Users can know the board's temperature status in time. The information will be refreshed per 1 second, and displays through diagram and number, as shown in **Figure 10-9**. **Figure 10-10** shows the location of the two temperature sensors of **Board** and **Board2** on the GUI.

Figure 10-9 Temperature section

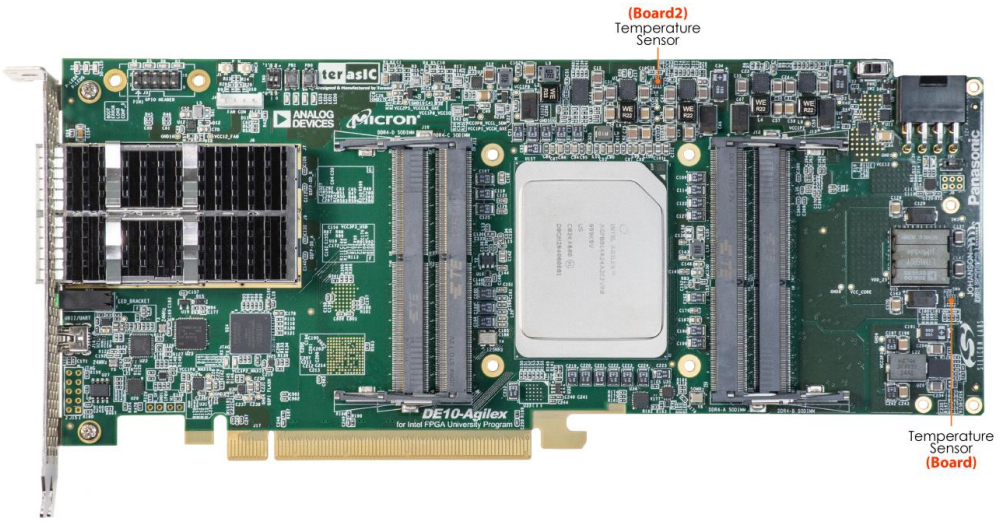


Figure 10-10 Location of the board's ambient temperature

- **Fan RPM**: It displays the real-time speed of the **two** fans (Fan and Fan2 in the GUI) on the DE10-Agilex board, as shown in **Figure 10-11**.

Figure 10-11 FAN RPM section

- **12V/Core Power monitor**: It displays the real-time 12V/Core Power (0.8V~0.85V) voltage and consumption current on the DE10-Agilex board.

  When the user clicks the "**12V Power**" button (See **Figure 10-12**), the GUI will display the voltage level and current number of 12V Power on the board.

  While user clicking the "**FPGA Core Power**" button (See **Figure 10-13**), the GUI will show the voltage level and current value of the FPGA core power on the board. Please note that the power module will output **two** channels of DC power to the core power part of the Agilex FPGA. Therefore, there will be two voltage and current value displayed on the GUI (See **Figure 10-14**). When the power consumption of the FPGA is very low, the current value of one power channel will

be 0(See **Figure 10-15**). If the FPGA power consumption is increasing, due to the slight difference in PCB impedance of the two power channels, the current of the two power channels may have a little difference instead of the same current value(See **Figure 10-16**).

