VEEK-MT

VEEK with Multi-touch Capacitive Panel

User Manual

terasIC
www.terasic.com

ALTERA

CONTENTS

Chapter 1

Introduction of the VEEK-MT

The Video and Embedded Evaluation Kit - Multi-touch (VEEK-MT) is a comprehensive design environment with everything embedded developers need to create processing-based systems. VEEK-MT delivers an integrated platform that includes hardware, design tools, intellectual property (IP) and reference designs for developing embedded software and hardware platform in a wide range of applications. The fully integrated kit allows developers to rapidly customize their processor and IP to best suit their specific application. The VEEK-MT features the DE2-115 development board targeting the Cyclone IV E FPGA, as well as a capacitive LCD multimedia color touch panel which natively supports multi-touch gestures. A 5-megapixel digital image sensor, ambient light sensor, and 3-axis accelerometer make up the rich feature-set

The VEEK-MT is preconfigured with an FPGA hardware reference design including several ready-to-run demonstration applications stored on the provided SD card. Software developers can use these reference designs as their platform to quickly architect, develop and build complex embedded systems. By simply scrolling through the demos of your choice on the LCD touch panel, you can evaluate numerous processor system designs.

The all-in-one embedded solution offered on the VEEK-MT, in combination of the LCD touch panel and digital image module, provides embedded developers the ideal platform for multimedia applications with unparallel processing performance. Developers can benefit from the use of FPGA-based embedded processing system such as mitigating design risk and obsolescence, design reuse, reducing bill of material (BOM) costs by integrating powerful graphics engines within the FPGA, and lower cost.

Figure 1-1 shows a photograph of VEEK-MT.

Figure 1-1 The VEEK-MT board overview

The key features of the board are listed below:

■ DE2-115 Development Board

- **Cyclone IV EP4CE115 FPGA**
  - 114,480 LEs
  - 432 M9K memory blocks
  - 3,888 Kb embedded memory
  - 4 PLLs
- **Configuration**
  - On-board USB-Blaster circuitry
  - JTAG and AS mode configuration supported
  - EPCS64 serial configuration device
- **Memory Devices**
  - 128MB SDRAM
  - 2MB SRAM
  - 8MB Flash with 8-bit mode
  - 32Kb EEPROM
- **Switches and Indicators**
  - 18 switches and 4 push-buttons
  - 18 red and 9 green LEDs
  - Eight 7-segment displays
- **Audio**
  - 24-bit encoder/decoder (CODEC)

  - 3.5mm line-in, line-out, and microphone-in jacks
- **Character Display**
  - 16x2 LCD module
- **On-board Clocking Circuitry**
  - Three 50MHz oscillator clock inputs
  - SMA connectors (external clock input/output)
- **SD Card Socket**
  - Provides SPI and 4-bit SD mode for SD Card access
- **Two Gigabit Ethernet Ports**
  - Integrated 10/100/1000 Ethernet
- **High Speed Mezzanine Card (HSMC)**
  - Configurable I/O standards (voltage levels: 3.3/2.5/1.8/1.5V)
- **USB Type A and B**
  - Provides host and device controller compliant with USB 2.0
  - Supports data transfer at full-speed and low-speed
  - PC driver available
- **40-pin Expansion Port**
  - Configurable I/O standards (voltage levels: 3.3/2.5/1.8/1.5V)
- **VGA-out Connector**
  - VGA DAC (high speed triple DACs)
- **DB9 Serial Connector**
  - RS232 port with flow control
- **PS/2 Connector**
  - PS/2 connector for connecting a PS2 mouse or keyboard
- **TV-in Connector**
  - TV decoder (NTSC/PAL/SECAM)
- **Remote Control**
  - Infrared receiver module
- **Power**
  - 12V DC input
  - Switching and step-down regulators LM3150MH

■ Capacitive LCD Touch Screen

- Equipped with an 7-inch Amorphous-TFT-LCD (Thin Film Transistor Liquid Crystal Display) module
- Module composed of LED backlight
- Support 24-bit parallel RGB interface
- Converting the X/Y coordination of touch point to its corresponding digital data via the Touch controller.

Table 1-1 shows the general physical specifications of the touch screen (Note*).

Table 1-1 General physical specifications of the LCD

Item	*Specification*	*Unit*
LCD size	7-inch (Diagonal)	-
Resolution	800 x3(RGB) x 480	dot
Dot pitch	0.1926(H) x0.1790 (V)	mm
Active area	154.08 (H) x 85.92 (V)	mm
Module size	164.9(H) x 100.0(V) x 5.7(D)	mm
Surface treatment	Glare	-
Color arrangement	RGB-stripe	-
Interface	Digital	-

■ 5-Megapixel Digital Image Sensor

- Superior low-light performance
- High frame rate
- Low dark current
- Global reset release, which starts the exposure of all rows simultaneously
- Bulb exposure mode, for arbitrary exposure times
- Snapshot-mode to take frames on demand
- Horizontal and vertical mirror image
- Column and row skip modes to reduce image size without reducing field-of-view
- Column and row binning modes to improve image quality when resizing
- Simple two-wire serial interface
- Programmable controls: gain, frame rate, frame size, exposure

Table 1-2 shows the key parameters of the CMOS sensor (Note*).

