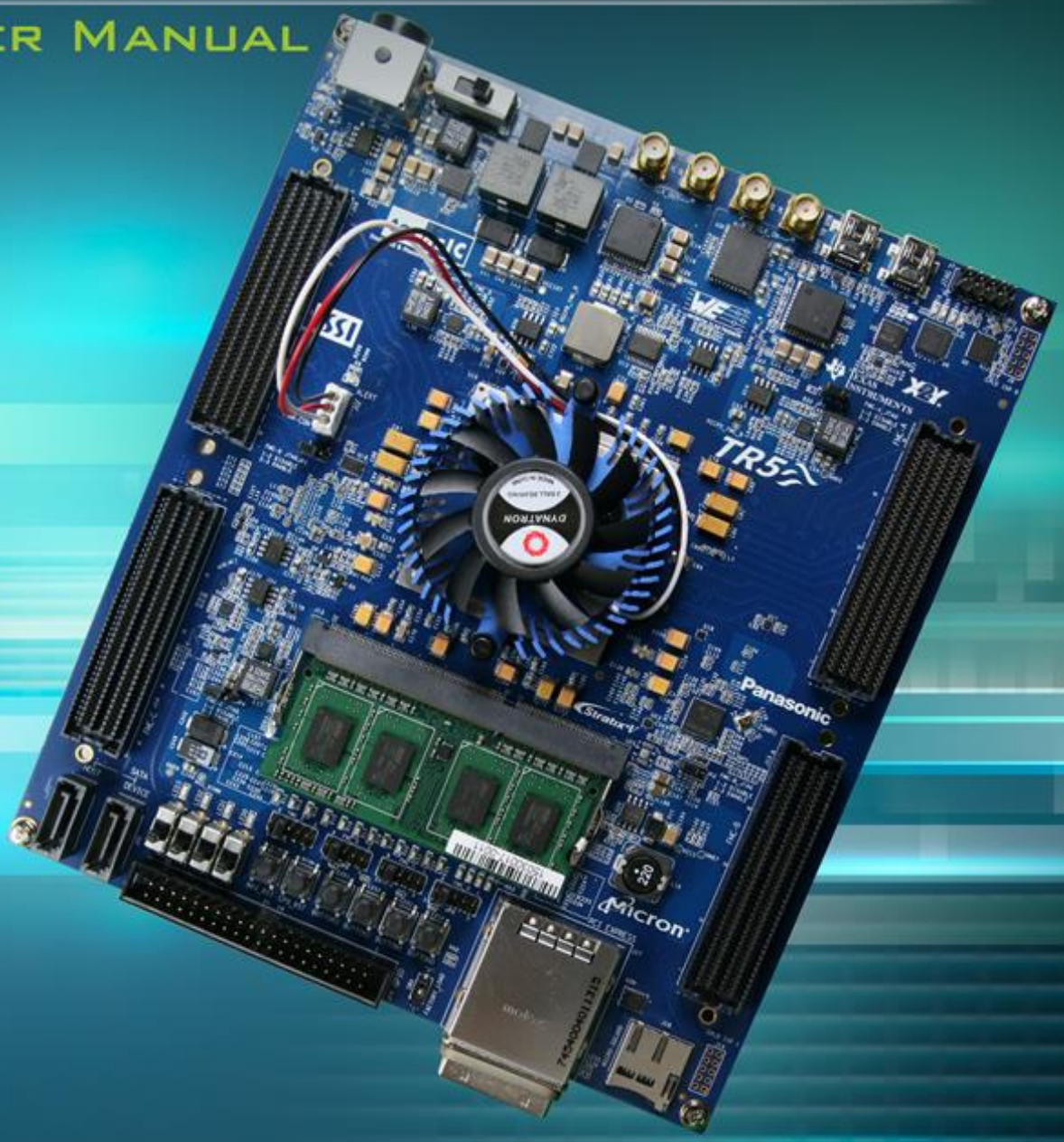
TR5

FPGA Development Kit

User Manual



terasIC
www.terasic.com

CONTENTS

Chapter 1

Overview

This chapter provides an overview of the TR5 Development Board and installation guide.

1.1 General Description

The Terasic TR5 Stratix V GX FPGA Development Kit provides the ideal hardware solution for designs that demand high capacity and bandwidth interface, ultra-low latency communication, high pin count and power efficiency. With an iPass PCIe gen3 connector, the TR5 is designed for the most demanding high-end applications, empowered with the Altera 28 nm Stratix V GX, delivering the best system-level integration and flexibility in the industry.

The Stratix® V GX FPGA features integrated transceivers that transfer at a maximum of 12.5Gbps, this allows the TR5 to be fully compliant with version 3.0 of the PCI Express standard. Not relying on an external PHY will accelerate mainstream development of network applications enabling customers to deploy designs for a broad range of high-speed connectivity applications. For designs that demand high capacity and high speed for memory and storage, the TR5 delivers with one independent bank of DDR3 SO-DIMM RAM, one ZBT SSRAM, and high-speed parallel flash memory. The feature-set of the TR5 fully supports all high-intensity applications such as ASIC verification, data acquisition, and signal processing.

1.2 Key Features

The following hardware is implemented on the TR5 board:

- FPGA
  - Altera Stratix® V GX FPGA
  - 5SGXEA7N2F45C2 /5SGXEABN3F45I3YY

■ FPGA Configuration

- On-Board USB Blaster II or JTAG header for FPGA programming
- Fast passive parallel (FPPx16) configuration via MAX II CPLD and flash memory

■ General user input/output:

- 4 LEDs
- 4 push-buttons
- 4 slide switches

■ Clock System

- 50MHz Oscillator
- CDCM6208 Programmable PLL
- LMK04096B Programmable PLL
- SMA connector pairs for differential clock input and output

■ Memory

- DDR3 SO-DIMM SDRAM
- QDRII+ SRAM
- FLASH
- SD Card

■ Communication Ports

- PCI Express (PCIe) x4 iPass connector
- Serial ATA host and device ports
- PCI Express (PCIe) x8 edge connector
- One mini Uart to USB connector

■ System Monitor and Control

- Temperature sensor
- Fan control
- Power monitor

■ Mechanical Specification

- 4 FPGA Mezzanine Card (FMC) Connectors
- One 40-pin Expansion Header

- Power
  - 12V DC Input

1.3 Block Diagram

Figure 1-1 shows the block diagram of the TR5 board. To provide maximum flexibility for the users, all key components are connected with the Stratix V GX FPGA device. Thus, users can configure the FPGA to implement any system design.

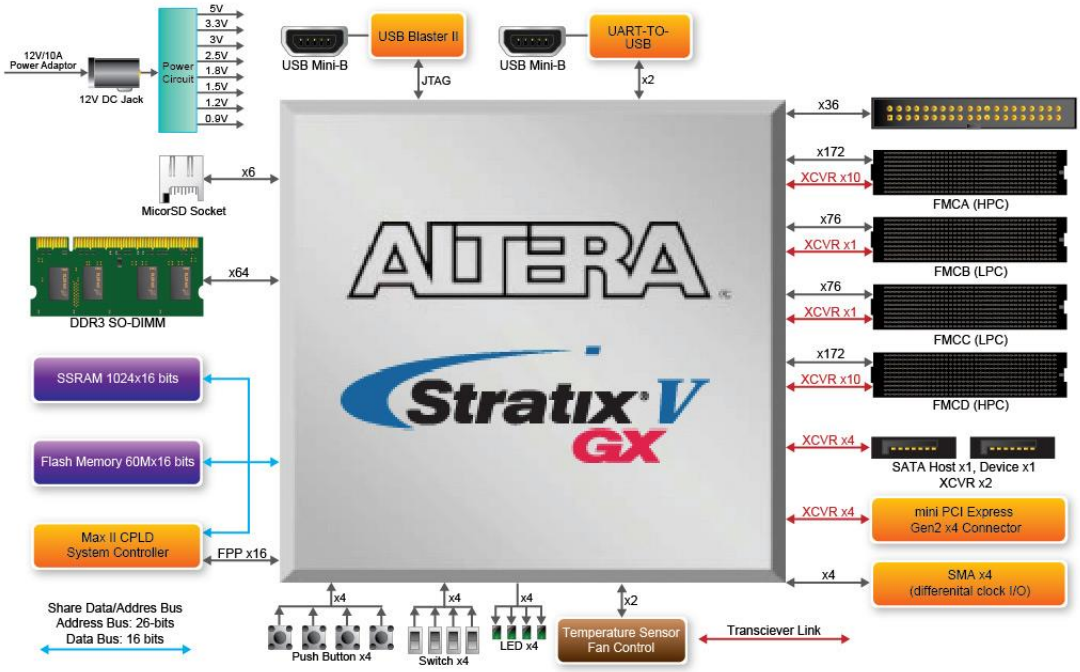


Figure 1-1 Block diagram of the TR5 board

Below is more detailed information regarding the blocks in Figure 1-1.

Stratix V GX FPGA

- 5SGXEA7N2F45C2
  - 622K logic elements (LEs)
  - 57.16-Mbits embedded memory

- 48 transceivers (12.5Gbps)
- 512 18 x18 multipliers
- 256 Variable-precision DSP blocks
- 28 Fractional PLLs and 4DLLs

- 5SGXEABN3F45I3YY
  - 952K logic elements (LEs)
  - 62.96-Mbits embedded memory
  - 48 transceivers (12.5Gbps)
  - 704 18 x18 multipliers
  - 352 Variable-precision DSP blocks
  - 28 Fractional PLLs and 4DLLs

JTAG Header and FPGA Configuration

- On-board USB Blaster II or JTAG header for use with the Quartus II Programmer
- MAXII CPLD EPM2210 System Controller and Fast Passive Parallel (FPP) configuration

Memory devices

- 2MB ZBT SSRAM
- Up to 8GB DDR3 SO-DIMM SDRAM
- 256MB FLASH

General user I/O

- 4 user controllable LEDs
- 4 user push buttons
- 4 user slide switches

On-Board Clock

- 50MHz oscillator
- Programming PLL providing clock for FMC transceivers
- Programming PLL providing clock for PCIe transceiver
- Programming PLL providing clocks for DDR3 SDRAM

Two Serial ATA ports

- SATA 3.0 standard at 6Gbps signaling rate

Four FMC Connectors

- 2 HPC (high-pin count) FMC connectors up to 172 x2 Single-end I/O
- 2 LPC (low-pin count) FMC connectors up to 76 x2 Single-end I/O
- 10 Transceiver Channels for HPC and 1 Transceiver Channel for LPC
- FMC VITA 57.1 Compliant
- Adjustable VADJ: 1.2V/1.5V/1.8V/2.5V/3.0V
- Don't support bidirectional LVDS due to Stratix V device only support single directional LVDS

One 40-pin GPIO Expansion Header

- 36 FPGA I/O pins; 4 power and ground lines
- I/O standards: 3.3V (with level shift from 2.5V to 3.3V)

External PCI Express x4 iPass Connector

- Support for PCIe x4 Gen1/2/3
- iPass connector with x4 PCI Express slot

Power Source

- DC 12V power adapter

Chapter 2

Board Components

This chapter introduces all the important components on the TR5.

2.1 Board Overview

Figure 2-1 is the top and bottom view of the TR5 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.

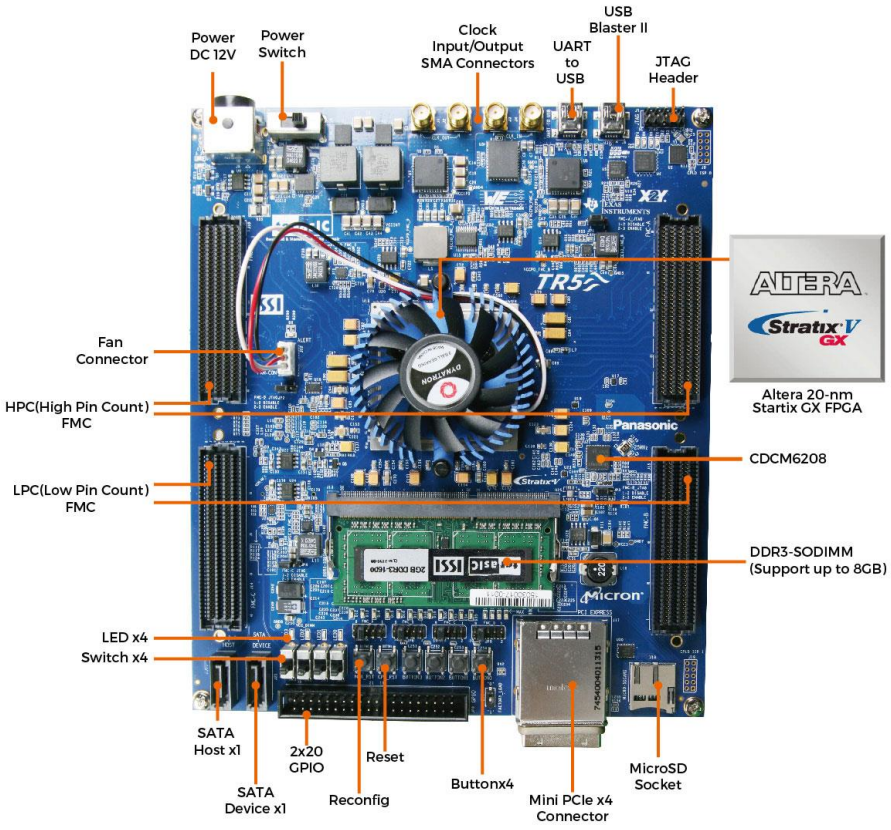


Figure 2-1 FPGA Board (Top)

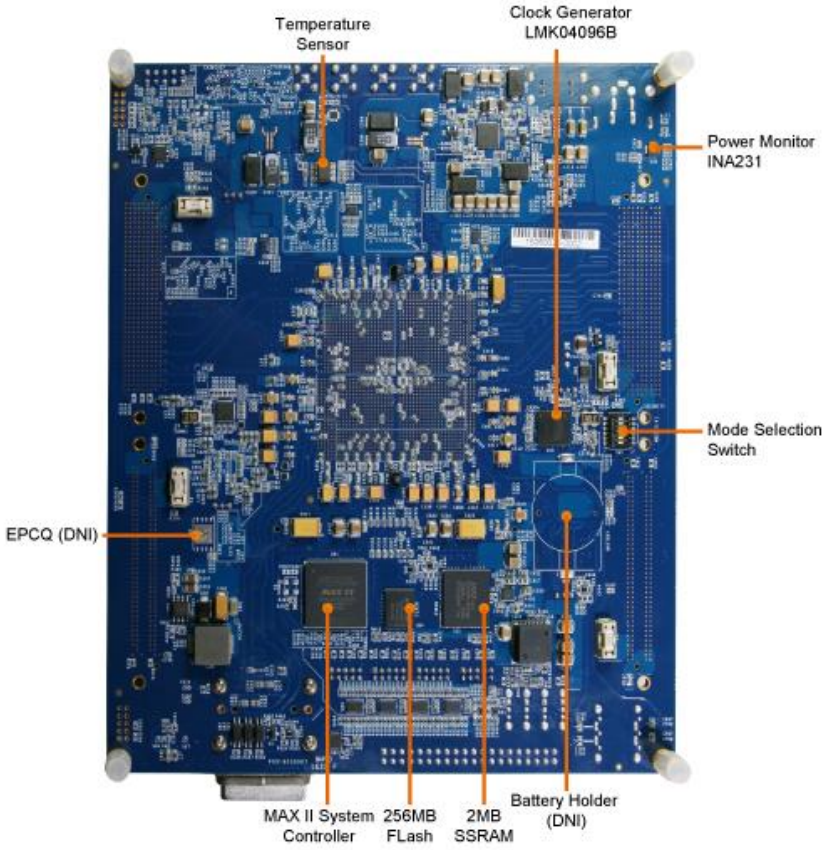


Figure 2-2 FPGA Board (Bottom)

2.2 Configuration, Status and Setup

■ Configure

The FPGA board supports two configuration methods for the Stratix V FPGA:

- Configure the FPGA using the on-board USB-Blaster II.
- Flash memory configuration of the FPGA using stored images from the flash memory on power-up.

For programming by on-board USB-Blaster II, the following procedures show how to download a configuration bit stream into the Stratix V GX 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 II programmer and make sure the USB-Blaster II is detected.
- In Quartus II Programmer, add the configuration bit stream file (.sof), check the associated "Program/Configure" item, and click "Start" to start FPGA programming.

■ Status LED

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

Table 2-1 Status LED

Board Reference	*LED Name*	*Description*
D6	12-V Power	Illuminates when 12-V power is active.
D1	3.3-V Power	Illuminates when 3.3-V power is active.
D21	CONF_DONE	Illuminates when the FPGA is successfully configured. Driven by the MAX II CPLD EPM2210 System Controller.
D22	LOAD	Illuminates when the MAX II CPLD EPM2210 System Controller is actively configuring the FPGA. Driven by the MAX II CPLD EPM2210 System Controller with the Embedded Blaster CPLD.
D23	ERROR	Illuminates when the MAX II CPLD EPM2210 System Controller fails to configure the FPGA. Driven by the MAX II CPLD EPM2210 System Controller.
D24	BOOT_PAGE	Illuminates when FPGA is configured by the factory configuration bit stream.
D12~D20,D33~D35	FMC Voltage Value Indicator	See Section 2.8 FMC Connectors

■ Setup Configure Mode Control DIP switch

The Configure Mode Control DIP switch (SW5) is provided to specify the configuration mode of the FPGA. As currently only one mode is supported, please set all positions as shown in Figure 2-3.

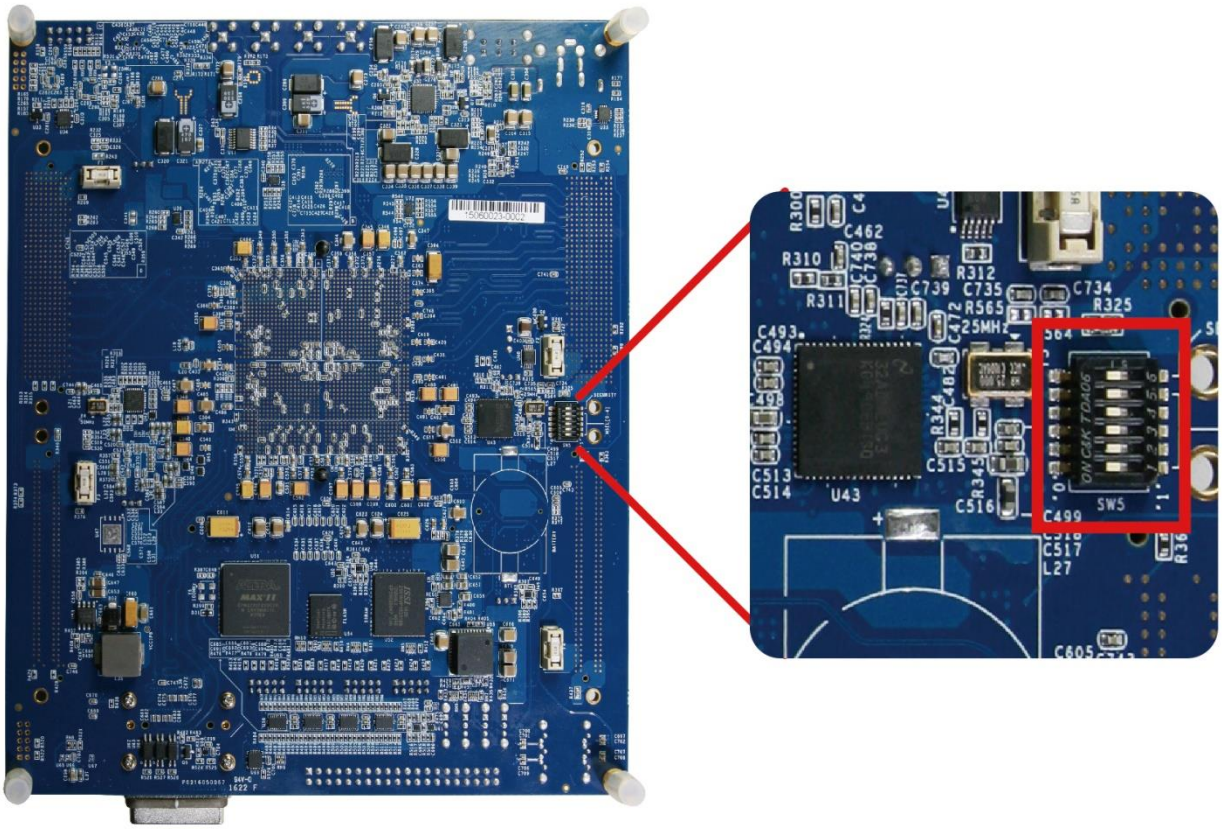


Figure 2-3 4-Position DIP switch for Configure Mode

■ Select Flash Image for Configuration

The Image Select DIP switch (SW4) is provided to specify the image for configuration of the FPGA. Setting SW4 to high (‘0’) specifies the default factory image to be loaded, setting SW4 to low (‘1’) specifies the TR5 to load a user-defined image, as shown in Figure 2-4.

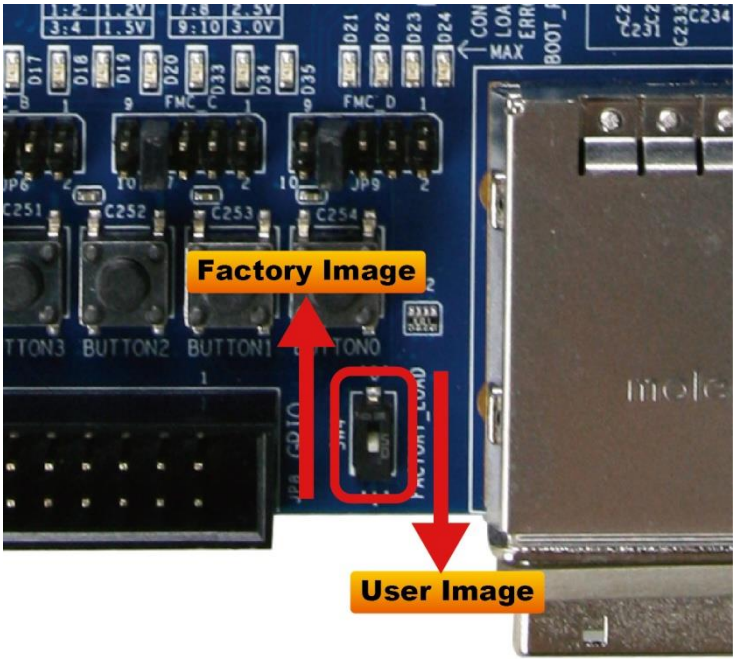


Figure 2-4 2-position DIP switch for Image Select

■ FMC VCCIO Voltage Setting Header

The I/O voltage of all the four FMC connectors is adjustable within 1.2/1.5/1.8/2.5/3.0V. For example, user can adjust the I/O voltage to 2.5V to support LVDS differential I/O stand. The user can independently control the voltage of FMCA~FMCD through JP5, JP6, JP7 and JP9. As shown in **Figure 2-5**, make short circuit onto JP5 pin 7 and pin 8, the status of D12, D13 and D14 will be set as "ON/OFF/ON" for representing the FMCA VCCIO is 2.5V. **Table 2-2, Table 2-3, Table 2-4** and **Table 2-5** lists the voltage settings of the FMCA~FMCD VCCIO and their corresponding LED display status.

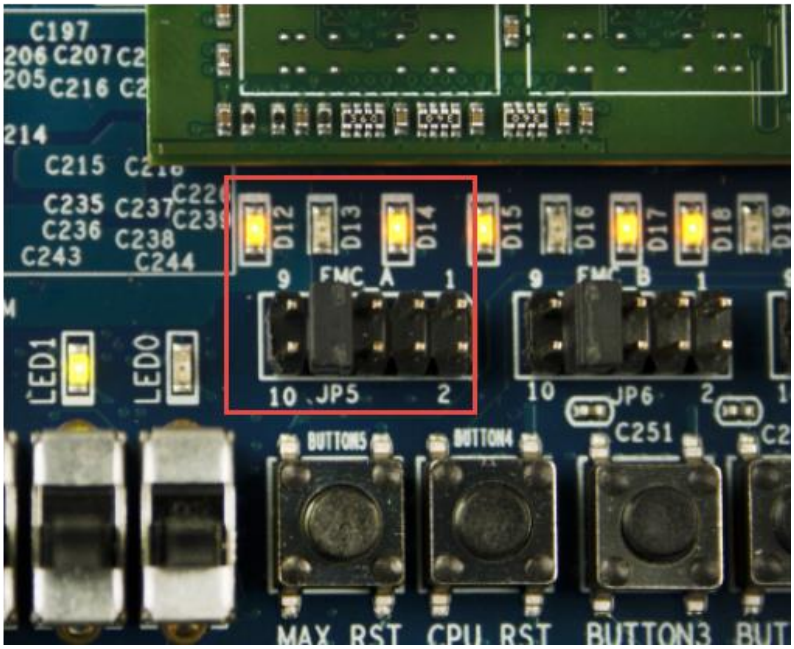


Figure 2-5 FMC A VCCIO Voltage Setting to 2.5V

Table 2-2 FMC A VCCIO Voltage Setting

JP5 Setting	*LED Status*			*FMC A VCCIO Voltage*
	D12	*D13*	*D14*	
Short Pin 1 & 2	**OFF**	**OFF**	**ON**	**1.2V**
Short Pin 3 & 4	**OFF**	**ON**	**OFF**	**1.5V**
Short Pin 5 & 6	**OFF**	**ON**	**ON**	**1.8V**
Short Pin 7 & 8	**ON**	**OFF**	**ON**	**2.5V(Default)**
Short Pin 9 & 10	**ON**	**ON**	**OFF**	**3.0V**

Table 2-3 FMC B VCCIO Voltage Setting

JP6 Setting	*LED Status*			*FMC B VCCIO Voltage*
	D15	*D16*	*D17*	
Short Pin 1 & 2	**OFF**	**OFF**	**ON**	**1.2V**
Short Pin 3 & 4	**OFF**	**ON**	**OFF**	**1.5V**
Short Pin 5 & 6	**OFF**	**ON**	**ON**	**1.8V**
Short Pin 7 & 8	**ON**	**OFF**	**ON**	**2.5V(Default)**
Short Pin 9 & 10	**ON**	**ON**	**OFF**	**3.0V**

Table 2-4 FMC C VCCIO Voltage Setting

JP7 Setting	LED Status			FMC C VCCIO Voltage
	D18	D19	D20	
Short Pin 1 & 2	OFF	OFF	ON	1.2V
Short Pin 3 & 4	OFF	ON	OFF	1.5V
Short Pin 5 & 6	OFF	ON	ON	1.8V
Short Pin 7 & 8	ON	OFF	ON	2.5V(Default)
Short Pin 9 & 10	ON	ON	OFF	3.0V

Table 2-5 FMC D VCCIO Voltage Setting

JP9 Setting	LED Status			FMC D VCCIO Voltage
	D33	D34	D35	
Short Pin 1 & 2	OFF	OFF	ON	1.2V
Short Pin 3 & 4	OFF	ON	OFF	1.5V
Short Pin 5 & 6	OFF	ON	ON	1.8V
Short Pin 7 & 8	ON	OFF	ON	2.5V(Default)
Short Pin 9 & 10	ON	ON	OFF	3.0V

■ FMC JTAG Header

The TR5 supports individual JTAG interfaces on each FMC connector. This feature allows users to extend the JTAG chain to FMC daughter cards. The JTAG signals on each FMC connector can be removed or included in the active JTAG chain via 3-Pin header (See **Figure 2-6**). **Table 2-6** lists the setting of the headers and their associated interfaces. Note that if the JTAG interface on FMC connector is enabled, make sure that the active JTAG chain must be a closed loop or the FPGA may not be detected.

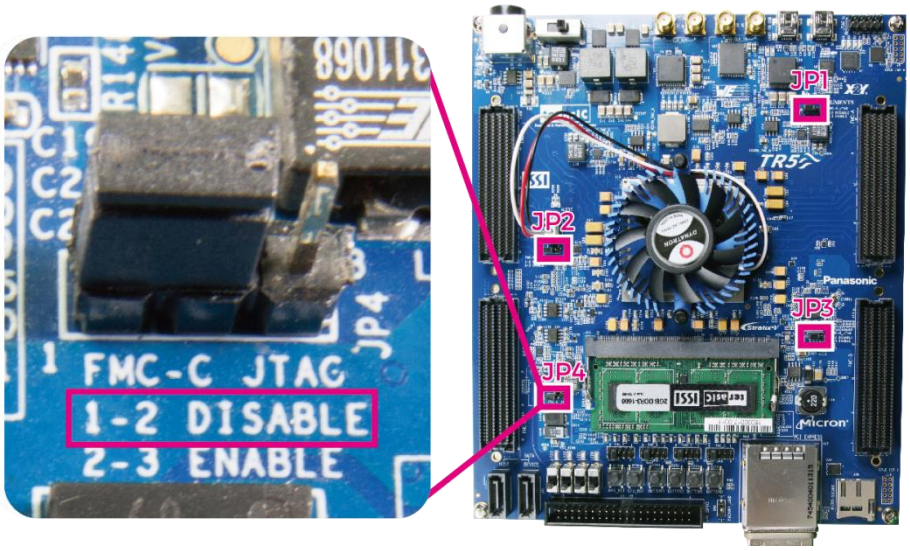


Figure 2-6 FMCC JTAG Header

Table 2-6 FMC JTAG Header Setting

Headers	*Setting*	*Description*
JP1	Short Pin 1 & 2	Disable FMCA JTAG
	Short Pin 2 & 3	Enable FMCA JTAG
JP2	Short Pin 1 & 2	Disable FMCD JTAG
	Short Pin 2 & 3	Enable FMCD JTAG
JP3	Short Pin 1 & 2	Disable FMCB JTAG
	Short Pin 2 & 3	Enable FMCB JTAG
JP4	Short Pin 1 & 2	Disable FMCC JTAG
	Short Pin 2 & 3	Enable FMCC JTAG

2.3 General User Input/Output

This section describes the user I/O interface to the FPGA.

■ User Defined Push-buttons

The FPGA board includes four user defined push-buttons that allow users to interact with the Stratix V GX device. Each push-button provides a high logic level or a low logic level when it is not pressed

or pressed, respectively. Table 2-7 lists the board references, signal names and their corresponding Stratix V GX device pin numbers.

Table 2-7 Push-button Pin Assignments, Schematic Signal Names, and Functions

Board Reference	*Schematic Signal Name*	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
BUTTON0	BUTTON0	High Logic Level when the button is not pressed	1.5-V	PIN_BC7
BUTTON1	BUTTON1		1.5-V	PIN_BD7
BUTTON 2	BUTTON2		1.5-V	PIN_BB8
BUTTON 3	BUTTON3		1.5-V	PIN_BB9

■ User-Defined Slide Switch

There are four slide switches on the FPGA board to provide additional FPGA input control. When a slide switch is in the DOWN position or the UPPER position, it provides a low logic level or a high logic level to the Stratix V GX FPGA. The down position provides a low logic level and the upper position provides a high logic level.

Table 2-8 lists the signal names and their corresponding Stratix V GX device pin numbers.

Table 2-8 Slide Switch Pin Assignments, Schematic Signal Names, and Functions

Board Reference	*Schematic Signal Name*	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
SW0	SW0	High logic level when SW in the UPPER position.	1.5-V	PIN_AT9
SW1	SW1		1.5-V	PIN_AU8
SW2	SW2		1.5-V	PIN_AK9
SW3	SW3		1.5-V	PIN_AL9

■ User-Defined LEDs

The FPGA board consists of four user-controllable LEDs to allow status and debugging signals to be driven to the LEDs from the designs loaded into the Stratix V GX device. Each LED is driven directly by the Stratix V GX FPGA. The LEDs are 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 2-9.

Table 2-9 User LEDs Pin Assignments, Schematic Signal Names, and Functions

Board Reference	*Schematic Signal Name*	*Description*	*I/O Standard*	*Stratix V GX 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.	1.5-V	PIN_AT32
LED1	LED1		1.5-V	PIN_BA31
LED2	LED2		1.5-V	PIN_AN27
LED3	LED3		1.5-V	PIN_AH27

■ UART-To-USB

The UART is designed to perform communication between the board and the PC, allowing a transmission speed of up to 3Mbps. This interface wouldn't support HW flow control signals. The physical interface is done using UART-USB on-board bridge from a FT232R chip and connects to the host using a USB Type-B connector. For detailed information on how to use the transceiver, please refer to the datasheet, which is available on the manufacturer's website, or under the Datasheets\FT232 folder on the Kit System CD. **Figure 2-7** shows the related schematics, and **Table 2-10** lists the UART pin assignments, signal names and functions.

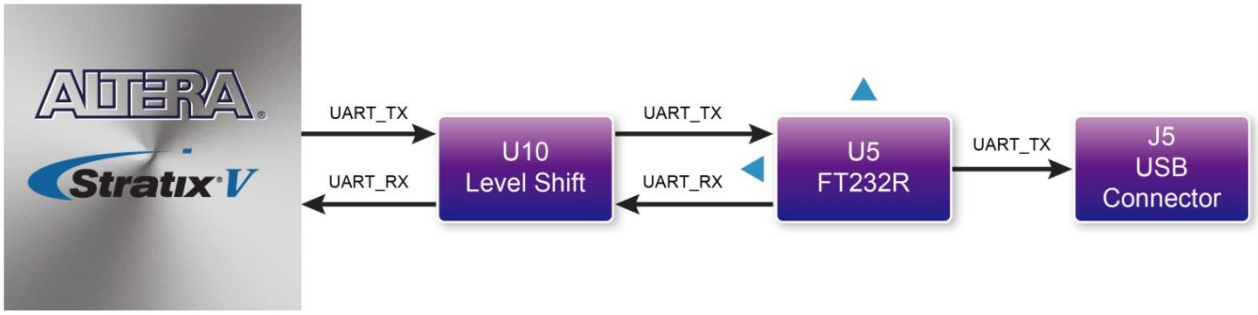


Figure 2-7 Connection between UART-To-USB and Stratix V GX FPGA

Table 2-10 UART-To-USB Pin Assignments, Schematic Signal Names, and Functions

Board Reference	*Schematic Signal Name*	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
UART_TX	UART_TX	Uart TX output from FPGA	1.2/1.5/1.8/2.5/3.0-	PIN_T26

			V	
UART_RX	UART_RX	Uart RX input to FPGA	1.2/1.5/1.8/2.5/3.0-V	PIN_T25

■ Micro SD-Card

The development board supports Micro SD card interface using 4 data lines. **Figure 2-8** shows the related signals connections between the SD Card and Stratix V GX FPGA. **Table 2-11** lists all the associated pins

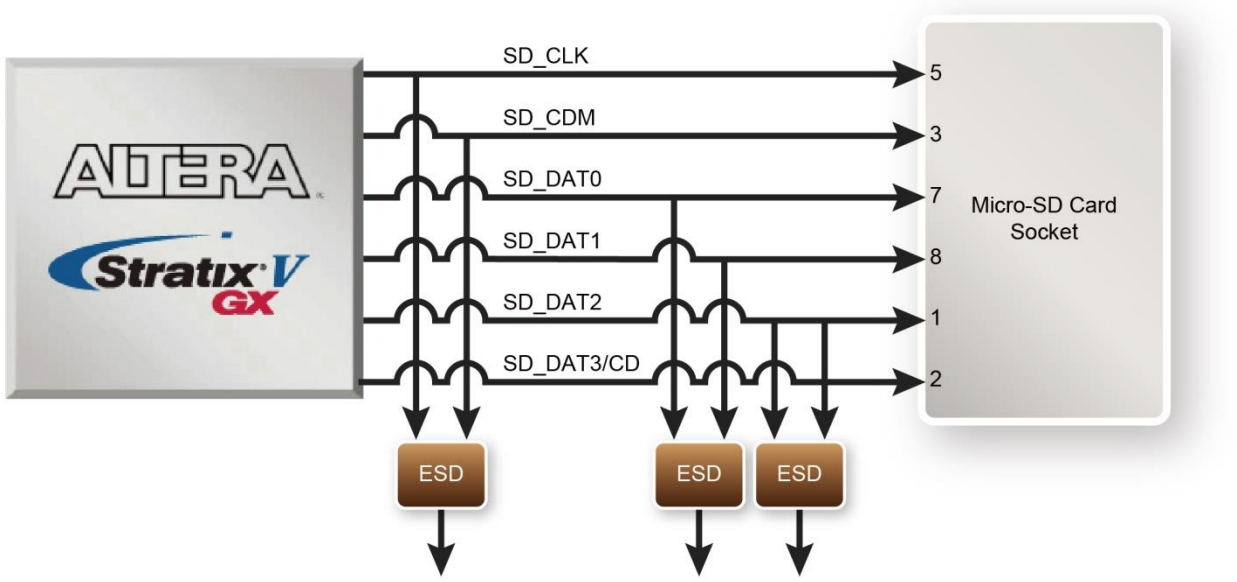


Figure 2-8 Connection between the SD Card Socket and Stratix V GX FPGA

Table 2-11 Micro SD Card Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
SD_CLK	Serial Clock	1.5-V	PIN_BB39
SD_CMD	Command, Response	1.5-V	PIN_BA36
SD_DAT0	Serial Data 0	1.5-V	PIN_AV37
SD_DAT1	Serial Data 1	1.5-V	PIN_AY37

SD_DAT2	Serial Data 2	1.5-V	PIN_BB36
SD_DAT3	Serial Data 3	1.5-V	PIN_AW37

2.4 Temperature Sensor, Fan Control and Power Monitor

The FPGA board is equipped with a temperature sensor, MAX1619, which provides temperature sensing and over-temperature alert. These functions are accomplished by connecting the temperature sensor to the internal temperature sensing diode of the Stratix V GX device. The temperature status and alarm threshold registers of the temperature sensor can be programmed by a two-wire SMBus, which is connected to the Stratix V GX FPGA. In addition, the 7-bit POR slave address for this sensor is set to '0011000b'.

A 3-pin +12V fan located on J12 of the FPGA board is intended to reduce the temperature of the FPGA. The board is equipped with a Fan-Speed regulator and monitor MAX6650 with an I2C/SMBus interfaces, Users regulate and monitor the speed of fan depending on the measured system temperature.

The TR5 has implemented a power monitor chip to monitor the board input power voltage and current. **Figure 2-9** shows the connection between the power monitor chip and the Stratix V GX FPGA. The power monitor chip monitors both shunt voltage drops and board input power voltage allows user to monitor the total board power consumption. Programmable calibration value, conversion times, and averaging, combined with an internal multiplier, enable direct readouts of current in amperes and power in watts. Note that, the temperature sensor, fan control and power monitor share the same I2C/SMBUS.

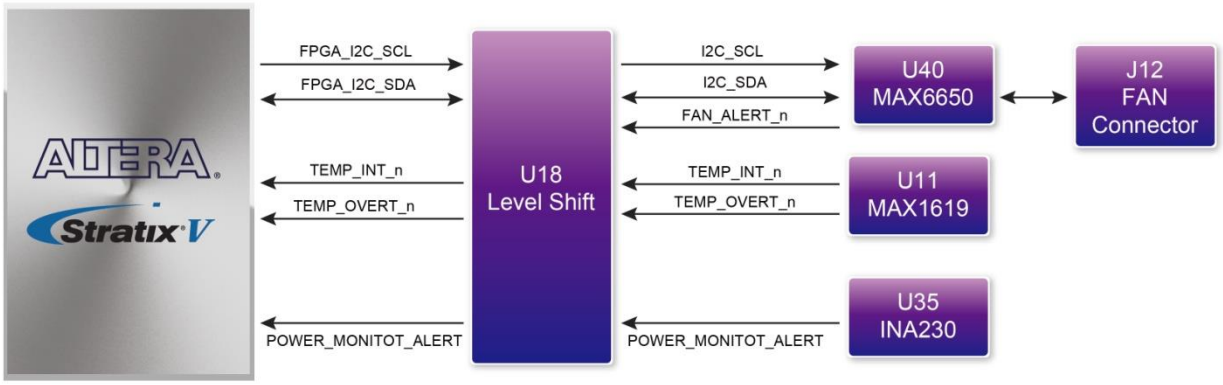


Figure 2-9 Connections between the temperature sensor/fan control/power monitor and the Stratix V GX FPGA

Table 2-12 Temperature Sensor and Fan Speed Control Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
TEMPDIODEp	Positive pin of temperature diode in Stratix V	-	PIN_P6
TEMPDIODEn	Negative pin of temperature diode in Stratix V	-	PIN_P7
FPGA_I2C_SCL	SMBus clock	1.5-V	PIN_AN11
FPGA_I2C_SDA	SMBus data	1.5-V	PIN_AP9
TEMP_OVERT_n	SMBus alert (interrupt)	1.5-V	PIN_AR9
TEMP_INT_n	SMBus alert (interrupt)	1.5-V	PIN_AT8
POWER_MONITOR_ALERT	Active-high ALERT input	1.5-V	PIN_AY9
FAN_ALERT_n	Active-low ALERT input	1.5-V	PIN_AM11

2.5 Clock Circuit

The development board includes one 50 MHz and two programmable clock generators. **Figure 2-10** shows the default frequencies of on-board all external clocks going to the Stratix V GX FPGA.

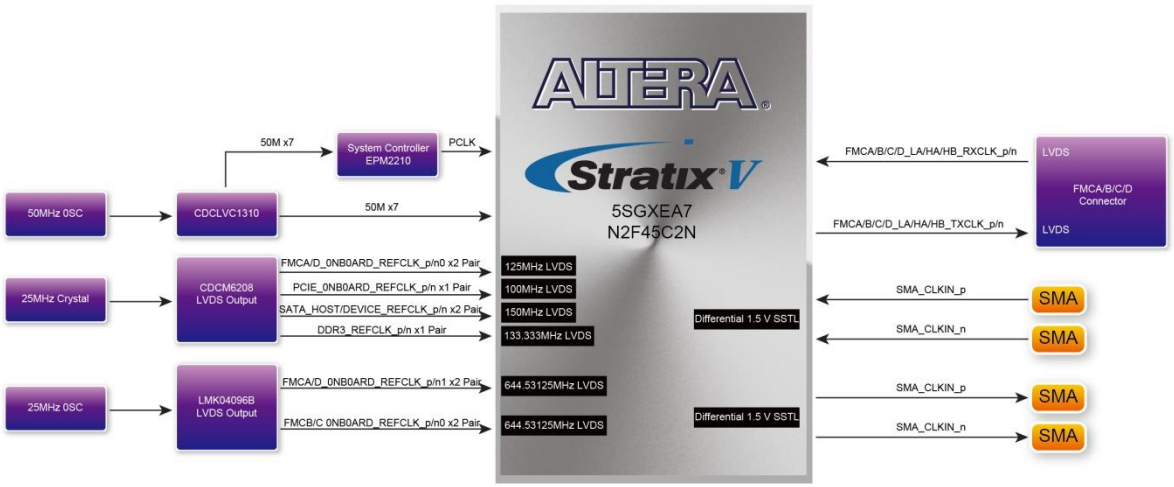


Figure 2-10 Clock circuit of the FPGA Board

A clock buffer is used to duplicate the 50 MHz oscillator, so there are seven 50MHz clocks fed into seven different FPGA banks. The two programming clock generators are low-jitter oscillators which are used to provide special and high quality clock signals for high-speed transceivers and high bandwidth memory. Through I2C serial interface, the clock generator controllers in the Stratix V GX FPGA can be used to program the CDCM6208 and LMK04096B to generate PCIe, SATA and high bandwidth memory reference clocks respectively. Two SMA connectors and Four FMC connectors provide external differential clock input(s) and clock output(s) respectively.