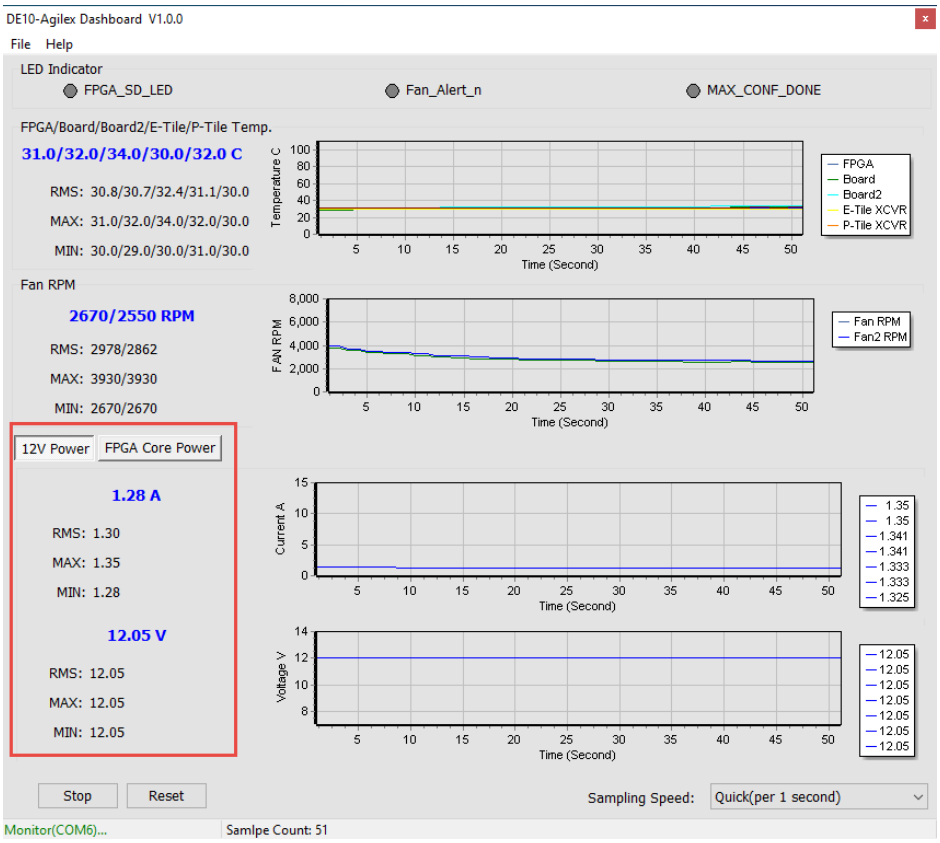


Figure 10-12 Select "12V Power"

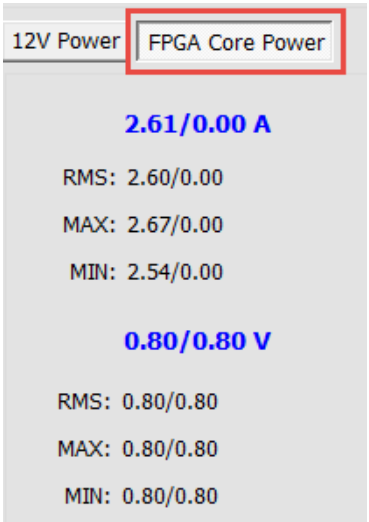


Figure 10-13 Select "FPGA Monitor Section"