Table 1-2 Key performance parameters of the CMOS sensor

Parameter		*Value*
Active Pixels		2592Hx1944V
Pixel size		2.2umx2.2um
Color filter array		RGB Bayer pattern
Shutter type		Global reset release(GRR)
Maximum data rate/master clock		96Mp/s at 96MHz
Frame rate	Full resolution	Programmable up to 15 fps
	VGA mode	Programmable up to 70 fps
ADC resolution		12-bit
Responsivity		1.4V/lux-sec(550nm)

Pixel dynamic range		**70.1dB**
SNRMAX		**38.1dB**
Supply Voltage	**Power**	**3.3V**
	I/O	**1.7V～3.1V**

■ Digital Accelerometer

- Up to 13-bit resolution at +/- 16g
- SPI (3- and 4-wire) digital interface
- Flexible interrupts modes

■ Ambient Light Sensor

- Approximates human-eye response
- Precise luminance measurement under diverse lighting conditions
- Programmable interrupt function with user-defined upper and lower threshold settings
- 16-bit digital output with I^2C fast-mode at 400 kHz
- Programmable analog gain and integration time
- 50/60-Hz lighting ripple rejection

Note: for more detailed information of the LCD touch panel and CMOS sensor module, please refer to their datasheets respectively.

1.1 About the Kit

The kit contains all users needed to run the demonstrations and develop custom designs, as shown in **Figure 1-2**.

The system CD contains technical documents of the VEEK-MT which includes component datasheets, demonstrations, schematic, and user manual.

Figure 1-2 VEEK-MT kit package contents

1.2 Setup License for Terasic Multi-touch IP

To utilize the multi-touch panel in a Quartus II project, a Terasic Multi-Touch IP is required. After a license file for Quartus II is installed, there is one more license file needed to implement Terasic's Multi-touch IP. Error messages will be displayed if the license file is not added before compiling projects using Terasic Multi-touch IP. The license file is located at:

VEEK-MT System CD\License\license_multi_touch.dat

There are two ways to install the License. The first one is to add the license file (license_multi_touch.dat) to the licensed file listed in Quartus II, as shown in **Figure 1-3**.

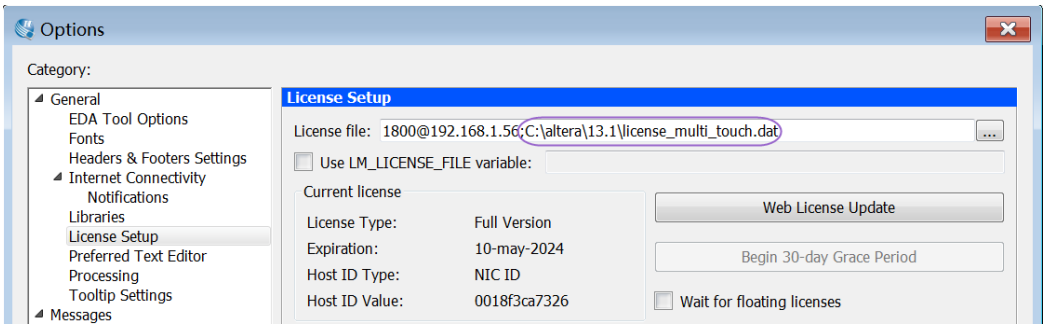

Figure 1-3 License Setup

The second way is to add license content to the existing license file. The procedures are listed below:

Use Notepad or other text editing software to open the file license_multi_touch.dat.

1. The license contains the FEATURE lines required to license the IP Cores as shown in **Figure 1-4**.

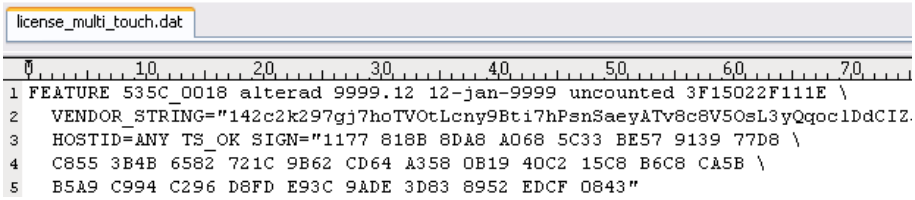

Figure 1-4 Content of license_multi_touch.dat

2. Open your Quartus II license.dat file in a text editor.
3. Copy everything under license_multi_touch.dat and paste it at the end of your Quartus II license file. (Note: Do not delete any FEATURE lines from the Quartus II license file. Doing so will result in an unusable license file.) .
4. Save the Quartus II license file.