Table 2-9 lists the clock source, signal names, default frequency and their corresponding Stratix V GX device pin numbers.

Table 2-9 Clock Source, Signal Name, Default Frequency, Pin Assignments and Functions

Source	*Schematic Signal Name*	*Default Frequency*	*I/O Standard*	*Stratix V GX Pin Number*	*Application*
Y1	CLK_50_B3B	50.0 MHz	1.5-V	PIN_AW35	
	CLK_50_B4A		1.5-V	PIN_AP10	
	CLK_50_B4D		1.2/1.5/1.8/2.5/3.0-V	PIN_AY18	
	CLK_50_B7A		1.2/1.5/1.8/2.5/3.0-V	PIN_M8	

	CLK_50_B7D		1.2/1.5/1.8/2.5/3.0-V	PIN_J18	
	CLK_50_B8A		1.2/1.5/1.8/2.5/3.0-V	PIN_R36	
	CLK_50_B8D		1.2/1.5/1.8/2.5/3.0-V	PIN_R25	
J3	SMA_CLKIN_p	User Defined	1.5-V	PIN_BC8	External Clock Input
J4	SMA_CLKIN_n	User Defined	1.5-V	PIN_BD8	Clock Output
J1	SMA_CLKOUT_p	User Defined	1.5-V	PIN_AV8	
J2	SMA_CLKOUT_n	User Defined	1.5-V	PIN_AW9	
U21	FMCA_ONBOARD_REFCLK_p0	125 MHz	LVDS	PIN_Y38	FMCA port xcvr reference clock
	FMCD_ONBOARD_REFCLK_p0	125 MHz	LVDS	PIN_Y7	FMCD port xcvr reference clock
	PCIE_ONBOARD_REFCLK_p	100 MHz	LVDS	PIN_AH39	PCIe reference clock
	SATA_DEVICE_REFCLK_p	150 MHz	LVDS	PIN_AK7	SATA Device reference clock
	SATA_HOST_REFCLK_p	150 MHz		PIN_BB33	SATA Host reference clock
	DDR3_REFCLK_p	133.333 MHz			DDR3 reference clock
U43	FMCA_ONBOARD_REFCLK_p1	644.53125 MHz	LVDS	PIN_T38	FMCA port xcvr reference clock
	FMCD_ONBOARD_REFCLK_p1	644.53125 MHz	LVDS	PIN_T7	FMCD port xcvr reference clock
	FMCC_ONBOARD_REFCLK_p0	644.53125 MHz	LVDS	PIN_AD39	FMCB port xcvr reference clock
	FMCC_ONBOARD_REFCLK_p1	644.53125 MHz	LVDS	PIN_AD6	FMCC port xcvr reference clock

Table 2-10 lists the programmable oscillator control pins, signal names, I/O standard and their corresponding Stratix V GX device pin numbers.

Table 2-10 Programmable oscillator control pin, Signal Name, I/O standard, Pin Assignments and Descriptions

Programmable Oscillator	*Schematic Signal Name*	*I/O Standard*	*Stratix V GX Pin Number*	*Description*
CDCM6208 (U21)	CLOCK_SCL	2.5-V	PIN_AR25	I2C bus, connected with CDCM6208
	CLOCK_SDA	2.5-V	PIN_BC25	
LMK04906B (U43)	LMK04906_CLK	2.5-V	PIN_AT24	I2C bus master output only, connected with LMK04096B
	LMK04906_DATAIN	2.5-V	PIN_BD25	
	LMK04906_DATAOUT	1.5-V	PIN_BC29	I2C bus master input signal
	LMK04906_LE	1.5-V	PIN_AT33	LMK04096B PLL locked signal

2.6 FLASH and SSRAM Memory

The development board has a 1G bit CFI-compatible synchronous flash device for non-volatile storage of FPGA configuration data, user application data, and user code space, and a 2M byte ZBT SSRAM for data Cache.

The flash has a 16-bit data bus and allow for FPP x16 configuration. This device is part of the shared flash and MAX (FM) bus, which connects to the flash memory and MAX II CPLD (EPM2210) System Controller. The SSRAM also has a 16-bit data bus and share address and data bus with the flash. **Figure 2-11** shows the connections between the Flash, SSRAM, MAX and Stratix V GX FPGA.

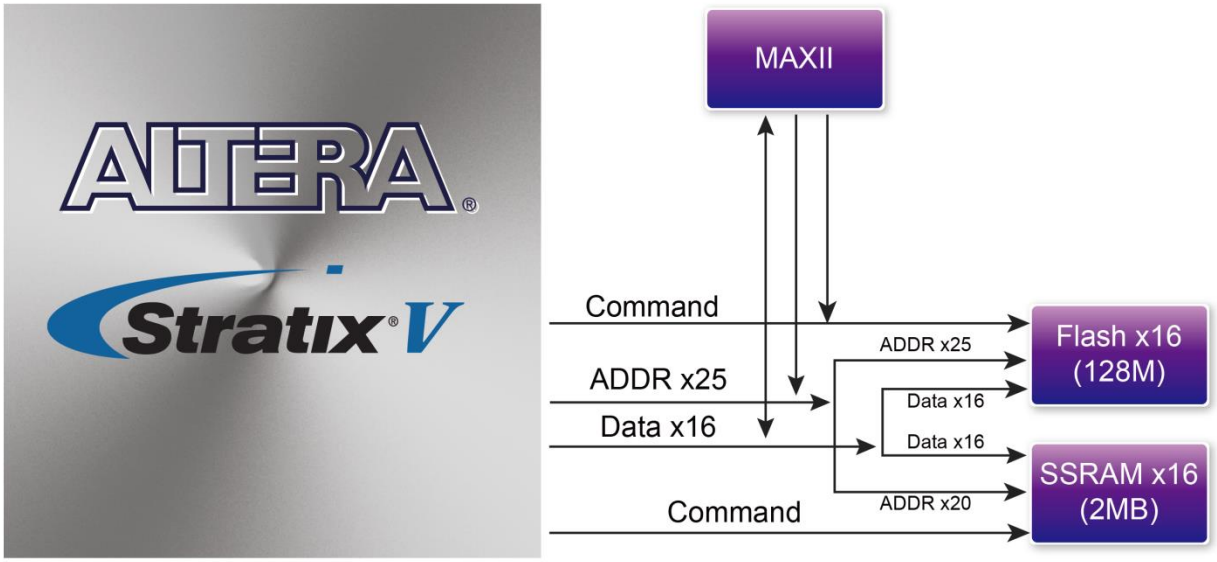


Figure 2-11 Connection between the Flash, Max and Stratix V GX FPGA

Table 2-11 lists the flash pin assignments, signal names, and functions.

Table 2-13 Flash Memory Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
FSM_A1	Address bus	2.5V	PIN_AE11
FSM_A2	Address bus	2.5V	PIN_AD14
FSM_A3	Address bus	2.5V	PIN_AE14
FSM_A4	Address bus	2.5V	PIN_AE10
FSM_A5	Address bus	2.5V	PIN_AF10
FSM_A6	Address bus	2.5V	PIN_AE12
FSM_A7	Address bus	2.5V	PIN_AF11
FSM_A8	Address bus	2.5V	PIN_AG13
FSM_A9	Address bus	2.5V	PIN_AJ10
FSM_A10	Address bus	2.5V	PIN_AF13
FSM_A11	Address bus	2.5V	PIN_AE13
FSM_A12	Address bus	2.5V	PIN_AJ11
FSM_A13	Address bus	2.5V	PIN_BD11
FSM_A14	Address bus	2.5V	PIN_AW10
FSM_A15	Address bus	2.5V	PIN_AF14

FSM_A16	Address bus	2.5V	PIN_AY12
FSM_A17	Address bus	2.5V	PIN_AY10
FSM_A18	Address bus	2.5V	PIN_BD10
FSM_A19	Address bus	2.5V	PIN_BB12
FSM_A20	Address bus	2.5V	PIN_BA12
FSM_A21	Address bus	2.5V	PIN_BA10
FSM_A22	Address bus	2.5V	PIN_BC11
FSM_A23	Address bus	2.5V	PIN_AE9
FSM_A24	Address bus	2.5V	PIN_AW11
FSM_A25	Address bus	2.5V	PIN_BC10
FSM_A26	Address bus	2.5V	PIN_BB11
FSM_D0	Data bus	2.5V	PIN_AG10
FSM_D1	Data bus	2.5V	PIN_AH10
FSM_D2	Data bus	2.5V	PIN_AG11
FSM_D3	Data bus	2.5V	PIN_AK12
FSM_D4	Data bus	2.5V	PIN_AV10
FSM_D5	Data bus	2.5V	PIN_AR12
FSM_D6	Data bus	2.5V	PIN_AL12
FSM_D7	Data bus	2.5V	PIN_AR13
FSM_D8	Data bus	2.5V	PIN_AG9
FSM_D9	Data bus	2.5V	PIN_AH12
FSM_D10	Data bus	2.5V	PIN_AG12
FSM_D11	Data bus	2.5V	PIN_AL11
FSM_D12	Data bus	2.5V	PIN_AN12
FSM_D13	Data bus	2.5V	PIN_AU9
FSM_D14	Data bus	2.5V	PIN_AM13
FSM_D15	Data bus	2.5V	PIN_AJ12
FLASH_CLK	Flash Clock	2.5V	PIN_AU11
FLASH_RESET_n	Flash Reset, active low	2.5V	PIN_AV25
FLASH_CE_n	Flash Chip enable,active low	2.5V	PIN_AU24
FLASH_OE_n	Flash Output enable, active low	2.5V	PIN_AP12
FLASH_WE_n	Flash Write enable, active low	2.5V	PIN_AT12
FLASH_ADV_n	Flash Address valid, active low	2.5V	PIN_BD26
FLASH_RDY_BSY_n	Flash ready output	2.5V	PIN_AU25
SSRAM_CLK	SSRAM Clock	2.5V	PIN_AP13
SSRAM_CKE_n	SSRAM Clock enable, active low	2.5V	PIN_AW24
SSRAM_CE_n	SSRAM Chip enable, active low	2.5V	PIN_AP24

SSRAM_WE_n	Flash Write enable, active low	2.5V	PIN_AV11
SSRAM_OE_n	Flash output enable, active low	2.5V	PIN_AU10
SSRAM_ADV	Flash Address valid, active high	2.5V	PIN_BC26
SSRAM_BWA_n	SSRAM Byte Write enable	2.5V	PIN_AY25
SSRAM_BWB_n	SSRAM Byte Write enable	2.5V	PIN_BA24
FLASH_CLK	Clock	2.5V	PIN_T9
FLASH_RESET_n	Reset	2.5V	PIN_C17
FLASH_CE_n	Chip enable of of flash-0	2.5V	PIN_H10
	Chip enable of of flash-1	2.5V	PIN_N16
FLASH_OE_n	Output enable	2.5V	PIN_C16
FLASH_WE_n	Write enable	2.5V	PIN_U10
FLASH_ADV_n	Address valid	2.5V	PIN_H7
FLASH_RDY_BSY_n	Ready of flash-0	2.5V	PIN_J8

2.7 DDR3 SO-DIMM

The development board supports DDR3 SDRAM SO-DIMM. The DDR3 SODIMM socket is wired to support a maximum capacity of 8GB with a 64-bit data bus. Using differential DQS signaling for the DDR3 SDRAM interfaces, it is capable of running at up to 800MHz memory clock for a maximum theoretical bandwidth up to 95.4Gbps. **Figure 2-12** shows the connections between the DDR3 SDRAM SO-DIMMs and Stratix V GX FPGA.

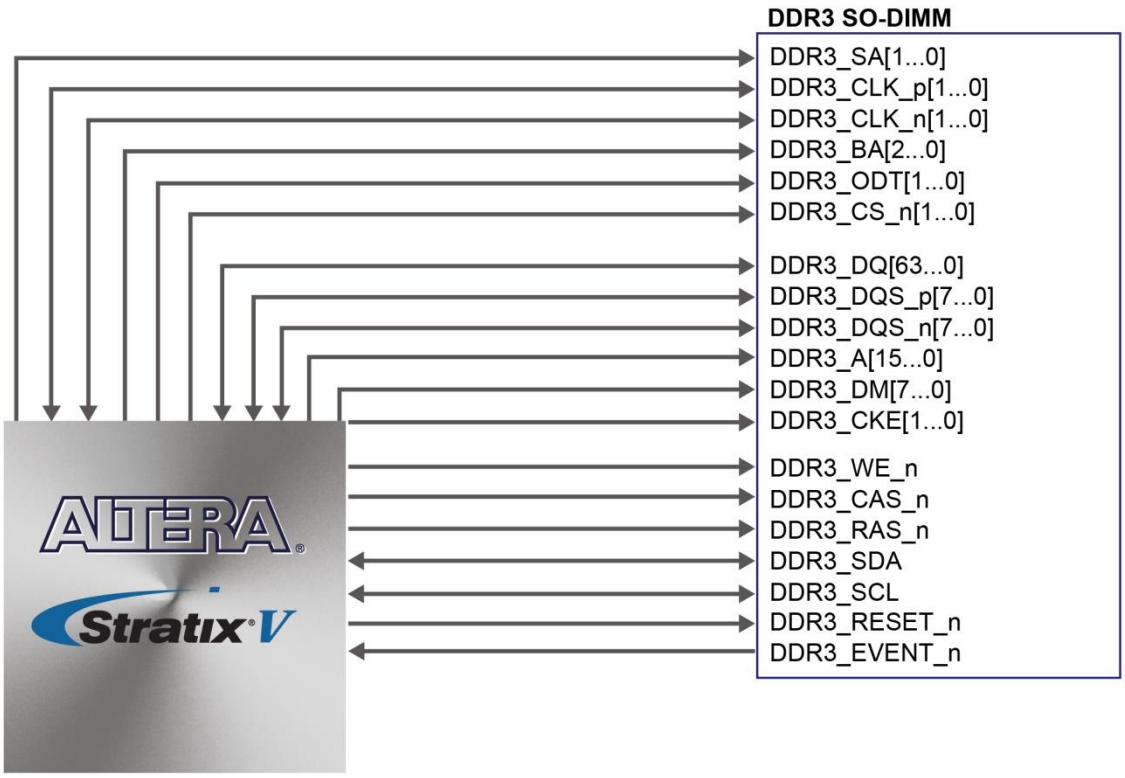


Figure 2-12 Connection between the DDR3 and Stratix V GX FPGA

The pin assignments for DDR3 SDRAM SO-DIMM are listed in **Table 2-14**.

Table 2-14 DDR3 Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
DDR3_DQ0	Data [0]	SSTL-15 Class I	PIN_AH31
DDR3_DQ1	Data [1]	SSTL-15 Class I	PIN_AJ31
DDR3_DQ2	Data [2]	SSTL-15 Class I	PIN_AN30
DDR3_DQ3	Data [3]	SSTL-15 Class I	PIN_AP30
DDR3_DQ4	Data [4]	SSTL-15 Class I	PIN_AH30
DDR3_DQ5	Data [5]	SSTL-15 Class I	PIN_AJ30
DDR3_DQ6	Data [6]	SSTL-15 Class I	PIN_AR30
DDR3_DQ7	Data [7]	SSTL-15 Class I	PIN_AT30
DDR3_DQ8	Data [8]	SSTL-15 Class I	PIN_AM29
DDR3_DQ9	Data [9]	SSTL-15 Class I	PIN_AN28
DDR3_DQ10	Data [10]	SSTL-15 Class I	PIN_AP28

DDR3_DQ11	Data [11]	SSTL-15 Class I	PIN_AR29
DDR3_DQ12	Data [12]	SSTL-15 Class I	PIN_AU31
DDR3_DQ13	Data [13]	SSTL-15 Class I	PIN_AV32
DDR3_DQ14	Data [14]	SSTL-15 Class I	PIN_AW32
DDR3_DQ15	Data [15]	SSTL-15 Class I	PIN_AV31
DDR3_DQ16	Data [16]	SSTL-15 Class I	PIN_AF28
DDR3_DQ17	Data [17]	SSTL-15 Class I	PIN_AF29
DDR3_DQ18	Data [18]	SSTL-15 Class I	PIN_AG30
DDR3_DQ19	Data [19]	SSTL-15 Class I	PIN_AG29
DDR3_DQ20	Data [20]	SSTL-15 Class I	PIN_AG28
DDR3_DQ21	Data [21]	SSTL-15 Class I	PIN_AG27
DDR3_DQ22	Data [22]	SSTL-15 Class I	PIN_AG26
DDR3_DQ23	Data [23]	SSTL-15 Class I	PIN_AG25
DDR3_DQ24	Data [24]	SSTL-15 Class I	PIN_BC31
DDR3_DQ25	Data [25]	SSTL-15 Class I	PIN_BC32
DDR3_DQ26	Data [26]	SSTL-15 Class I	PIN_BB30
DDR3_DQ27	Data [27]	SSTL-15 Class I	PIN_BD31
DDR3_DQ28	Data [28]	SSTL-15 Class I	PIN_BD32
DDR3_DQ29	Data [29]	SSTL-15 Class I	PIN_BA30
DDR3_DQ30	Data [30]	SSTL-15 Class I	PIN_AY31
DDR3_DQ31	Data [31]	SSTL-15 Class I	PIN_AW30
DDR3_DQ32	Data [32]	SSTL-15 Class I	PIN_BB29
DDR3_DQ33	Data [33]	SSTL-15 Class I	PIN_BB27
DDR3_DQ34	Data [34]	SSTL-15 Class I	PIN_BA27
DDR3_DQ35	Data [35]	SSTL-15 Class I	PIN_AW27
DDR3_DQ36	Data [36]	SSTL-15 Class I	PIN_AY28
DDR3_DQ37	Data [37]	SSTL-15 Class I	PIN_BA28
DDR3_DQ38	Data [38]	SSTL-15 Class I	PIN_AW29
DDR3_DQ39	Data [39]	SSTL-15 Class I	PIN_AY27
DDR3_DQ40	Data [40]	SSTL-15 Class I	PIN_AT27
DDR3_DQ41	Data [41]	SSTL-15 Class I	PIN_AN25
DDR3_DQ42	Data [42]	SSTL-15 Class I	PIN_AM25
DDR3_DQ43	Data [43]	SSTL-15 Class I	PIN_AL25
DDR3_DQ44	Data [44]	SSTL-15 Class I	PIN_AW26
DDR3_DQ45	Data [45]	SSTL-15 Class I	PIN_AV26
DDR3_DQ46	Data [46]	SSTL-15 Class I	PIN_AU27
DDR3_DQ47	Data [47]	SSTL-15 Class I	PIN_AM26
DDR3_DQ48	Data [48]	SSTL-15 Class I	PIN_AU28
DDR3_DQ49	Data [49]	SSTL-15 Class I	PIN_AU29
DDR3_DQ50	Data [50]	SSTL-15 Class I	PIN_AM28
DDR3_DQ51	Data [51]	SSTL-15 Class I	PIN_AL27

DDR3_DQ52	Data [52]	SSTL-15 Class I	PIN_AV28
DDR3_DQ53	Data [53]	SSTL-15 Class I	PIN_AV29
DDR3_DQ54	Data [54]	SSTL-15 Class I	PIN_AL28
DDR3_DQ55	Data [55]	SSTL-15 Class I	PIN_AK27
DDR3_DQ56	Data [56]	SSTL-15 Class I	PIN_AK24
DDR3_DQ57	Data [57]	SSTL-15 Class I	PIN_AJ24
DDR3_DQ58	Data [58]	SSTL-15 Class I	PIN_AH24
DDR3_DQ59	Data [59]	SSTL-15 Class I	PIN_AH25
DDR3_DQ60	Data [60]	SSTL-15 Class I	PIN_AH28
DDR3_DQ61	Data [61]	SSTL-15 Class I	PIN_AJ28
DDR3_DQ62	Data [62]	SSTL-15 Class I	PIN_AL26
DDR3_DQ63	Data [63]	SSTL-15 Class I	PIN_AK26
DDR3_DQS0	Data Strobe p[0]	Differential 1.5-V SSTL Class I	PIN_AL30
DDR3_DQS_n0	Data Strobe n[0]	Differential 1.5-V SSTL Class I	PIN_AL31
DDR3_DQS1	Data Strobe p[1]	Differential 1.5-V SSTL Class I	PIN_AK30
DDR3_DQS_n1	Data Strobe n[1]	Differential 1.5-V SSTL Class I	PIN_AL29
DDR3_DQS2	Data Strobe p[2]	Differential 1.5-V SSTL Class I	PIN_AE27
DDR3_DQS_n2	Data Strobe n[2]	Differential 1.5-V SSTL Class I	PIN_AE28
DDR3_DQS3	Data Strobe p[3]	Differential 1.5-V SSTL Class I	PIN_AY30
DDR3_DQS_n3	Data Strobe n[4]	Differential 1.5-V SSTL Class I	PIN_BA29
DDR3_DQS4	Data Strobe p[4]	Differential 1.5-V SSTL Class I	PIN_BC28
DDR3_DQS_n4	Data Strobe n[4]	Differential 1.5-V SSTL Class I	PIN_BD28
DDR3_DQS5	Data Strobe p[5]	Differential 1.5-V SSTL Class I	PIN_AT26
DDR3_DQS_n5	Data Strobe n[5]	Differential 1.5-V SSTL Class I	PIN_AU26
DDR3_DQS6	Data Strobe p[6]	Differential 1.5-V SSTL Class I	PIN_AR27
DDR3_DQS_n6	Data Strobe n[6]	Differential 1.5-V SSTL Class I	PIN_AR28
DDR3_DQS7	Data Strobe p[7]	Differential 1.5-V SSTL Class I	PIN_AJ25
DDR3_DQS_n7	Data Strobe n[7]	Differential 1.5-V SSTL Class I	PIN_AJ26
DDR3_DM0	Data Mask [0]	SSTL-15 Class I	PIN_AU32
DDR3_DM1	Data Mask [1]	SSTL-15 Class I	PIN_AU30
DDR3_DM2	Data Mask [2]	SSTL-15 Class I	PIN_AK29
DDR3_DM3	Data Mask [3]	SSTL-15 Class I	PIN_BB32
DDR3_DM4	Data Mask [4]	SSTL-15 Class I	PIN_BD29
DDR3_DM5	Data Mask [5]	SSTL-15 Class I	PIN_AR26
DDR3_DM6	Data Mask [6]	SSTL-15 Class I	PIN_AP27
DDR3_DM7	Data Mask [7]	SSTL-15 Class I	PIN_AJ27
DDR3_A0	Address [0]	SSTL-15 Class I	PIN_AM32
DDR3_A1	Address [1]	SSTL-15 Class I	PIN_AF31
DDR3_A2	Address [2]	SSTL-15 Class I	PIN_AJ33
DDR3_A3	Address [3]	SSTL-15 Class I	PIN_AE31
DDR3_A4	Address [4]	SSTL-15 Class I	PIN_AP33

DDR3_A5	Address [5]	SSTL-15 Class I	PIN_AG32
DDR3_A6	Address [6]	SSTL-15 Class I	PIN_AN33
DDR3_A7	Address [7]	SSTL-15 Class I	PIN_AK33
DDR3_A8	Address [8]	SSTL-15 Class I	PIN_AF32
DDR3_A9	Address [9]	SSTL-15 Class I	PIN_AH33
DDR3_A10	Address [10]	SSTL-15 Class I	PIN_AE30
DDR3_A11	Address [11]	SSTL-15 Class I	PIN_BA33
DDR3_A12	Address [12]	SSTL-15 Class I	PIN_AG33
DDR3_A13	Address [13]	SSTL-15 Class I	PIN_AD32
DDR3_A14	Address [14]	SSTL-15 Class I	PIN_BA34
DDR3_A15	Address [15]	SSTL-15 Class I	PIN_AY33
DDR3_RAS_n	Row Address Strobe	SSTL-15 Class I	PIN_AJ32
DDR3_CAS_n	Column Address Strobe	SSTL-15 Class I	PIN_AE33
DDR3_BA0	Bank Address [0]	SSTL-15 Class I	PIN_AE29
DDR3_BA1	Bank Address [1]	SSTL-15 Class I	PIN_AK32
DDR3_BA2	Bank Address [2]	SSTL-15 Class I	PIN_AE34
DDR3_CK0	Clock p0	Differential 1.5-V SSTL Class I	PIN_AR31
DDR3_CK_n0	Clock n0	Differential 1.5-V SSTL Class I	PIN_AR32
DDR3_CK1	Clock p1	Differential 1.5-V SSTL Class I	PIN_AV34
DDR3_CK_n1	Clock n1	Differential 1.5-V SSTL Class I	PIN_AW33
DDR3_CKE0	Clock Enable pin 0	SSTL-15 Class I	PIN_AF34
DDR3_CKE1	Clock Enable pin 1	SSTL-15 Class I	PIN_AY34
DDR3_ODT0	On Die Termination[0]	SSTL-15 Class I	PIN_AN31
DDR3_ODT1	On Die Termination[1]	SSTL-15 Class I	PIN_AM31
DDR3_WE_n	Write Enable	SSTL-15 Class I	PIN_AE32
DDR3_CS_n0	Chip Select [0]	SSTL-15 Class I	PIN_AP31
DDR3_CS_n1	Chip Select [1]	SSTL-15 Class I	PIN_AD33
DDR3_RESET_n	Chip Reset	SSTL-15 Class I	PIN_AR33
DDR3_EVENT_n	Chip Temperature Event	SSTL-15 Class I	PIN_AU35
DDR3_SDA	Chip I2C Serial Clock	1.5V	PIN_AJ29
DDR3_SCL	Chip I2C Serial Data Bus	1.5V	PIN_AT29

2.8 FMC Connectors

The FPGA Mezzanine Card (FMC) interface provides a mechanism to extend the peripheral-set of an FPGA host board by means of add-on daughter cards, which can address today's high speed signaling requirements as well as low-speed device interface support. The FMC interfaces support JTAG, clock

outputs and inputs, high-speed serial I/O (transceivers), and single-ended or differential signaling. The detailed specifications of the FMC connectors are described below:

■ 4 FMC Connector

There are four FMC connectors on the TR5 board are: FMCA, FMCB, FMCC, FMCD. Both FMCA and FMCD are High Pin Count (HPC) size of connectors and FMCB and FMCC are Low Pin Count (LPC) size of connectors (See **Figure 2-12**). The HPC connector on TR5 board can provides 172 user-define, single-ended signals (include clock signals) and 10 serial transceiver pairs (See **Figure 2-14**). The LPC connector can provides 76 user-define, single-ended signals (include clock signals) and 1 serial transceiver pairs (See **Figure 2-15**). The HPC and LPC connectors use the same mechanical connector. The only difference is which signals are actually populated. Thus, cards with LPC connectors can be plugged into HPC sites. Please note that some standard FMC cards may not work with TR5 due to unidirectional LVDS due to Stratix V device.

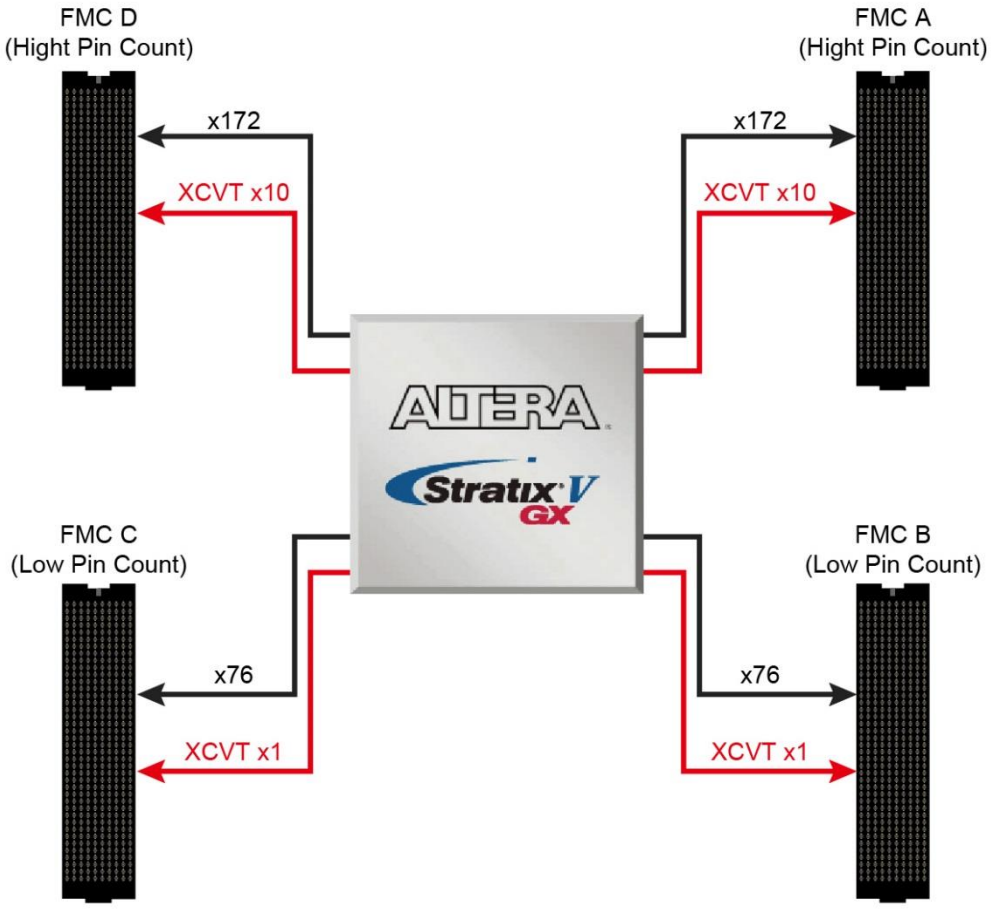
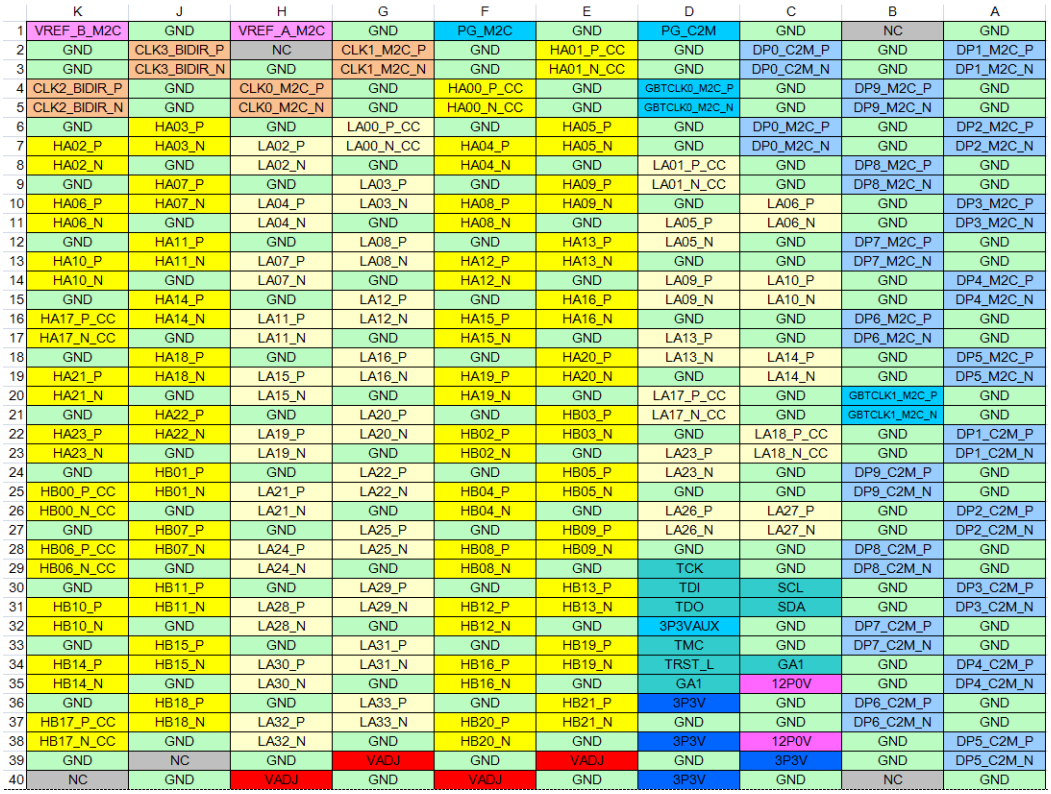
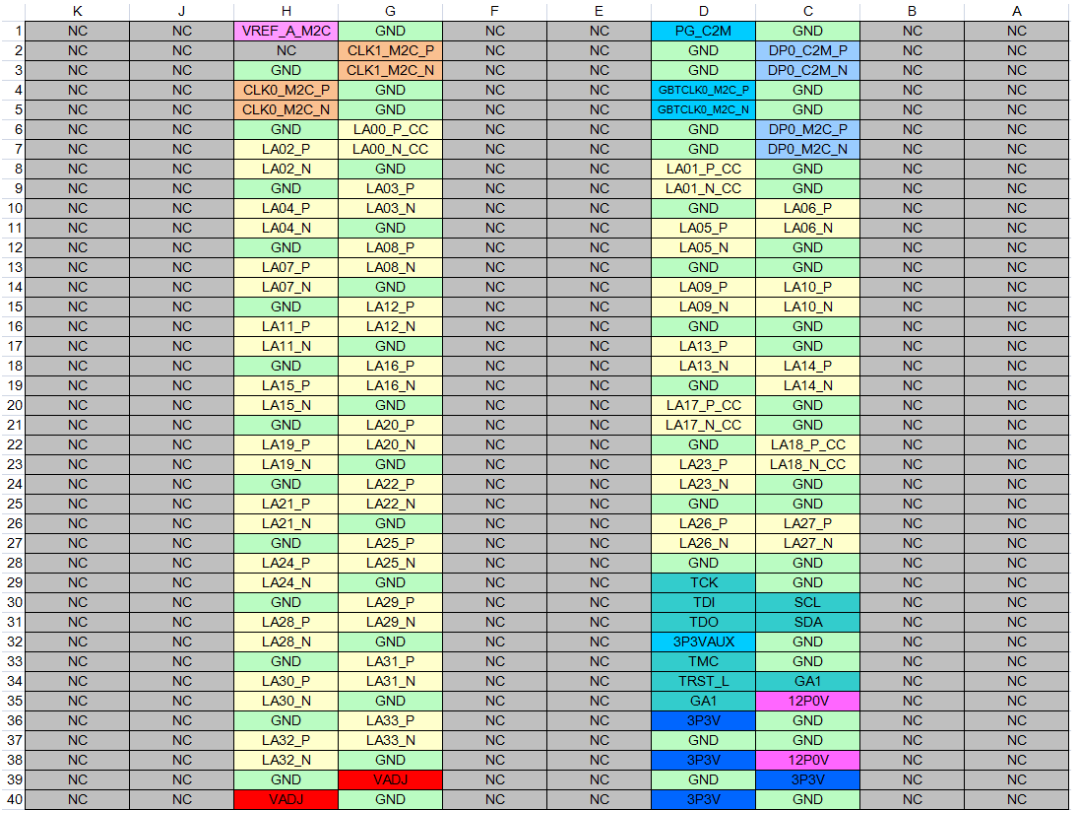


Figure 2-13 FMC connectors on TR5 board



	K	J	H	G	F	E	D	C	B	A
1	VREF_B_M2C	GND	VREF_A_M2C	GND	PG_M2C	GND	PG_C2M	GND	NC	GND
2	GND	CLK3_BIDIR_P	NC	CLK1_M2C_P	GND	HA01_P_CC	GND	DP0_C2M_P	GND	DP1_M2C_P
3	GND	CLK3_BIDIR_N	GND	CLK1_M2C_N	GND	HA01_N_CC	GND	DP0_C2M_N	GND	DP1_M2C_N
4	CLK2_BIDIR_P	GND	CLK0_M2C_P	GND	HA00_P_CC	GND	GBTCLK0_M2C_P	GND	DP9_M2C_P	GND
5	CLK2_BIDIR_N	GND	CLK0_M2C_N	GND	HA00_N_CC	GND	GBTCLK0_M2C_N	GND	DP9_M2C_N	GND
6	GND	HA03_P	GND	LA00_P_CC	GND	HA05_P	GND	DP0_M2C_P	GND	DP2_M2C_P
7	HA02_P	HA03_N	LA02_P	LA00_N_CC	HA04_P	HA05_N	GND	DP0_M2C_N	GND	DP2_M2C_N
8	HA02_N	GND	LA02_N	GND	HA04_N	GND	LA01_P_CC	GND	DP8_M2C_P	GND
9	GND	HA07_P	GND	LA03_P	GND	HA09_P	LA01_N_CC	GND	DP8_M2C_N	GND
10	HA06_P	HA07_N	LA04_P	LA03_N	HA08_P	HA09_N	GND	LA06_P	GND	DP3_M2C_P
11	HA06_N	GND	LA04_N	GND	HA08_N	GND	LA05_P	LA06_N	GND	DP3_M2C_N
12	GND	HA11_P	GND	LA08_P	GND	HA13_P	LA05_N	GND	DP7_M2C_P	GND
13	HA10_P	HA11_N	LA07_P	LA08_N	HA12_P	HA13_N	GND	GND	DP7_M2C_N	GND
14	HA10_N	GND	LA07_N	GND	HA12_N	GND	LA09_P	LA10_P	GND	DP4_M2C_P
15	GND	HA14_P	GND	LA12_P	GND	HA16_P	LA09_N	LA10_N	GND	DP4_M2C_N
16	HA17_P_CC	HA14_N	LA11_P	LA12_N	HA15_P	HA16_N	GND	GND	DP6_M2C_P	GND
17	HA17_N_CC	GND	LA11_N	GND	HA15_N	GND	LA13_P	GND	DP6_M2C_N	GND
18	GND	HA18_P	GND	LA16_P	GND	HA20_P	LA13_N	LA14_P	GND	DP5_M2C_P
19	HA21_P	HA18_N	LA15_P	LA16_N	HA19_P	HA20_N	GND	LA14_N	GND	DP5_M2C_N
20	HA21_N	GND	LA15_N	GND	HA19_N	GND	LA17_P_CC	GND	GBTCLK1_M2C_P	GND
21	GND	HA22_P	GND	LA20_P	GND	HB03_P	LA17_N_CC	GND	GBTCLK1_M2C_N	GND
22	HA23_P	HA22_N	LA19_P	LA20_N	HB02_P	HB03_N	GND	LA18_P_CC	GND	DP1_C2M_P
23	HA23_N	GND	LA19_N	GND	HB02_N	GND	LA23_P	LA18_N_CC	GND	DP1_C2M_N
24	GND	HB01_P	GND	LA22_P	GND	HB05_P	LA23_N	GND	DP9_C2M_P	GND
25	HB00_P_CC	HB01_N	LA21_P	LA22_N	HB04_P	HB05_N	GND	GND	DP9_C2M_N	GND
26	HB00_N_CC	GND	LA21_N	GND	HB04_N	GND	LA26_P	LA27_P	GND	DP2_C2M_P
27	GND	HB07_P	GND	LA25_P	GND	HB09_P	LA26_N	LA27_N	GND	DP2_C2M_N
28	HB06_P_CC	HB07_N	LA24_P	LA25_N	HB08_P	HB09_N	GND	GND	DP8_C2M_P	GND
29	HB06_N_CC	GND	LA24_N	GND	HB08_N	GND	TCK	GND	DP8_C2M_N	GND
30	GND	HB11_P	GND	LA29_P	GND	HB13_P	TDI	SCL	GND	DP3_C2M_P
31	HB10_P	HB11_N	LA28_P	LA29_N	HB12_P	HB13_N	TDO	SDA	GND	DP3_C2M_N
32	HB10_N	GND	LA28_N	GND	HB12_N	GND	3P3VAUX	GND	DP7_C2M_P	GND
33	GND	HB15_P	GND	LA31_P	GND	HB19_P	TMC	GND	DP7_C2M_N	GND
34	HB14_P	HB15_N	LA30_P	LA31_N	HB16_P	HB19_N	TRST_L	GA1	GND	DP4_C2M_P
35	HB14_N	GND	LA30_N	GND	HB16_N	GND	GA1	12P0V	GND	DP4_C2M_N
36	GND	HB18_P	GND	LA33_P	GND	HB21_P	3P3V	GND	DP6_C2M_P	GND
37	HB17_P_CC	HB18_N	LA32_P	LA33_N	HB20_P	HB21_N	GND	GND	DP6_C2M_N	GND
38	HB17_N_CC	GND	LA32_N	GND	HB20_N	GND	3P3V	12P0V	GND	DP5_C2M_P
39	GND	NC	GND	VADJ	GND	VADJ	GND	3P3V	GND	DP5_C2M_N
40	NC	GND	VADJ	GND	VADJ	GND	3P3V	GND	NC	GND

Figure 2-14 Pin-Out of the high pin count FMC connector



	K	J	H	G	F	E	D	C	B	A
1	NC	NC	VREF_A_M2C	GND	NC	NC	PG_C2M	GND	NC	NC
2	NC	NC	NC	CLK1_M2C_P	NC	NC	GND	DP0_C2M_P	NC	NC
3	NC	NC	GND	CLK1_M2C_N	NC	NC	GND	DP0_C2M_N	NC	NC
4	NC	NC	CLK0_M2C_P	GND	NC	NC	GBTCLK0_M2C_P	GND	NC	NC
5	NC	NC	CLK0_M2C_N	GND	NC	NC	GBTCLK0_M2C_N	GND	NC	NC
6	NC	NC	GND	LA00_P_CC	NC	NC	GND	DP0_M2C_P	NC	NC
7	NC	NC	LA02_P	LA00_N_CC	NC	NC	GND	DP0_M2C_N	NC	NC
8	NC	NC	LA02_N	GND	NC	NC	LA01_P_CC	GND	NC	NC
9	NC	NC	GND	LA03_P	NC	NC	LA01_N_CC	GND	NC	NC
10	NC	NC	LA04_P	LA03_N	NC	NC	GND	LA06_P	NC	NC
11	NC	NC	LA04_N	GND	NC	NC	LA05_P	LA06_N	NC	NC
12	NC	NC	GND	LA08_P	NC	NC	LA05_N	GND	NC	NC
13	NC	NC	LA07_P	LA08_N	NC	NC	GND	GND	NC	NC
14	NC	NC	LA07_N	GND	NC	NC	LA09_P	LA10_P	NC	NC
15	NC	NC	GND	LA12_P	NC	NC	LA09_N	LA10_N	NC	NC
16	NC	NC	LA11_P	LA12_N	NC	NC	GND	GND	NC	NC
17	NC	NC	LA11_N	GND	NC	NC	LA13_P	GND	NC	NC
18	NC	NC	GND	LA16_P	NC	NC	LA13_N	LA14_P	NC	NC
19	NC	NC	LA15_P	LA16_N	NC	NC	GND	LA14_N	NC	NC
20	NC	NC	LA15_N	GND	NC	NC	LA17_P_CC	GND	NC	NC
21	NC	NC	GND	LA20_P	NC	NC	LA17_N_CC	GND	NC	NC
22	NC	NC	LA19_P	LA20_N	NC	NC	GND	LA18_P_CC	NC	NC
23	NC	NC	LA19_N	GND	NC	NC	LA23_P	LA18_N_CC	NC	NC
24	NC	NC	GND	LA22_P	NC	NC	LA23_N	GND	NC	NC
25	NC	NC	LA21_P	LA22_N	NC	NC	GND	GND	NC	NC
26	NC	NC	LA21_N	GND	NC	NC	LA26_P	LA27_P	NC	NC
27	NC	NC	GND	LA25_P	NC	NC	LA26_N	LA27_N	NC	NC
28	NC	NC	LA24_P	LA25_N	NC	NC	GND	GND	NC	NC
29	NC	NC	LA24_N	GND	NC	NC	TCK	GND	NC	NC
30	NC	NC	GND	LA29_P	NC	NC	TDI	SCL	NC	NC
31	NC	NC	LA28_P	LA29_N	NC	NC	TDO	SDA	NC	NC
32	NC	NC	LA28_N	GND	NC	NC	3P3VAUX	GND	NC	NC
33	NC	NC	GND	LA31_P	NC	NC	TMC	GND	NC	NC
34	NC	NC	LA30_P	LA31_N	NC	NC	TRST_L	GA1	NC	NC
35	NC	NC	LA30_N	GND	NC	NC	GA1	12P0V	NC	NC
36	NC	NC	GND	LA33_P	NC	NC	3P3V	GND	NC	NC
37	NC	NC	LA32_P	LA33_N	NC	NC	GND	GND	NC	NC
38	NC	NC	LA32_N	GND	NC	NC	3P3V	12P0V	NC	NC
39	NC	NC	GND	VADJ	NC	NC	GND	3P3V	NC	NC
40	NC	NC	VADJ	GND	NC	NC	3P3V	GND	NC	NC

Figure 2-15 Pin-Out of the low pin count FMC connector

■ Clock Interface

Due to the limitation of the FPGA clock input pin numbers, not all the FMC ports have same clock interface. **Table 2-15** shows the FPGA dedicated clock input pin placement on each FMC port.