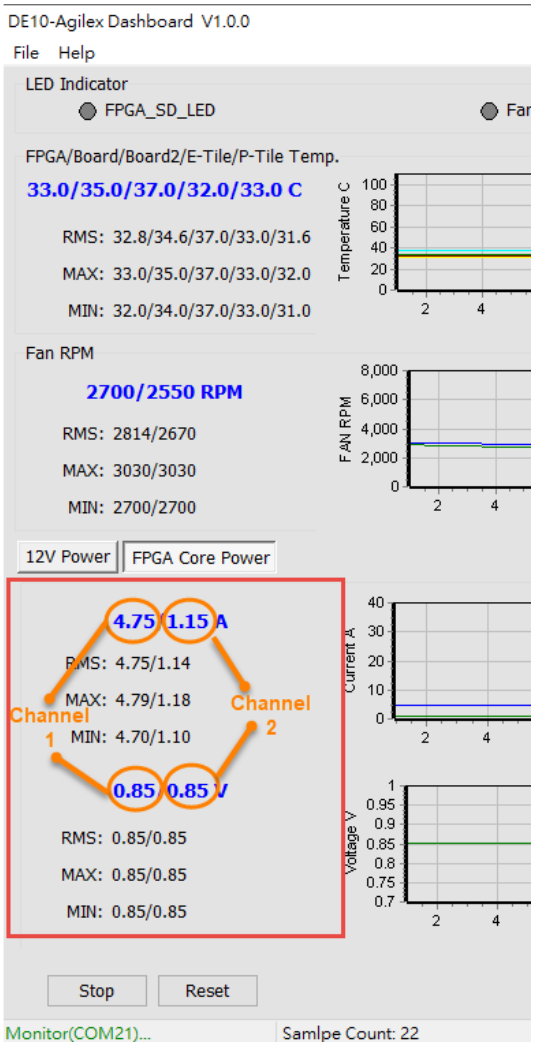


Figure 10-14 Two power channels of the FPGA core power

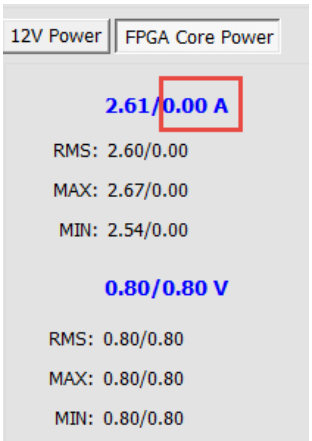


Figure 10-15 One of the core power channel's current is 0

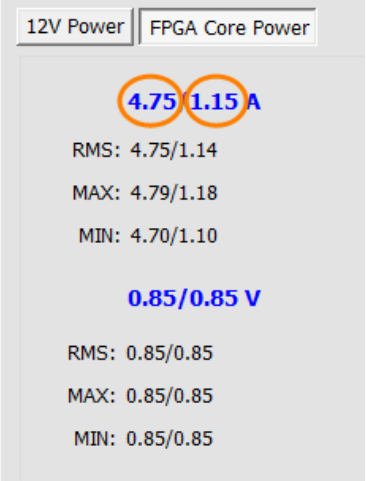


Figure 10-16 Two core power channel's current is different

- **Sampling Speed**: It can change interval time that the Dashboard GUI sample the board status. Users can adjust it to 1s/10s/1min/Full Speed (0.1s) to sample the board status, as shown in **Figure 10-17** and **Figure 10-18**.