1.3 Getting Help

Here is information of how to get help if you encounter any problem:

- **Terasic Technologies**
- **Tel: +886-3-5750-880**
- **Email: support@terasic.com**

Chapter 2

Architecture

This chapter describes the architecture of the Video and Embedded Evaluation Kit – Multi-touch (VEEK-MT) including block diagram and components.

2.1 Layout and Components

The picture of the VEEK-MT is shown in **Figure 2-1** and **Figure 2-2**. It depicts the layout of the board and indicates the locations of the connectors and key components.

Figure 2-1 VEEK-MT PCB and Component Diagram (top view)

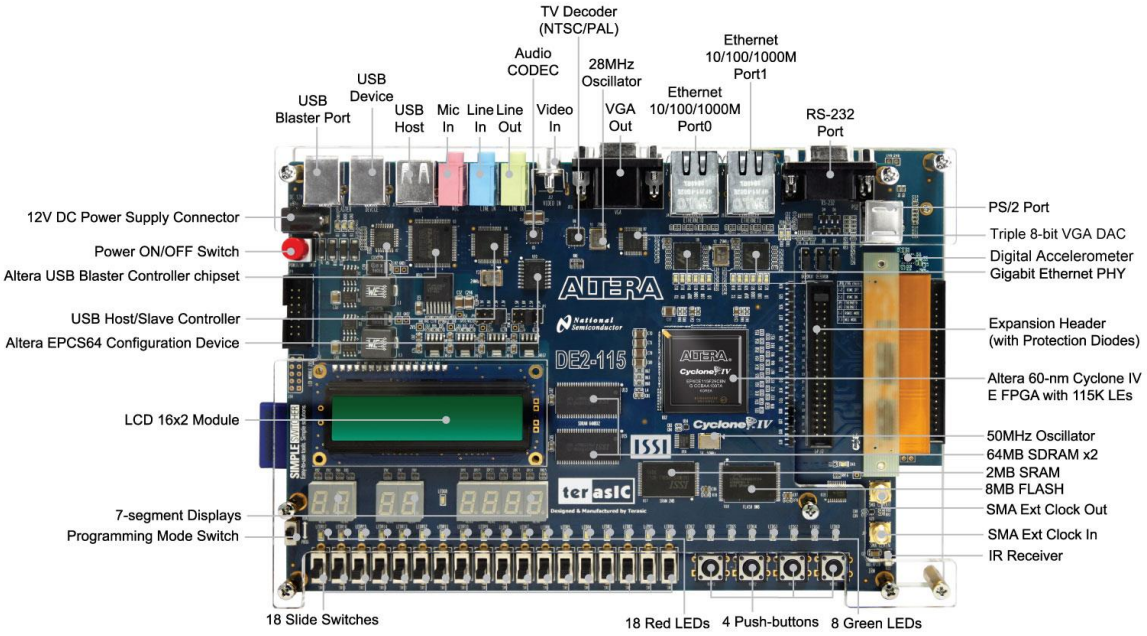

Figure 2-2 VEEK-MT PCB and Component Diagram (bottom view)

2.2 Block Diagram of the VEEK-MT

Figure 2-3 gives the block diagram of the VEEK-MT board. To provide maximum flexibility for the user, all connections are made through the Cyclone IV E FPGA device. Thus, the user can configure the FPGA to implement any system design.

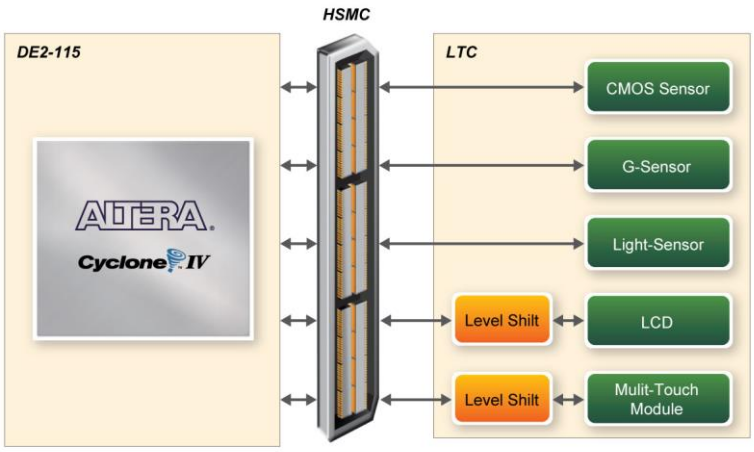

Figure 2-3 Block Diagram of VEEK-MT

Chapter 3

Using VEEK-MT

This section describes the detailed information of the components, connectors, and pin assignments of the VEEK-MT.