Table 2-15 FMC clock interface distribution

FMC Clock in/out pin name	*FPGA Clock Input Pin Placement*			
	FMCA	*FMCB*	*FMCC*	*FMCD*
CLK0_M2C_P	I/O	CLK22p	I/O	I/O
CLK0_M2C_N	I/O	CLK22n	I/O	I/O
CLK1_M2C_P	I/O	CLK23p	I/O	CLK15p
CLK1_M2C_N	I/O	CLK23n	I/O	CLK15n
HA01_P_CC	CLK17p	N/A	N/A	CLK14p
HA01_N_CC	CLK17n	N/A	N/A	CLK14n
LA01_P_CC	CLK19p	CLK21p	CLK7p	CLK13p
LA01_N_CC	CLK19n	CLK21n	CLK7n	CLK13n

Please note that, all the dedicated clock pin of the FPGA are connected to external termination resistors (See **Figure 2-16**). IF users want to us these I/O as single-end standard, pleaser remove these resistors.

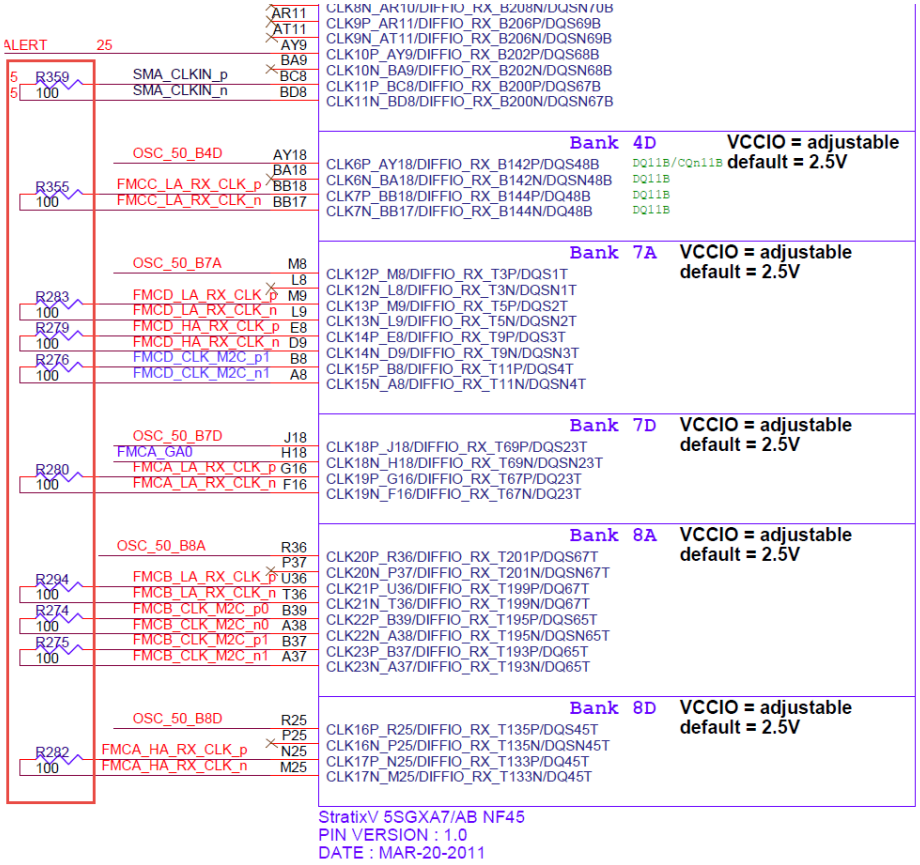


Figure 2-16 Termination resistors on FMC clock I/Os

■ Power Supply

The TR5 board provides 12V, 3.3V and VCCIO_FMC power through FMC ports. **Table 2-16** indicates the maximum power consumption for all FMC ports. Please note that this table shows the total max current limit for all six ports, not just for one.

Also, the 12V DC power supplies from the FMC ports have fuses for protection. Users who don't need the power from the FMC can remove these fuses to cut the power on connector.

CAUTION. Before powering on the TR5 board with a daughter card, please check to see if there is a short circuit between the power pins and FPGA I/O.

Table 2-16 Power Supply of the FMC

Supplied Voltage	*Max. Current Limit*
12V	3A
3.3V	3A
VCCIO_FMC	FMCA : 12A ; FMCB/FMCC/FMCD : 6A

■ Adjustable I/O Standards

The FPGA I/O standards of the FMC ports can be adjusted by configuring the header position. Each port can be individually adjusted to 1.5V, 1.8V, 2.5V or 3.0V via jumpers on the TR5 board. For detailed setting, please refer to Section 2.2.

■ JTAG Chain on FMC

Figure 2-17 shows the JTAG chain loop of the TR5 board. The JTAG interface on the FMC connectors can be activated through four 3-pin headers. For detailed setting, please refer to Section 2.2.

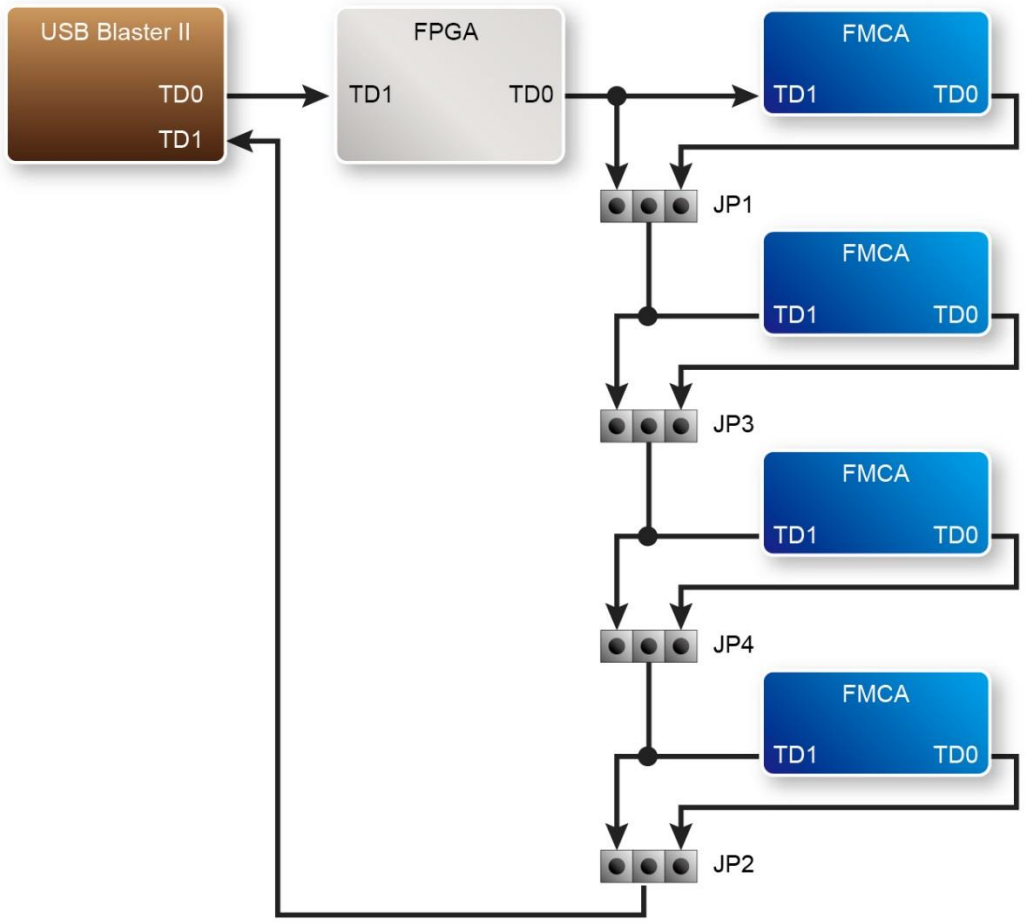


Figure 2-17 JTAG Chain for a Standalone TR5

■ Component Information of the FMC Connector

Table 2-17 shows the manufacture and part numbers of the FMC connector.

Table 2-17 Component information of the FMC connector

Connector Type		*Manufacturer*	*Part Number*
Female (For Mother Board)	**High Pin Count(HPC)**	**SAMTEC**	**ASP-134486-01**
	Low Pin Count(LPC)	**SAMTEC**	**ASP-134603-01**
Male	**High Pin Count(HPC)**	**SAMTEC**	**ASP-134488-01**

(For Daughter Board)	Low Pin Count(LPC)	SAMTEC	ASP-134604-01

2.9 SATA

The two Serial ATA (SATA) ports are available on the FPGA development board which are computer bus standard with a primary function of transferring data between the motherboard and mass storage devices (such as hard drives, optical drives, and solid-state disks). Supporting a storage interface is just one of many different applications for which an FPGA can be used in storage appliances. The Stratix V GX device can bridge different protocols such as bridging simple bus I/Os like PCI Express (PCIe) to SATA or network interfaces such as Gigabit Ethernet (GbE) to SATA. The SATA interface supports SATA 3.0 standard with connection speed of 6 Gbps based on Stratix V GX device with integrated transceivers compliant to SATA electrical standards.

The two Serial ATA (SATA) ports include one available port for device and one available port for host capable of implementing SATA solution with a design that consists of both host and target (device side) functions. **Figure 2-18** shows the connections between the SATA and Stratix V GX FPGA.

Figure 2-18 Connection between the SATA and Stratix V GX FPGA

Table 2-18, list the SATA pin assignments and signal names relative to the Stratix V GX device.

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

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
SATA_HOST_REFCLK_p	SATA Host reference clock	1.4-V PCML	PIN_AH6
SATA_HOST_TX_p	SATA Host transmitter data	1.4-V PCML	PIN_AU4
SATA_HOST_RX_p	SATA Host Receiver data	1.4-V PCML	PIN_AY2
SATA_DEVICE_REFCLK_p	SATA Device reference clock	1.4-V PCML	PIN_AK7
SATA_DEVICE_TX_p	SATA Device transmitter data	1.4-V PCML	PIN_AY6
SATA_DEVICE_RX_p	SATA Device Receiver data	1.4-V PCML	PIN_BB2

2.10 GPIO

The TR5 Board provides a 40-pin expansion header. The header connects directly to 36 pins of the Stratix V GX FPGA, and also provides DC +5V (VCC5), DC +3.3V (VCC3P3), and two GND pins. Figure 2-19 shows the I/O distribution of the GPIO connector. The maximum power consumption of the daughter card that connects to GPIO port is shown in Table 2-19.

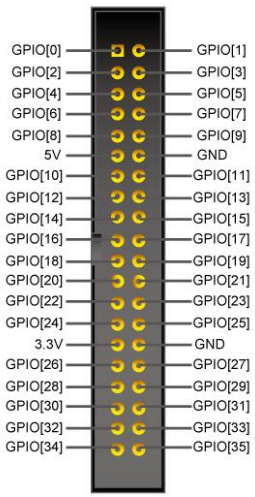


Figure 2-19 GPIO Pin Arrangement

Table 2-19 Power Supply of the Expansion Header

Supplied Voltage	Max. Current Limit
5V	2A
3.3V	3A

Each pin on the expansion headers is connected to a level shift that provides an I/O voltage level shift from 2.5V to 3.3V for the daughter card. **Figure 2-20** shows the level-shift circuitry for only one of the pin on the header, but this circuitry is included for all 36 data pins.

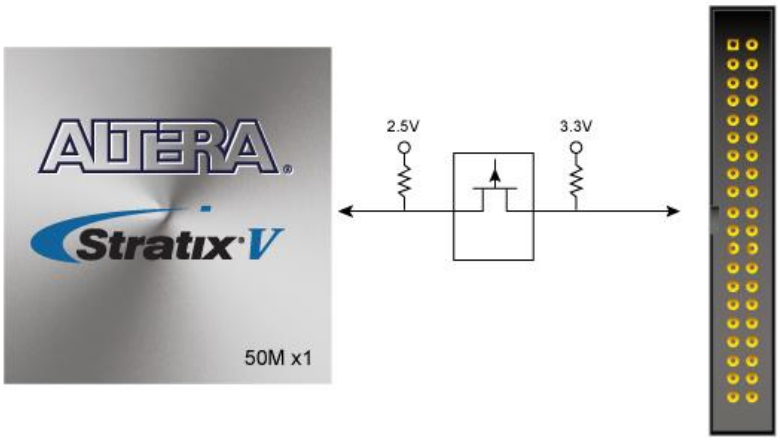


Figure 2-20 Connections between the GPIO connector and Stratix V GX FPGA

Table 2-20 shows all the pin assignments of the GPIO connector.

Table 2-20 GPIO Pin Assignments, Schematic Signal Names, and Functions

Signal Name	Description	I/O Stand	Stratix V GX Pin Number
GPIO[0]	GPIO Connection DATA[0]	2.5V	PIN_AU36
GPIO[1]	GPIO Connection DATA[1]	2.5V	PIN_AE36
GPIO[2]	GPIO Connection DATA[2]	2.5V	PIN_AF35
GPIO[3]	GPIO Connection DATA[3]	2.5V	PIN_AE35
GPIO[4]	GPIO Connection DATA[4]	2.5V	PIN_AN36
GPIO[5]	GPIO Connection DATA[5]	2.5V	PIN_AP36

GPIO[6]	GPIO Connection DATA[6]	2.5V	PIN_AG34
GPIO[7]	GPIO Connection DATA[7]	2.5V	PIN_AK35
GPIO[8]	GPIO Connection DATA[8]	2.5V	PIN_AN34
GPIO[9]	GPIO Connection DATA[9]	2.5V	PIN_AH34
GPIO[10]	GPIO Connection DATA[10]	2.5V	PIN_AL35
GPIO[11]	GPIO Connection DATA[11]	2.5V	PIN_AH22
GPIO[12]	GPIO Connection DATA[12]	2.5V	PIN_AP34
GPIO[13]	GPIO Connection DATA[13]	2.5V	PIN_AJ23
GPIO[14]	GPIO Connection DATA[14]	2.5V	PIN_AJ34
GPIO[15]	GPIO Connection DATA[15]	2.5V	PIN_AJ22
GPIO[16]	GPIO Connection DATA[16]	2.5V	PIN_AK23
GPIO[17]	GPIO Connection DATA[17]	2.5V	PIN_AL23
GPIO[18]	GPIO Connection DATA[18]	2.5V	PIN_AL24
GPIO[19]	GPIO Connection DATA[19]	2.5V	PIN_AK21
GPIO[20]	GPIO Connection DATA[20]	2.5V	PIN_AM23
GPIO[21]	GPIO Connection DATA[21]	2.5V	PIN_AL21
GPIO[22]	GPIO Connection DATA[22]	2.5V	PIN_AN23
GPIO[23]	GPIO Connection DATA[23]	2.5V	PIN_AU23
GPIO[24]	GPIO Connection DATA[24]	2.5V	PIN_AR24
GPIO[25]	GPIO Connection DATA[25]	2.5V	PIN_BA25
GPIO[26]	GPIO Connection DATA[26]	2.5V	PIN_AR23

GPIO[27]	GPIO Connection DATA[27]	2.5V	PIN_BB24
GPIO[28]	GPIO Connection DATA[28]	2.5V	PIN_BC23
GPIO[29]	GPIO Connection DATA[29]	2.5V	PIN_AT23
GPIO[30]	GPIO Connection DATA[30]	2.5V	PIN_AV23
GPIO[31]	GPIO Connection DATA[31]	2.5V	PIN_BD23
GPIO[32]	GPIO Connection DATA[32]	2.5V	PIN_BB26
GPIO[33]	GPIO Connection DATA[33]	2.5V	PIN_AW23
GPIO[34]	GPIO Connection DATA[34]	2.5V	PIN_AY24
GPIO[35]	GPIO Connection DATA[35]	2.5V	PIN_BB23

2.11 PCI Express

The TR5 development board features one PCIe Express **downstream** interfaces (x4 lane) which are designed to interface with a PC motherboard x4 slot via PCIe cable and PCIe adapter card. Utilizing built-in transceivers on a Stratix V GX device, it is able to provide a fully integrated PCI Express-compliant solution for multi-lane (x4) applications. With the PCI Express hard IP block incorporated in the Stratix V GX device, it will allow users to implement simple and fast protocols, as well as saving logic resources for logic applications.

The PCI Express interface supports complete PCI Express Gen1 at 2.5Gbps/lane, Gen2 at 5.0Gbps/lane, and Gen3 at 8.0Gbps/lane protocol stack solution compliant to PCI Express base specification 3.0 that includes PHY-MAC, Data Link, and transaction layer circuitry embedded in PCI Express hard IP blocks.

To use PCIe interface, two external associated devices will be needed to establish a link with PC. First, a PCIe half-height add-in host card with a PCIe x4 cable connector called PCA (PCIe Cabling Adapter Card and See **Figure 2-21**) will be used to plug into the PCIe slot on a mother board. Then,

a PCIe x4 cable (See **Figure 2-22**) will be used to connect TR5 board and PCIe add-in card as shown in **Figure 2-23**, the longest length is up to 3 meters. These two associated devices are not included in TR5 kit. To purchase the PCA card as well as the external cable, please refer to Terasic website ***pca.terasic.com*** and ***PCIe_Cable.terasic.com***.

Figure 2-21 PCIe Cabling Adaptor(PCA) card

Figure 2-22 PCIe External Cable

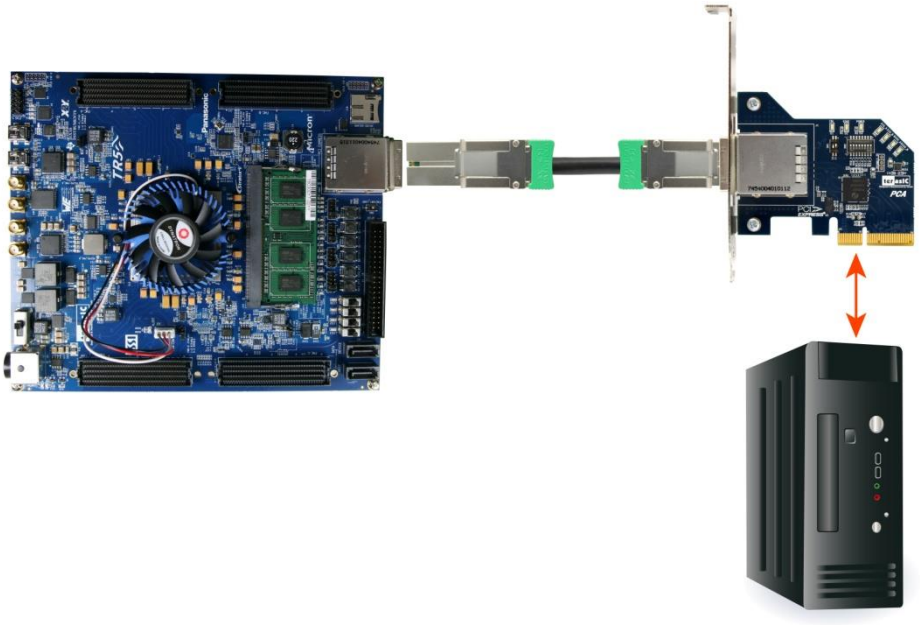


Figure 2-23 PCIe Link Setup between TR5 and PC

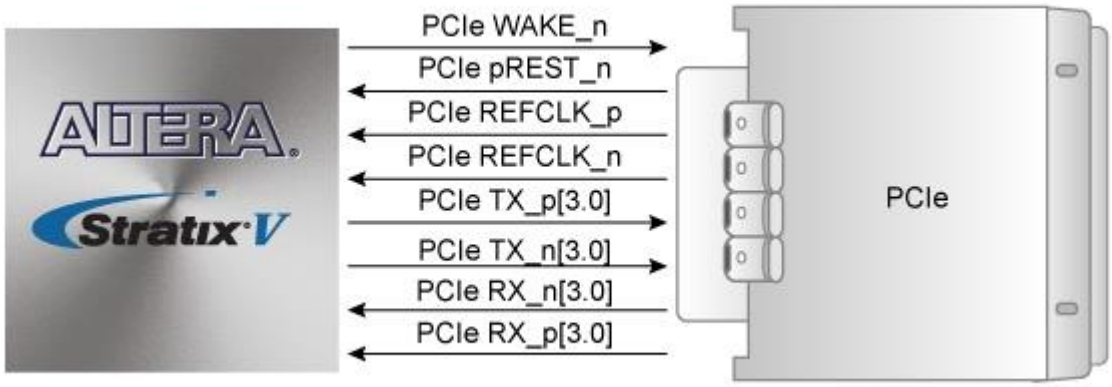


Figure 2-24 PCI Express Pin Connection

Table 2-22 summarizes the PCI Express pin assignments of the signal names relative to the Stratix

V GX FPGA.

Table 2-22 PCI Express Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
PCIE_REFCLK_p	PCIe reference clock	HCSL	PIN_AK38
PCIE_PREST_n	PCIe present ， active low	1.5-V	PIN_AU33
PCIE_WAKE_n	PCIe wake	1.5-V	PIN_BD35
PCIE_TX_p[0]	PCIe Transmitter data p0	1.4-V PCML	PIN_AY39
PCIE_RX_p[0]	PCIe Receiver data p0	1.4-V PCML	PIN_BB43
PCIE_TX_p[1]	PCIe Transmitter data p1	1.4-V PCML	PIN_AV39
PCIE_RX_p[1]	PCIe Receiver data p1	1.4-V PCML	PIN_BA41
PCIE_TX_p[2]	PCIe Transmitter data p2	1.4-V PCML	PIN_AT39
PCIE_RX_p[2]	PCIe Receiver data p2	1.4-V PCML	PIN_AW41
PCIE_TX_p[3]	PCIe Transmitter data p3	1.4-V PCML	PIN_AU41
PCIE_RX_p[3]	PCIe Receiver data p3	1.4-V PCML	PIN_AY43

Chapter 3

System Builder

This chapter describes how users can create a custom design project on the FPGA board by using the Software Tools – System Builder.

3.1 Introduction

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

- Quartus II Project File (.qpf)
- Quartus II 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 that has 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 the **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 II setting file (.qsf).

The top-level design file contains top-level Verilog wrapper for users to add their own design/logic. The Quartus II 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 II programmer must be used to download SOF file to the FPGA board using JTAG interface.

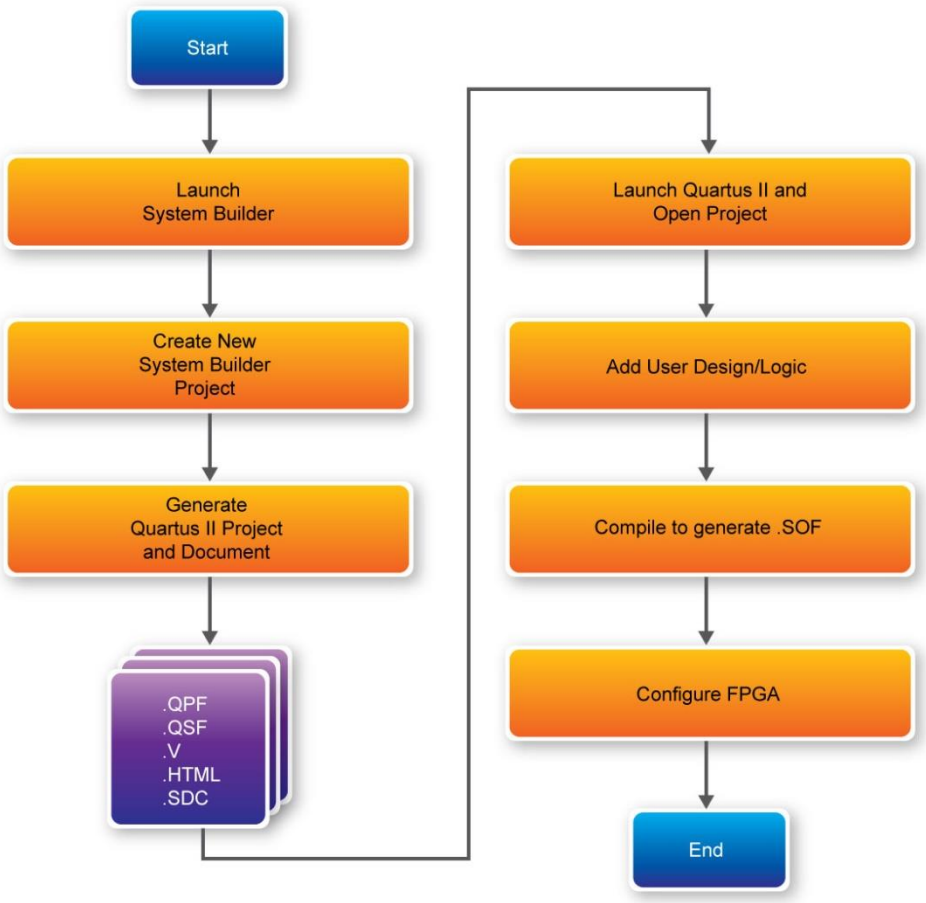


Figure 3-1 The general design flow of building a design

3.3 Using System Builder

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

■ Install and launch the System Builder

The System Builder is located in the directory: "**Tools\SystemBuilder**" in the System CD. Users can copy the whole folder to a host computer without installing the utility. Before using the System Builder, execute the **SystemBuilder.exe** on the host computer as appears in **Figure 3-2**.

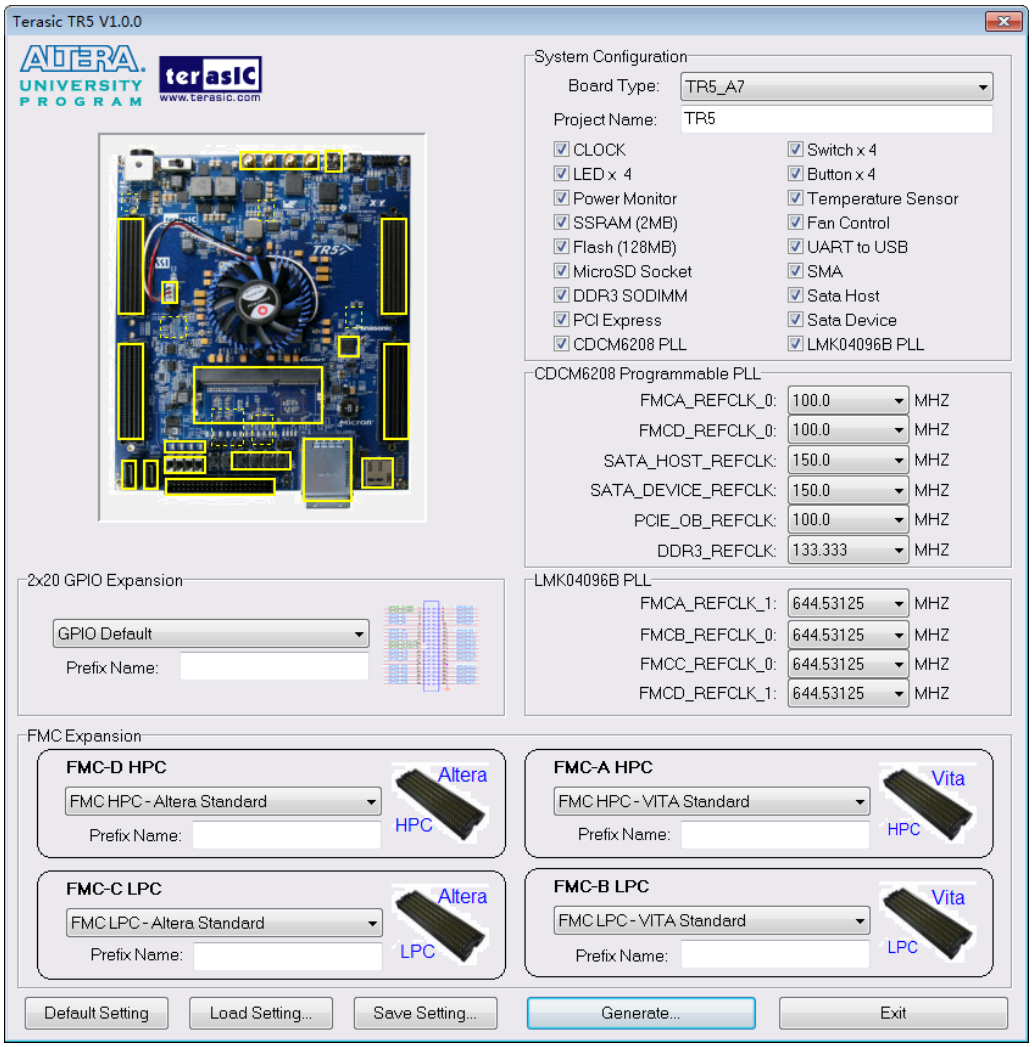


Figure 3-2 The System Builder window

■ Select Board Type and Input Project Name

Select the target board type (TR5_A7 or TR5_AB) and input project name as show in **Figure 3-3**.

■ Project Name:

Specify the project name as it is automatically assigned to the name of the top-level design entity.

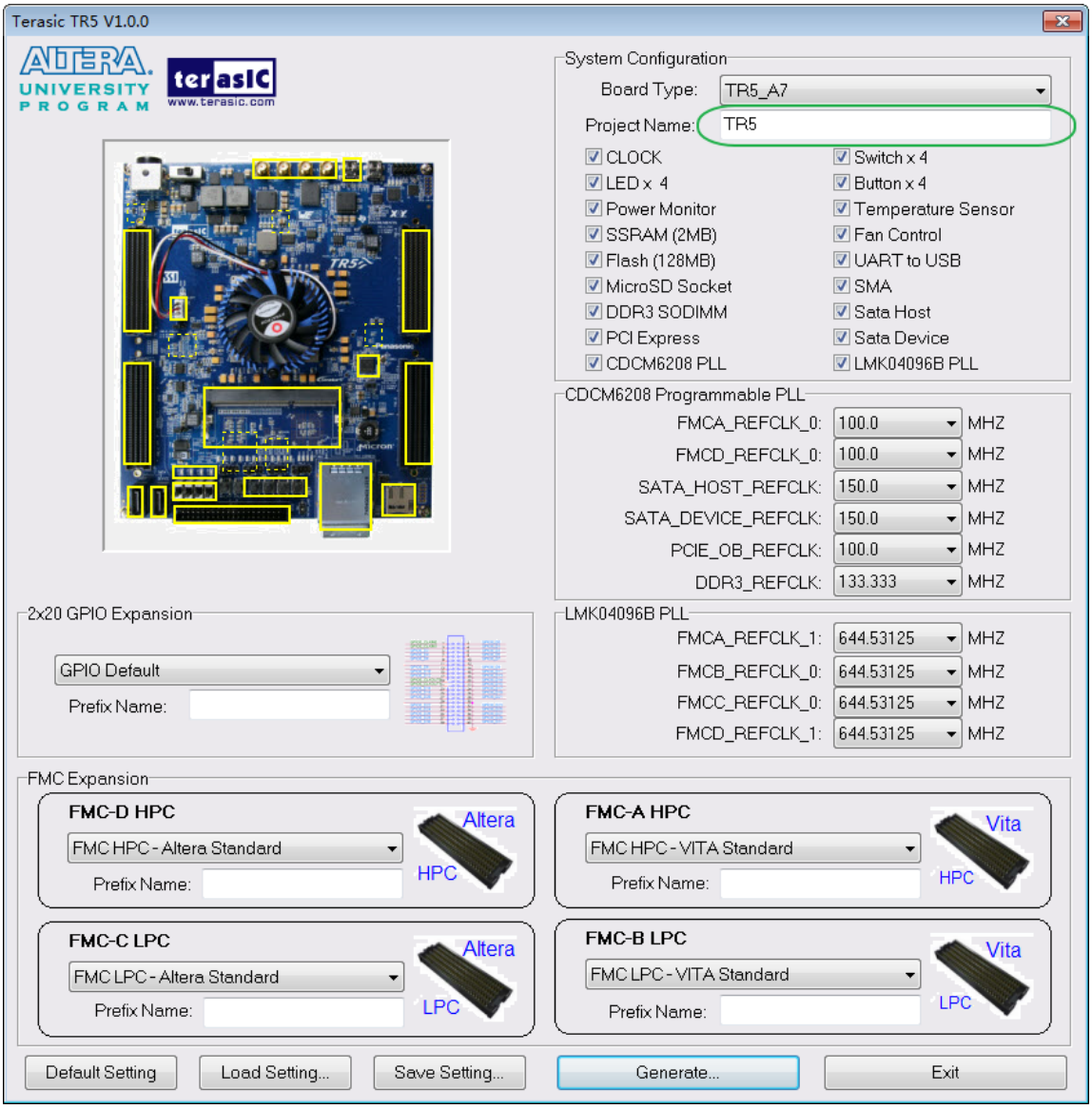


Figure 3-3 The Quartus Project Name

■ System Configuration

Under System Configuration users are given the flexibility of enabling their choice of components on the FPGA as shown in **Figure 3-4**. Each component of the FPGA board is listed where users can enable or disable a component according to their design by simply marking a check or removing the check in the field provided. If the component is enabled, the System Builder will automatically generate the associated pin assignments including the pin name, pin location, pin direction, and I/O standards.

Note: The pin assignments for some components (e.g. DDR3 and SATA) require associated controller codes in the Quartus project otherwise Quartus will result in compilation errors. Therefore, do not select them if they are not necessary in your design. To use the DDR3 controller, please refer to the DDR3 SDRAM demonstration in Chapter 6.

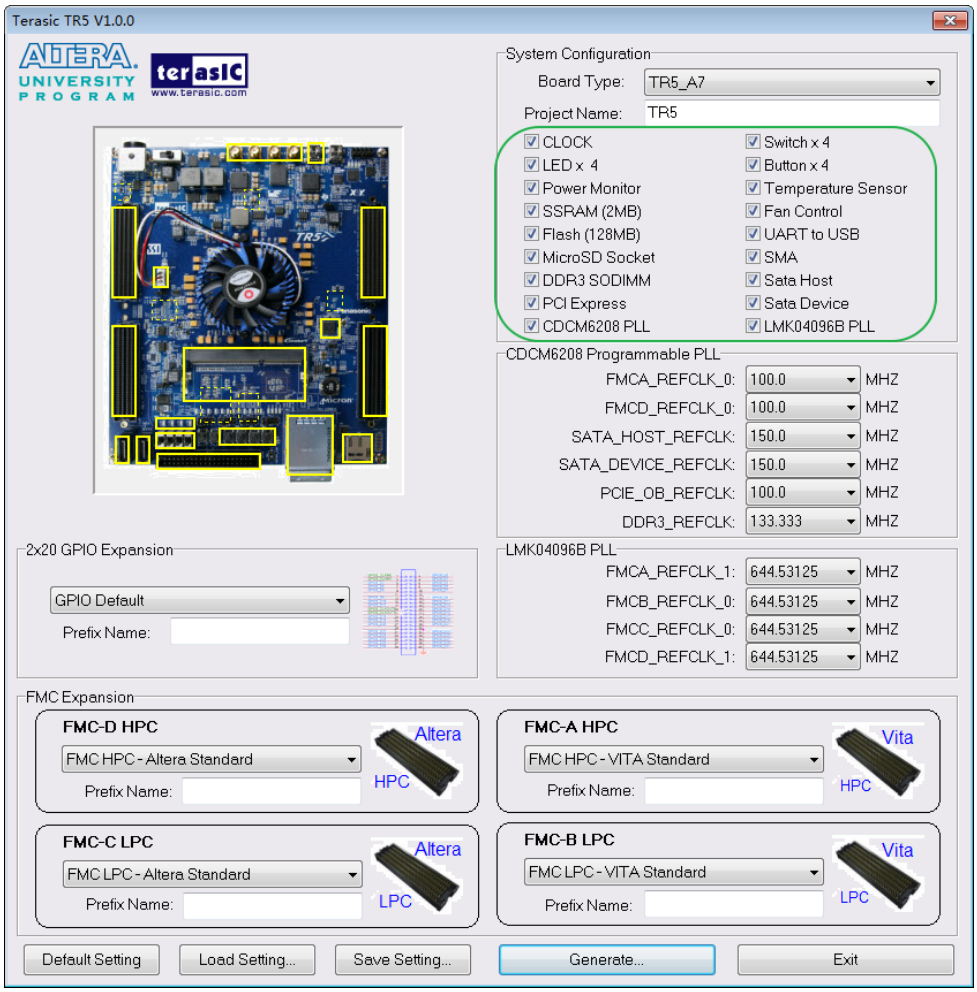


Figure 3-4 System Configuration Group

■ Programmable PLL

There are two external plls on-board that provide reference clocks for the following signals:

- FMCA_ONBOARD_REFCLK
- FMCD_ONBOARD_REFCLK
- PCIE_ONBOARD_REFCLK

- DDR3_REFCLK
- SATA_DEVICE_REFCLK
- SATA_HOST_REFCLK
- FMCB_ONBOARD_REFCLK
- FMCC_ONBOARD_REFCLK

To use these clocks, users can select the desired frequency on the Programmable Oscillator group, as shown in **Figure 3-5**. FMC, DDR3, PCIe or SATA must be checked before users can start to specify the desired frequency in the programmable oscillators.

As the Quartus project is created, the 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.

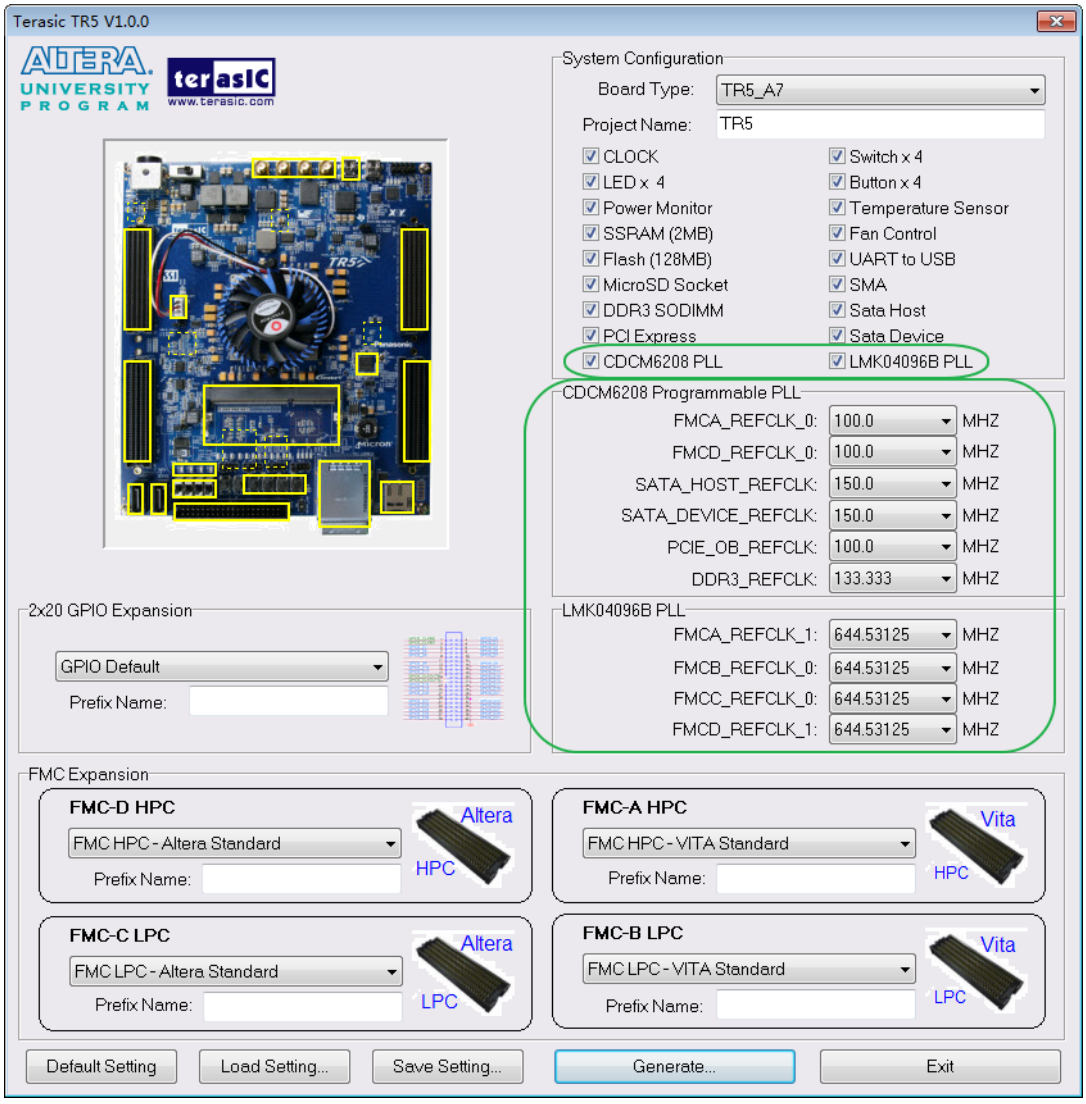


Figure 3-5 External Programmable PLLs

■ Project Setting Management

The System Builder also provides functions to restore default setting, loading a setting, and saving users' board configuration(s) file, as shown in **Figure 3-6**. Users can save the current board configuration information into a .cfg file and load it to the System Builder.