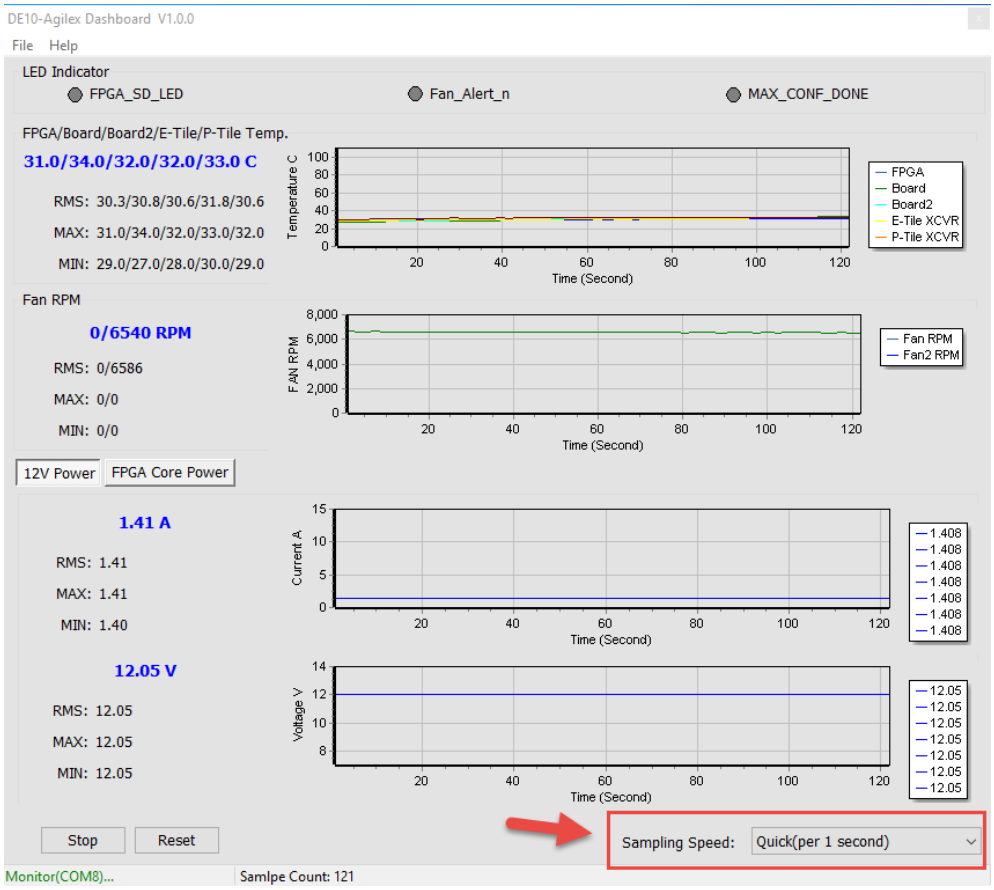


Figure 10-17 Sampling Speed section

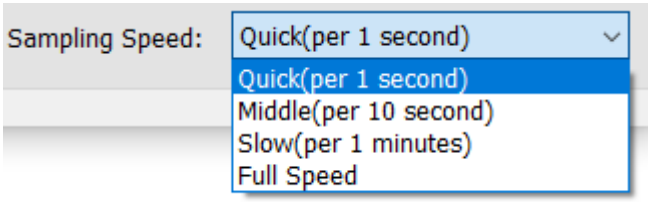


Figure 10-18 Options of Sampling Speed

- **File Manu:** The user can click "**File**" menu at the top left of the GUI (See Figure 10.x) and some options such as board information and status export will appear. Note that to active these functions, you will need to **stop** obtaining the board status (i.e. Don't Press "**Start**" button or Press "**Stop**" button) in the GUI. Detailed introductions of these functions are described in below.

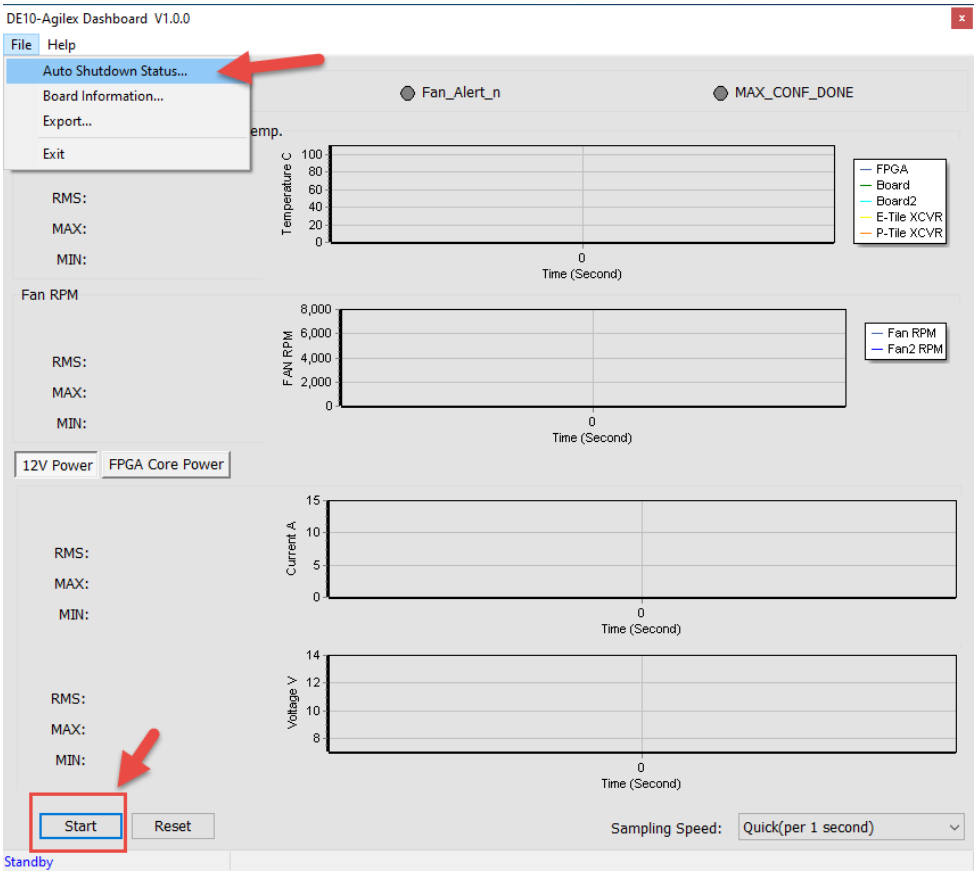


Figure 10-19 Options of Sampling Speed

- **Auto Shutdown Status**: This option will report whether the board entered "Auto shutdown status" because the FPGA temperature is too high or the fan

speed is abnormal.

- **Board Information**: Click the “Board Information” to get the current MAX 10 FPGA software version and the DE10-Agilex board version, as shown in **Figure 10-20**.