3.1 Configuring the Cyclone IV E FPGA

The Video and Embedded Evaluation Kit (VEEK-MT) contains a serial configuration device that stores configuration data for the Cyclone IV E FPGA. This configuration data is automatically loaded from the configuration device into the FPGA every time while power is applied to the board. Using the Quartus II software, it is possible to reconfigure the FPGA at any time, and it is also possible to change the non-volatile data that is stored in the serial configuration device. Both types of programming methods are described below.

1. JTAG programming: In this method of programming, named after the IEEE standards Joint Test Action Group, the configuration bit stream is downloaded directly into the Cyclone IV E FPGA. The FPGA will retain this configuration as long as power is applied to the board; the configuration information will be lost when the power is turned off.
2. AS programming: In this method, called Active Serial programming, the configuration bit stream is downloaded into the Altera EPCS64 serial configuration device. It provides non-volatile storage of the bit stream, so that the information is retained even when the power supply to the VEEK-MT is turned off. When the board's power is turned on, the configuration data in the EPCS64 device is automatically loaded into the Cyclone IV E FPGA.

■ JTAG Chain on VEEK-MT

To use the JTAG interface for configuring FPGA device, the JTAG chain on the VEEK-MT must form a closed loop that allows Quartus II programmer to detect the FPGA device. **Figure 3-1** illustrates the JTAG chain on the VEEK-MT. Shorting pin1 and pin2 on JP3 can disable the JTAG signals on the HSMC connector that will form a close JTAG loopback on DE2-115 (See **Figure 3-2**). Thus, only the on-board FPGA device (Cyclone IV E) will be detected by Quartus II programmer. By default, a jumper is placed on pin1 and pin2 of JP3. To prevent any changes to the bus controller (Max II EPM240) described in later sections, users should not adjust the jumper on JP3.

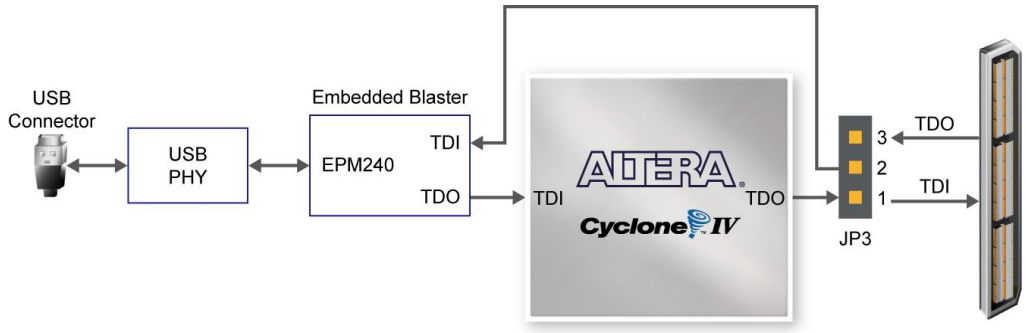

Figure 3-1 JTAG Chain

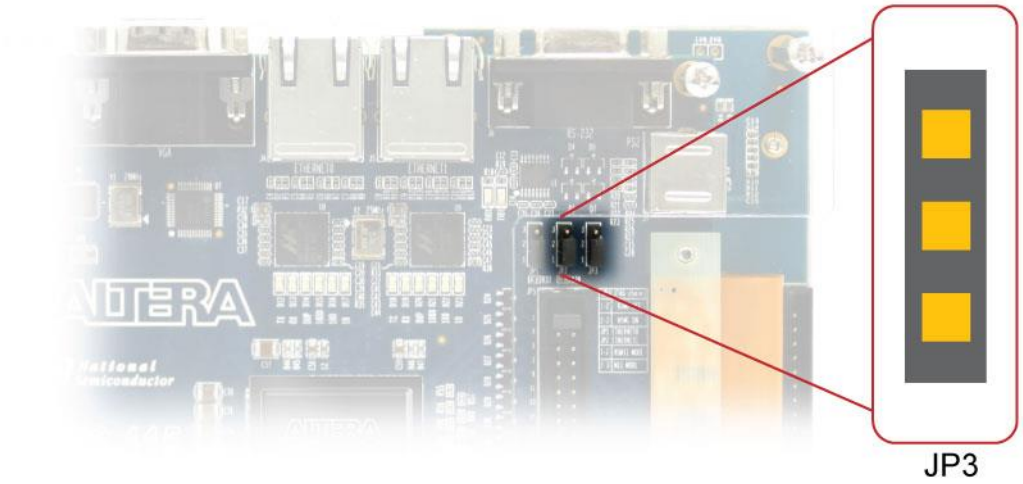

Figure 3-2 JTAG Chain Configuration Header

■ Configuring the FPGA in JTAG Mode

Figure 3-3 illustrates the JTAG configuration setup. To download a configuration bit stream into the Cyclone IV E FPGA, perform the following steps:

- Ensure that power is applied to the VEEK-MT
- Configure the JTAG programming circuit by setting the RUN/PROG slide switch (SW19) to the RUN position (See Figure 3-4**)**
- Connect the supplied USB cable to the USB-Blaster port on the VEEK-MT
- The FPGA can now be programmed by using the Quartus II Programmer module to select a configuration bit stream file with the .sof filename extension

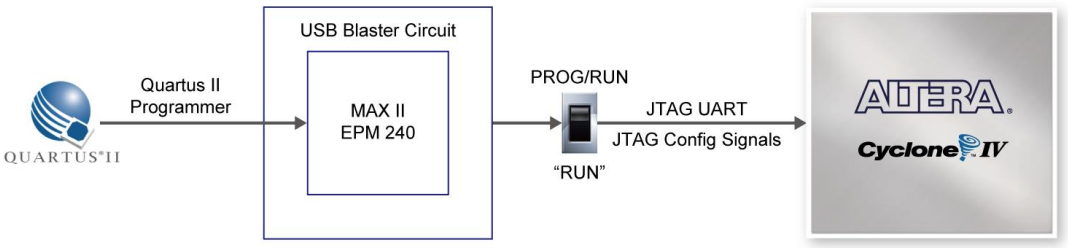

Figure 3-3 JTAG Chain Configuration Scheme

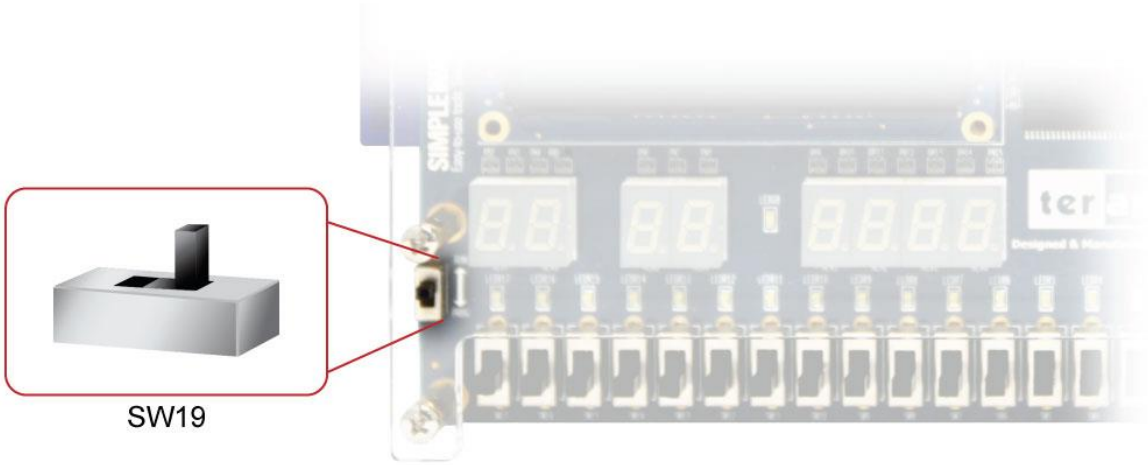

Figure 3-4 The RUN/PROG switch (SW19) is set to JTAG mode

■ Configuring the EPCS64 in AS Mode

Figure 3-5 illustrates the AS configuration set up. To download a configuration bit stream into the EPCS64 serial configuration device, perform the following steps:

- Ensure that power is applied to the VEEK-MT
- Connect the supplied USB cable to the USB-Blaster port on the VEEK-MT
- Configure the JTAG programming circuit by setting the RUN/PROG slide switch (SW19) to the PROG position
- The EPCS64 chip can now be programmed by using the Quartus II Programmer module to select a configuration bit stream file with the .pof filename extension
- Once the programming operation is finished, set the RUN/PROG slide switch back to the RUN position and then reset the board by turning the power switch off and back on; this action causes the new configuration data in the EPCS64 device to be loaded into the FPGA chip