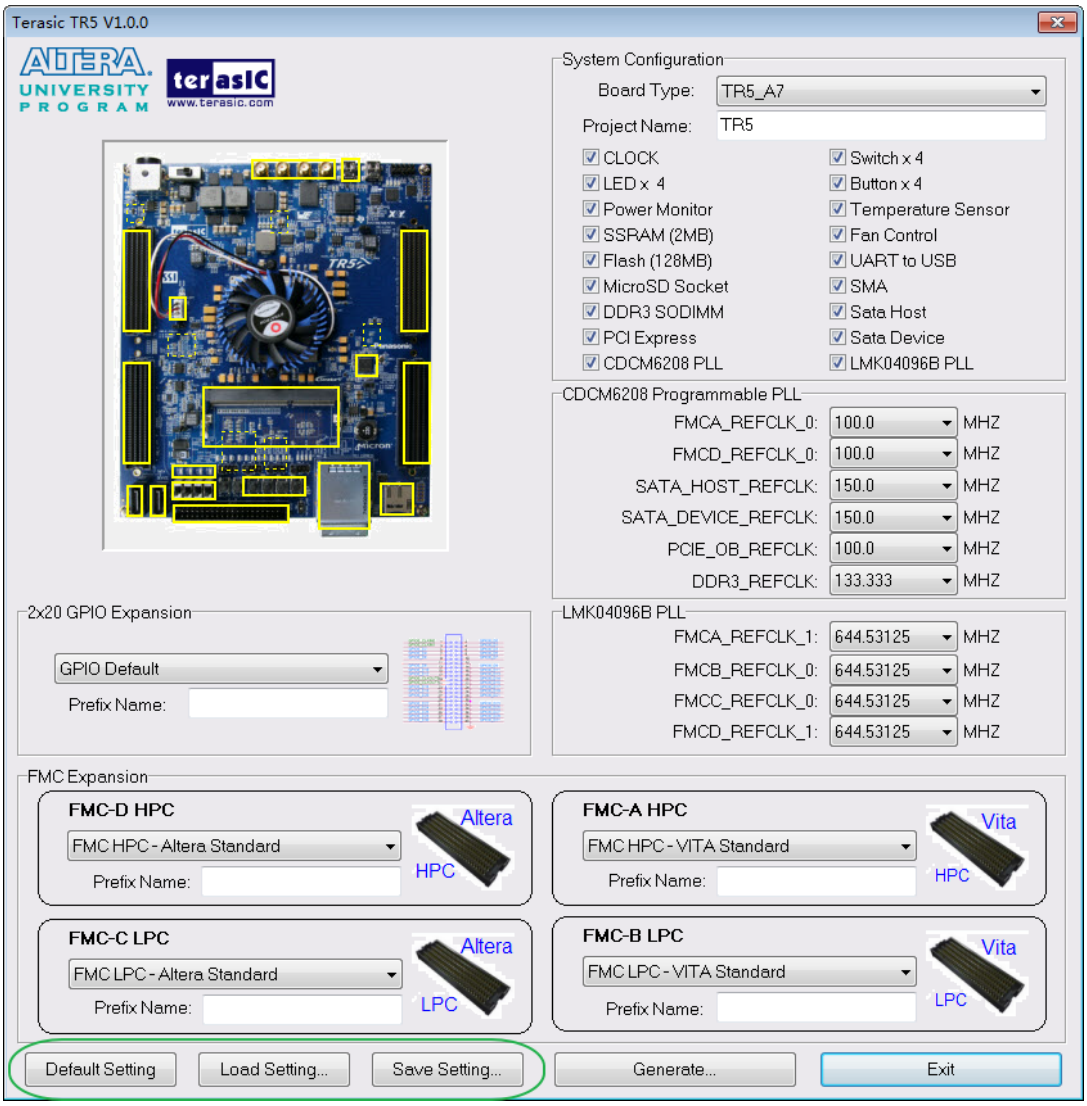


Figure 3-6 Project Settings

■ Project Generation

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

Table 3-1 The files generated by System Builder

No.	*Filename*	*Description*
1	<Project name>.v	Top level Verilog file for Quartus II
2	CDCM6208_controller (*)	CDCM6208 External PLL controller IP

3	LMK04096B_controller(*)	LMK04096B External PLL controller IP
4	<Project name>.qpf	Quartus II Project File
5	<Project name>.qsf	Quartus II Setting File
6	<Project name>.sdc	Synopsis Design Constraints file for Quartus II
7	<Project name>.htm	Pin Assignment Document

(*) The CDCM6208_controller is a folder which contains the verilog files for CDCM6208 configuration.

(*) The LMK04096B_controller is a folder which contains the verilog files for LMK04096B configuration.

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

For CDCM6208, the Controller will be instantiated in the Quartus II top-level file as listed below:

```
//=======================================================
//  External PLL CDCM6208 Configuration ==================
//=======================================================

//  Signal declarations
`define CDCM6208_MODE_0  2'd0   //FMCA(100),FMCD(100),PCIE(100),SATA_HOST(150),SATA_DEVICE(150),DDR3(133.333)
`define CDCM6208_MODE_1  2'd1   //FMCA(125),FMCD(125),PCIE(100),SATA_HOST(150),SATA_DEVICE(150),DDR3(133.333)
`define CDCM6208_MODE_2  2'd2   //FMCA(150),FMCD(150),PCIE(100),SATA_HOST(150),SATA_DEVICE(150),DDR3(133.333)

`define CDCM6208_FMCA_DISABLE_BIT       6'b000001
`define CDCM6208_FMCD_DISABLE_BIT       6'b000010
`define CDCM6208_SATA_HOST_DISABLE_BIT      6'b000100
`define CDCM6208_SATA_DEVICE_DISABLE_BIT        6'b001000
`define CDCM6208_PCIE_DISABLE_BIT       6'b010000
`define CDCM6208_DDR3_DISABLE_BIT       6'b100000

wire    [1:0]   cdcm6208_freq_select;
wire    [5:0]   cdcm6208_freq_disable;
wire            cdcm6208_reset_n;
wire            cdcm6208_i2c_done;

//  Structural coding
assign cdcm6208_reset_n = CPU_RESET_n;
assign cdcm6208_freq_select = `CDCM6208_MODE_1;
assign cdcm6208_freq_disable = 6'b000000;

I2C_CDCM6208_Config I2C_CDCM6208_Config_inst(
    .iCLK(OSC_50_B3B), //50MHZ
    .iRST_N(cdcm6208_reset_n),

    .iFREQ_SELECT(cdcm6208_freq_select),
    .iFREQ_DISABLE(cdcm6208_freq_disable),

    // i2c
    .I2C_SCLK(CLOCK_SCL),
    .I2C_SDAT(CLOCK_SDA),

    .I2C_DONE(cdcm6208_i2c_done)
);
```

For LMK04096B, the Controller will be instantiated in the Quartus II top-level file as listed below:

```
//=======================================================
//  External PLL LMK04906 Configuration ================
//=======================================================

//  Signal declarations
`define FMCA_644M53125_BIT  4'h8   //644.53125 MHz
`define FMCB_644M53125_BIT  4'h2   //644.53125 MHz
`define FMCC_644M53125_BIT  4'h1   //644.53125 MHz
`define FMCD_644M53125_BIT  4'h4   //644.53125 MHz

wire [3:0]      lmk04906_freq_select;
wire            lmk04906_reset_n;
wire            lmk04906_spi_done;

//  Structural coding
assign lmk04906_reset_n = CPU_RESET_n;
assign lmk04906_freq_select = `FMCA_644M53125_BIT;

SPI_LMK04906_Config  SPI_LMK04906_Config_inst(
    .clk50(OSC_50_B4A), //50MHZ
    .rst_n(lmk04906_reset_n),
    .iFREQ_SELECT(lmk04906_freq_select),

    // spi
    .LMK04906_CLK(LMK04906_CLK),
    .LMK04906_DATAIN(LMK04906_DATAIN),
    .LMK04906_LE(LMK04906_LE),
    .SPI_DONE(lmk04906_spi_done)
);
```

If dynamic configuration for the oscillator is required, users need to modify the code according to users' desired behavior.

Chapter 4

Flash Programming

As you develop your own project using the Altera tools, you can program the flash memory device so that your own design loads from flash memory into the FPGA on power up. This chapter will describe how to use Altera Quartus II Programmer Tool to program the common flash interface (CFI) flash memory device on the FPGA board. The Stratix V X GX FPGA development board ships with the CFI flash device preprogrammed with a default factory FPGA configuration for running the Parallel Flash Loader design example.

4.1 CFI Flash Memory Map

Table 4-1 shows the default memory contents of a 1Gb (128MB) CFI flash device. The flash device has a 16-bit data bus. For the factory default code to run correctly and update designs in the user memory, this memory map 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 hardware	**41984**	**0x00040000 – 0x0293FFFF**
User hardware	**41984**	**0x02940000 – 0x0523FFFF**
Factory software	**23424**	**0x05240000 – 0x0691FFFF**
User software and data	**23424**	**0x06920000 – 0x07FFFFFF**

For user application, user hardware can be stored with start address **0x02940000**, and the user's software is suggested to be stored with start address **0x06920000**. The **QuartusII Programmer** is used for programming the flash. Before programming, users need to bundle their *.sof* files and NIOS II *.elf* files together and then convert them into the *.pof* file which is used by the **Convert**

Programming File tool. Before Bundle, user should translate the *.elf* to .hex at first with NIOS II EDS tool and **nios2-elf-objcopy** tool. For convenience, the System CD contains a batch file for file translation and flash programming with users given *.sof* and *.elf* files.

4.2 FPGA Configure Operation

Here is the procedure to enable FPGA configuration from Flash:

1. Please make sure the FPGA configuration data has been stored in the CFI flash.
2. Set the FPGA configuration mode to FPPx16 mode by setting **SW5** MSEL[0:4] as 00000 as shown in **Figure 4-1.**
3. Specify the configuration of the FPGA using the default Factory Configuration or User Configuration by setting **SW4** according to **Figure 4-2.**
4. Power on the FPGA board or press MAX_RST button if board is already powered on
5. When configuration is completed, the green Configure Done LED **D21** will light. If there is error, the red Configure Error LED **D23** will light.

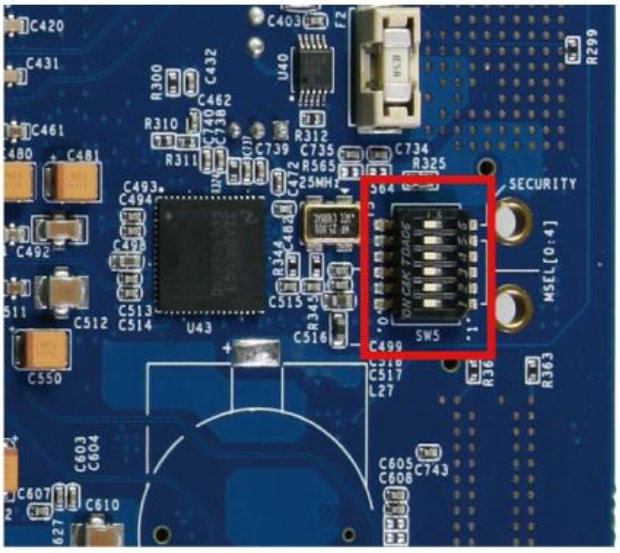


Figure 4-1 MSEL[0:4] set to "00000"

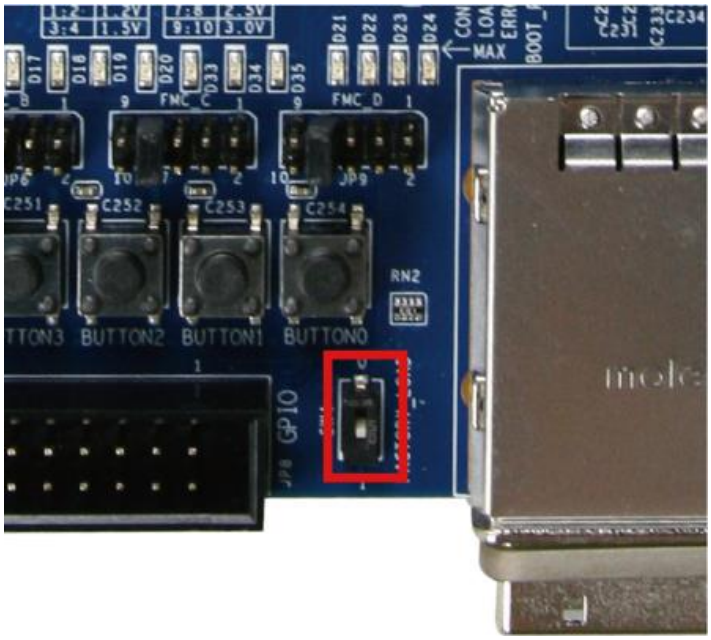


Figure 4-2 Configuration Image Selection

4.3 Flash Programming with Users Design

Users can program the flash memory device so that a custom design loads from flash memory into the FPGA on power up. For convenience, the translation and programming batch files are available on the Demonstrations/flash_programming/flash_programming_batch folder in the System CD. There folder contains five files as shown in Table 4-2

Table 4-2 Content of flash_programming_batch folder

Files Name	*Description*
TR5_PFL.sof	Parallel Flash Loader Configuration File
flash_program.bat	Top batch file to generate and download the .pof file
build_hex.sh	Translate .elf into .hex file
output_file.cof	input file for convert
program_flash.cdf	Input file for download
factory.sof	Factory Hardware design file for Hello Demo
factory.elf	Factory Software design file for Hello Demo
user.sof	User Hardware design file

User.elf	**User Software design file**

To apply the batch file to users' .sof and .elf file, users can change the .sof filename in the **output_file.cof** file and .elf filename in the **build_hex.sh** as shown in Figure 4-3.

```
<sof_data>
        <start_address>02940000</start_address>
        <end_address>0523FFFF</end_address>
        <user_name>Page_1</user_name>
        <page_flags>2</page_flags>
        <bit0>
                <sof_filename>user.sof</sof_filename>
        </bit0>
</sof_data>
```

```
# convert to .hex as uer image
"$SOPC_KIT_NIOS2/nios2_command_shell.sh" elf2flash --base=0x00000000 --end=0x07FFFFFF --reset=0x06920000 --input=user.elf --output=user.flash --boot=$SOPC_KIT_NIOS2/components/altera_nios2/boot_loader_cfi.srec
"$SOPC_KIT_NIOS2/nios2_command_shell.sh" nios2-elf-objcopy  -I srec -O ihex user.flash  user.hex
exit
```

Figure 4-3 Change to usrs' .sof and .elf filename

If your design does not contain a NIOS II processor, users can change the content "Child_OpMask(6 1 0 1 **1** 0 0)" to "Child_OpMask(6 1 0 1 **0** 0 0)" of **program_flash.cdf** file as shown in Figure 4-4.

```
/* Quartus II 64-Bit Version 15.0.0 Build 145 04/22/2015 Patches 0.32 SJ Full Version */
JedecChain;
        FileRevision(JESD32A);
        DefaultMfr(6E);

        P ActionCode(Cfg)
                Device PartName(5SGXEA7N2F45) Path("") File("TR5_PFL.sof") MfrSpec(OpMask(1) SEC_Device(CFI_1GB) Child_OpMask(6 1 0 1 1 0 0) PFLPath("output_file.pof"));

ChainEnd;

AlteraBegin;
        ChainType(JTAG);
AlteraEnd;
```

Figure 4-4 Disable .elf translation and programming

If your design includes a NIOS II processor and the NIOS II program is stored on external memory, users must to perform following items so the NIOS II program can be boot from flash successfully:

1. QSYS should include a Flash controller for the CFI Flash on the development board. Please ensure that the base address of the controller is 0x00, as shown in Figure 4-5.

2. In NIOS II processor options, select FLASH as reset vector memory and specify 0x06920000 as reset vector, as shown in **Figure 4-6**.

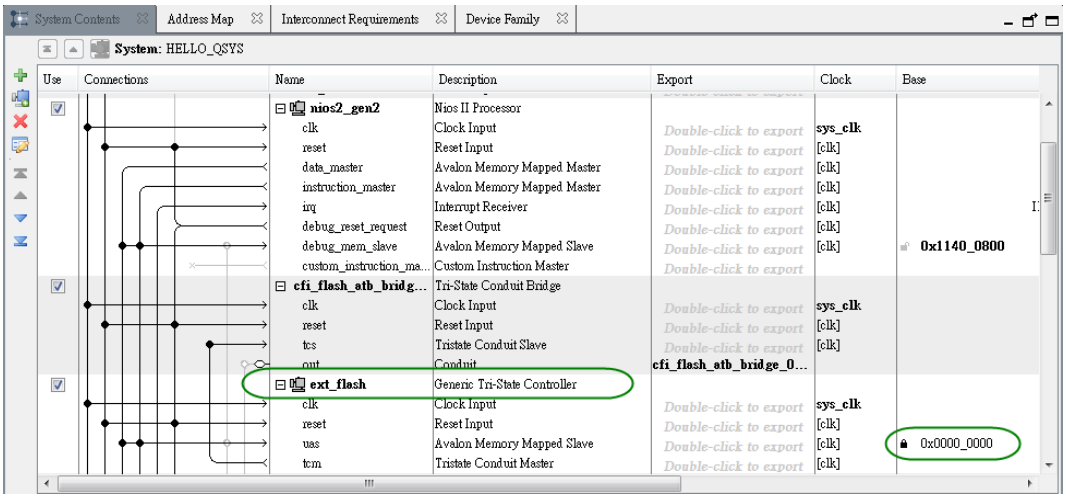


Figure 4-5 Flash Controller Settings in QSYS

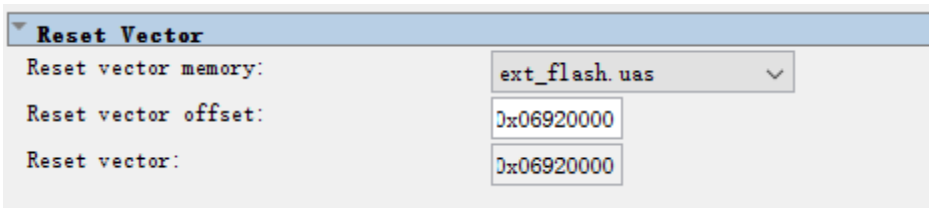


Figure 4-6 Reset Vector Settings for NIOS II Processor

For implementation detail, users can refer the Hello example located in the CD folder:

Demonstrations/ Hello

4.4 Restore Factory Settings

This section describes how to restore the original factory contents to the flash memory device on the FPGA development board. Perform the following instructions:

1. Make sure the Nios II EDS and USB-Blaster II driver are installed.
2. Make sure the FPGA board and PC are connected with a UBS Cable.
3. Power on the FPGA board.
4. Copy the "Demonstrations/flash_programming/factory_programming_batch" folder under the CD to your PC's local drive.
5. Execute the batch file flash_program.bat to start flash programming.
6. Power off the FPGA Board.
7. Set FPGA configure mode as FPPx16 Mode by setting SW5 MSEL[0:4] to 00000.
8. Specify configuration of the FPGA to Factory Hardware by setting the FACTORY_LOAD dip in SW4 to the '0' position.
9. Power on the FPGA Board, and the Configure Done LED should light.

Except for programming the Flash with the default code PFL, the batch file also writes PFL (Parallel Flash Loader) Option Bits data into the address 0x30000. The option bits data specifies 0x2940000 as start address of your hardware design.

The Quartus II program tool **quartus_pgm** programs the Flash based on the Parallel Flasher Loader design in the FPGA.

Chapter 5

Programmable PLL

This chapter introduces TR5 peripheral interface reference designs. It mainly introduces CDCM6208 and LMK04096B chips which are programmable clock generators. We provide two ways (Pure RTL IP and NIOS/Qsys System) respectively to show how to control CDCM6208 and LMK04096B to output desired frequencies. The source codes and tool of these examples are all available on the System CD.

5.1 Configure CDCM6208 and LMK04096B in RTL

There are two clock generators: CDCM6208 and LMK04096B on TR5 FPGA board can provide adjustable frequency reference clock (See **Figure 5-1**) for FMC, SATA , DDR3 and PCIE interfaces, etc. The CDCM6208 clock generator can output six differential frequencies from 100Hz ~ 800Mhz though I2C interface configuration. The LMK04096B clock generator can output four differential frequencies from 100Hz ~ 2600Mhz though SPI interface configuration. This section will show you how to use FPGA RTL IP to configure the PLLs and generate users desired output frequency to each peripheral.

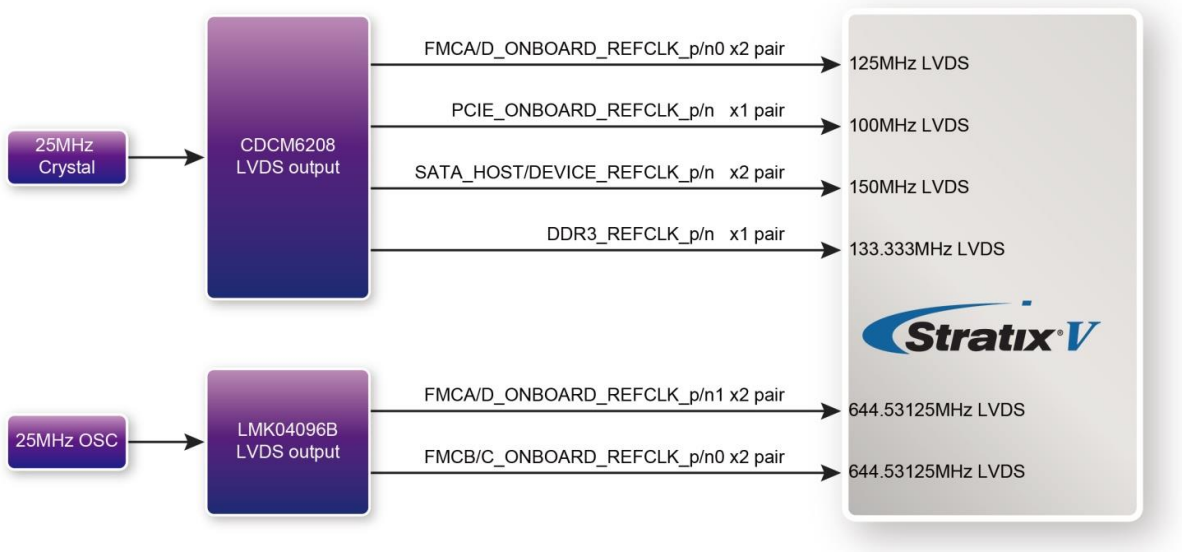


Figure 5-1 Programmable PLL Block diagram

■ Creating CDCM6208 Control IP

System Builder tool (locate in System CD) can be used to help users to set CDCM6208 and LMK04096B to output desired frequencies, and generate a Quartus project with control IP. In System Builder window, when checking the boxes of FMC, SATA, DDR3 and PCIE interfaces, CDCM6208 and LMK04096B corresponding output channels will become available. For example, when select "CDCM6208 PLL" and "LMK0496B PLL"(See Figure 5-2), all the clock channels controlled by the CDCM6208 and LMK0496B will be active and numbers of the frequencies can be chosen.

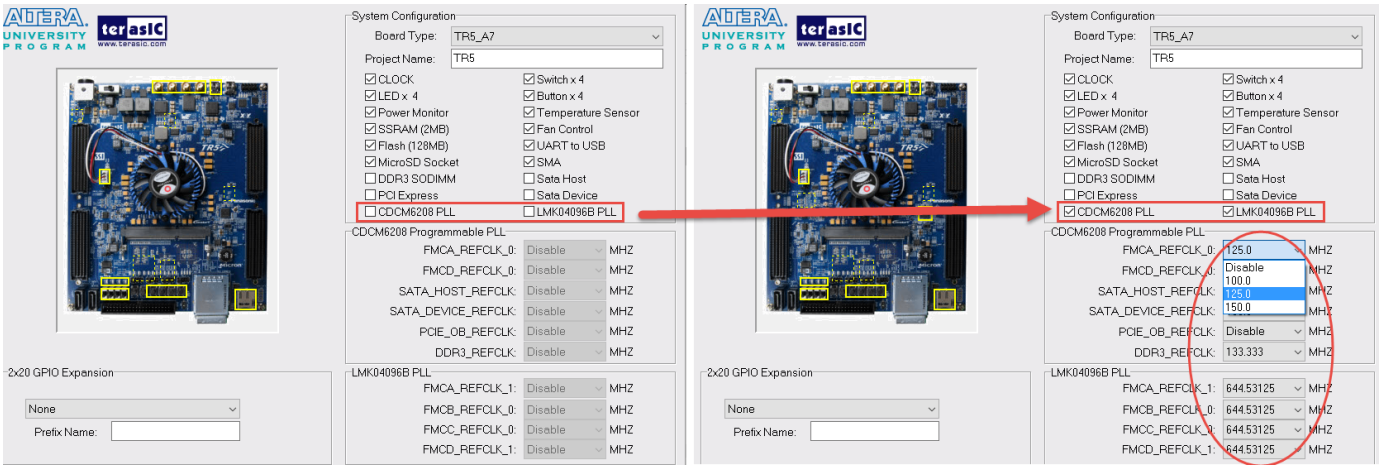


Figure 5-2 Enable CDCM6208 clock on System Builder

```
//=============================================
//  External PLL CDCM6208 Configuration======
//  Signal declarations
`define CDCM6208_MODE_0   2'd0
`defineCDCM6208_MODE_1   2'd1
`define CDCM6208_MODE_2   2'd2
`define CDCM6208_FMCA_DISABLE_BIT          6'b000001
`define CDCM6208_FMCD_DISABLE_BIT          6'b000010
`define CDCM6208_SATA_HOST_DISABLE_BIT 6'b000100
`define CDCM6208_SATA_DEVICE_DISABLE_BIT
        6'b001000
`define CDCM6208_PCIE_DISABLE_BIT          6'b010000
`define CDCM6208_DDR3_DISABLE_BIT          6'b100000
wire      [1:0]     cdcm6208_freq_select;
wire      [5:0]     cdcm6208_freq_disable;
wire                cdcm6208_reset_n, cdcm6208_i2c_done;
//  Structural coding
assign cdcm6208_reset_n = CPU_RESET_n;
assign cdcm6208_freq_select = `CDCM6208_MODE_0;
assign                    cdcm6208_freq_disable                    =
`CDCM6208_FMCA_DISABLE_BIT                                         |
`CDCM6208_FMCD_DISABLE_BIT                                         |
`CDCM6208_SATA_HOST_DISABLE_BIT                                    |
`CDCM6208_SATA_DEVICE_DISABLE_BIT                                  |
`CDCM6208_PCIE_DISABLE_BIT                                         |
`CDCM6208_DDR3_DISABLE_BIT;
I2C_CDCM6208_Config I2C_CDCM6208_Config_inst(
        .iCLK(OSC_50_B3B), //50MHZ
        .iRST_N(cdcm6208_reset_n),
        .iFREQ_SELECT(cdcm6208_freq_select),
        .iFREQ_DISABLE(cdcm6208_freq_disable),
        // i2c
        .I2C_SCLK(CLOCK_SCL),
        .I2C_SDAT(CLOCK_SDA),
        .I2C_DONE(cdcm6208_i2c_done) );
```

```
//=============================================
//  External PLL LMK04906 Configuration======
//=============================================
//  Signal declarations
`define FMCA_644M53125_BIT   4'h8     //644.53125 MHz
`define FMCB_644M53125_BIT   4'h2     //644.53125 MHz
`define FMCC_644M53125_BIT   4'h1     //644.53125 MHz
`define FMCD_644M53125_BIT   4'h4     //644.53125 MHz

wire [3:0]          lmk04906_freq_select;
wire                lmk04906_reset_n;
wire                lmk04906_spi_done;
//  Structural coding
assign lmk04906_reset_n = CPU_RESET_n;
assign lmk04906_freq_select = 4'b0000;

SPI_LMK04906_Config   SPI_LMK04906_Config_inst(
        .clk50(OSC_50_B4A), //50MHZ
        .rst_n(lmk04906_reset_n),
        .iFREQ_SELECT(lmk04906_freq_select),
        // spi
        .LMK04906_CLK(LMK04906_CLK),
        .LMK04906_DATAIN(LMK04906_DATAIN),
        .LMK04906_LE(LMK04906_LE),
        .SPI_DONE(lmk04906_spi_done)
);
```

When user finish the clock setting, click "**Generate**" button, then, open the Quartus Project generated by the System Builder, the control IPs for CDCM6208 and LMK04096B can be found in the top level file.

If the output frequency doesn't need to be modified, users can just add their own User Logic and compile it, then the CDCM6208 and LMK04096B can output the desired frequencies. At the same time, System Builder will set Clock constrain according user's preset frequency in a SDC file (as shown in **Figure 5-3**).