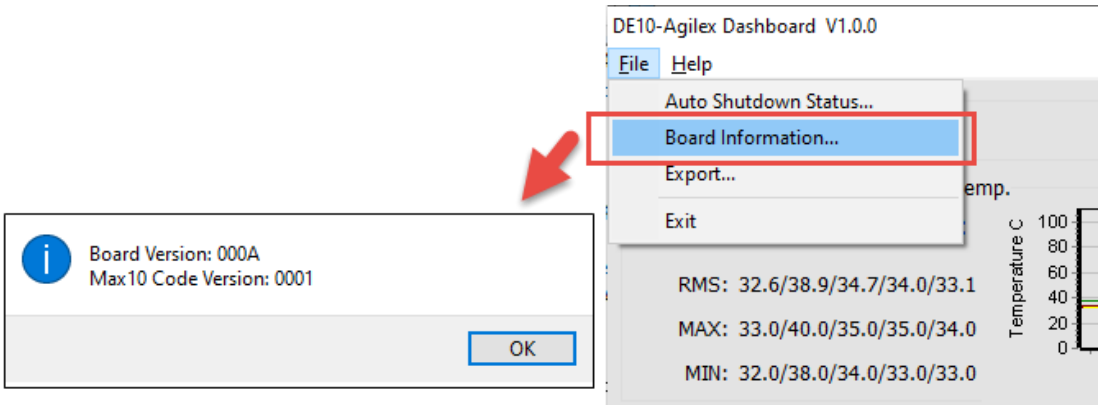


Figure 10-20 Board Information

- **Log File**: Click the Export in the File page to save the board temperature, fan speed and voltage data in .csv format document, as shown in **Figure 10-21** and Figure 10-22.

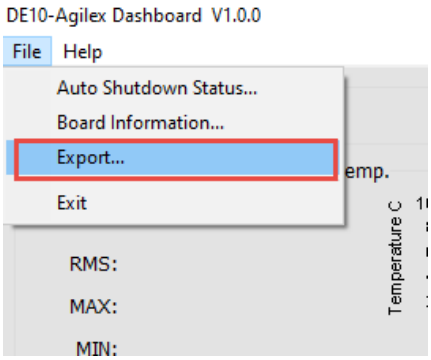


Figure 10-21 Export the log file

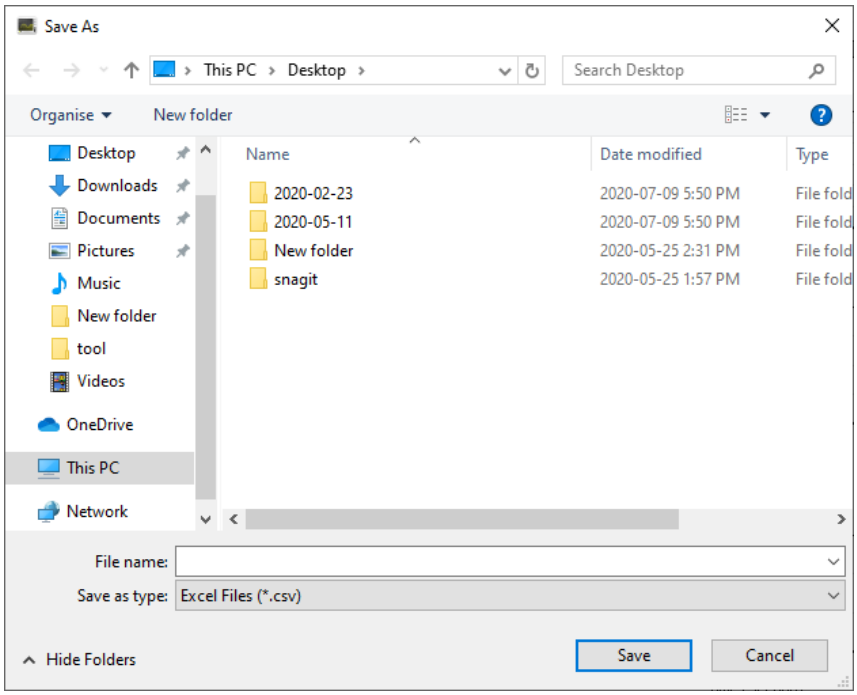


Figure 10-22 Export the log file in .csv format

Chapter 11

Additional Information

11.1 Getting Help

Here are the addresses where you can get help if you encounter problems:

■ Terasic Technologies

No.80, Fenggong Rd., Hukou Township, Hsinchu County 303035. Taiwan
Email: support@terasic.com
Web: www.terasic.com
DE10-Agilex Web: DE10-Agilex.terasic.com

■ Revision History

Date	Version	Changes
2021.08	First publication	
2021.09	V1.1	Modify Figure 1-2
2022.04	V1.2	Modify descriptions of the SW9
2022.12	V1.3	Modify Figure 2-8

■