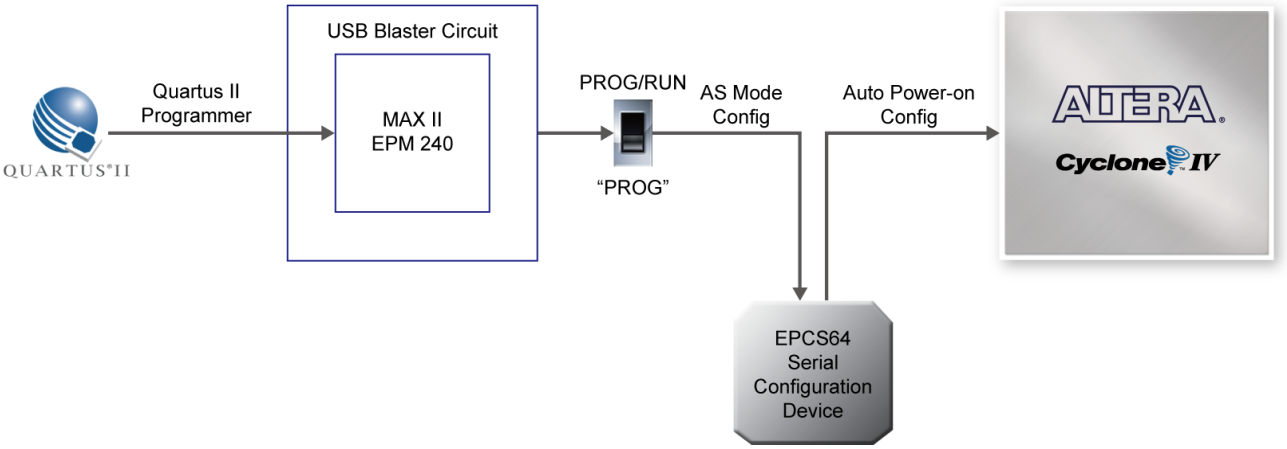

Figure 3-5 The AS Configuration Scheme

3.2 Bus Controller

The VEEK-MT comes with a bus controller using the Max II EPM240 that allows user to access the touch screen module through the HSMC connector. This section describes its structure in block diagram-form and its capabilities.

■ Bus Controller Introduction

The bus controller provides level shifting functionality from 2.5V (HSMC) to 3.3V domains.

■ Block Diagram of the Bus Controller

Figure 3-6 gives the block diagram of the connection setup from the HSMC connector to the bus controller on the Max II EPM240 to the touch screen module. To provide maximum flexibility for the user, all connections are established through the HSMC connector. Thus, the user can configure the Cyclone IV E FPGA on the VEEK-MT to implement any system design.

Figure 3-6 Block Diagram of the Bus Controller

3.3 Using the 7” LCD Capacitive Touch Screen

The VEEK-MT features a 7-inch capacitive amorphous TFT-LCD panel. The LCD touch screen offers resolution of (800x480) to provide users the best display quality for developing applications. The LCD panel supports 24-bit parallel RGB data interface.

The VEEK-MT is also equipped with a Touch controller touch controller, which can read the coordinates of the touch points through the serial port interface of Touch controller

To display images on the LCD panel correctly, the RGB color data along with the data enable and clock signals must act according to the timing specification of the LCD touch panel as shown in **Table 3-1**

Table 3-2 gives the pin assignment information of the LCD touch panel.

Table 3-1 LCD timing specifications

ITEM		SYMBOL	MIN.	TYP.	MAX.	UNIT	NOTE
	Dot Clock	1/tCLK		33		MHZ	
DCLK	DCLK pulse duty	Tcwh	40	50	60	%	
	Setup time	Tesu	8			ns	
	Hold time	Tehd	8			ns	
	Horizontal period	tH		1056		tCLK	
	Horizontal Valid	tHA		800		tCLK	
	Horizontal Blank	tHB		256		tCLK	
	Vertical Period	tV		525		tH	
	Vertical Valid	tVA		480		tH	
DE	Vertical Blank	tVB		45		tH	
	HSYNC setup time	Thst	8			ns	
	HSYNC hold time	Thhd	8			ns	
	VSYNC Setup Time	Tvst	8			ns	
	VSYNC Hold Time	Tvhd	8			ns	
	Horizontal Period	th		1056		tCLK	
	Horizontal Pulse Width	thpw		30		tCLK	thb+thpw=46DCLK
	Horizontal Back Porch	thb		16		tCLK	is fixed
	Horizontal Front Porch	thfp		210		tCLK	
	Horizontal Valid	thd		800		tCLK	
	Vertical Period	tv		525		th	
SYNC	Vertical Pulse Width	tvpw		13		th	tvpw + tvb =

	Vertical Back Porch	tvb		10		th	23th is fixed
	Vertical Front Porch	tvfp		22		th	
	Vertical Valid	tvd		480		th	
DATA	Setup time	Tdsu	8			ns	
	Hold time	Tdsu	8			ns	