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

#**************************************************************
# Create Clock
#**************************************************************
create_clock -period "50.000000 MHz" [get_ports OSC_50_B3B]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B4A]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B4D]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B7A]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B7D]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B8A]
create_clock -period "50.000000 MHz" [get_ports OSC_50_B8D]
create_clock -period "322.265625 MHz" [get_ports FMCA_ONBOARD_REFCLK_p[1]]
create_clock -period "322.265625 MHz" [get_ports FMCB_ONBOARD_REFCLK_p[0]]
create_clock -period "644.531250 MHz" [get_ports FMCC_ONBOARD_REFCLK_p[0]]
create_clock -period "644.531250 MHz" [get_ports FMCD_ONBOARD_REFCLK_p[1]]
create_clock -period "133.333328 MHz" [get_ports DDR3_REFCLK_p]
create_clock -period "125.000000 MHz" [get_ports FMCA_ONBOARD_REFCLK_p[0]]
create_clock -period "125.000000 MHz" [get_ports FMCD_ONBOARD_REFCLK_p[0]]
create_clock -period "100.000000 MHz" [get_ports PCIE_ONBOARD_REFCLK_p]
create_clock -period "150.000000 MHz" [get_ports SATA_DEVICE_REFCLK_p]
create_clock -period "150.000000 MHz" [get_ports SATA_HOST_REFCLK_p]
```

Figure 5-3 SDC file created by System Builder

■ Using CDCM6208 control IP

Table 5-1 lists the instruction ports of CDCM6208 Controller IP.

Table 5-1 CDCM6208 Controller Instruction Ports

Port	Direction	Description
iCLK	input	System Clock (50Mhz)
iRST_n	input	Synchronous Reset (0: Module Reset, 1: Normal)
iFREQ_DISABLE	input	Disable the CDCM6208 output frequency
iFREQ_SELECT	input	Setting CDCM6208 Output Channel Frequency combination mode

I2C_DONE	output	CDCM6208 Configuration status (0: Configuration in Progress, 1: Configuration Complete)
I2C_DATA	inout	I2C Serial Data to/from CDCM6208
I2C_CLK	output	I2C Serial Clock to CDCM6208

As shown in Table 5-2, the CDCM6208 control IPs have preset three output frequency combinations, if users want to change frequency, users can fill in the input port " iFREQ_SELEC" with a desired Frequency combination mode and recompile the project. For example, in CDCM6208 control IP, change

assign cdcm6208_freq_select = `CDCM6208_MODE_0;

to *assign cdcm6208_freq_select = `CDCM6208_MODE_1;*

Recompile project, the CDCM6208 output frequency combination will change from mode 0 to 1.

Table 5-2 CDCM6208 Controller Frequency Setting

iFREQ_SELECT MODE Setting	FMCA Freq(MHz)	FMCD Freq(MHz)	PCIE Freq(MHz)	SATA_HOST Freq(MHz)	SATA _DEVICE Freq(MHz)	DDR3 Freq(MHz)
2'b00	100	100	100	150	150	133.333
2'b01	125	125	100	150	150	133.333
2'b10	150	150	100	150	150	133.333

Users can also dynamically modify the input parameters, and input a positive edge trigger for “iRST_N”, and then the CDCM6208 output frequency can be modified.

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

■ Modify Clock Parameter For Your Own Frequency

If the CDCM6208 control IP build-in frequencies are not the users’ desired frequencies, users can refer to the below steps to modify control the IP register parameter settings to modify the IP to output a desired frequency.

1. Firstly, download ClockBuider Pro Software(See **Figure 5-4**), which is provided by TI. This tool can help users to set the CDCM6208’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://www.ti.com/lit/sw/scac134d/scac134d.zip

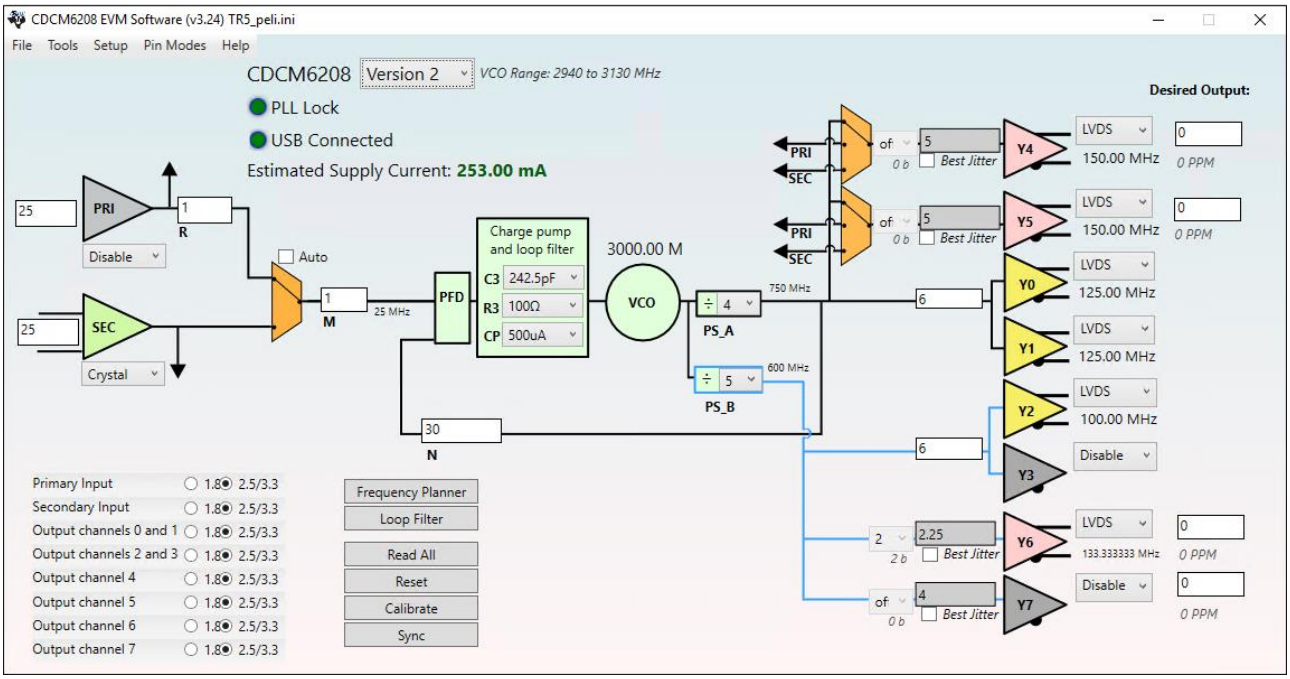


Figure 5-4 ClockBuilder Pro Wizard

2. After the installation, select CDCM6208, and configure the input frequency and output frequency as shown in **Figure 5-5**.

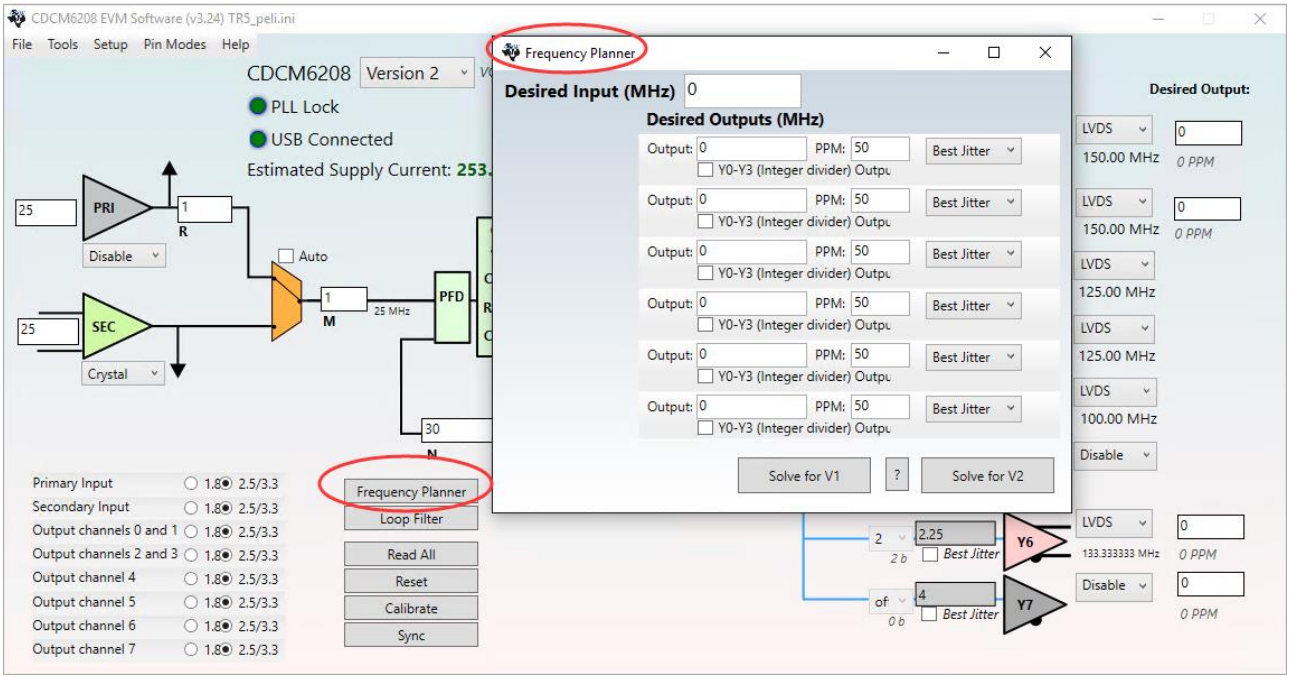


Figure 5-5 Define Output Clock Frequencies on CDCM6208 EVM Software

3. After the setting is completed, CDCM6208 EVM Software generates a register table, which contains users setting frequency corresponding register value (See **Figure 5-6**).

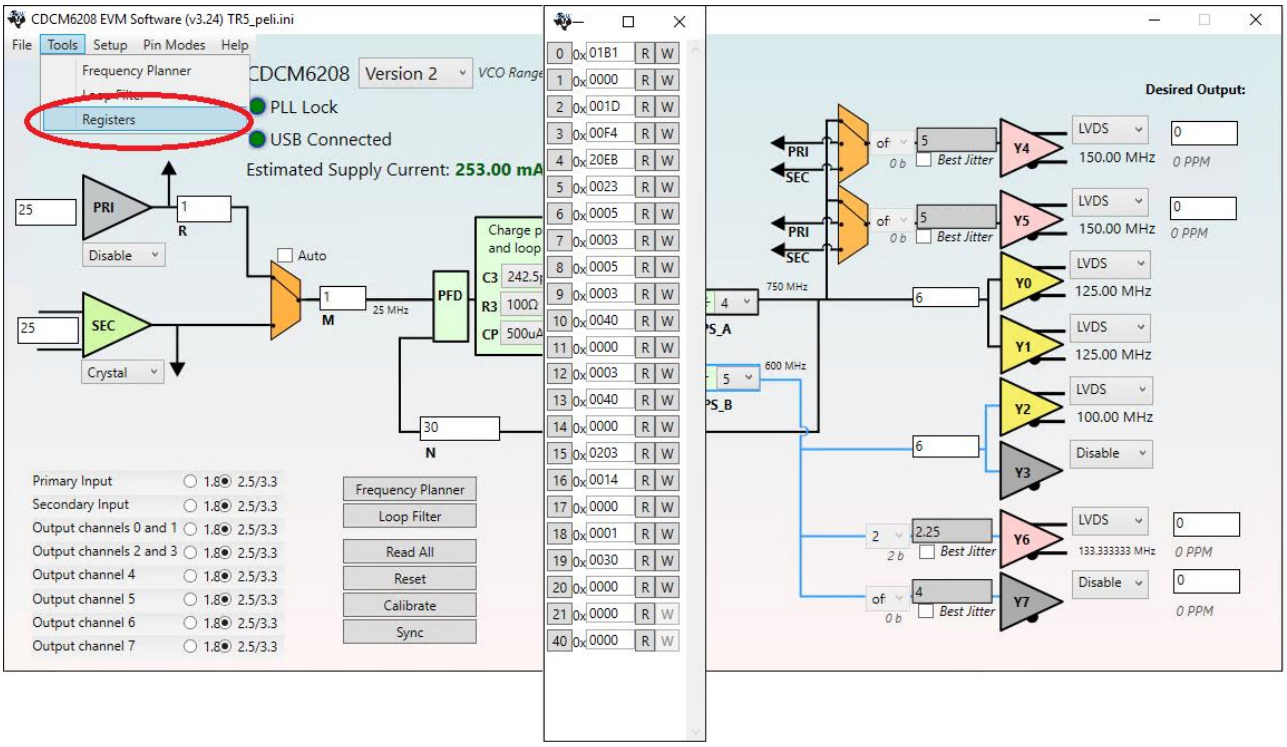


Figure 5-6 Open Register Table on CDCM6208 EVM Software

4. Open CDCM6208 control IP sub-module "I2C_CDCM6208_Config.v "as shown in **Figure 5-7**, refer to Register Table to modify all the sub-module corresponding register values (See **Figure 5-8**).

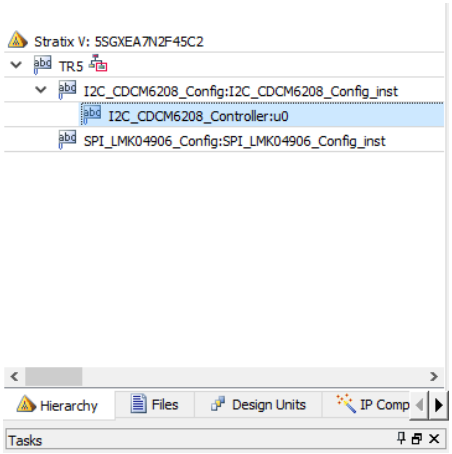


Figure 5-7 Sub-Module file "I2C_CDCM6208_Config.v"

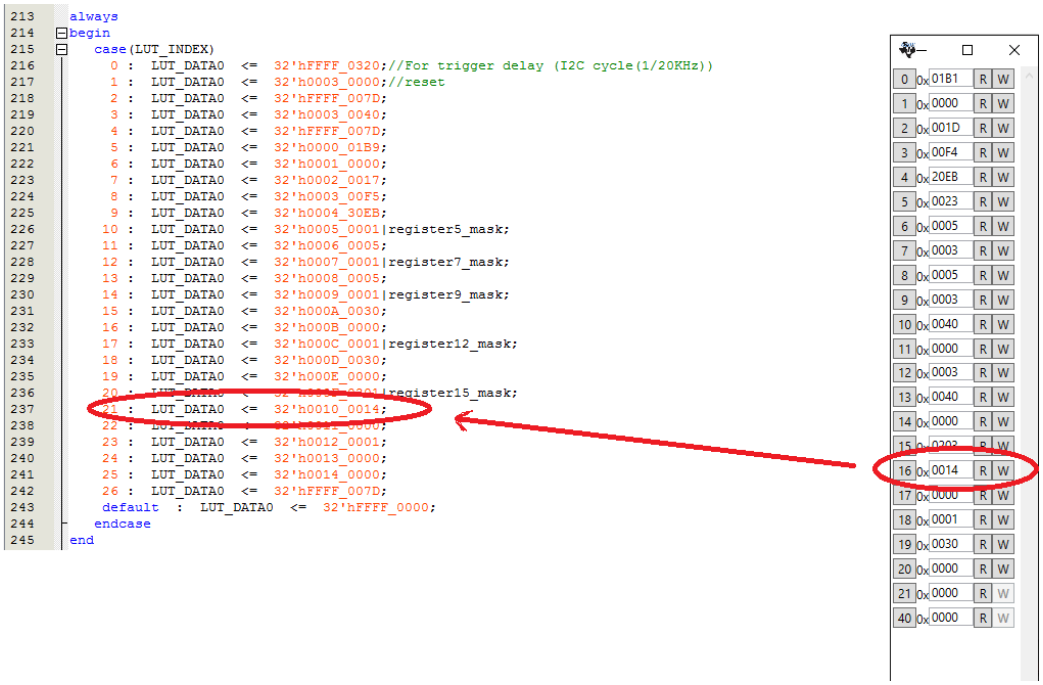


Figure 5-8 Modify CDCM6208 Control IP Base on Register Table

After modifying and compiling, CDCM6208 can output new frequencies according to the users' setting.

Note :

(1) No need to modify all Design Report parameters in I2C_CDCM6208_Config.v, users can ignore parameters which have nothing to do with the frequency setting.

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

■ Using LMK04096B control IP

Table 5-3 lists the instruction ports of LMK04096B Controller IP.

Table 5-3 LMK04096B Controller Instruction Ports

Port	Direction	Description
clk50	input	System Clock (50Mhz)
rst_n	input	Synchronous Reset (0: Module Reset, 1: Normal)
iFREQ_SELECT	input	Setting LMK04096B Output Channel Frequency combination mode
SPI_DONE	output	LMK04096B Configuration status (0: Configuration in Progress, 1: Configuration Complete)
LMK04906_CLK	output	SPI Clock to LMK04096B
LMK04906_DATAIN	output	SPI Data to CDCM6208
LMK04906_LE	output	SPI Latch Enable

As shown in Table 5-4, the LMK04096B control IP has preset three output frequency combinations, if users want to change frequencies, users can fill in the input port "iFREQ_SELEC" with a desired Frequency combination mode and recompile the project. For example, in LMK04096B control IP, change

assign lmk04906_freq_select = `FMCC_644M53125_BIT;

to

assign lmk04906_freq_select = `FMCC_644M53125_BIT | `FMCD_644M53125_BIT;

Recompile project, the LMK04096B output frequency combination will change from mode 1 to 5.

Table 5-4 LMK04096B Controller Frequency Setting

iFREQ_SELECT MODE Setting	FMCA Freq(MHz)	FMCD Freq(MHz)	FMCB Freq(MHz)	FMCC Freq(MHz)
4'd0	322.265625	322.265625	322.265625	322.265625
4'd1	322.265625	322.265625	322.265625	644.53125
4'd2	322.265625	322.265625	644.53125	322.265625
4'd3	322.265625	322.265625	644.53125	644.53125
4'd4	322.265625	644.53125	322.265625	322.265625
4'd5	322.265625	644.53125	322.265625	644.53125
4'd6	322.265625	644.53125	644.53125	322.265625
4'd7	322.265625	644.53125	644.53125	644.53125
4'd8	644.53125	322.265625	322.265625	322.265625
4'd9	644.53125	322.265625	322.265625	644.53125
4'd10	644.53125	322.265625	644.53125	322.265625

4'd11	644.53125	322.265625	644.53125	644.53125
4'd12	644.53125	644.53125	322.265625	322.265625
4'd13	644.53125	644.53125	322.265625	644.53125
4'd14	644.53125	644.53125	644.53125	322.265625
4'd15	644.53125	644.53125	644.53125	644.53125

Users can also dynamically modify the input parameters, and input a positive edge trigger for "**rst_n**", then, LMK04096B 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 LMK04096B control IP build-in frequencies are not the users' desired, users can refer to the below steps to modify control IP register parameter settings to modify the IP to output a desired frequency.

5. Firstly, download "CodeLoader" Software(See **Figure 5-9**), which is provided by TI. This tool can help users to set the LMK04096B'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://www.ti.com/tool/codeloader?keyMatch=CodeLoader&tisearch=Search-EN-Everything

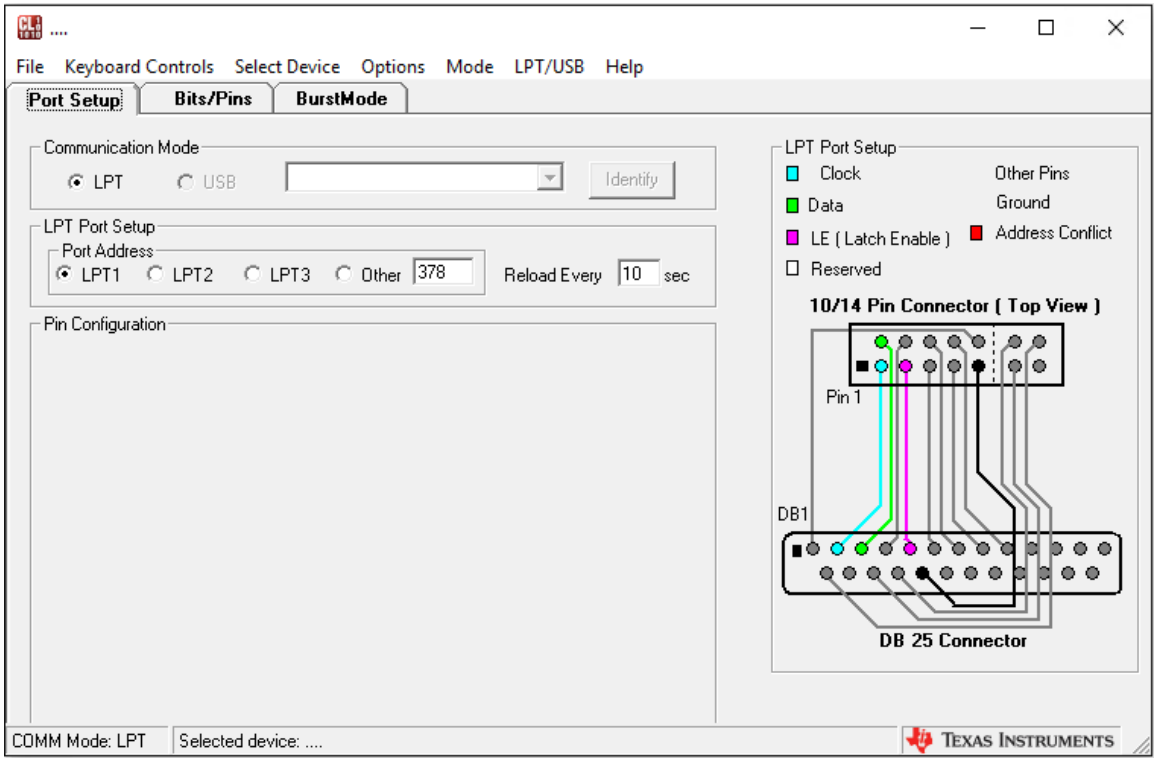


Figure 5-9 Codeloader

6. After the installation, select LMK04096B, and configure the input frequency and the output frequency as shown in **Figure 5-10**.

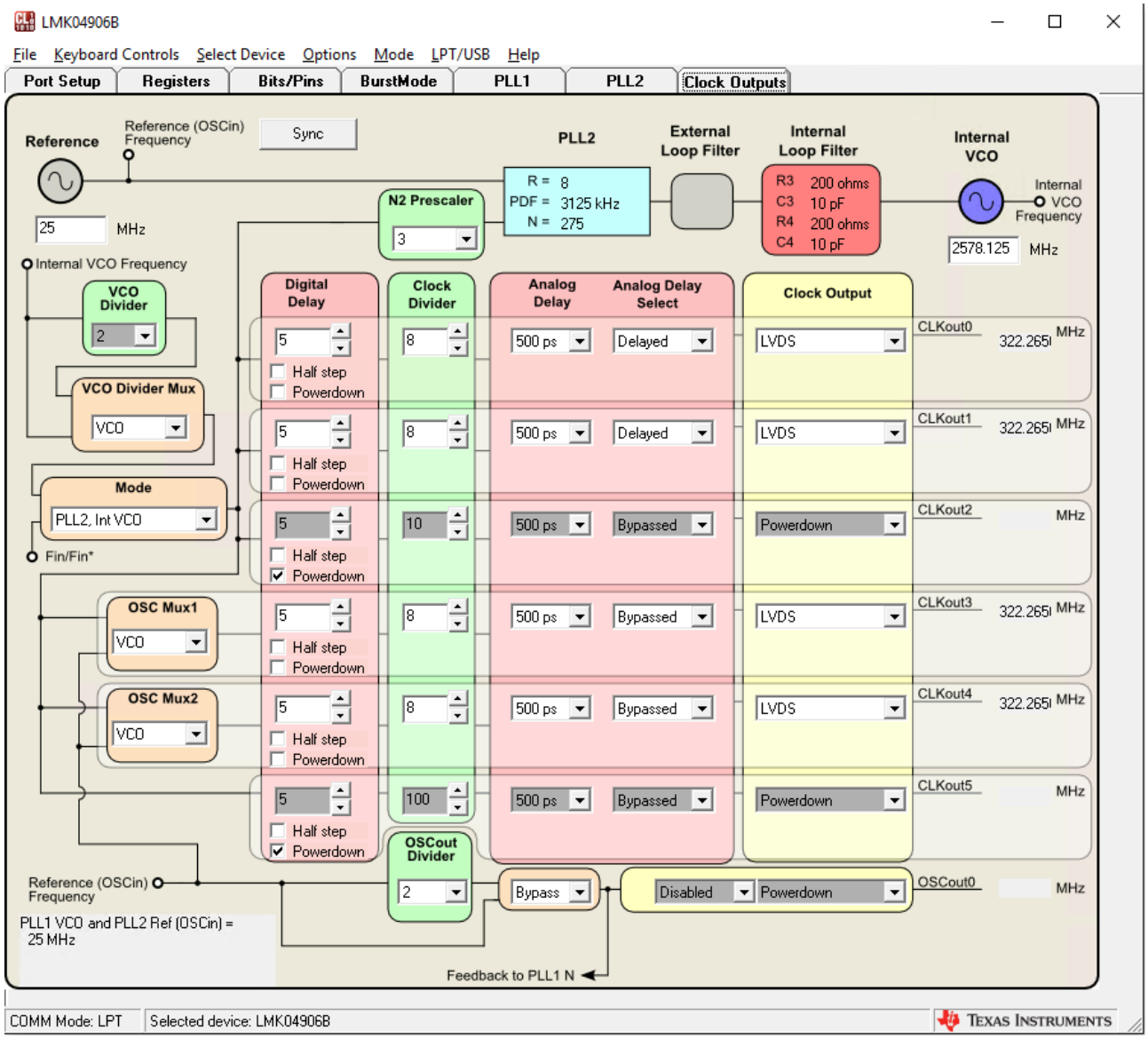


Figure 5-10 Define Output Clock Frequencies on Codeloader

7. After the setting is completed, CodeLoader generates a Register Table, which contains users setting frequency corresponding register values (See **Figure 5-11**).

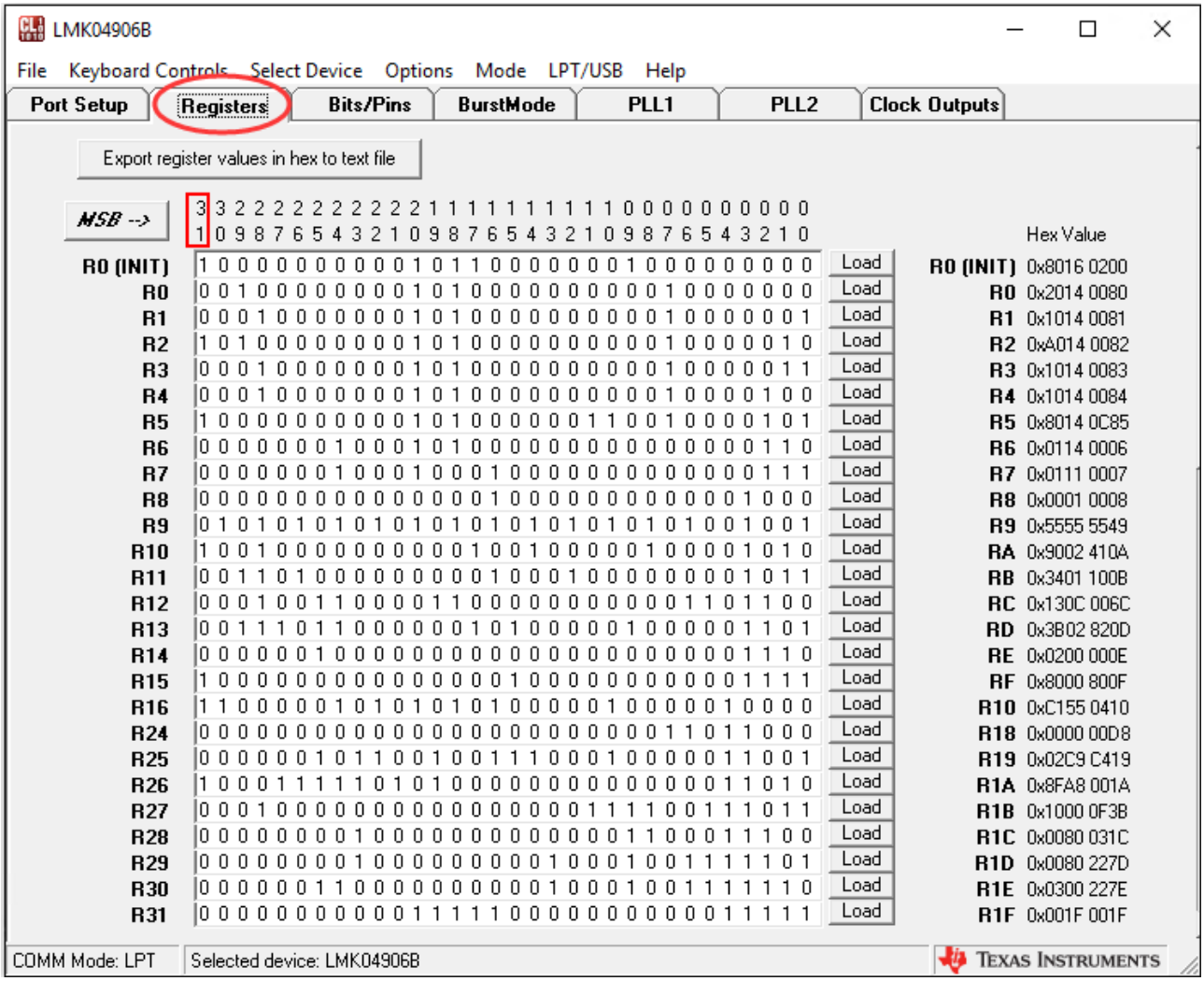


Figure 5-11 Open Register Table on Codeloder

8. Open LMK04096B control IP sub-module "SPI_LMK04906_Config.v "as shown in **Figure 5-12**, refer Design Report parameter to modify all the sub-module corresponding register values (See **Figure 5-13**).

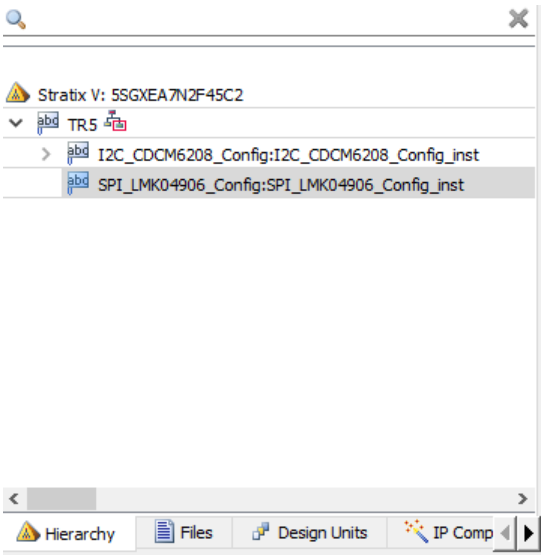


Figure 5-12 Sub-Module file " SPI_LMK04906_Config.v"