Table 3-2 Pin assignment of the LCD touch panel

Signal Name	FPGA Pin No.	Description	I/O Standard
LCD_B0	P28	LCD blue data bus bit 0	2.5V
LCD_B1	P27	LCD blue data bus bit 1	2.5V
LCD_B2	J24	LCD blue data bus bit 2	2.5V
LCD_B3	J23	LCD blue data bus bit 3	2.5V
LCD_B4	T26	LCD blue data bus bit 4	2.5V
LCD_B5	T25	LCD blue data bus bit 5	2.5V
LCD_B6	R26	LCD blue data bus bit 6	2.5V
LCD_B7	R25	LCD blue data bus bit 7	2.5V
LCD_DCLK	V24	LCD Clock	2.5V
LCD_DE	H23	Data Enable signal	2.5V
LCD_DIM	P21	LCD backlight enable	2.5V
LCD_DITH	L23	Dithering setting	2.5V
LCD_G0	P26	LCD green data bus bit 0	2.5V
LCD_G1	P25	LCD green data bus bit 1	2.5V
LCD_G2	N26	LCD green data bus bit 2	2.5V
LCD_G3	N25	LCD green data bus bit 3	2.5V
LCD_G4	L22	LCD green data bus bit 4	2.5V
LCD_G5	L21	LCD green data bus bit 5	2.5V
LCD_G6	U26	LCD green data bus bit 6	2.5V
LCD_G7	U25	LCD green data bus bit 7	2.5V
LCD_HSD	U22	Horizontal sync input.	2.5V
LCD_MODE	L24	DE/SYNC mode select	2.5V
LCD_POWER_CTL	M25	LCD power control	2.5V
LCD_R0	V28	LCD red data bus bit 0	2.5V
LCD_R1	V27	LCD red data bus bit 1	2.5V
LCD_R2	U28	LCD red data bus bit 2	2.5V
LCD_R3	U27	LCD red data bus bit 3	2.5V
LCD_R4	R28	LCD red data bus bit 4	2.5V
LCD_R5	R27	LCD red data bus bit 5	2.5V

LCD_R6	V26	LCD red data bus bit 6	2.5V
LCD_R7	V25	LCD red data bus bit 7	2.5V
LCD_RSTB	K22	Global reset pin	2.5V
LCD_SHLR	H24	Left or Right Display Control	2.5V
LCD_UPDN	K21	Up / Down Display Control	2.5V
LCD_VSD	V22	Vertical sync input.	2.5V
TOUCH _I2C_SCL	T22	touch I2C clock	2.5V
TOUCH _I2C_SDA	T21	touch I2C data	2.5V
TOUCH _INT_n	R23	touch interrupt	2.5V

3.4 Using 5-megapixel Digital Image Sensor

The VEEK-MT is equipped with a 5-megapixel digital image sensor that provides an active imaging array of 2,592H x 1,944V. It features low-noise CMOS imaging technology that achieves CCD image quality. In addition, it incorporates sophisticated camera functions on-chip such as windowing, column and row skip mode, and snapshot mode.

The sensor can be operated in its default mode or programmed by the user through a simple two-wire serial interface for frame size, exposure, gain settings, and other parameters. Table 3-3 contains the pin names and descriptions of the image sensor module.

Table 3-3 Pin assignment of the CMOS sensor

Signal Name	FPGA Pin No.	Description	I/O Standard
CAMERA_PIXCLK	J27	Pixel clock	2.5V
CAMERA_D0	F24	Pixel data bit 0	2.5V
CAMERA_D1	F25	Pixel data bit 1	2.5V
CAMERA_D2	D26	Pixel data bit 2	2.5V
CAMERA_D3	C27	Pixel data bit 3	2.5V
CAMERA_D4	F26	Pixel data bit 4	2.5V
CAMERA_D5	E26	Pixel data bit 5	2.5V
CAMERA_D6	G25	Pixel data bit 6	2.5V
CAMERA_D7	G26	Pixel data bit 7	2.5V
CAMERA_D8	H25	Pixel data bit 8	2.5V
CAMERA_D9	H26	Pixel data bit 9	2.5V
CAMERA_D10	K25	Pixel data bit 10	2.5V
CAMERA_D11	K26	Pixel data bit 11	2.5V
CAMERA_STROBE	E27	Snapshot strobe	2.5V
CAMERA_LVAL	D28	Line valid	2.5V

CAMERA_FVAL	D27	Frame valid	2.5V
CAMERA_RESET_n	F27	Image sensor reset	2.5V
CAMERA_SCLK	AE26	Serial clock	2.5V
CAMERA_TRIGGER	E28	Snapshot trigger	2.5V
CAMERA_SDATA	AE27	Serial data	2.5V
CAMERA_XCLKIN	G23	External input clock	2.5V

3.5 Using the Digital Accelerometer

The VEEK-MT is equipped with a digital accelerometer sensor module. The ADXL345 is a small, thin, ultralow power assumption 3-axis accelerometer with high resolution measurement. Digitalized output is formatted as 16-bit twos complement and could be accessed either using SPI interface or I2C interface. This chip uses the 3.3V CMOS signaling standard. Main applications include medical instrumentation, industrial instrumentation, personal electronic aid and hard disk drive protection etc. Some of the key features of this device are listed below. For more detailed information of better using this chip, please refer to its datasheet which is available on manufacturer's website or under the /datasheet folder of the system CD.