```
////////////////////////////////////////////////////////////////////
////////////////////   Registers map    ////////////////////////////
//  for FMCA/B/C/D 644.53125MHz
always
begin
   case(lut_index)
     0 :  lut_data15  <=  33'h1_000061A8;//bit32 high for trigger delay
     1 :  lut_data15  <=  33'h0_80160200;
     2 :  lut_data15  <=  33'h1_000061A8;
     3 :  lut_data15  <=  33'h0_20140080;
     4 :  lut_data15  <=  33'h1_000061A8;
     5 :  lut_data15  <=  33'h0_10140081;
     6 :  lut_data15  <=  33'h0_A0140082;
     7 :  lut_data15  <=  33'h0_00140083;
     8 :  lut_data15  <=  33'h0_00140084;
     9 :  lut_data15  <=  33'h0_80140C85;
    10 :  lut_data15  <=  33'h0_01140006;
    11 :  lut_data15  <=  33'h0_01110007;
    12 :  lut_data15  <=  33'h0_00010008;
    13 :  lut_data15  <=  33'h0_55555549;
    14 :  lut_data15  <=  33'h0_9002410A;
    15 :  lut_data15  <=  33'h0_3401100B;
    16 :  lut_data15  <=  33'h0_130C006C;
    17 :  lut_data15  <=  33'h0_3B02820D;
    18 :  lut_data15  <=  33'h0_0200000E;
    19 :  lut_data15  <=  33'h0_8000800F;
    20 :  lut_data15  <=  33'h0_C1550410;
    21 :  lut_data15  <=  33'h0_000000D8;
    22 :  lut_data15  <=  33'h0_02C9C419;
    23 :  lut_data15  <=  33'h0_8FA8001A;
    24 :  lut_data15  <=  33'h0_10000F3B;
    25 :  lut_data15  <=  33'h0_0080031C;
    26 :  lut_data15  <=  33'h0_0800227D;
    27 :  lut_data15  <=  33'h0_0300227E;
    28 :  lut_data15  <=  33'h0_001F001F;
    29 :  lut_data15  <=  33'h1_000061A8;
    default  :  lut_data15  <=  33'h1_00000000;
   endcase
end
```

	Hex Value
R0 (INIT)	0x8016 0200
R0	0x2014 0080
R1	0x1014 0081
R2	0xA014 0082
R3	0x1014 0083
R4	0x1014 0084
R5	0x8014 0C85
R6	0x0114 0006
R7	0x0111 0007
R8	0x0001 0008
RA	0x9002 410A
RB	0x3401 100B
RC	0x130C 006C
RD	0x3B02 820D
RE	0x0200 000E
RF	0x8000 800F
R10	0xC155 0410
R18	0x0000 00D8
R19	0x02C9 C419
R1A	0x8FA8 001A
R1B	0x1000 0F3B
R1C	0x0080 031C
R1D	0x0080 227D
R1E	0x0300 227E
R1F	0x001F 001F

Figure 5-13 Modify LMK04096B Control IP Based on the Design Report

After modifying and compiling, LMK04096B can output new frequencies according to the users' settings.

Note:

(1) No need to modify all Design Report parameters in SPI_LMK04906_Config.v, users can ignore parameters which have nothing to do with the frequency setting

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

5.2 Nios II control for PLL/Temperature/Power

This demonstration shows how to use the Nios II processor to program two programmable oscillators (CDCM6208 and LMK04096B) 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.

■ System Block Diagram

Figure 5-14 shows the system block diagram of this demonstration. The system requires a 50 MHz clock provided from the board. The three peripherals (including temperature sensor, CDCM6208 and INA230) are all controlled by the Nios II through the PIO controller, the I2C pins from chip are connected to Qsys System Interconnect Fabric through the PIO controllers, and all of them are programmed through the I2C protocol which is implemented in the C code. The LMK04096B is controlled by the Nios II through the SPI controller. The Nios II program is running in the on-chip memory.

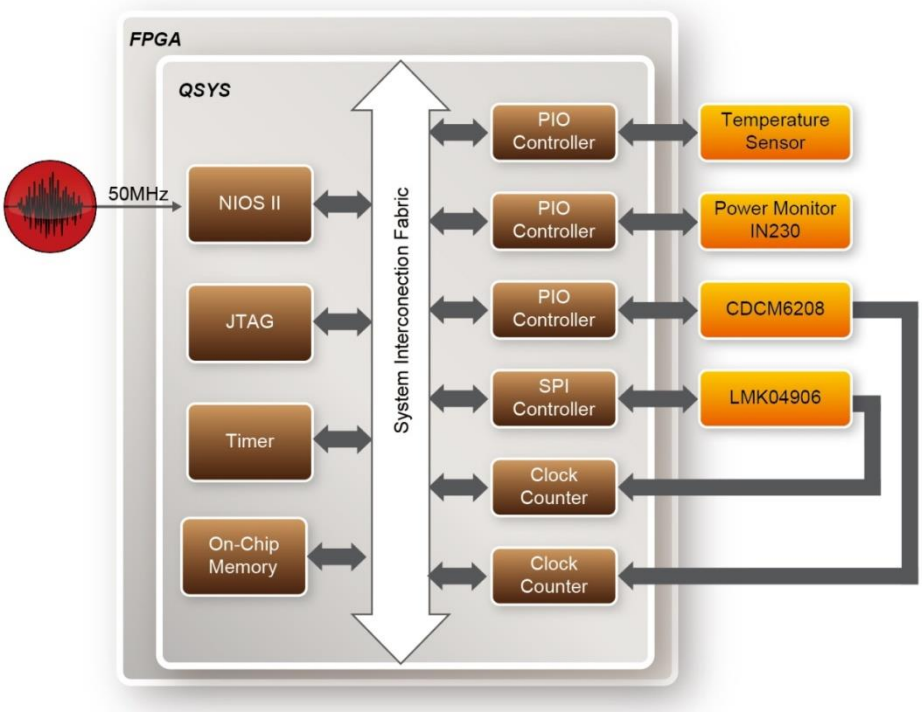


Figure 5-14 Block diagram of the Nios II Basic Demonstration

The program provides a menu in the nios-terminal, as shown in **Figure 5-15** to provide an interactive interface. With the menu, users can perform the test for the temperatures sensor, external PLL and power monitor. Note, pressing 'ENTER' should be followed with the choosing of a number.

Figure 5-15 Menu of Demo Program

In temperature test, the program will display the local temperature and the remote temperature. The remote temperature is the FPGA temperature, and the local temperature is the board temperature where the temperature sensor located (or at the temperature sensors location).

A power monitor IC (INA230AIRGTT) embedded on the board can monitor TR5 real-time current

and power. This IC can work out current/power value as multiplier and divider are embedded in it. There is a shunt resistor R35 (RSHUNT =0.003 Ω) for INA230AIRGTT in the circuit, when the TR5 board is powered on, there will be a voltage drop (named Shut Voltage) on R35. Based on sense resistors, the program of the power monitor can calculate the associated voltage, current and power consumption from the IN230 through the I2C interface. Please note the device I2C address is 0x80.

In the external PLL programming test, the program will program the PLL first, and subsequently it will use the CLOCK_COUNTER IP to count the clock count in a specified period to check whether the output frequency is changed as configured. To avoid a Quartus II compilation error, dummy transceiver controllers are created to receive the clock from the external PLL. Users can ignore the functionality of the transceiver controller in the demonstration.
For CDCM6208 programming, Please note the device I2C address is 0xA8. The program can control the CDCM6208 to configure the output frequency of FMCA/FMCD/DDR3/PCIE/SATA REFCLK according to your choice.
For LMK04096B programming, the program can control the LMK04096B to configure the output frequency of the FMCA/FMCB/FMCC/FMCD REFCLK to 644.53125MHz.

■ Demonstration File Locations

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

■ Demonstration Setup and Instructions

- Make sure Quartus II 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 “NIOS_BASIC_DEMO.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 temperature test, please input key ‘0’ and press ‘Enter’ in the nios-terminal, , as shown in **Figure 5-16**.
- For power monitor test, please input key ‘1’ and press ‘Enter’ in the nios-terminal, the Nios II console will display the current values of voltage, current and power as shown in **Figure 5-17**.
- For programmable PLL CDCM6208 test, please input key ‘2’ and press ‘Enter’ in the nios-terminal first, then select the desired output frequency of FMCA/FMCD/DDR3/PCIE/SATA, as shown in **Figure 5-18**.
- For programmable PLL LMK04906 test, please input key ‘3’ and press ‘Enter’ in the nios-terminal, as shown in **Figure 5-19**.

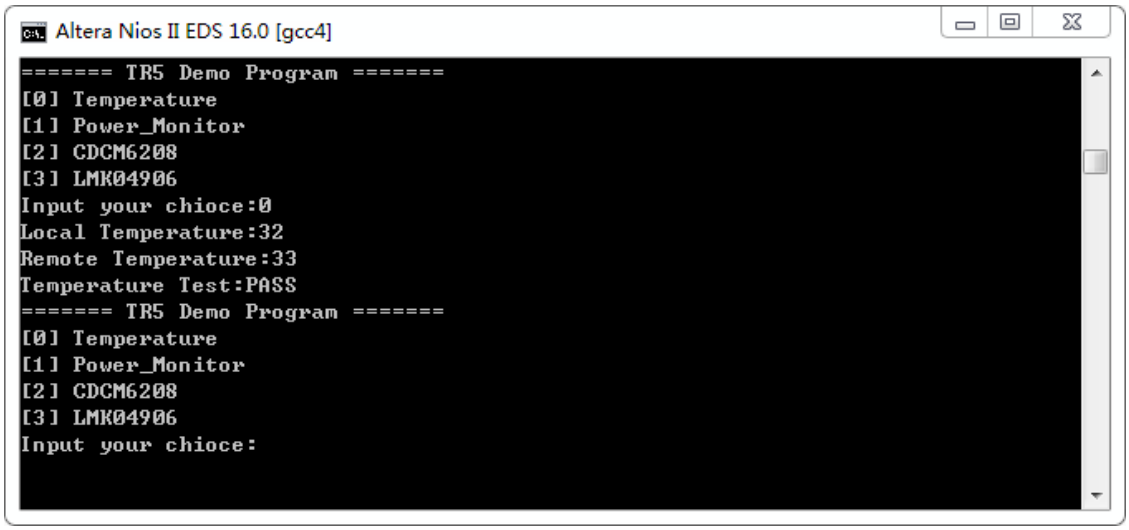


Figure 5-16 Temperature Demo

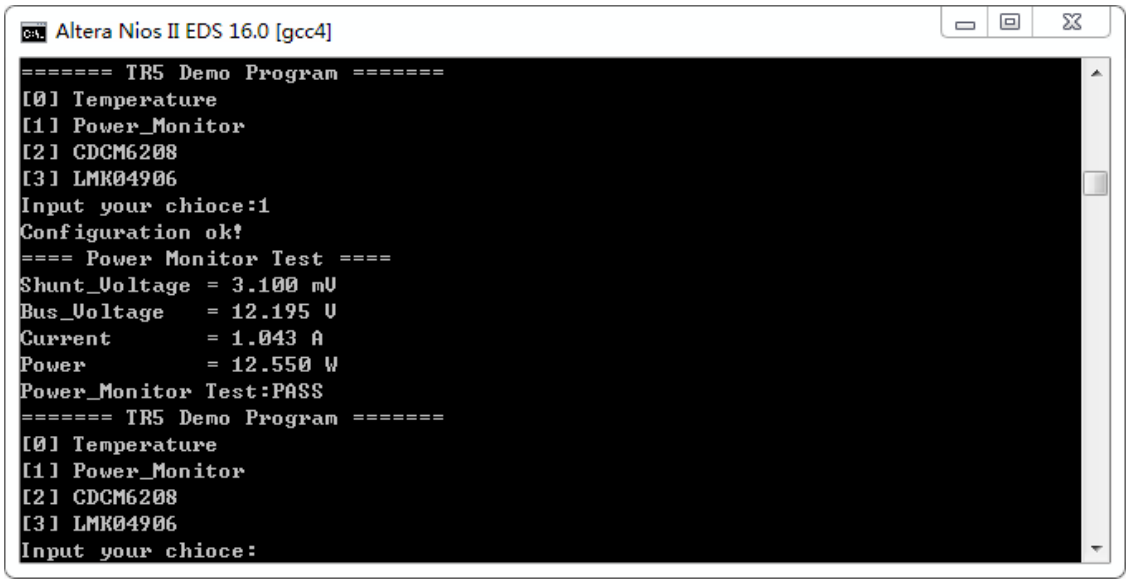


Figure 5-17 power monitor Demo

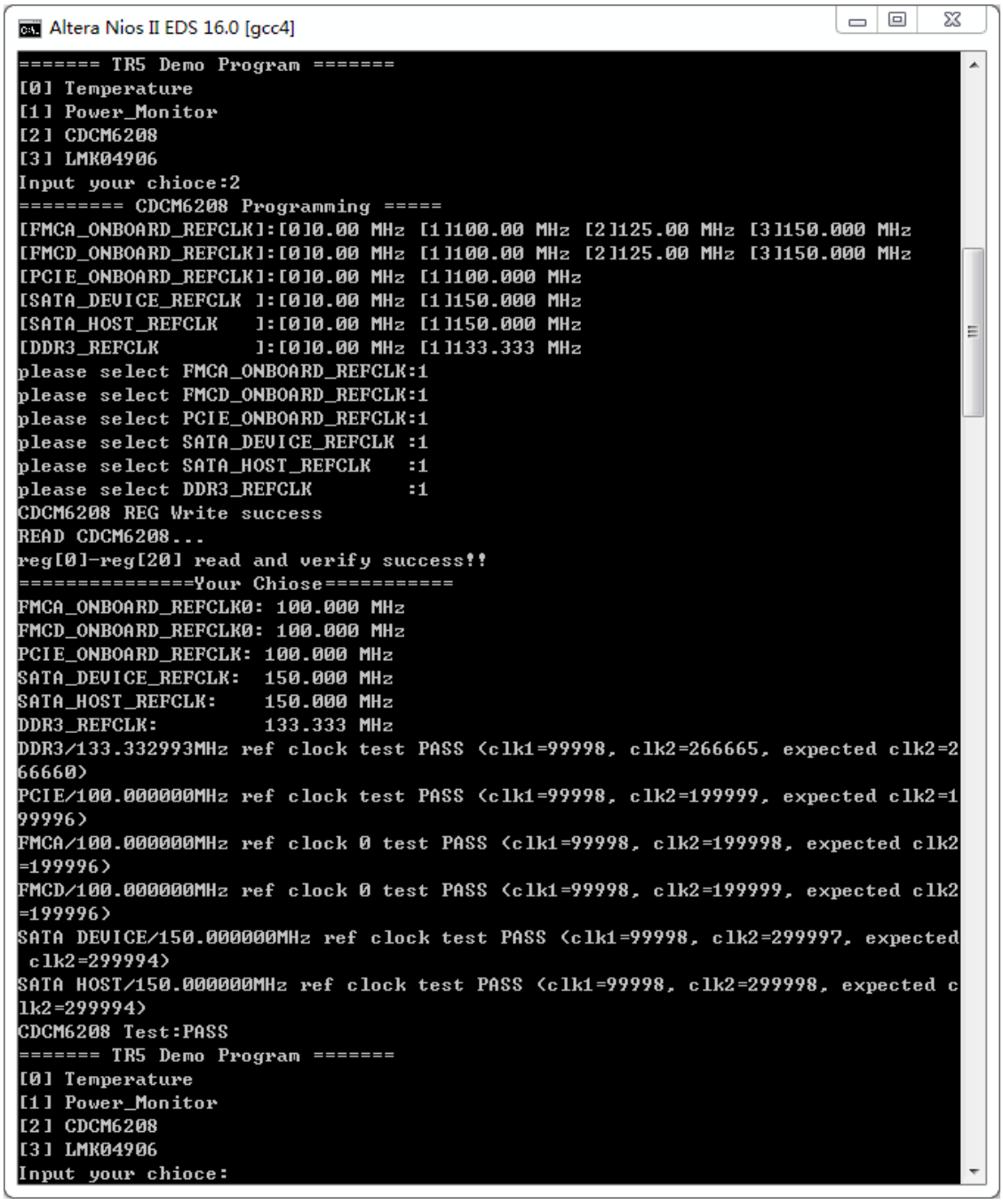


Figure 5-18 CDCM6208 Demo

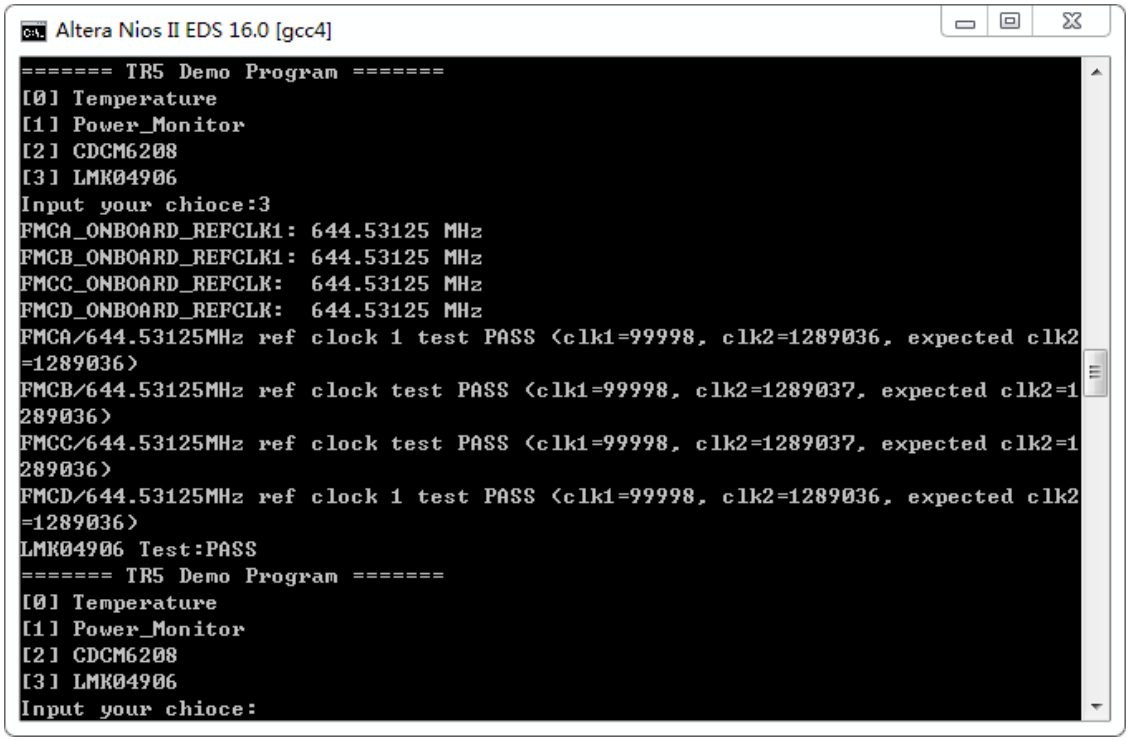


Figure 5-19 LMK04906 Demo

Chapter 6

Examples of Advanced Demonstration

This chapter introduces several advanced designs that demonstrate Stratix V GX features using the TR5 board. The provided designs include the major features on the board; such as the DDR3, fan control and USB to Uart interface. For each demonstration the Stratix V GX FPGA configuration file is provided, as well as full source code in Verilog HDL. All of the associated files can be found in the demonstrations folder from the TR5 System CD.

6.1 Flash and SSRAM Test

In this demonstration hardware and software designs are provided to illustration how to perform Flash and SSRAM memory access in QSYS.

■ Function Block Diagram

Figure 6-1 shows the System block diagram of this demonstration. The QSYS system requires one 50MHz clock source. There are two Generic Tristate Controllers in this demonstration. One Generic Tri-state Controller is configured as a 1Gb Flash controller and another one is configured as 16Mb SSRAM controller. The Tristate Conduit Pin Sharer multiplexes between the signals of the two connected tri-state controllers. Nios II processor is used to perform memory test. The Nios II program is running on the On-Chip memory..

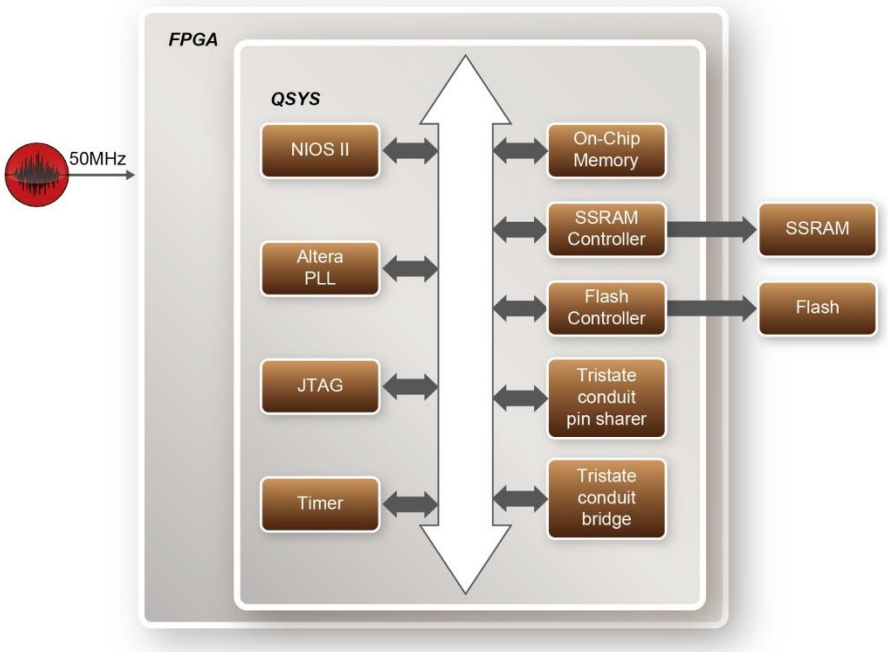


Figure 6-1 Function Block Diagram of the Flash and SSRAM Test

The system flow is controlled by a Nios II program. For Flash test, the Nios II program erases the whole block at first. The Nios II program writes test words into the whole size of SSRAM and Flash at first. Then, it calls Nios II system function *alt_dcache_flush_all()* to make sure that all data has been written to SSRAM and Flash. Finally, it reads back all data from SSRAM and Flash for data verification. The program will show the progress in JTAG-Terminal when writing or reading data from or to SSRAM and Flash. When verification is completed, the result is displayed in the JTAG-Terminal.

Design Tools

- Quartus II 16.0
- Nios II Eclipse 16.0

Demonstration Source Code

- Quartus II project directory: TR5_Flash_SSRAM
- Nios II Eclipse: TR5_Flash_SSRAM\software

Nios 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: TR5_Flash_SSRAM\demo_batch

The demo batch file includes following files:

- Batch file for USB-Blaster II: test.bat, test.sh
- FPGA configure file: TR5_Flash_SSRAM.sof
- Nios II program: mem_test.elf

Demonstration Setup

Please follow below procedures to setup the demonstrations.

- Make sure Quartus II and Nios II are installed on your PC.
- Power on the FPGA board.
- Use the USB Cable to connect the PC and the FPGA board and install the USB Blaster II driver if necessary.
- Execute the demo batch file “test.bat” under the folder TR5_Flash_SSRAM\demo_batch.
- After the Nios II program is downloaded and executed successfully, a prompt message will be displayed in the nios2-terminal.
- The program will display progressing and resulting information, as shown in **Figure 6-2**.

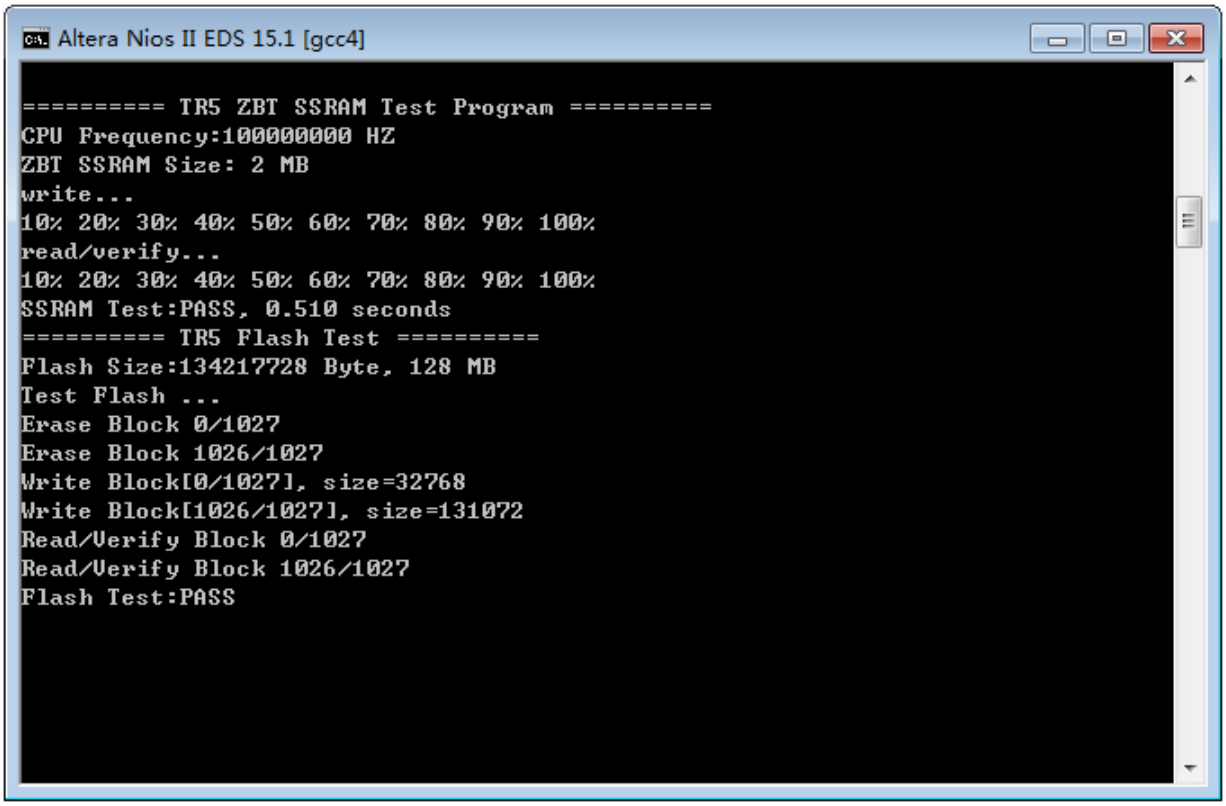


Figure 6-2 Display progress and result information for the flash and SSRAM test demo

6.2 DDR3 SDRAM Test

This demonstration performs a memory test function for two DDR3-SDRAM SO-DIMMs by RTL code on the TR5. The memory size of each DDR3 SDRAM SO-DIMM used in this test is 2 GB.

■ Function Block Diagram

Figure 6-3 shows the function block diagram of this demonstration. There are two DDR3 SDRAM controllers. The controller uses 50 MHz as a reference clock. It generates one 800MHz clock as memory clock from the FPGA to the memory and the controller itself runs at quarter-rate in the FPGA i.e. 200MHz.

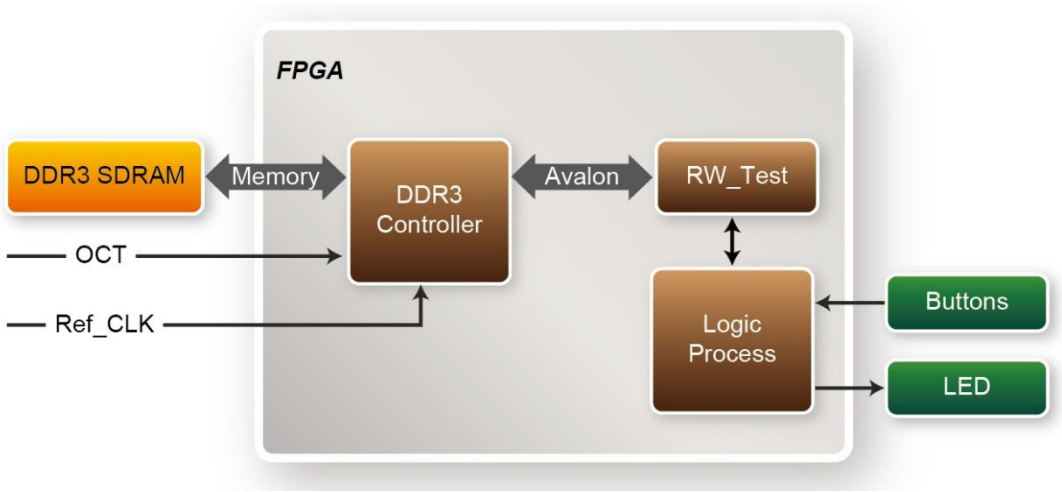


Figure 6-3 Block diagram of DDR3 SDRAM (2GB) demonstration

■ Altera DDR3 SDRAM Controller with UniPHY

To use Altera DDR3 controller, please perform the three major steps below:

1. Create correct pin assignments for DDR3.
2. Setup correct parameters in the dialog of DDR3 controller.
3. Go to the Quartus II menu and select Process→Start→Start Analysis & Synthesis to perform "Analysis and Synthesis".
4. Go to the Quartus II menu and select Tools -> TCL Scripts … to run the TCL files generated by the DDR3 IP.

■ Design Tools

- 64-bit Quartus II v16.0

■ Demonstration Source Code

- Project Directory: Demonstration\DDR3_Test
- Bit Stream: DDR3_Test.sof

■ Demonstration Batch File

- Demo Batch File Folder: DDR3_Test \demo_batch
- The demo batch file includes following files:
  - Batch File: DDR3_Test.bat
  - FPGA Configuration File: DDR3_Test.sof

■ Demonstration Setup

- Make sure Quartus II is installed on the host PC.
- Connect the TR5 board to the host PC via the USB cable. Install the USB-Blaster II driver if necessary.
- Install the DDR3 SODDIM and Power on the TR5 board.
- Execute the demo batch file "DDR3_Test.bat" under the batch file folder \DDR3_Test\demo_batch.
- Press BUTTON0 on TR5 to start the verification process. When BUTTON0 is pressed, all LEDs (LED [3:0]) should light. At the instant of releasing BUTTON0, LED1, LED2, and LED3 should start blinking. After approximately 2 seconds, LED1 should stop blinking and stay on to indicate the DDR3 have passed the test, respectively. **Table 6-1** lists the LED indicators.
- If LED2 or LED3 is not blinking, it means the 50MHz clock source is not working.
- If LED1 does not start blinking upon releasing BUTTON0, it indicates "local_cal_success" of the DDR3 fails.
- If LED1 fails to remain on after 2 seconds, the corresponding DDR3 test has failed.
- Press BUTTON0 again to regenerate the test control signals for a repeat test.
-

Table 6-1 LED **Indicators**

NAME	Description
LED0	**Reset**
LED1	**DDR3 test result**
LED2	**50MHz clock source**
LED3	**50MHz clock source**

6.3 DDR3 SDRAM Test by Nios II

Many applications use a high performance RAM, such as a DDR3 SDRAM, to provide temporary storage. In this demonstration hardware and software designs are provided to illustrate how to perform DDR3 memory access in QSYS. We describe how Altera's "DDR3 SDRAM Controller with UniPHY" IP is used to access the DDR3-Sodimm on the FPGA board, and how the Nios II processor is used to read and write the SDRAM for hardware verification. The DDR3 SDRAM controller handles the complex aspects of using DDR3 SDRAM by initializing the memory devices, managing the SDRAM banks, and keeping the devices refreshed at appropriate intervals.

■ System Block Diagram

Figure 6-4 shows the system block diagram of this demonstration. The QSYS system requires one 50 MHz clock source provided from the board. The DDR3 controller is configured as a 2 GB DDR3-800Mhz controller. The DDR3 IP generates one 800 MHz clock as SDRAM's data clock and one quarter-rate system clock 800/4=200 MHz for those host controllers, e.g. Nios II processor, accessing the SDRAM. In the QSYS, Nios II and the On-Chip memory are designed running with the 200 MHz clock, and 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.

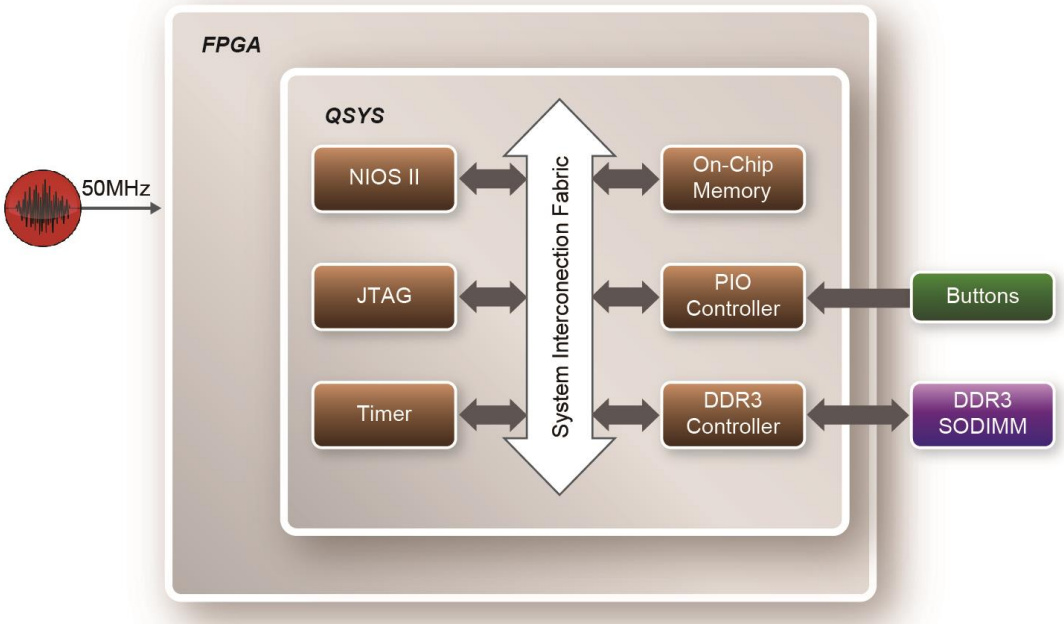


Figure 6-4 Block diagram of the DDR3 Basic Demonstration

The system flow is controlled by an Nios II program. First, the Nios II program writes test patterns into the whole 2 GB 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. The program will show progress in JTAG-Terminal when writing/reading data to/from the SDRAM. When verification process is completed, the result is displayed in the JTAG-Terminal.

■ Altera DDR3 SDRAM Controller with UniPHY

To use the Altera DDR3 controller, users need to perform three major steps:

1. Create correct pin assignments for the DDR3.
2. Setup correct parameters in the DDR3 controller dialog.
3. Perform "Analysis and Synthesis" by selecting from the Quartus II menu: Process→Start→Start Analysis & Synthesis.
4. Run the TCL files generated by DDR3 IP by selecting from the Quartus II menu: Tools→TCL Scripts…

■ Design Tools

- Quartus II 16.0
- Nios II Eclipse 16.0

■ Demonstration Source Code

- Quartus Project directory: DDR3_Nios
- Nios II Eclipse: DDR3_Nios\software
- Nios II Project Compilation

■ 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: *DDR3_Nios\demo_batch*

The demo batch file includes following files:

- Batch File for USB-Blaseter II: test.bat, test.sh
- FPGA Configure File: DDR3_Nios.sof
- Nios II Program: DDR3_Nios.elf

■ Demonstration Setup

Please follow the below procedures to setup the demonstration.

- Make sure Quartus II and Nios II are installed on your PC.
- Make sure the DDR3 SODIMMs are installed on the FPGA board.
- Power on the FPGA board.
- Use the USB Cable to connect PC and the FPGA board and install USB Blaster II driver if necessary.

- Execute the demo batch file “test.bat” under the folder “DDR3_Nios\demo_batch”.
- After Nios II program is downloaded and executed successfully, a prompt message will be displayed in nios2-terminal.
- Press Button3~Button0 of the FPGA board to start DDR3 SDRAM verify process. Press Button0 for continued test.
- The program will display progressing and result information, as shown in **Figure 6-5**.

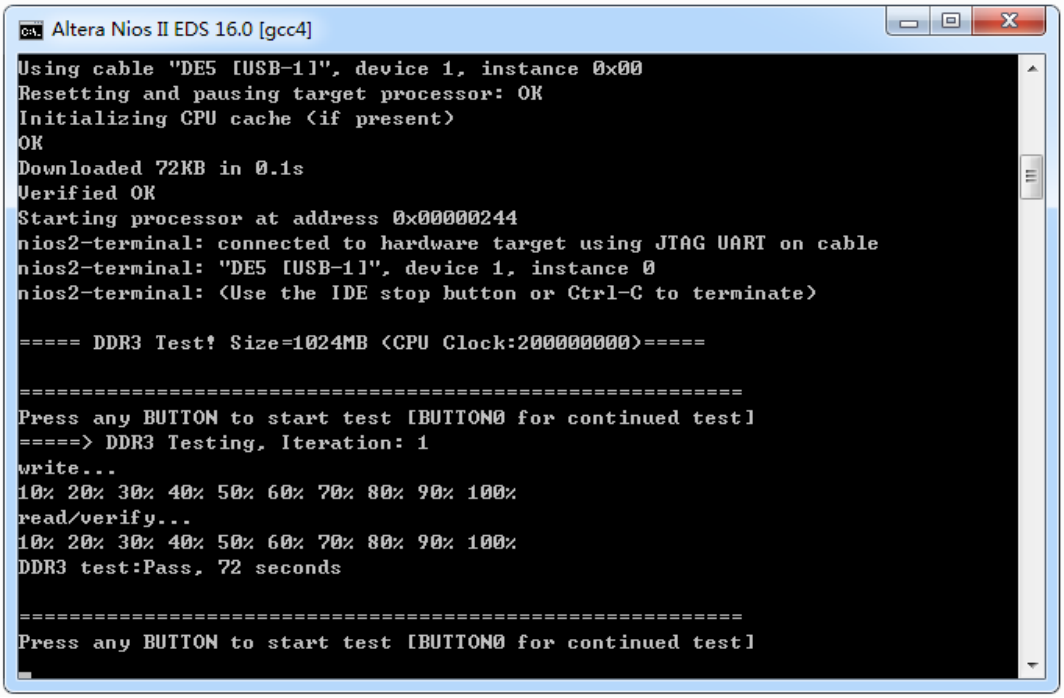


Figure 6-5 Display Progress and Result Information for the DDR3 Demonstration

6.4 Fan Speed Control

This demo helps users quickly understand how to set the MAX6650 chip from the FPGA to control the fan sink. The MAX6650 chip can set or retrieve the RPM of the fan sink. It can also monitor if there is any unexpected error and determine which type of error it is. The following section will save lots of time for the development of user application.

■ System Block Diagram

Figure 6-6 shows the system block diagram of this demo. It is necessary to configure the MAX6650

chip prior upon the initialization of fan sink control. The MAX6650 chip uses standard I2C protocol for communication. The functions I2C_Config and I2C_Bus_Controller set and monitor the RPM of the fan sink, respectively. A pre-scaler is used as frequency divider for the clock frequency of I2C. Users need to calculate the frequency based on the equations from the datasheet to control the RPM of the fan sink. There are three equations in the datasheet and this demo uses one of them. For other equations, please refer to the datasheet MAX6650-MAX6651.pdf in the system CD.

The SW0 (Switch 0) controls the RPM in this demo. When the SW0 is set to 0, the speed is around 2000 RPM. The speed would reach about 5000 RPM if the SW0 is set to 1. It would take 10 ~ 30 secs as the buffer time for the conversion. If an error is detected, the LED would light. Users need to press Button1 to reset the LED to turn it off.

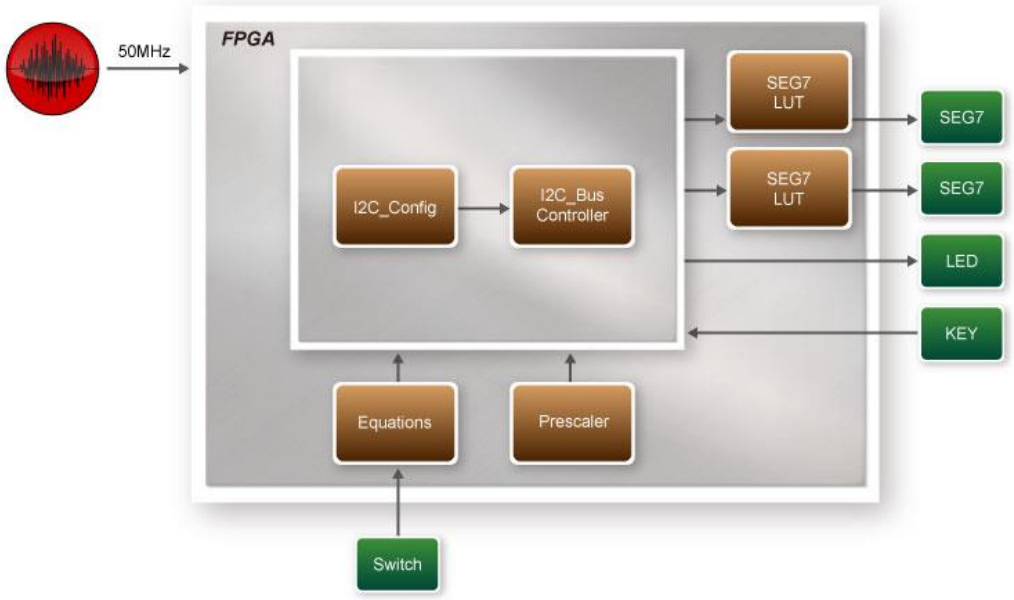


Figure 6-6 Block diagram of the fan speed control demonstration

■ Alarm Status Register Bit Assignments

When the fan is abnormal, the LED will light. Users can refer to Table **6-2** and get a better understanding about the malfunction of the fan sink accordingly. The status of BIT 4 ~ 7 can be ignored because BIT 4 is for MAX6651 only and BIT 5 ~ 7 are always low.

Table 6-2 Alarm-Enable Resgister Bit Masks

BIT	NAME	POR(DEFAULT) STATE	FUNCTION
7(MSB) to 5	---	0	Always 0
4	GPIO2(MAX6651 only)	0	GPIO2 Alarm. Set when GPIO2 is low (MAX6651 only)
3(LED[3])	GPIO1	0	GPIO1 Alarm. Set when GPIO1 is low
2(LED[2])	TACH	0	Tachometer Overflow Alarm
1(LED[1])	MIN	0	Minimum Output Level Alarm
0(LED[0])	MAX	0	Maximum Output Level Alarm

■ Design Tools

- 64-bit Quartus II v16.0

■ Demonstration Source Code

- Project Directory: Demonstration/Fan
- Bit Stream: Golden_Top.sof

■ Demonstration Batch File

- Demo Batch File Folder: *\Fan\demo_batch*
- The demo batch file includes following files:
  - Batch File: test.bat
  - FPGA Configure File: Golden_Top.sof

■ Demonstration Setup

- Make sure Quartus II is installed on the host PC.
- Connect the TR5 and the host PC via the USB cable. Install the USB-Blaster II driver if necessary.

- Power on the FPGA Board.
- Execute the demo batch file “test.bat” under the batch file folder \Fan\demo_batch.
- When SW0 is set to 0, the RPM would slowly be adjusted to ~2000. When SW0 is set to 1, the RPM would slowly be adjusted to ~5000.

6.5 UART to USB Control

Many applications need communication with the computer through the common port. The traditional connector is RS232 which needs to connect to a RS232 cable. But today many personal computers don't have the RS232 connector which makes it very inconvenient to develop projects. The TR5 board is designed to support UART communication through a USB cable. The UART to USB circuit is responsible for converting the data format. Developers can use a USB cable rather than an RS232 cable to enable the communication between the FPGA and the host computer. In this demonstration we will show you how to control the LEDs by sending a command on the computer putty terminal. The command is sent and received through a USB cable to the FPGA. Note that in FPGA, the information was received and sent through a UART IP.

Figure 6-7 shows the hardware block diagram of this demonstration. The system requires a 50 MHz clock provided from the board. The PLL generates a 100MHz clock for Nios II processor and the controller IP. The LEDs are controlled by the PIO IP. The UART controller sends and receives command data and the command is sent through Putty terminal on the computer.

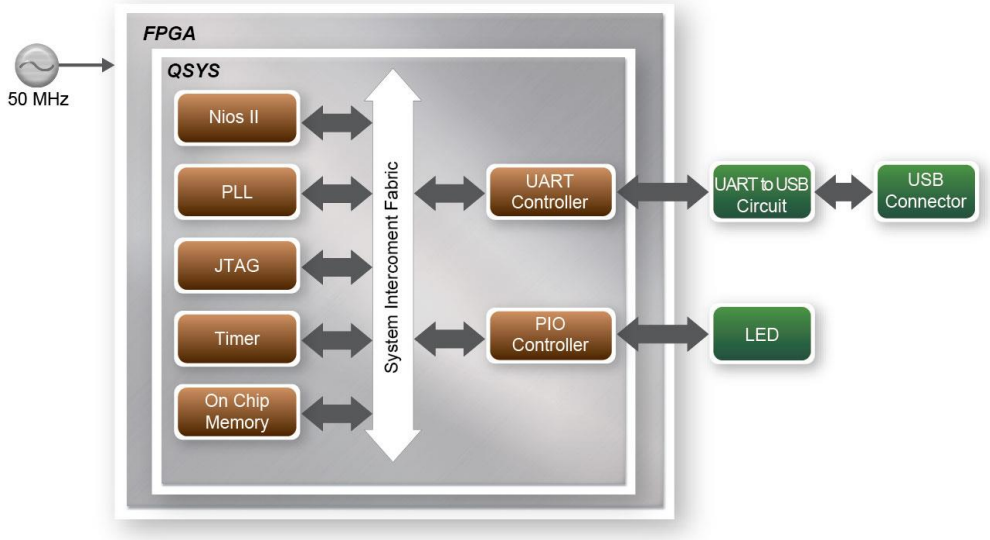


Figure 6-7 Block diagram of UART Control LED demonstration

■ Design Tools

- Quartus II 16.0

■ Nios II Eclipse 16.0

■ Demonstration Source Code

- Quartus Project directory: uart_control
- Nios II Eclipse: uart_control\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: uart_control\demo_batch

- The demo batch file includes following files:
  - Batch File for USB-Blaster: test.bat, test.sh
  - FPGA Configure File : uart_control.sof
  - Nios II Program: *uart_led*.elf

Demonstration Setup

- Connect a USB cable between your computer and the USB connector (J5) on the TR5 board.
- Power on your TR5 board, if you find an unrecognized USB Serial Port in Device Manager as shown in **Figure 6-8 .**you should install the UART to the USB driver before you run the demonstration.

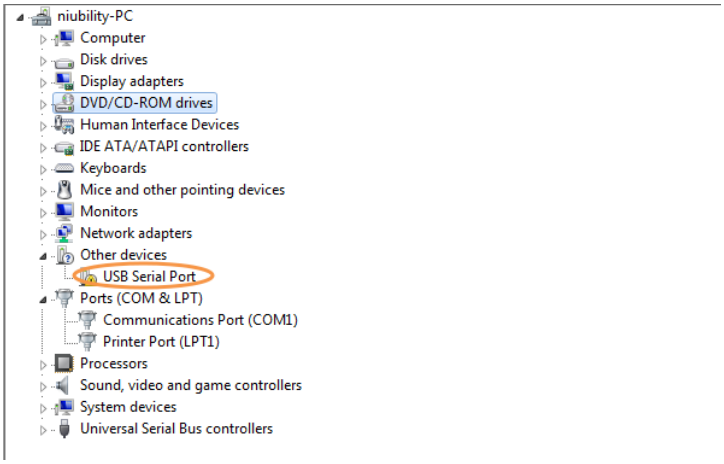


Figure 6-8 Unrecognized USB Serial Port on PC

- To install UART_TO_USB driver on your computer please select the USB Serial Port to update the driver software. The driver file can be downloaded from the following website: http://www.ftdichip.com/Drivers/VCP.htm.
- Open the Device Manager to ensure which common port is assigned to the UART to USB port as shown in **Figure 6-9**. The common number 9 (COM9) is assigned on this computer.

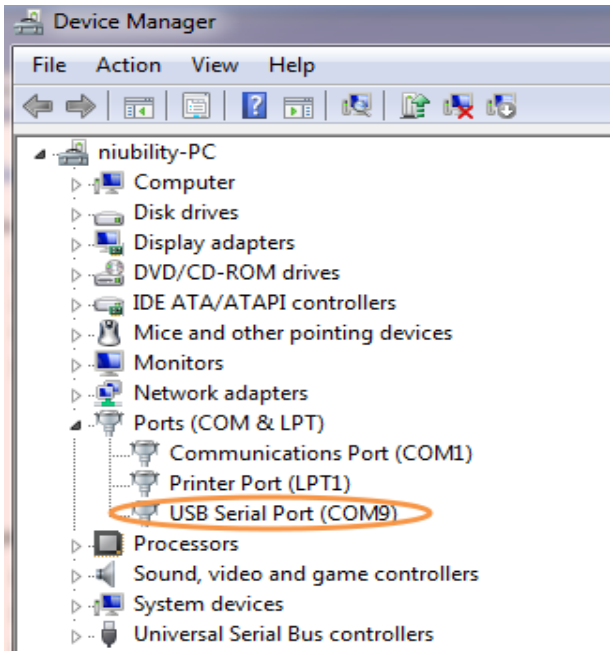


Figure 6-9 Check the assigned Com Port number On PC

- Open the putty software, type in the parameter as shown in **Figure 6-10** and click open button to open the terminal.(Here is a link for you to download the putty terminal: Download Putty)

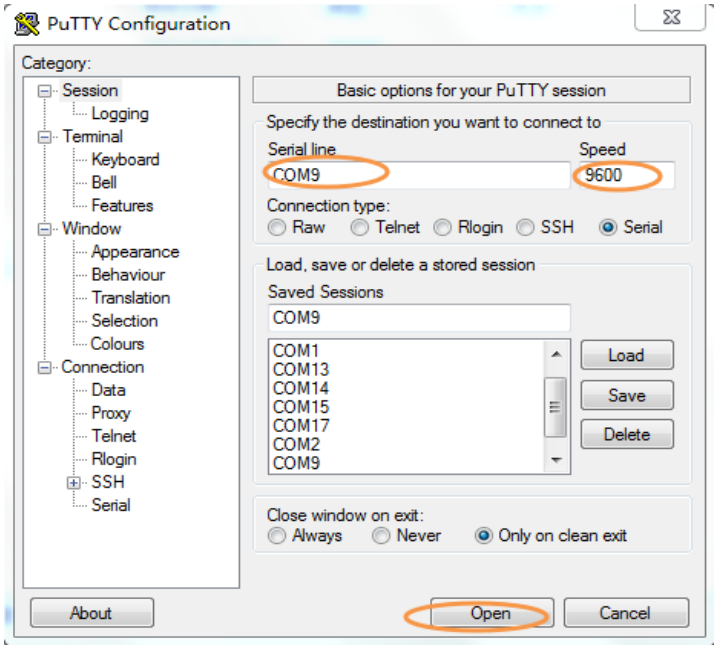


Figure 6-10 putty terminal setup

- Make sure Quartus II and Nios II are installed on your PC.
- Connect USB Blaster to the TR5 board (J6) and install USB Blaster driver if necessary.
- Execute the demo batch file " test.bat" under the batch file folder uart_control\demo_batch.
- The Nios II-terminal and putty terminal running result as shown in **Figure 6-11.**

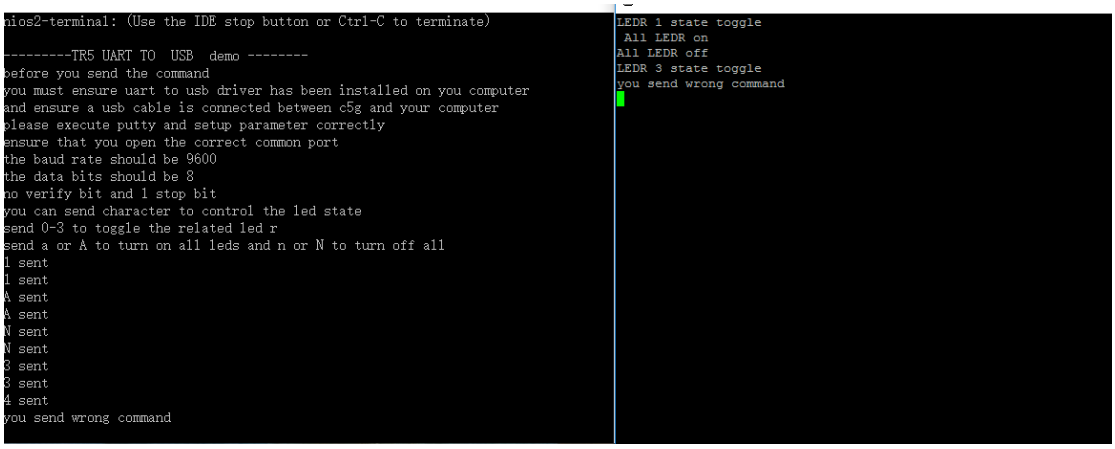


Figure 6-11 Running result of uart_usb demo

- In the putty terminal, type character to change the LED state. Type a digital number to toggle the LED [4..0] state and type a/A or n/N to turn on/off all LED.

Chapter 7

PCI Express Reference Design

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 and FPGA communicate with each other through the PCI Express interface. V-Series Avalon-MM DMA for PCI Express IP is used in this demonstration. For detail about this IP, please refer to Altera document : ug_pcie_avmm_dma.pdf.

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 V-Series Avalon-MM DMA for PCI Express. The application software on the PC side is developed by Terasic based on Altera's PCIe kernel mode driver.

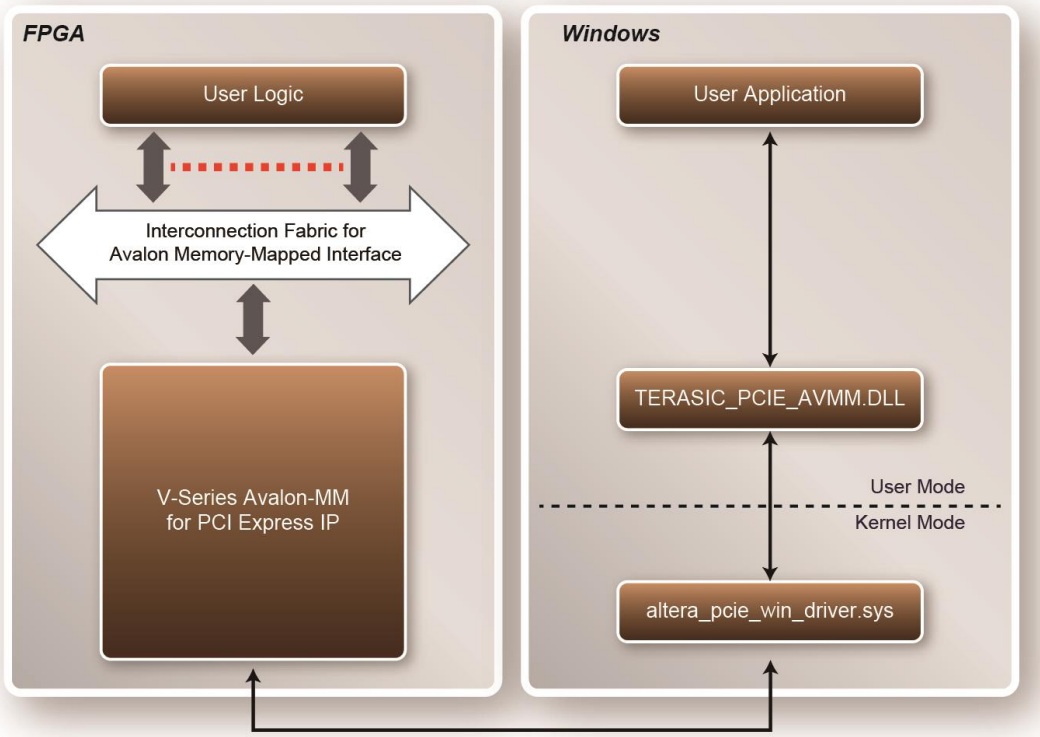


Figure 7-1 PCI Express System Infrastructure

7.2 PC PCI Express Software SDK

The FPGA System CD contains a PC Windows based SDK to allow users to develop their 64-bits software application on Windows XP/7/10 64-bits. The SDK is located in the "CDROM \demonstrations\PCIe_SW_KIT" folder which includes:

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

The kernel mode driver assumes the PCIe vender ID (VID) is 0x1172 and the device ID (DID) is 0xE003. If different VID and DID are used in the design, users need to modify the PCIe vender ID (VID) and device ID (DID) in the driver INF file accordingly.

The PCI Express Library is implemented as a single DLL called TERASIC_PCIE_AVMM.DLL. This file is a 64-bits DLL. With the DLL 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, DMA is required as the read and write operations are specified under the hardware design on the FPGA.

■ PCI Express Software Stack

Figure 7-8 shows the software stack for the PCI Express application software on 64-bit Windows. The PCI Express driver incorporated in the DLL library is called TERASIC_PCIE_AVMM.dll. Users can develop their applications based on this DLL. The altera_pcie_win_driver.sys kernel driver is provided by Altera.

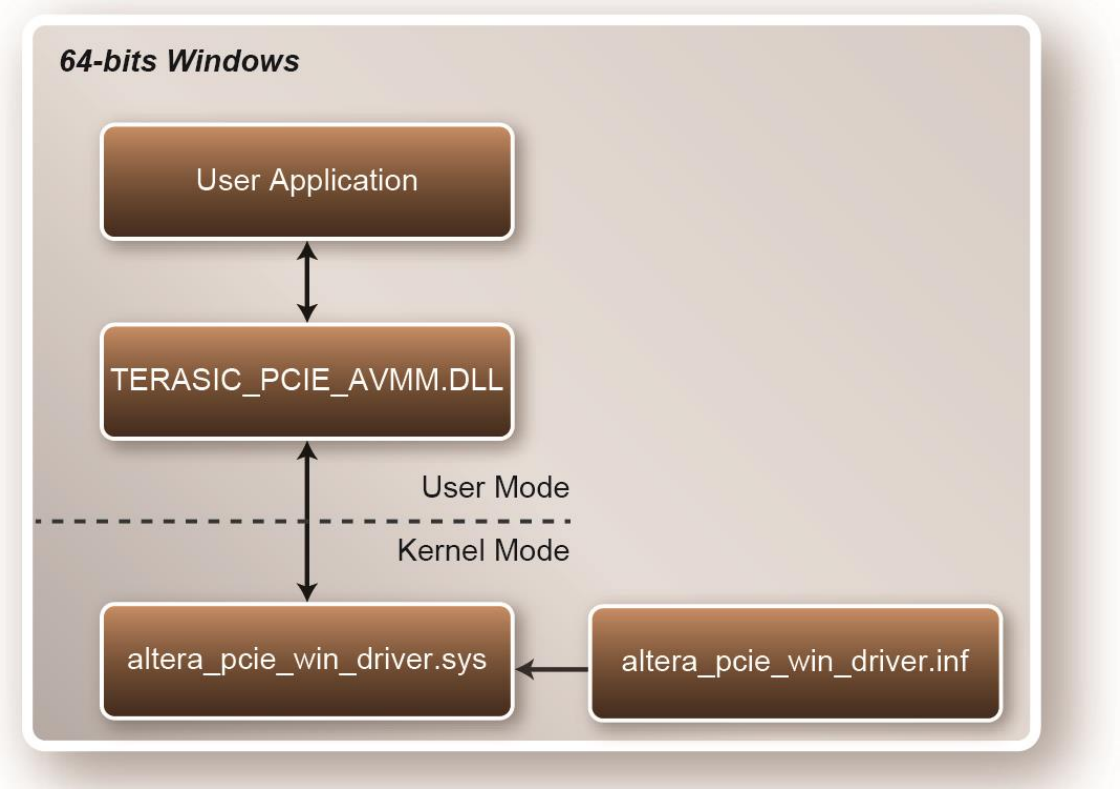


Figure 7-2 PCI Express Software Stack

■ Install PCI Express Driver on Windows

The PCIe driver is located in the folder: “CDROM\Demonstrations\PCIe_SW_KIT\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, execute the steps below:

1. Make sure the TR5 and the PC are both powered off.
2. Plug the PCIe adapter card into the PCIe slot on the PC motherboard. Use the PCIe cable to connect to the TR5 PCIE connector and the PCIe adapter card (See **Figure 7-3**)

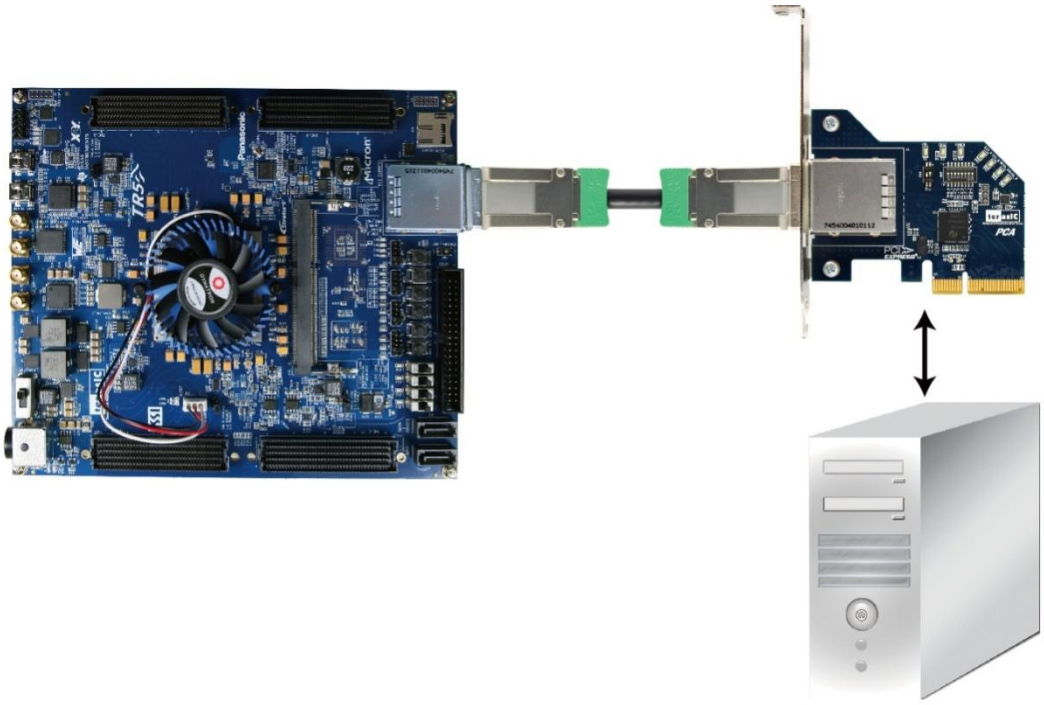


Figure 7-3 FPGA board connect to PC

3. Power on your TR5 board and the host PC
4. Make sure Altera Programmer and USB-Blaster II driver are installed
5. Execute test.bat in "CDROM\Demonstrations\PCIe_Fundamental\demo_batch" to configure the FPGA
6. Restart windows operation system
7. Click the Control Panel menu from the Windows Start menu. Click the Hardware and the 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 7-4**. Move the mouse cursor to the PCI Device item and right click it to select the Update Driver Software... item.

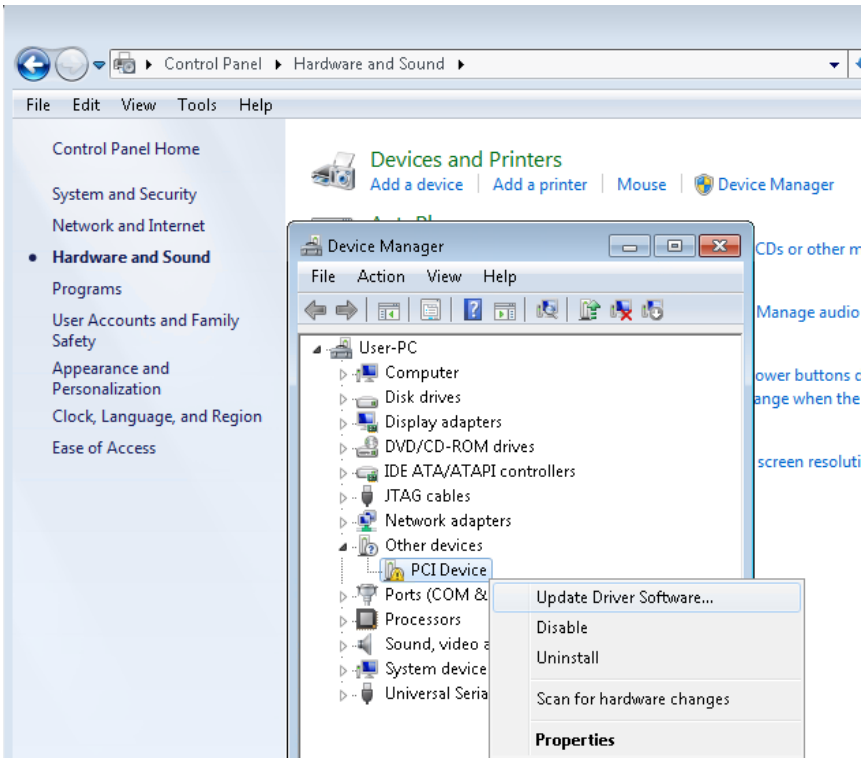


Figure 7-4 Screenshot of launching Update Driver Software dialog

8. In the **How do you want to search for driver software** dialog, click **Browse my computer for driver software** item, as shown in **Figure 7-5**. Click "OK" and then "Exit" to close the installation program.

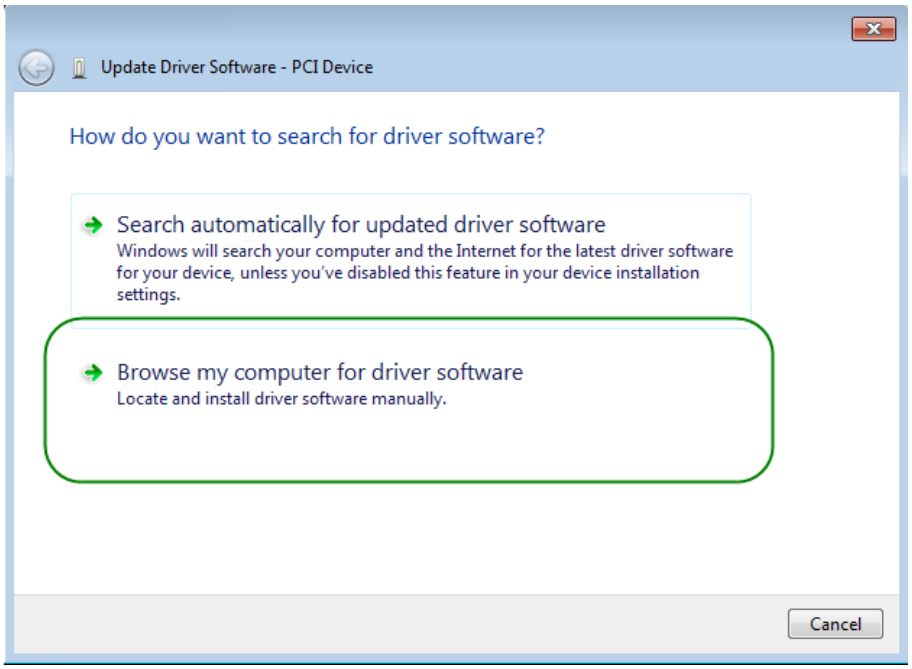


Figure 7-5 Dialog of Browse my computer for driver software

9. 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 7-6** Click the **Next** button.

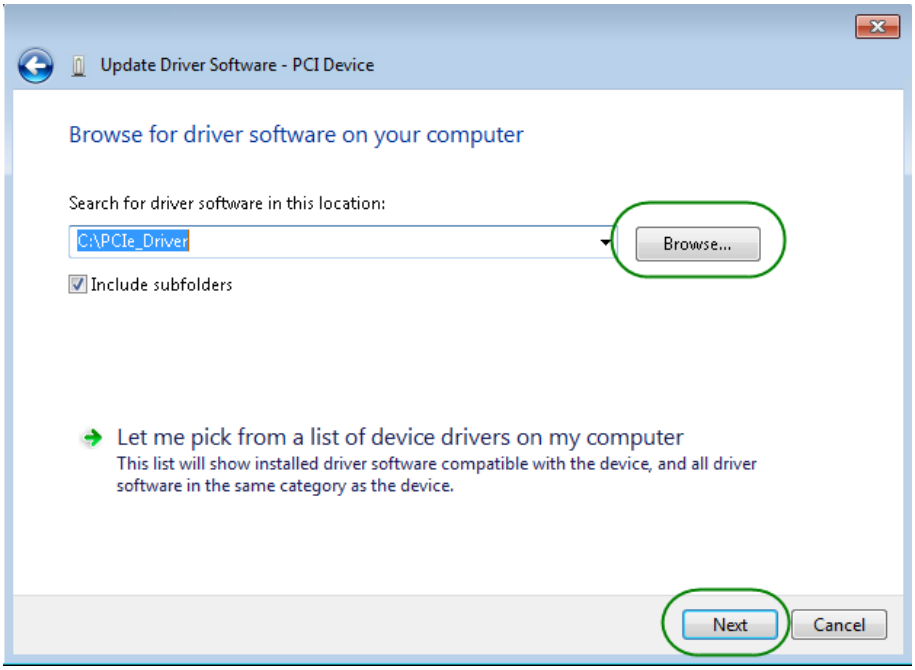


Figure 7-6 Browse for driver software on your computer

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

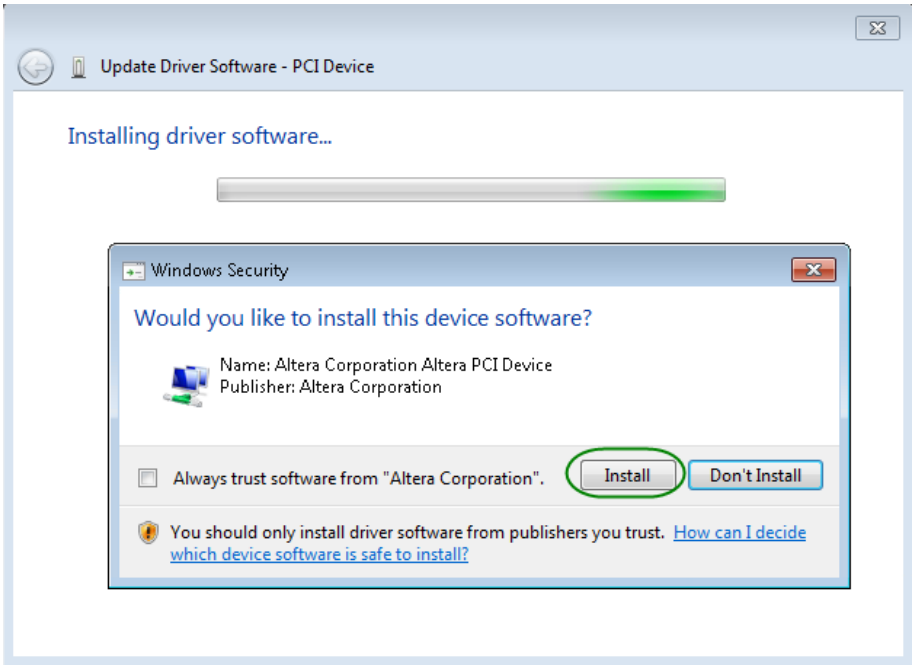


Figure 7-7 Click Install in the dialog of Windows Security

11. When the driver is installed successfully, the successfully dialog will appears, as shown in

Figure 7-8. Click the **Close** button.

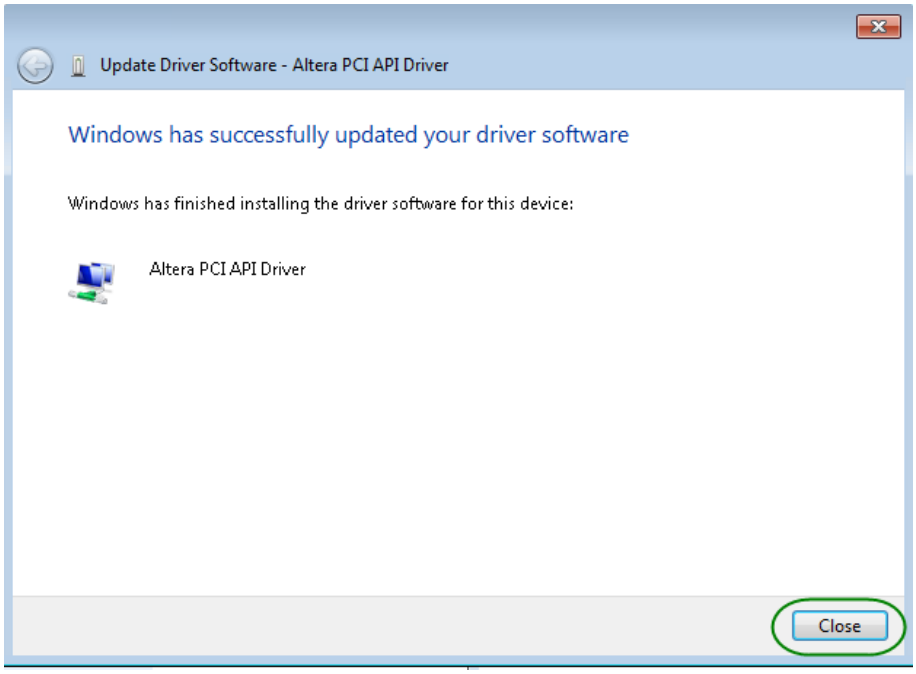


Figure 7-8 Click Close when the installation of Altera PCI API Driver is complete

12. Once the driver is successfully installed, users can see the **Altera PCI API Driver** under the device manager window, as shown in **Figure 7-9**.

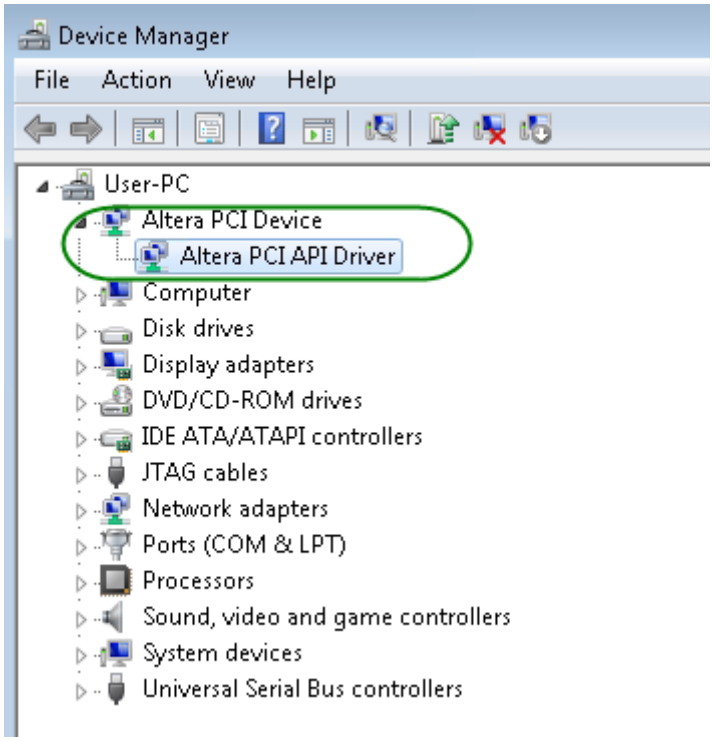


Figure 7-9 Altera PCI API Driver in Device Manager

■ Create a Software Application

All necessary files to create a PCIe software application are located in the *CDROM\demonstration\PCIe_SW_KIT\PCIe_Library* which includes the following files:

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

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

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

- TERASIC_PCIE.DLL/TERASIC_PCIEx64.DLL Software API

Users can easily communicate with the FPGA through the PCIe bus through the TERASIC_PCIE_AVMM.DLL API. The details of API are described below:

PCIE_Open

Function:

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

Prototype:

```
PCIE_HANDLE PCIE_Open(
    WORD wVendorID,
    WORD wDeviceID,
    WORD 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 verder 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 more 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, DWORD * 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, DWORD 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_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,
    DWORD dwBufSize
    );
```

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.

dwBufSize:

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,
    DWORD dwDataSize
    );
```

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.

dwDataSize:

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,
    DWORD Offset,
    DWORD *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.

7.3 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 DMA.

■ Demonstration Files Location

- The demo file is located in the batch folder:
  - CDROM\demonstrations\PCIe_funcdamental\Demo_batch

- The folder includes following files:
  - FPGA Configuration File: PCIe_funcdamental.sof
  - Download Batch file: test.bat
  - Windows Application Software folder : windows_app, includes
    - PCIE_FUNDAMENTAL.exe
    - TERASIC_PCIE_AVMM.dll

■ Demonstration Setup

1. Use the PCIe cable to connect to the TR5 PCIE connector and PCIe adapter card as shown in

Figure 7-10.

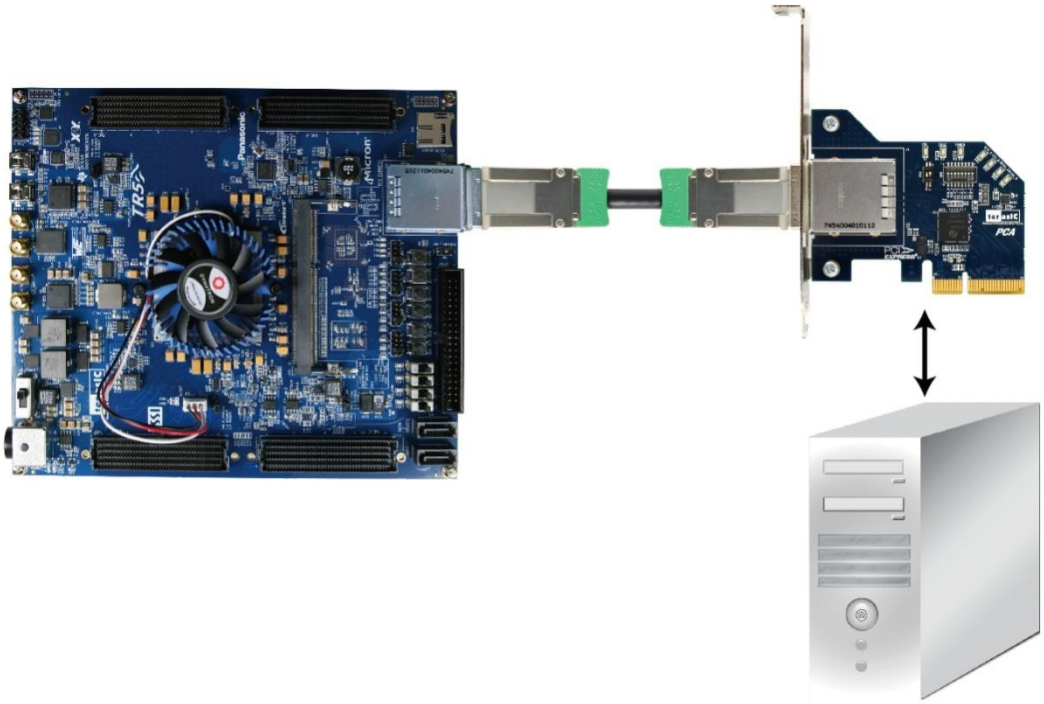


Figure 7-10 FPGA board connect to PC

2. Configure FPGA with PCIE_Fundamental.sof by executing the test.bat.
3. Install PCIe driver if necessary. The driver is located in the folder: CDROM\Demonstration \PCIe_SW_KIT\PCIe_Driver.
4. Restart Windows
5. Make sure the Windows has detected the FPGA Board by checking the Windows Control panel as shown in Figure 7-11.

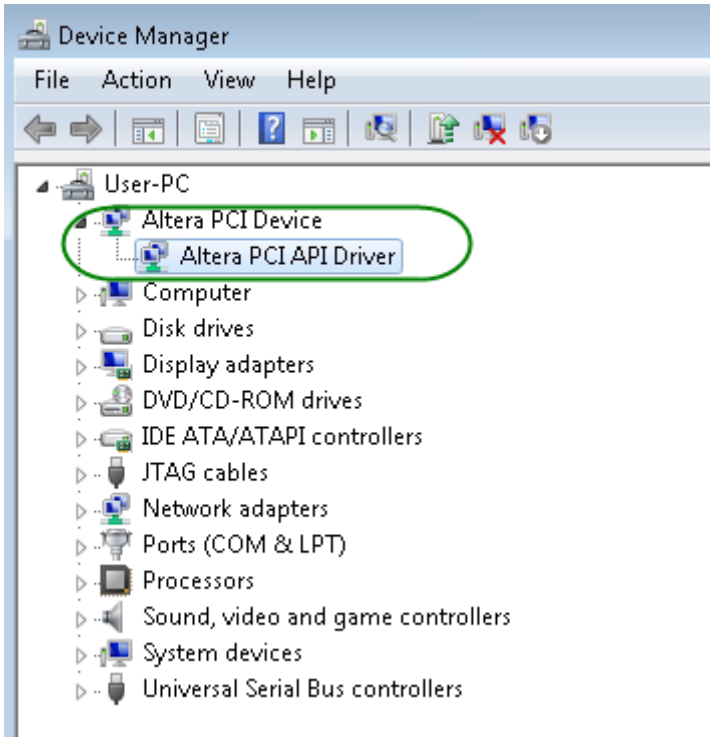


Figure 7-11 Screenshot for PCIe Driver

6. Goto windows_app folder, execute PCIE_FUNDMENTAL.exe. A menu will appear as shown in Figure 7-12.

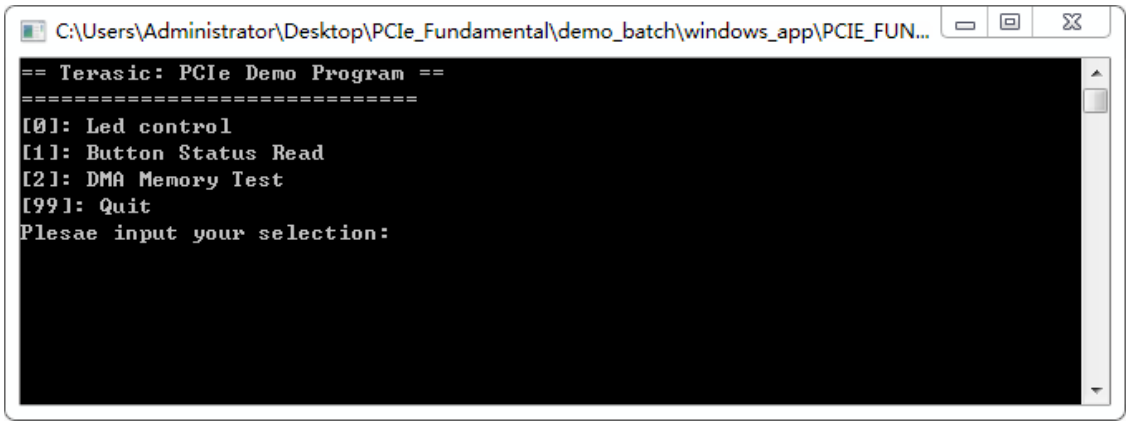


Figure 7-12 Screenshot of Program Menu

7. Type 0 followed by a ENTER key to select Led Control item, then input 15 (hex 0x0f) will make all led on as shown in Figure 7-13. If input 0(hex 0x00), all led will be turned off.

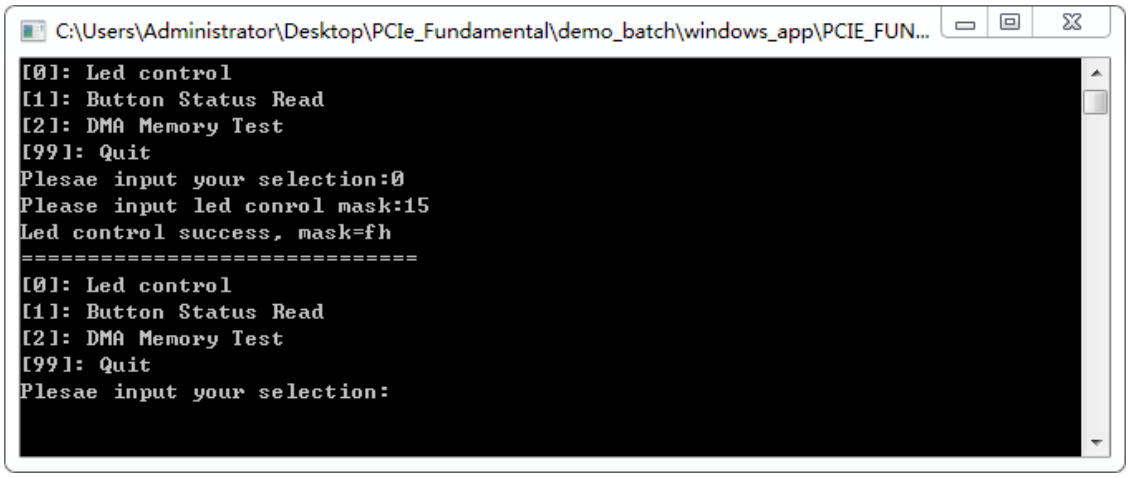


Figure 7-13 Screenshot of LED Control

8. Type 1 followed by an ENTER key to select Button Status Read item. The button status will be report as shown in Figure 7-14.

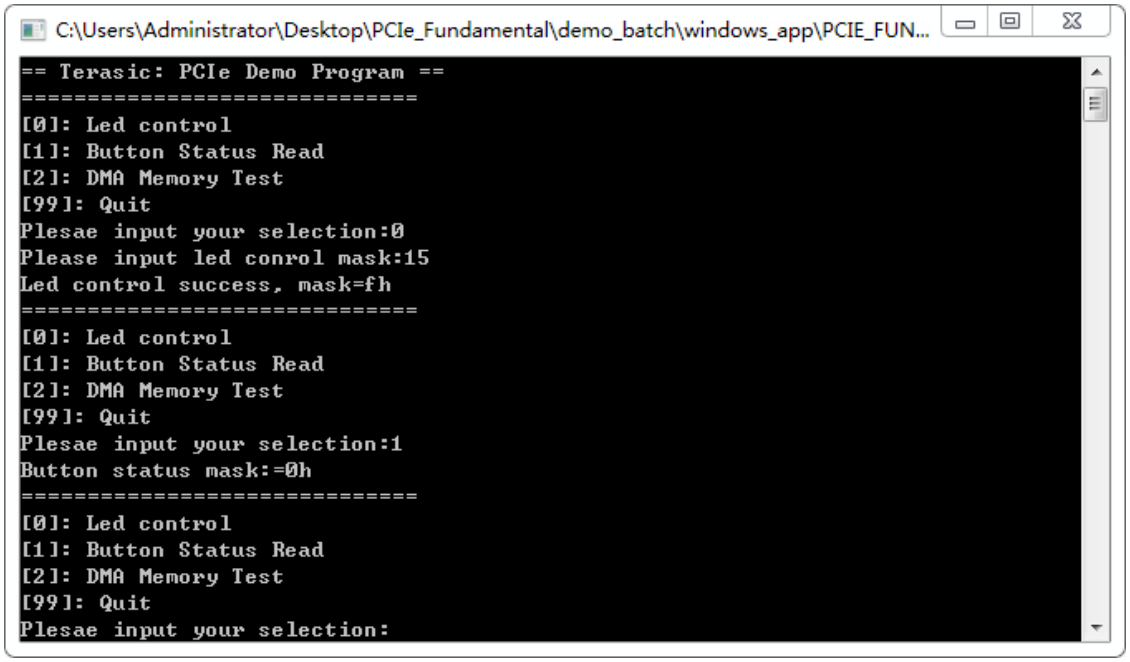


Figure 7-14 Screenshot of Button Status Report

9. Type-2 followed by an ENTER key to select DMA Testing item. The DMA test result will be report as shown in Figure 7-15.

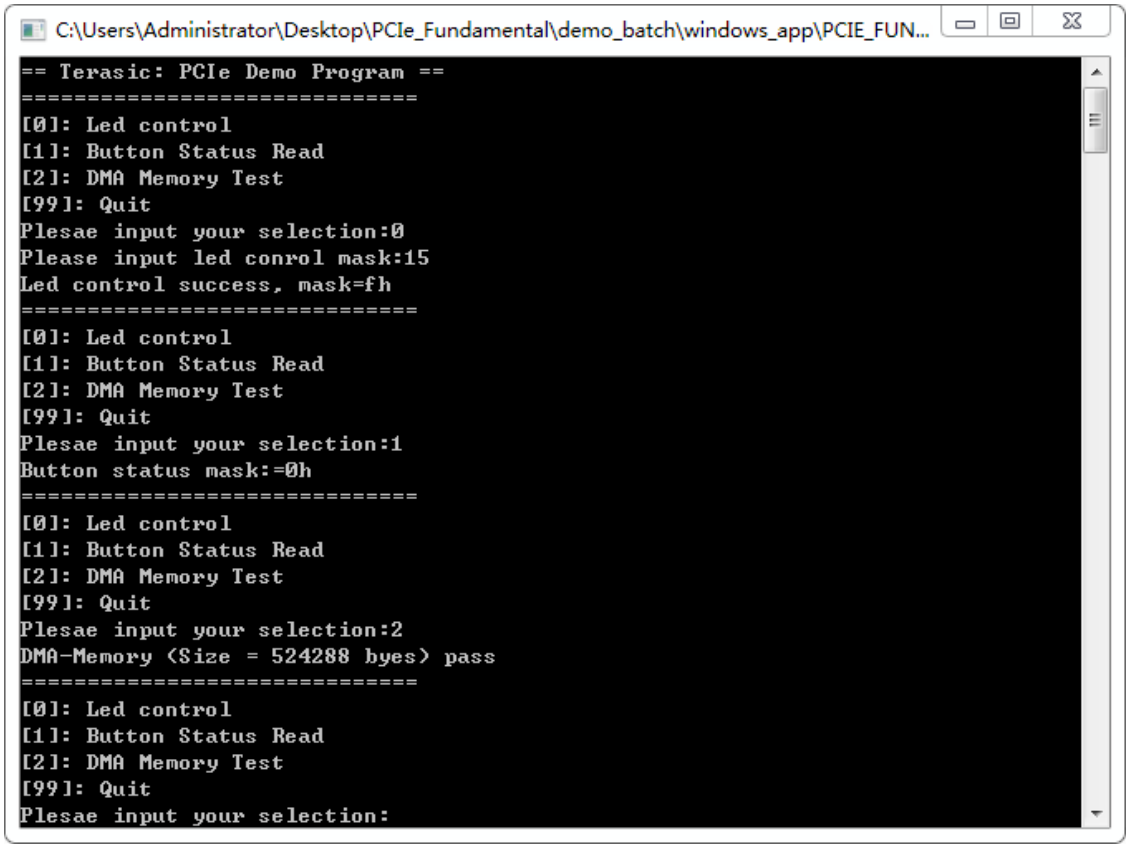


Figure 7-15 Screenshot of DMA Memory Test Result

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

■ Development Tools

- Quartus II 16.0
- Visual C++ 2012

■ Demonstration Source Code Location

- Quartus Project: Demonstrations\PCIE_Fundamental
- Visual C++ Project: Demonstrations\PCIe_SW_KIT\PCIE_FUNDAMENTAL

■ FPGA Application Design

Figure 7-16 shows the system block diagram in the FPGA system. In the Qsys, Altera 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.

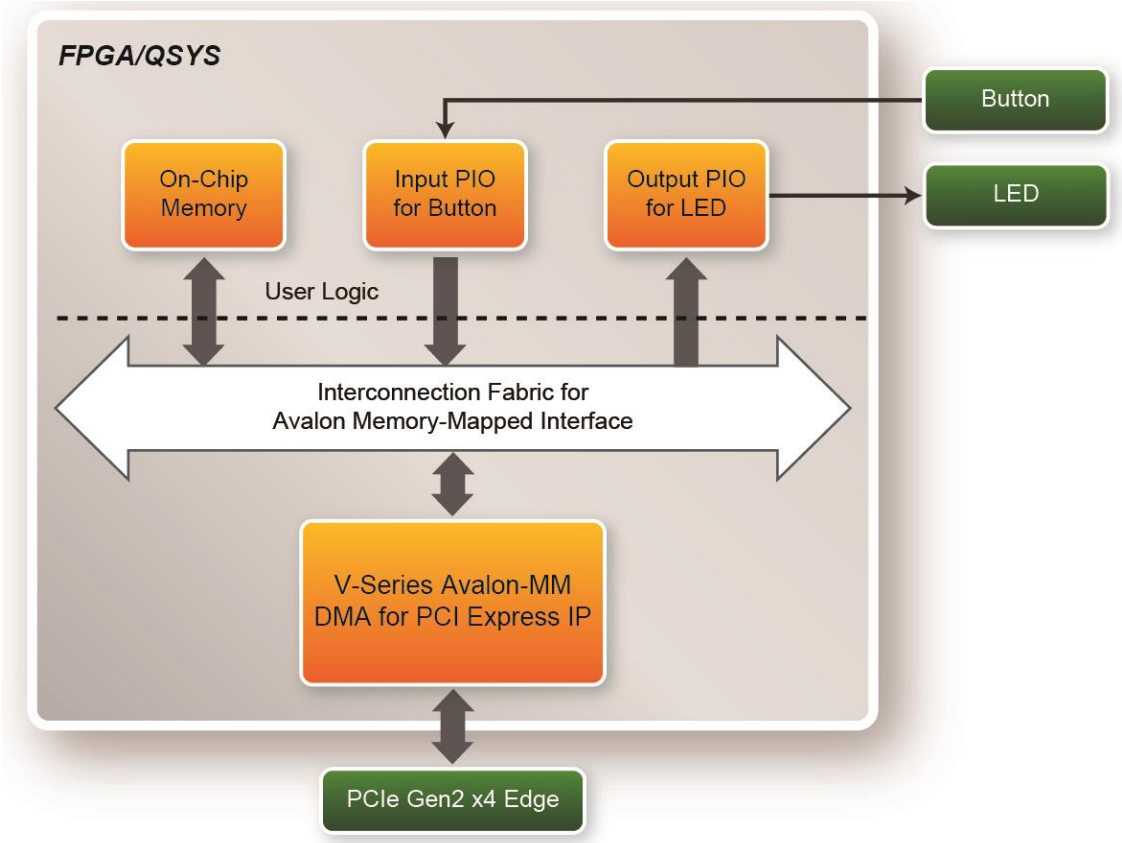


Figure 7-16 Hardware block diagram of the PCIe reference design

■ Windows Based Application Software Design

The application software project is built by Visual C++ 2012. 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_AVMM.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       0x4000010
#define DEMO_PCIE_IO_BUTTON_ADDR    0x4000020
#define DEMO_PCIE_MEM_ADDR          0x00000000

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

The base address of BUTTON and LED controllers are 0x4000010 and 0x4000020 based on PCIE_BAR4, respectively. The on-chip memory base address is 0x00000000 relative to the DMA controller.

Before accessing the FPGA through PCI Express, the application first calls PCIE_Load to dynamically load the TERASIC_PCIE_AVMM.DLL. Then, it calls the 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_AVMM.h. If developer changes the Vender ID, the Device ID, and the PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM.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,(DWORD)Mask);
```

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

```
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:

```
PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);

PCIE_DmaRead(hPCIe, LocalAddr, pRead, nTestSize);
```

7.4 PCIe Reference Design – DDR3

The application reference design shows how to add DDR3 Memory Controllers into the PCIe Quartus project based on the PCI_Fundamental Quartus project and perform 2GB 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_DDR3\Demo_batch

- The folder includes following files:
  - FPGA Configuration File: PCIe_DDR3.sof
  - Download Batch file: test.bat
  - Windows Application Software folder : windows_app, includes
    - PCIE_DDR3.exe
    - TERASIC_PCIE_AVMM.dll

■ Demonstration Setup

1. Install both DDR3 1600 2GB SODIMM on the FPGA board.
2. Install the FPGA board on your PC.

3. Configure FPGA with PCIE_DDR3.sof by executing the test.bat.
4. Install PCIe driver if necessary.
5. Restart Windows
6. Make sure the Windows has detected the FPGA Board by checking the Windows Control panel.
7. Goto windows_app folder, execute PCIE_DDR3.exe. A menu will appear as shown in **Figure 7-17**.

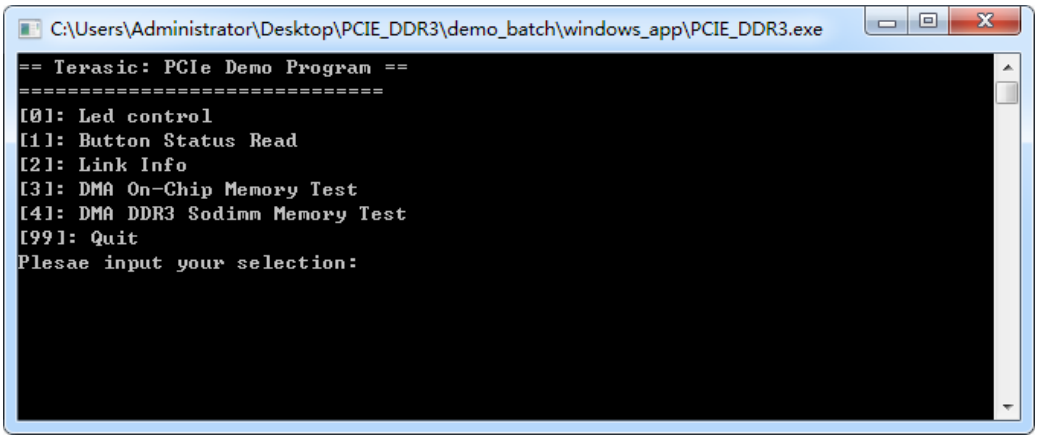


Figure 7-17 Screenshot of Program Menu

8. Type 2 followed by the ENTER key to select Link Info item. The PICe link information will be shown as in **Figure 7-18**. Gen2 link speed and x4 link width are expected.

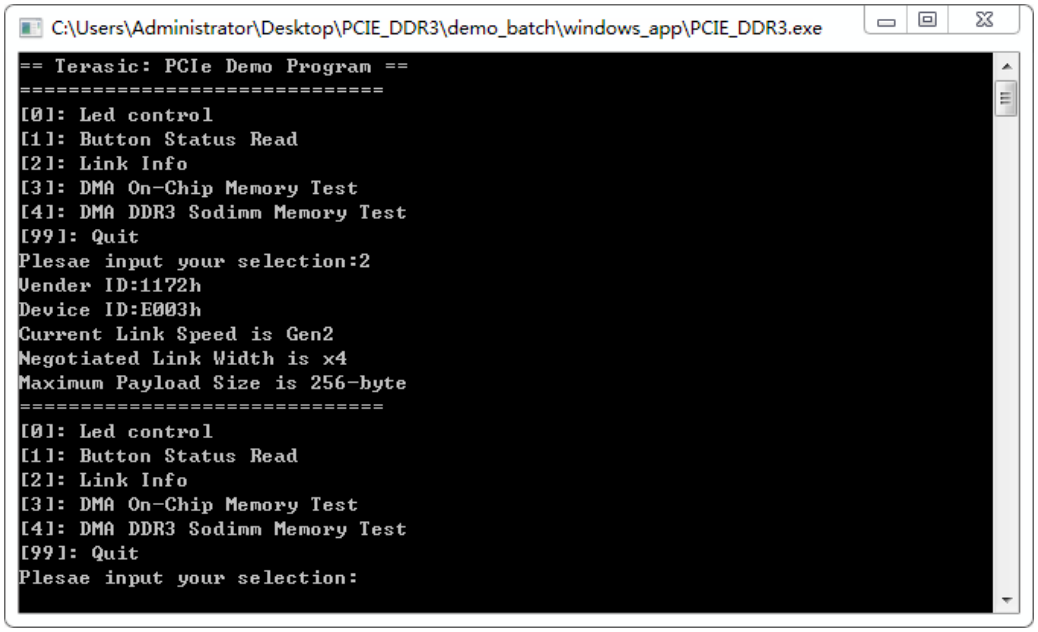


Figure 7-18 Screenshot of Link Info

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 7-19.

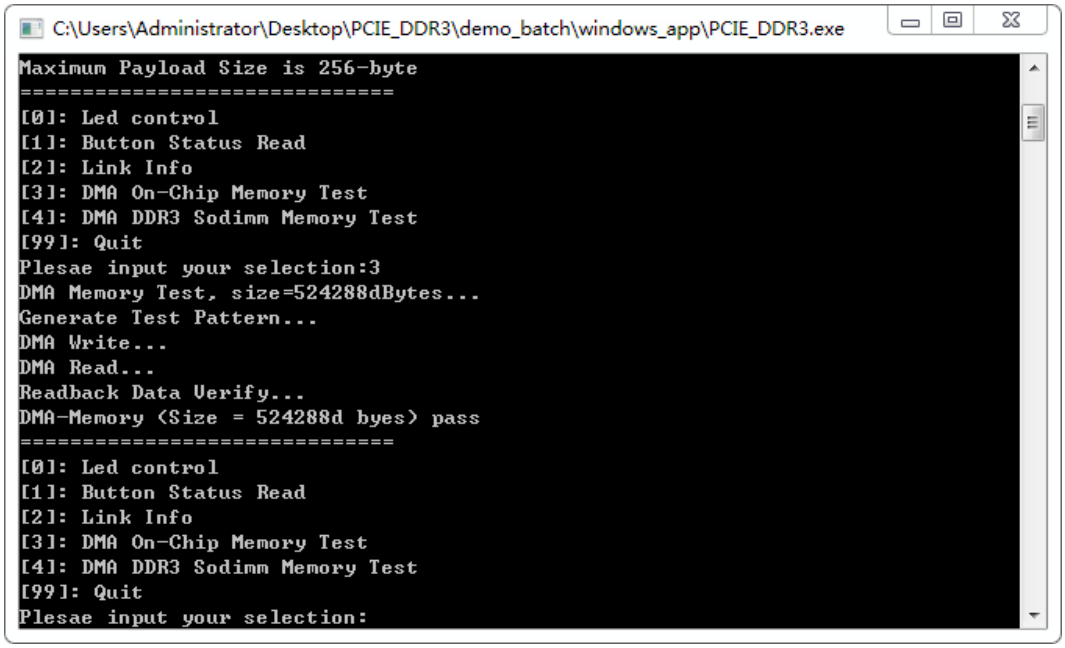


Figure 7-19 Screenshot of On-Chip Memory DMA Test Result

10. Type-4 followed by the ENTER key to select DMA DDR3 SODIMM Memory Test item. The DMA write and read test result will be report as shown in Figure 7-20.

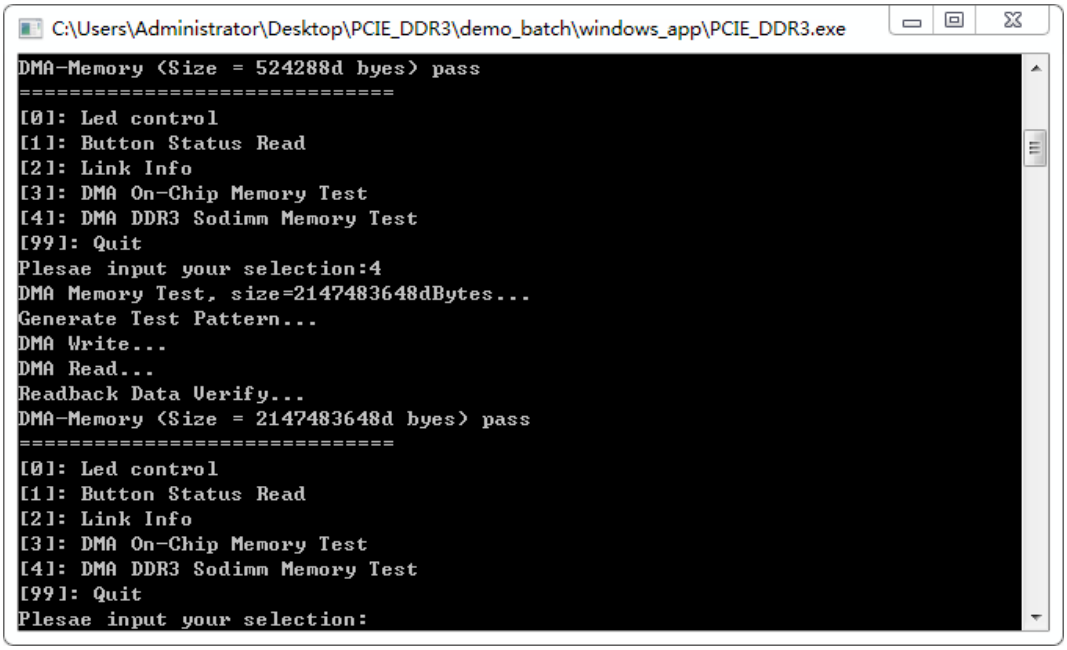


Figure 7-20 Screenshot of DDR3 SOSIMM Memory DAM Test Result

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

Development Tools

- Quartus II 16.0
- Visual C++ 2012

Demonstration Source Code Location

- Quartus Project: Demonstrations\PCIE_DDR3
- Visual C++ Project: Demonstrations\PCIe_SW_KIT\PCIe_DDR3

FPGA Application Design

Figure 7-21 shows the system block diagram in the FPGA system. In the Qsys, Altera PIO controller is used to control the LED and monitor the Button Status, and the On-Chip memory and DDR3 SOSIMM Memory are used for performing DMA testing. The PIO controllers、the On-Chip memory and DDR3 SOSIMM Memory are connected to the PCI Express Hard IP controller through the Memory-Mapped Interface.

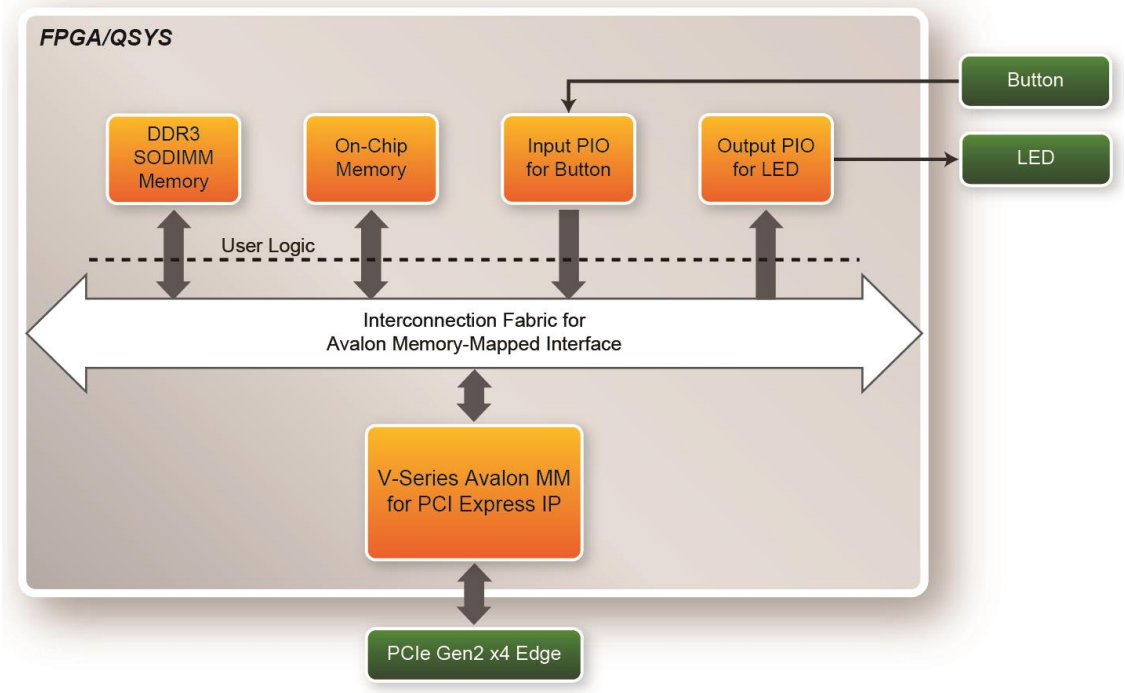


Figure 7-21 Hardware block diagram of the PCIe DDR3 reference design

■ Windows Based Application Software Design

The application software project is built by Visual C++ 2012. The project includes the following major files:

Name	Description
PCIE_DDR3.cpp	Main program
PCIE.c	Implement dynamically load for TERAISC_PCIE_AVMM.DLL
PCIE.h	
TERASIC_PCIE_AVMM.h	SDK library file, defines constant and data structure

The main program PCIE_DDR3.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       0x4000010
#define DEMO_PCIE_IO_BUTTON_ADDR    0x4000020
#define DEMO_PCIE_ONCHIP_MEM_ADDR   0x00000000
#define DEMO_PCIE_DDR3_MEM_ADDR     0x1000000000

#define ONCHIP_MEM_TEST_SIZE        (512*1024) //512KB
#define DDR3_MEM_TEST_SIZE          (2*1024*1024*1024) //2GB
```

The base address of BUTTON and LED controllers are 0x4000010 and 0x4000020 based on PCIE_BAR4, respectively. The on-chip memory base address is 0x00000000 relative to the DMA controller. The above definition is 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_AVMM.DLL. Then, it calls 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_AVMM.h. If the developer changes the Vender ID, Device ID, and PCI Express IP, they also need to change the ID value defined in TERASIC_PCIE_AVMM.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,(DWORD)Mask);
```

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

```
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:

```
PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);

PCIE_DmaWrite(hPCIe, LocalAddr, pWrite, nTestSize);
```

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

```
// read config - link status
if (PCIE_ConfigRead32(hPCIe,  0x90, &Data32)){
  switch((Data32 >> 16) & 0x0F){
              case 1:
                  printf("Current Link Speed is Gen1\r\n");
                  break;
              case 2:
                  printf("Current Link Speed is Gen2\r\n");
                  break;
              case 3:
                  printf("Current Link Speed is Gen3\r\n");
                  break;
              default:
                  printf("Current Link Speed is Unknown\r\n");
                  break;
  }
  switch((Data32 >> 20) & 0x3F){
              case 1:
                  printf("Negotiated Link Width is x1\r\n");
                  break;
              case 2:
                  printf("Negotiated Link Width is x2\r\n");
                  break;
              case 4:
                  printf("Negotiated Link Width is x4\r\n");
                  break;
              case 8:
                  printf("Negotiated Link Width is x8\r\n");
                  break;
              case 16:
                  printf("Negotiated Link Width is x16\r\n");
                  break;
              default:
                  printf("Negotiated Link Width is Unknown\r\n");
                  break;
  }
}else{
  bPass = false;
}
```

Chapter 8

Transceiver Verification

This chapter describes how to verify the FPGA transceivers for the SATA and FMC XCVRs by using the test code provided in the TR5 system CD.

8.1 Function of the Transceiver Test Code

The transceiver test code is used to verify the transceiver channels for the SATA and FMC XCVR ports through an external loopback method. The transceiver channels are verified with PRBS31 test pattern and with the data rates:

- For 5SGXEA7N2F45C2 Devic**e: 12.5 Gbps for FMC XCVRs** and 6Gbps for SATA.
- For 5SGXEABN3F45I3YY Device: **10.3125 Gbps for FMC XCVRs** and 6Gbps for SATA.

8.2 Function of the Transceiver Test Code

To enable an external loopback of transceiver channels, one of the following two fixtures are required:

- SATA Cable, as shown in **Figure 8-1** (General SATA 3.0 cable)
- FMC Loopback Card, as shown in **Figure 8-2**

Figure 8-1 SATA Cable

Figure 8-2 SATA Cable

Figure 8-3 shows the FPGA board with SATA Cable four FMC Loopback Cards installed.

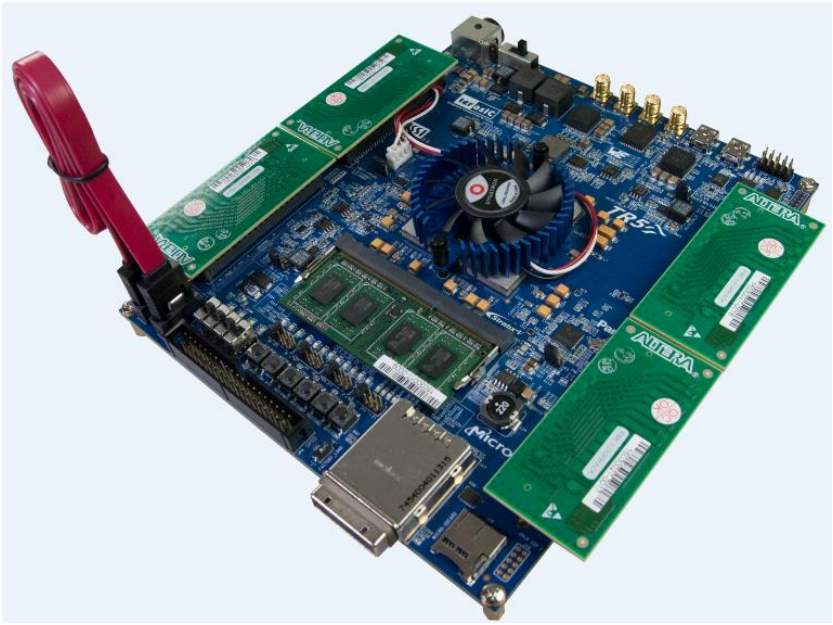


Figure 8-3 SATA Cable and Four FMC Loopback Card Installed

8.1 Testing

The transceiver test code is available in the folder System CD\Tool\Transceiver_Test. Here are the procedures to perform transceiver channel test:

1. Copy Transceiver_Test folder to your local disk.
2. Ensure that the FPGA board is NOT powered on.
3. Plug-in the FMC loopback Card.
   - Note: There is only one FMC loopback card in the Kit.
4. Plug-in the SATA cable to loop SATA HOST and DEVICE connector
   - *HOST TX-> DEVICE RX, DEVICE TX -> HOST RX*
5. Connect your FPGA board to your PC with a mini USB cable.
6. Power on the FPGA board
7. Execute 'test.bat" in the Transceiver_Test folder under your local disk.
8. The batch file will download .sof and .elf files, and start the test immediately. The test result is shown in the Nios-Terminal, as shown in **Figure 8-4**.
9. To terminate the test, press one of the BUTTON0~3 buttons on the FPGA board. The loopback test will terminate and show the summary test results as shown in **Figure 8-5**.

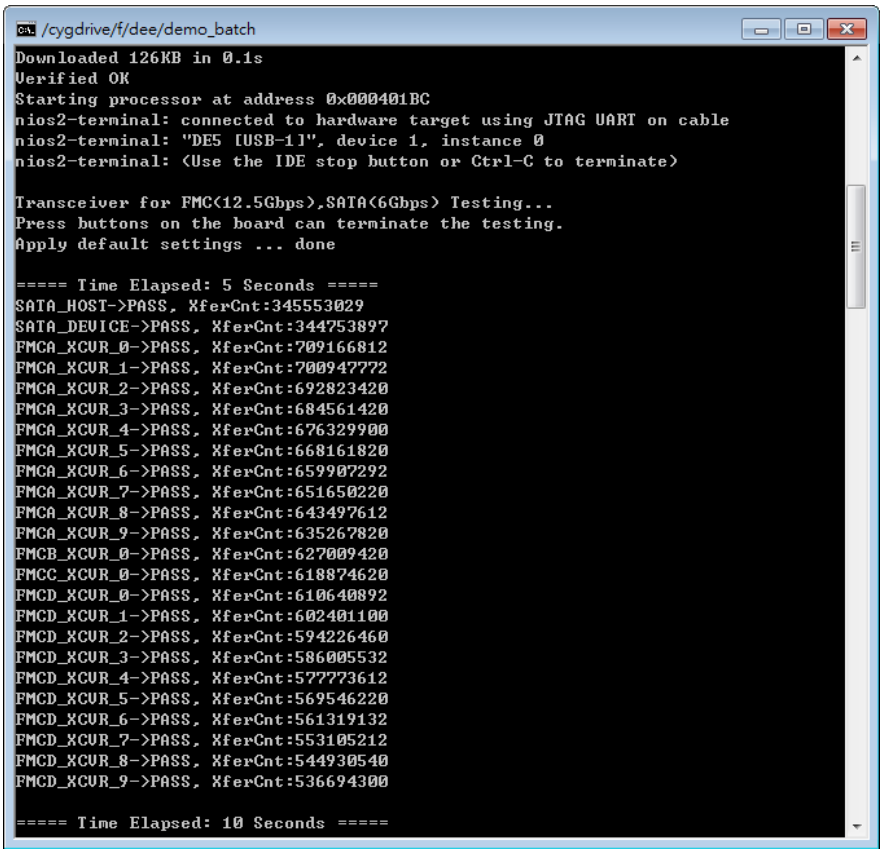


Figure 8-4 Transceiver Loopback Test in Progress

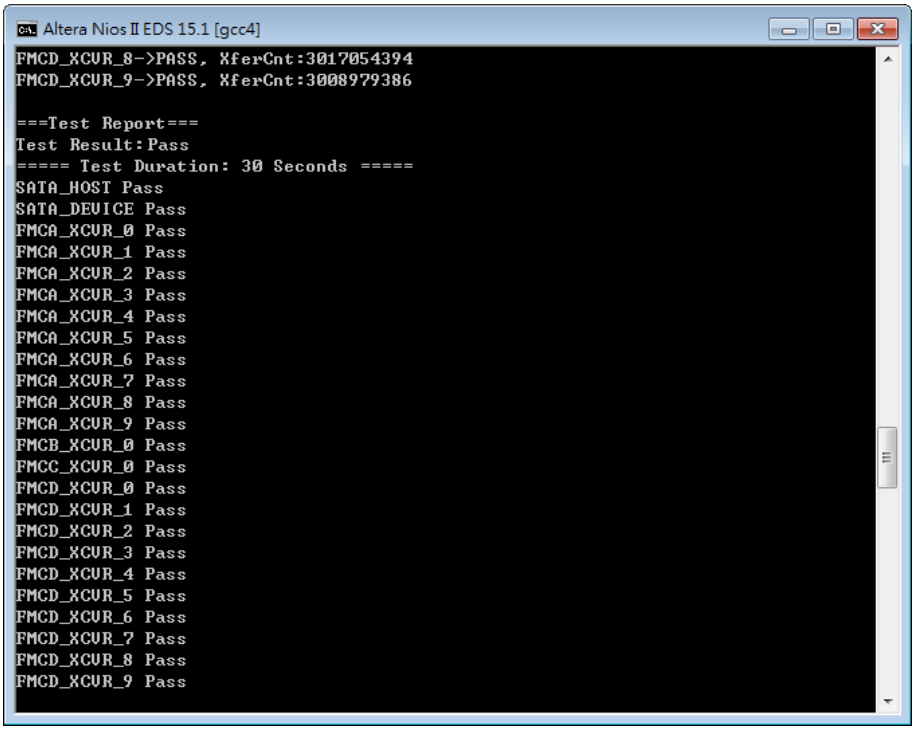


Figure 8-5 Transceiver Loopback Done

Chapter 9

FMC Connectors Pin Out

This chapter gives all the pin assignments of the FMC connectors on the TR5.

Table 9-1 FMCA (HPC) Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
FMCA_LA_TX_CLK_p	FMCA LA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B16
FMCA_LA_TX_CLK_n	FMCA LA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A16
FMCA_LA_TX_p0	FMCA LA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R19
FMCA_LA_TX_n0	FMCA LA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R18
FMCA_LA_TX_p1	FMCA LA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M23
FMCA_LA_TX_n1	FMCA LA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L24
FMCA_LA_TX_p2	FMCA LA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W29
FMCA_LA_TX_n2	FMCA LA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V30
FMCA_LA_TX_p3	FMCA LA bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K28
FMCA_LA_TX_n3	FMCA LA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J28
FMCA_LA_TX_p4	FMCA LA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V27
FMCA_LA_TX_n4	FMCA LA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V26
FMCA_LA_TX_p5	FMCA LA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H29
FMCA_LA_TX_n5	FMCA LA bank tx	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H28

	data n5		
FMCA_LA_TX_p6	FMCA LA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P28
FMCA_LA_TX_n6	FMCA LA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P29
FMCA_LA_TX_p7	FMCA LA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G31
FMCA_LA_TX_n7	FMCA LA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F31
FMCA_LA_TX_p8	FMCA LA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N29
FMCA_LA_TX_n8	FMCA LA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M30
FMCA_LA_TX_p9	FMCA LA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E26
FMCA_LA_TX_n9	FMCA LA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D26
FMCA_LA_TX_p10	FMCA LA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C28
FMCA_LA_TX_n10	FMCA LA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B28
FMCA_LA_TX_p11	FMCA LA bank tx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P20
FMCA_LA_TX_n11	FMCA LA bank tx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N20
FMCA_LA_TX_p12	FMCA LA bank tx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F25
FMCA_LA_TX_n12	FMCA LA bank tx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F24
FMCA_LA_TX_p13	FMCA LA bank tx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L20
FMCA_LA_TX_n13	FMCA LA bank tx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K19
FMCA_LA_TX_p14	FMCA LA bank tx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E24
FMCA_LA_TX_n14	FMCA LA bank tx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D24
FMCA_LA_TX_p15	FMCA LA bank tx data p15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E23
FMCA_LA_TX_n15	FMCA LA bank tx data n15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D23
FMCA_LA_TX_p16	FMCA LA bank tx	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B19

	data p16		
FMCA_LA_TX_n16	FMCA LA bank tx data n16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A19
FMCA_LA_RX_CLK_p	FMCA LA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G16
FMCA_LA_RX_CLK_n	FMCA LA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F16
FMCA_LA_RX_p0	FMCA LA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K30
FMCA_LA_RX_n0	FMCA LA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K29
FMCA_LA_RX_p1	FMCA LA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V25
FMCA_LA_RX_n1	FMCA LA bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U26
FMCA_LA_RX_p2	FMCA LA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J30
FMCA_LA_RX_n2	FMCA LA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H30
FMCA_LA_RX_p3	FMCA LA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T28
FMCA_LA_RX_n3	FMCA LA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R28
FMCA_LA_RX_p4	FMCA LA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J27
FMCA_LA_RX_n4	FMCA LA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H26
FMCA_LA_RX_p5	FMCA LA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P26
FMCA_LA_RX_n5	FMCA LA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N26
FMCA_LA_RX_p6	FMCA LA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U24
FMCA_LA_RX_n6	FMCA LA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T24
FMCA_LA_RX_p7	FMCA LA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T20
FMCA_LA_RX_n7	FMCA LA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T19
FMCA_LA_RX_p8	FMCA LA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C31
FMCA_LA_RX_n8	FMCA LA bank rx	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B31

	data n8		
FMCA_LA_RX_p9	FMCA LA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A32
FMCA_LA_RX_n9	FMCA LA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A31
FMCA_LA_RX_p10	FMCA LA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H25
FMCA_LA_RX_n10	FMCA LA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G25
FMCA_LA_RX_p11	FMCA LA bank rx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P19
FMCA_LA_RX_n11	FMCA LA bank rx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P18
FMCA_LA_RX_p12	FMCA LA bank rx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N17
FMCA_LA_RX_n12	FMCA LA bank rx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P17
FMCA_LA_RX_p13	FMCA LA bank rx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G19
FMCA_LA_RX_n13	FMCA LA bank rx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F19
FMCA_LA_RX_p14	FMCA LA bank rx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E17
FMCA_LA_RX_n14	FMCA LA bank rx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D17
FMCA_HA_TX_CLK_p	FMCA HA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T31
FMCA_HA_TX_CLK_n	FMCA HA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R31
FMCA_HA_TX_p0	FMCA HA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T27
FMCA_HA_TX_n0	FMCA HA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U27
FMCA_HA_TX_p1	FMCA HA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y32
FMCA_HA_TX_n1	FMCA HA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W31
FMCA_HA_TX_p2	FMCA HA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W28
FMCA_HA_TX_n2	FMCA HA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V28
FMCA_HA_TX_p3	FMCA HA bank	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R27

	tx data p3		
FMCA_HA_TX_n3	FMCA HA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P27
FMCA_HA_TX_p4	FMCA HA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P31
FMCA_HA_TX_n4	FMCA HA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P30
FMCA_HA_TX_p5	FMCA HA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W18
FMCA_HA_TX_n5	FMCA HA bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V18
FMCA_HA_TX_p6	FMCA HA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P23
FMCA_HA_TX_n6	FMCA HA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N23
FMCA_HA_TX_p7	FMCA HA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J25
FMCA_HA_TX_n7	FMCA HA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J24
FMCA_HA_TX_p8	FMCA HA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V29
FMCA_HA_TX_n8	FMCA HA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U29
FMCA_HA_TX_p9	FMCA HA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L30
FMCA_HA_TX_n9	FMCA HA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L29
FMCA_HA_TX_p10	FMCA HA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H32
FMCA_HA_TX_n10	FMCA HA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H31
FMCA_HA_RX_CLK_p	FMCA HA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N25
FMCA_HA_RX_CLK_n	FMCA HA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M25
FMCA_HA_RX_p0	FMCA HA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y30
FMCA_HA_RX_n0	FMCA HA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y29
FMCA_HA_RX_p1	FMCA HA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W32
FMCA_HA_RX_n1	FMCA HA bank	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V31

	rx data n1		
FMCA_HA_RX_p2	FMCA HA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U30
FMCA_HA_RX_n2	FMCA HA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T29
FMCA_HA_RX_p3	FMCA HA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T30
FMCA_HA_RX_n3	FMCA HA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R30
FMCA_HA_RX_p4	FMCA HA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M27
FMCA_HA_RX_n4	FMCA HA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L26
FMCA_HA_RX_p5	FMCA HA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y28
FMCA_HA_RX_n5	FMCA HA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y27
FMCA_HA_RX_p6	FMCA HA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L23
FMCA_HA_RX_n6	FMCA HA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K24
FMCA_HA_RX_p7	FMCA HA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H27
FMCA_HA_RX_n7	FMCA HA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G26
FMCA_HA_RX_p8	FMCA HA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L27
FMCA_HA_RX_n8	FMCA HA bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K27
FMCA_HA_RX_p9	FMCA HA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G32
FMCA_HA_RX_n9	FMCA HA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F32
FMCA_HA_RX_p10	FMCA HA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E29
FMCA_HA_RX_n10	FMCA HA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D29
FMCA_HB_TX_CLK_p	FMCA HB bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B23
FMCA_HB_TX_CLK_n	FMCA HB bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A23
FMCA_HB_TX_p0	FMCA HB bank	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N28

	tx data p0		
FMCA_HB_TX_n0	FMCA HB bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M28
FMCA_HB_TX_p1	FMCA HB bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M18
FMCA_HB_TX_n1	FMCA HB bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M17
FMCA_HB_TX_p2	FMCA HB bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K18
FMCA_HB_TX_n2	FMCA HB bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K17
FMCA_HB_TX_p3	FMCA HB bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H17
FMCA_HB_TX_n3	FMCA HB bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H16
FMCA_HB_TX_p4	FMCA HB bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G17
FMCA_HB_TX_n4	FMCA HB bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F17
FMCA_HB_TX_p5	FMCA HB bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C30
FMCA_HB_TX_n5	FMCA HB bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B29
FMCA_HB_TX_p6	FMCA HB bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B26
FMCA_HB_TX_n6	FMCA HB bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A26
FMCA_HB_TX_p7	FMCA HB bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E18
FMCA_HB_TX_n7	FMCA HB bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D18
FMCA_HB_TX_p8	FMCA HB bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E30
FMCA_HB_TX_n8	FMCA HB bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D30
FMCA_HB_TX_p9	FMCA HB bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A29
FMCA_HB_TX_n9	FMCA HB bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A28
FMCA_HB_TX_p10	FMCA HB bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B17
FMCA_HB_TX_n10	FMCA HB bank	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A17

	tx data n10		
FMCA_HB_RX_CLK_p	FMCA HB bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E32
FMCA_HB_RX_CLK_n	FMCA HB bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D32
FMCA_HB_RX_p0	FMCA HB bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R24
FMCA_HB_RX_n0	FMCA HB bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P24
FMCA_HB_RX_p1	FMCA HB bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U18
FMCA_HB_RX_n1	FMCA HB bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T18
FMCA_HB_RX_p2	FMCA HB bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N19
FMCA_HB_RX_n2	FMCA HB bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M20
FMCA_HB_RX_p3	FMCA HB bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J19
FMCA_HB_RX_n3	FMCA HB bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H19
FMCA_HB_RX_p4	FMCA HB bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C19
FMCA_HB_RX_n4	FMCA HB bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C18
FMCA_HB_RX_p5	FMCA HB bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D27
FMCA_HB_RX_n5	FMCA HB bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C27
FMCA_HB_RX_p6	FMCA HB bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B25
FMCA_HB_RX_n6	FMCA HB bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A25
FMCA_HB_RX_p7	FMCA HB bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K26
FMCA_HB_RX_n7	FMCA HB bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K25
FMCA_HB_RX_p8	FMCA HB bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F26
FMCA_HB_RX_n8	FMCA HB bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E27
FMCA_HB_RX_p9	FMCA HB bank	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C25

	rx data p9		
FMCA_HB_RX_n9	FMCA HB bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C24
FMCA_HB_RX_p10	FMCA HB bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H24
FMCA_HB_RX_n10	FMCA HB bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H23
FMCA_SCL	FMCA serial clock	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F29
FMCA_SDA	FMCA serial data	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F28
FMCA_GA0	FMCA GA0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H18
FMCA_GA1	FMCA GA1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T23
FMCA_CLK_M2C_p0	FMCA card to carrier clock p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G23
FMCA_CLK_M2C_n0	FMCA card to carrier clock n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F23
FMCA_CLK_M2C_p1	FMCA card to carrier clock p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G29
FMCA_CLK_M2C_n1	FMCA card to carrier clock n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G28
FMCA_GBTCLK_M2C_p0	FMCA DP Reference clock 0	1.4-V PCML / LVDS	PIN_AB39
FMCA_GBTCLK_M2C_p1	FMCA DP Reference clock 1	1.4-V PCML / LVDS	PIN_AB40
FMCA_DP_C2M_p0	FMCA Transmitter DP 0	1.4-V PCML	PIN_W41
FMCA_DP_M2C_p0	FMCA Receiver DP 0	1.4-V PCML	PIN_AB43
FMCA_DP_C2M_p1	FMCA Transmitter DP 1	1.4-V PCML	PIN_U41
FMCA_DP_M2C_p1	FMCA Receiver DP 1	1.4-V PCML	PIN_Y43
FMCA_DP_C2M_p2	FMCA Transmitter DP 2	1.4-V PCML	PIN_R41
FMCA_DP_M2C_p2	FMCA Receiver DP 2	1.4-V PCML	PIN_V43
FMCA_DP_C2M_p3	FMCA Transmitter DP 3	1.4-V PCML	PIN_N41
FMCA_DP_M2C_p3	FMCA Receiver DP 3	1.4-V PCML	PIN_T43
FMCA_DP_C2M_p4	FMCA Transmitter DP 4	1.4-V PCML	PIN_J41

FMCA_DP_M2C_p4	FMCA Receiver DP 4	1.4-V PCML	PIN_M43
FMCA_DP_C2M_p5	FMCA Transmitter DP 5	1.4-V PCML	PIN_K39
FMCA_DP_M2C_p5	FMCA Receiver DP 5	1.4-V PCML	PIN_K43
FMCA_DP_C2M_p6	FMCA Transmitter DP 6	1.4-V PCML	PIN_H39
FMCA_DP_M2C_p6	FMCA Receiver DP 6	1.4-V PCML	PIN_H43
FMCA_DP_C2M_p7	FMCA Transmitter DP 7	1.4-V PCML	PIN_G41
FMCA_DP_M2C_p7	FMCA Receiver DP 7	1.4-V PCML	PIN_F43
FMCA_DP_C2M_p8	FMCA Transmitter DP 8	1.4-V PCML	PIN_E41
FMCA_DP_M2C_p8	FMCA Receiver DP 8	1.4-V PCML	PIN_D43
FMCA_DP_C2M_p9	FMCA Transmitter DP 9	1.4-V PCML	PIN_D39
FMCA_DP_M2C_p9	FMCA Receiver DP 9	1.4-V PCML	PIN_C41

Table 9-2 FMCB (LPC) Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
FMCB_LA_TX_CLK_p	FMCB LA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G37
FMCB_LA_TX_CLK_n	FMCB LA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F36
FMCB_LA_TX_p0	FMCB LA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P33
FMCB_LA_TX_n0	FMCB LA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P32
FMCB_LA_TX_p1	FMCB LA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V34
FMCB_LA_TX_n1	FMCB LA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V33
FMCB_LA_TX_p2	FMCB LA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W35

FMCB_LA_TX_n2	FMCB LA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V36
FMCB_LA_TX_p3	FMCB LA bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R33
FMCB_LA_TX_n3	FMCB LA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P34
FMCB_LA_TX_p4	FMCB LA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T34
FMCB_LA_TX_n4	FMCB LA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R34
FMCB_LA_TX_p5	FMCB LA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P39
FMCB_LA_TX_n5	FMCB LA bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P38
FMCB_LA_TX_p6	FMCB LA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N38
FMCB_LA_TX_n6	FMCB LA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N37
FMCB_LA_TX_p7	FMCB LA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M34
FMCB_LA_TX_n7	FMCB LA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L33
FMCB_LA_TX_p8	FMCB LA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M39
FMCB_LA_TX_n8	FMCB LA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M38
FMCB_LA_TX_p9	FMCB LA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M37
FMCB_LA_TX_n9	FMCB LA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M36
FMCB_LA_TX_p10	FMCB LA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J34
FMCB_LA_TX_n10	FMCB LA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J33
FMCB_LA_TX_p11	FMCB LA bank tx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K35
FMCB_LA_TX_n11	FMCB LA bank tx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K34
FMCB_LA_TX_p12	FMCB LA bank tx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E33
FMCB_LA_TX_n12	FMCB LA bank tx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D33

FMCB_LA_TX_p13	FMCB LA bank tx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G34
FMCB_LA_TX_n13	FMCB LA bank tx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F34
FMCB_LA_TX_p14	FMCB LA bank tx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C37
FMCB_LA_TX_n14	FMCB LA bank tx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B38
FMCB_LA_TX_p15	FMCB LA bank tx data p15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D36
FMCB_LA_TX_n15	FMCB LA bank tx data n15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D35
FMCB_LA_TX_p16	FMCB LA bank tx data p16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C33
FMCB_LA_TX_n16	FMCB LA bank tx data n16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B32
FMCB_LA_RX_CLK_p	FMCB LA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U36
FMCB_LA_RX_CLK_n	FMCB LA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T36
FMCB_LA_RX_p0	FMCB LA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W34
FMCB_LA_RX_n0	FMCB LA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V35
FMCB_LA_RX_p1	FMCB LA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U33
FMCB_LA_RX_n1	FMCB LA bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U32
FMCB_LA_RX_p2	FMCB LA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U35
FMCB_LA_RX_n2	FMCB LA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T35
FMCB_LA_RX_p3	FMCB LA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T33
FMCB_LA_RX_n3	FMCB LA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T32
FMCB_LA_RX_p4	FMCB LA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N32
FMCB_LA_RX_n4	FMCB LA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M33
FMCB_LA_RX_p5	FMCB LA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L32

FMCB_LA_RX_n5	FMCB LA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K32
FMCB_LA_RX_p6	FMCB LA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L36
FMCB_LA_RX_n6	FMCB LA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L35
FMCB_LA_RX_p7	FMCB LA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J36
FMCB_LA_RX_n7	FMCB LA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H36
FMCB_LA_RX_p8	FMCB LA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K37
FMCB_LA_RX_n8	FMCB LA bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K36
FMCB_LA_RX_p9	FMCB LA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H35
FMCB_LA_RX_n9	FMCB LA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G35
FMCB_LA_RX_p10	FMCB LA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H34
FMCB_LA_RX_n10	FMCB LA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H33
FMCB_LA_RX_p11	FMCB LA bank rx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F35
FMCB_LA_RX_n11	FMCB LA bank rx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E35
FMCB_LA_RX_p12	FMCB LA bank rx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C36
FMCB_LA_RX_n12	FMCB LA bank rx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B35
FMCB_LA_RX_p13	FMCB LA bank rx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C34
FMCB_LA_RX_n13	FMCB LA bank rx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B34
FMCB_LA_RX_p14	FMCB LA bank rx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A35
FMCB_LA_RX_n14	FMCB LA bank rx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A34
FMCB_SCL	FMCB serial clock	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E36
FMCB_SDA	FMCB serial data	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D37
FMCB_GA0	FMCB GA0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J37
FMCB_GA1	FMCB GA1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H37

FMCB_CLK_M2C_p0	FMCB card to carrier clock p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B39
FMCB_CLK_M2C_n0	FMCB card to carrier clock n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A38
FMCB_CLK_M2C_p1	FMCB card to carrier clock p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B37
FMCB_CLK_M2C_n1	FMCB card to carrier clock n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A37
FMCB_GBTCLK_M2C_p0	FMCB DP Reference clock 0	1.4-V PCML / LVDS	PIN_AF38
FMCB_DP_C2M_p0	FMCB Transmitter DP 0	1.4-V PCML	PIN_AL41
FMCB_DP_M2C_p0	FMCB Receiver DP 0	1.4-V PCML	PIN_AP43

Table 9-3 FMCC (LPC) Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
FMCC_LA_TX_CLK_p	FMCC LA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY19
FMCC_LA_TX_CLK_n	FMCC LA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA19
FMCC_LA_TX_p0	FMCC LA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU20
FMCC_LA_TX_n0	FMCC LA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV20
FMCC_LA_TX_p1	FMCC LA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU16
FMCC_LA_TX_n1	FMCC LA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV16
FMCC_LA_TX_p2	FMCC LA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG17
FMCC_LA_TX_n2	FMCC LA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG18
FMCC_LA_TX_p3	FMCC LA bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ17
FMCC_LA_TX_n3	FMCC LA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AK17
FMCC_LA_TX_p4	FMCC LA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ19
FMCC_LA_TX_n4	FMCC LA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ20

FMCC_LA_TX_p5	FMCC LA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU17
FMCC_LA_TX_n5	FMCC LA bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV17
FMCC_LA_TX_p6	FMCC LA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL18
FMCC_LA_TX_n6	FMCC LA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL19
FMCC_LA_TX_p7	FMCC LA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV19
FMCC_LA_TX_n7	FMCC LA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW19
FMCC_LA_TX_p8	FMCC LA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW20
FMCC_LA_TX_n8	FMCC LA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW21
FMCC_LA_TX_p9	FMCC LA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC20
FMCC_LA_TX_n9	FMCC LA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD20
FMCC_LA_TX_p10	FMCC LA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU21
FMCC_LA_TX_n10	FMCC LA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU22
FMCC_LA_TX_p11	FMCC LA bank tx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY21
FMCC_LA_TX_n11	FMCC LA bank tx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA21
FMCC_LA_TX_p12	FMCC LA bank tx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR17
FMCC_LA_TX_n12	FMCC LA bank tx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT17
FMCC_LA_TX_p13	FMCC LA bank tx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL20
FMCC_LA_TX_n13	FMCC LA bank tx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AM20
FMCC_LA_TX_p14	FMCC LA bank tx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN19
FMCC_LA_TX_n14	FMCC LA bank tx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AP19
FMCC_LA_TX_p15	FMCC LA bank tx data p15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AP21

FMCC_LA_TX_n15	FMCC LA bank tx data n15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR21
FMCC_LA_TX_p16	FMCC LA bank tx data p16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AM22
FMCC_LA_TX_n16	FMCC LA bank tx data n16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN22
FMCC_LA_RX_CLK_p	FMCC LA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB18
FMCC_LA_RX_CLK_n	FMCC LA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB17
FMCC_LA_RX_p0	FMCC LA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY22
FMCC_LA_RX_n0	FMCC LA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA22
FMCC_LA_RX_p1	FMCC LA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG21
FMCC_LA_RX_n1	FMCC LA bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AH21
FMCC_LA_RX_p2	FMCC LA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG19
FMCC_LA_RX_n2	FMCC LA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG20
FMCC_LA_RX_p3	FMCC LA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AK20
FMCC_LA_RX_n3	FMCC LA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ21
FMCC_LA_RX_p4	FMCC LA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB20
FMCC_LA_RX_n4	FMCC LA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB21
FMCC_LA_RX_p5	FMCC LA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC22
FMCC_LA_RX_n5	FMCC LA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD22
FMCC_LA_RX_p6	FMCC LA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AM19
FMCC_LA_RX_n6	FMCC LA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN20
FMCC_LA_RX_p7	FMCC LA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AP18
FMCC_LA_RX_n7	FMCC LA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR19

FMCC_LA_RX_p8	FMCC LA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN17
FMCC_LA_RX_n8	FMCC LA bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AP16
FMCC_LA_RX_p9	FMCC LA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR18
FMCC_LA_RX_n9	FMCC LA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT18
FMCC_LA_RX_p10	FMCC LA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR20
FMCC_LA_RX_n10	FMCC LA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT20
FMCC_LA_RX_p11	FMCC LA bank rx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU18
FMCC_LA_RX_n11	FMCC LA bank rx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU19
FMCC_LA_RX_p12	FMCC LA bank rx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW16
FMCC_LA_RX_n12	FMCC LA bank rx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW17
FMCC_LA_RX_p13	FMCC LA bank rx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR22
FMCC_LA_RX_n13	FMCC LA bank rx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT21
FMCC_LA_RX_p14	FMCC LA bank rx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV22
FMCC_LA_RX_n14	FMCC LA bank rx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW22
FMCC_SCL	FMCC serial clock	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AH18
FMCC_SDA	FMCC serial data	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AH19
FMCC_GA0	FMCC GA0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY16
FMCC_GA1	FMCC GA1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA16
FMCC_CLK_M2C_p0	FMCC card to carrier clock p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC19
FMCC_CLK_M2C_n0	FMCC card to carrier clock n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD19
FMCC_CLK_M2C_p1	FMCC card to carrier clock p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ15
FMCC_CLK_M2C_n1	FMCC card to carrier clock n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ16
FMCC_GBTCLK_M2C_p0	FMCC DP Reference clock 0	1.4-V PCML / LVDS	PIN_AF7

FMCC_DP_C2M_p0	FMCC Transmitter DP 0	1.4-V PCML	PIN_AL4
FMCC_DP_M2C_p0	FMCC Receiver DP 0	1.4-V PCML	PIN_AP2

Table 9-4 FMCD (HPC) Pin Assignments, Schematic Signal Names, and Functions

Schematic Signal Name	*Description*	*I/O Standard*	*Stratix V GX Pin Number*
FMCD_LA_TX_CLK_p	FMCD LA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F9
FMCD_LA_TX_CLK_n	FMCD LA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E9
FMCD_LA_TX_p0	FMCD LA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D11
FMCD_LA_TX_n0	FMCD LA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C10
FMCD_LA_TX_p1	FMCD LA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B10
FMCD_LA_TX_n1	FMCD LA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A10
FMCD_LA_TX_p2	FMCD LA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D10
FMCD_LA_TX_n2	FMCD LA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C9
FMCD_LA_TX_p3	FMCD LA bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B7
FMCD_LA_TX_n3	FMCD LA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A7
FMCD_LA_TX_p4	FMCD LA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E12
FMCD_LA_TX_n4	FMCD LA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E11
FMCD_LA_TX_p5	FMCD LA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K13
FMCD_LA_TX_n5	FMCD LA bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J13
FMCD_LA_TX_p6	FMCD LA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H12
FMCD_LA_TX_n6	FMCD LA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H11
FMCD_LA_TX_p7	FMCD LA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K12

FMCD_LA_TX_n7	FMCD LA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J12
FMCD_LA_TX_p8	FMCD LA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K8
FMCD_LA_TX_n8	FMCD LA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J9
FMCD_LA_TX_p9	FMCD LA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K10
FMCD_LA_TX_n9	FMCD LA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K9
FMCD_LA_TX_p10	FMCD LA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P8
FMCD_LA_TX_n10	FMCD LA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N8
FMCD_LA_TX_p11	FMCD LA bank tx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M12
FMCD_LA_TX_n11	FMCD LA bank tx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L12
FMCD_LA_TX_p12	FMCD LA bank tx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T10
FMCD_LA_TX_n12	FMCD LA bank tx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R10
FMCD_LA_TX_p13	FMCD LA bank tx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N11
FMCD_LA_TX_n13	FMCD LA bank tx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M11
FMCD_LA_TX_p14	FMCD LA bank tx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U12
FMCD_LA_TX_n14	FMCD LA bank tx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U11
FMCD_LA_TX_p15	FMCD LA bank tx data p15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P12
FMCD_LA_TX_n15	FMCD LA bank tx data n15	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R12
FMCD_LA_TX_p16	FMCD LA bank tx data p16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U14
FMCD_LA_TX_n16	FMCD LA bank tx data n16	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T14
FMCD_LA_RX_CLK_p	FMCD LA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M9
FMCD_LA_RX_CLK_n	FMCD LA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L9

FMCD_LA_RX_p0	FMCD LA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H15
FMCD_LA_RX_n0	FMCD LA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G14
FMCD_LA_RX_p1	FMCD LA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H14
FMCD_LA_RX_n1	FMCD LA bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H13
FMCD_LA_RX_p2	FMCD LA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D12
FMCD_LA_RX_n2	FMCD LA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C12
FMCD_LA_RX_p3	FMCD LA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B11
FMCD_LA_RX_n3	FMCD LA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A11
FMCD_LA_RX_p4	FMCD LA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G11
FMCD_LA_RX_n4	FMCD LA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F11
FMCD_LA_RX_p5	FMCD LA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G10
FMCD_LA_RX_n5	FMCD LA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F10
FMCD_LA_RX_p6	FMCD LA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L15
FMCD_LA_RX_n6	FMCD LA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K14
FMCD_LA_RX_p7	FMCD LA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K11
FMCD_LA_RX_n7	FMCD LA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L11
FMCD_LA_RX_p8	FMCD LA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J10
FMCD_LA_RX_n8	FMCD LA bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H10
FMCD_LA_RX_p9	FMCD LA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P14
FMCD_LA_RX_n9	FMCD LA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N14
FMCD_LA_RX_p10	FMCD LA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T12

FMCD_LA_RX_n10	FMCD LA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T11
FMCD_LA_RX_p11	FMCD LA bank rx data p11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P13
FMCD_LA_RX_n11	FMCD LA bank rx data n11	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N13
FMCD_LA_RX_p12	FMCD LA bank rx data p12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U9
FMCD_LA_RX_n12	FMCD LA bank rx data n12	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T9
FMCD_LA_RX_p13	FMCD LA bank rx data p13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V10
FMCD_LA_RX_n13	FMCD LA bank rx data n13	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V9
FMCD_LA_RX_p14	FMCD LA bank rx data p14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V12
FMCD_LA_RX_n14	FMCD LA bank rx data n14	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V11
FMCD_HA_TX_CLK_p	FMCD HA bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T17
FMCD_HA_TX_CLK_n	FMCD HA bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T16
FMCD_HA_TX_p0	FMCD HA bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G13
FMCD_HA_TX_n0	FMCD HA bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F13
FMCD_HA_TX_p1	FMCD HA bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L21
FMCD_HA_TX_n1	FMCD HA bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K20
FMCD_HA_TX_p2	FMCD HA bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M14
FMCD_HA_TX_n2	FMCD HA bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_L14
FMCD_HA_TX_p3	FMCD HA bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R21
FMCD_HA_TX_n3	FMCD HA bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_P21
FMCD_HA_TX_p4	FMCD HA bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U21
FMCD_HA_TX_n4	FMCD HA bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T21

FMCD_HA_TX_p5	FMCD HA bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G20
FMCD_HA_TX_n5	FMCD HA bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F20
FMCD_HA_TX_p6	FMCD HA bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B13
FMCD_HA_TX_n6	FMCD HA bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A13
FMCD_HA_TX_p7	FMCD HA bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N16
FMCD_HA_TX_n7	FMCD HA bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M15
FMCD_HA_TX_p8	FMCD HA bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D14
FMCD_HA_TX_n8	FMCD HA bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C13
FMCD_HA_TX_p9	FMCD HA bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K16
FMCD_HA_TX_n9	FMCD HA bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J16
FMCD_HA_TX_p10	FMCD HA bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_T22
FMCD_HA_TX_n10	FMCD HA bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_R22
FMCD_HA_RX_CLK_p	FMCD HA bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E8
FMCD_HA_RX_CLK_n	FMCD HA bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D9
FMCD_HA_RX_p0	FMCD HA bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F22
FMCD_HA_RX_n0	FMCD HA bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F21
FMCD_HA_RX_p1	FMCD HA bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E15
FMCD_HA_RX_n1	FMCD HA bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_D15
FMCD_HA_RX_p2	FMCD HA bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H21
FMCD_HA_RX_n2	FMCD HA bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H20
FMCD_HA_RX_p3	FMCD HA bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K15

FMCD_HA_RX_n3	FMCD HA bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J15
FMCD_HA_RX_p4	FMCD HA bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_V19
FMCD_HA_RX_n4	FMCD HA bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_U20
FMCD_HA_RX_p5	FMCD HA bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_H22
FMCD_HA_RX_n5	FMCD HA bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_G22
FMCD_HA_RX_p6	FMCD HA bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_F14
FMCD_HA_RX_n6	FMCD HA bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_E14
FMCD_HA_RX_p7	FMCD HA bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_N22
FMCD_HA_RX_n7	FMCD HA bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_M22
FMCD_HA_RX_p8	FMCD HA bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C16
FMCD_HA_RX_n8	FMCD HA bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C15
FMCD_HA_RX_p9	FMCD HA bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_K21
FMCD_HA_RX_n9	FMCD HA bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_J21
FMCD_HA_RX_p10	FMCD HA bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y17
FMCD_HA_RX_n10	FMCD HA bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W17
FMCD_HB_TX_CLK_p	FMCD HB bank tx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU14
FMCD_HB_TX_CLK_n	FMCD HB bank tx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV14
FMCD_HB_TX_p0	FMCD HB bank tx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AE15
FMCD_HB_TX_n0	FMCD HB bank tx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AE16
FMCD_HB_TX_p1	FMCD HB bank tx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ14
FMCD_HB_TX_n1	FMCD HB bank tx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AK15

FMCD_HB_TX_p2	FMCD HB bank tx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN15
FMCD_HB_TX_n2	FMCD HB bank tx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AM16
FMCD_HB_TX_p3	FMCD HB bank tx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT14
FMCD_HB_TX_n3	FMCD HB bank tx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU13
FMCD_HB_TX_p4	FMCD HB bank tx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW13
FMCD_HB_TX_n4	FMCD HB bank tx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AW14
FMCD_HB_TX_p5	FMCD HB bank tx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AK14
FMCD_HB_TX_n5	FMCD HB bank tx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL14
FMCD_HB_TX_p6	FMCD HB bank tx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC16
FMCD_HB_TX_n6	FMCD HB bank tx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD16
FMCD_HB_TX_p7	FMCD HB bank tx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY13
FMCD_HB_TX_n7	FMCD HB bank tx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA13
FMCD_HB_TX_p8	FMCD HB bank tx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AF16
FMCD_HB_TX_n8	FMCD HB bank tx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG15
FMCD_HB_TX_p9	FMCD HB bank tx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR15
FMCD_HB_TX_n9	FMCD HB bank tx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR14
FMCD_HB_TX_p10	FMCD HB bank tx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB14
FMCD_HB_TX_n10	FMCD HB bank tx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BB15
FMCD_HB_RX_CLK_p	FMCD HB bank rx clock positive	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AH13
FMCD_HB_RX_CLK_n	FMCD HB bank rx clock negative	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AJ13
FMCD_HB_RX_p0	FMCD HB bank rx data p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG16

FMCD_HB_RX_n0	FMCD HB bank rx data n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AF17
FMCD_HB_RX_p1	FMCD HB bank rx data p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AG14
FMCD_HB_RX_n1	FMCD HB bank rx data n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AH15
FMCD_HB_RX_p2	FMCD HB bank rx data p2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AM14
FMCD_HB_RX_n2	FMCD HB bank rx data n2	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AN14
FMCD_HB_RX_p3	FMCD HB bank rx data p3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AT15
FMCD_HB_RX_n3	FMCD HB bank rx data n3	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU15
FMCD_HB_RX_p4	FMCD HB bank rx data p4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AY15
FMCD_HB_RX_n4	FMCD HB bank rx data n4	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BA15
FMCD_HB_RX_p5	FMCD HB bank rx data p5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AE17
FMCD_HB_RX_n5	FMCD HB bank rx data n5	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AE18
FMCD_HB_RX_p6	FMCD HB bank rx data p6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AP15
FMCD_HB_RX_n6	FMCD HB bank rx data n6	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AR16
FMCD_HB_RX_p7	FMCD HB bank rx data p7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC13
FMCD_HB_RX_n7	FMCD HB bank rx data n7	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD13
FMCD_HB_RX_p8	FMCD HB bank rx data p8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL15
FMCD_HB_RX_n8	FMCD HB bank rx data n8	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AL16
FMCD_HB_RX_p9	FMCD HB bank rx data p9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AU12
FMCD_HB_RX_n9	FMCD HB bank rx data n9	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_AV13
FMCD_HB_RX_p10	FMCD HB bank rx data p10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BC14
FMCD_HB_RX_n10	FMCD HB bank rx data n10	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_BD14

FMCD_SCL	FMCD serial clock	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A20
FMCD_SDA	FMCD serial data	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B20
FMCD_GA0	FMCD GA0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C22
FMCD_GA1	FMCD GA1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_C21
FMCD_CLK_M2C_p0	FMCD card to carrier clock p0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_Y16
FMCD_CLK_M2C_n0	FMCD card to carrier clock n0	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_W16
FMCD_CLK_M2C_p1	FMCD card to carrier clock p1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_B8
FMCD_CLK_M2C_n1	FMCD card to carrier clock n1	1.2/1.5/1.8/2.5/3.0V/LVDS	PIN_A8
FMCD_GBTCLK_M2C_p0	FMCD DP Reference clock 0	1.4-V PCML / LVDS	PIN_AB6
FMCD_GBTCLK_M2C_p1	FMCD DP Reference clock 1	1.4-V PCML / LVDS	PIN_V6
FMCD_DP_C2M_p0	FMCD Transmitter DP 0	1.4-V PCML	PIN_W4
FMCD_DP_M2C_p0	FMCD Receiver DP 0	1.4-V PCML	PIN_AB2
FMCD_DP_C2M_p1	FMCD Transmitter DP 1	1.4-V PCML	PIN_U4
FMCD_DP_M2C_p1	FMCD Receiver DP 1	1.4-V PCML	PIN_Y2
FMCD_DP_C2M_p2	FMCD Transmitter DP 2	1.4-V PCML	PIN_R4
FMCD_DP_M2C_p2	FMCD Receiver DP 2	1.4-V PCML	PIN_V2
FMCD_DP_C2M_p3	FMCD Transmitter DP 3	1.4-V PCML	PIN_N4
FMCD_DP_M2C_p3	FMCD Receiver DP 3	1.4-V PCML	PIN_T2
FMCD_DP_C2M_p4	FMCD Transmitter DP 4	1.4-V PCML	PIN_J4
FMCD_DP_M2C_p4	FMCD Receiver DP 4	1.4-V PCML	PIN_M2
FMCD_DP_C2M_p5	FMCD Transmitter DP 5	1.4-V PCML	PIN_K6
FMCD_DP_M2C_p5	FMCD Receiver DP 5	1.4-V PCML	PIN_K2
FMCD_DP_C2M_p6	FMCD Transmitter DP 6	1.4-V PCML	PIN_H6

FMCD_DP_M2C_p6	**FMCD Receiver DP 6**	**1.4-V PCML**	PIN_H2
FMCD_DP_C2M_p7	**FMCD Transmitter DP 7**	**1.4-V PCML**	PIN_G4
FMCD_DP_M2C_p7	**FMCD Receiver DP 7**	**1.4-V PCML**	PIN_F2
FMCD_DP_C2M_p8	**FMCD Transmitter DP 8**	**1.4-V PCML**	PIN_E4
FMCD_DP_M2C_p8	**FMCD Receiver DP 8**	**1.4-V PCML**	PIN_D2
FMCD_DP_C2M_p9	**FMCA Transmitter DP 9**	**1.4-V PCML**	PIN_D6
FMCD_DP_M2C_p9	**FMCD Receiver DP 9**	**1.4-V PCML**	PIN_C4

Additional Information

Getting Help

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

- Terasic Technologies
  9F., No.176, Sec.2, Gongdao 5th Rd,
  East Dist, HsinChu City, Taiwan, 30070
  Email: support@terasic.com
  Web: www.terasic.com
  TR5 Web: tr5.terasic.com

Revision History

Date	Version	Changes
2016.10	First publication	
2016.12	V1.0.1	Modify Section 2.8 FMC Part
2016.12	V1.0.2	Add AB device information
2017.01	V1.0.3	Add FMC don't support bidirectional LVDS note
2017.02	V1.0.4	Modify Table2-7 FMC component infromations