Table 3-4 contains the pin names and descriptions of the G sensor module.

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
GSENSOR_INT1	G27	Interrupt 1 output	2.5V
GSENSOR_INT2	G28	Interrupt 2 output	2.5V
GSENSOR_CS_n	F28	Chip Select	2.5V
GSENSOR_ALT_ADDR	K27	I2C Address Select	2.5V
GSENSOR_SDA_SDI_SDIO	K28	Serial Data	2.5V
GSENSOR_SCL_SCLK	M27	Serial Communications Clock	2.5V

3.6 Using the Ambient Light Sensor

The APDS-9300 is a low-voltage digital ambient light sensor that converts light intensity to digital signal output capable of direct I2C communication. Each device consists of one broadband photodiode (visible plus infrared) and one infrared photodiode. Two integrating ADCs convert the photodiode currents to a digital output that represents the irradiance measured on each channel. This digital output can be input to a microprocessor where illuminance (ambient light level) in lux is derived using an empirical formula to approximate the human-eye response. For more detailed information of better using this chip, please refer to its datasheet which is available on manufacturer's website or under the /datasheet folder of the system CD.

Table 3-5 contains the pin names and descriptions of the ambient light sensor module.

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
LSENSOR_ADDR_SEL	J25	Chip select	2.5V
LSENSOR_INT	L28	Interrupt output	2.5V
LSENSOR_SCL	J26	Serial Communications Clock	2.5V
LSENSOR_SDA	L27	Serial Data	2.5V

3.7 Using Terasic Multi-touch IP

Terasic Multi-touch IP is provided for developers to retrieve user inputs, including multi-touch gestures and single-touch. The file name of this IP is **i2c_touch_config** and it is encrypted. To compile projects with the IP, users need to install the IP license first. For license installation, please refer to section **1.2 Setup License for Terasic Multi-touch IP** in this document. The license file is located at:

VEEK-MT System CD\License\license_multi_touch.dat

The IP decodes I2C information and outputs coordinate and gesture information. The IP interface is shown below:

```
module i2c_touch_config  (
            // Host Side
            iCLK,
            iRSTN,
            iTRIG,
            oREADY,
            oREG_X1,
            oREG_Y1,
            oREG_X2,
            oREG_Y2,
            oREG_TOUCH_COUNT,
            oREG_GESTURE,
            // I2C Side
            I2C_SCLK,
            I2C_SDAT
);
```

The signal purpose of the IP is described in **Table 3-6.** The IP requires a 50 MHz signal as a reference clock to the **iCLK** pin and system reset signal to **iRSTN**. **iTRIG**, **I2C_SCLK**, and **IC2_SDAT** pins should be connected of the TOUCH_INT_n, TOUCH_I2C_SCL, and TOUCH_I2C_SDA signals in the 2x20 GPIO header respectively. When **oREADY** rises, it means there is touch activity, and associated information is given in the oREG_X1, oREG_Y1, oREG_X2, oREG_Y2, oREG_TOUCH_COUNT, and oREG_GESTURE pins.

For the control application, when touch activity occurs, it should check whether the value of oREG_GESTURE matched a pre-defined gesture ID defined in **Table 3-7.** If it is not a gesture, it means a single-touch has occurred and the relative X/Y coordinates can be derived from oREG_X1 and oREG_Y1.

Table 3-6 Interface Definitions of Terasic Multi-touch IP

Pin Name	*Direction*	*Description*
iCLK	Input	Connect to 50MHz Clock
iRSTN	Input	Connect to system reset signal
iTRIG	Input	Connect to Interrupt Pin of Touch IC
oREADY	Output	Rising Trigger when following six output data is valid
oREG_X1	Output	10-bits X coordinate of first touch point
oREG_Y1	Output	9-bits Y coordinate of first touch point
oREG_X2	Output	10-bits X coordinate of second touch point
oREG_Y2	Output	9-bits Y coordinate of second touch point
oREG_TOUCH_COUNT	Output	2-bits touch count. Valid value is 0, 1, or 2.
oREG_GESTURE	Output	8-bits gesture ID (See Table 3-7)
I2C_SCLK	Output	Connect to I2C Clock Pin of Touch IC
I2C_SDAT	Inout	Connect to I2C Data Pin of Touch IC

The supported gestures and IDs are shown in **Table 3-7.**

Table 3-7 Gestures

Gesture	*ID (hex)*
One Point Gesture	
North	0x10
North-East	0x12
East	0x14
South-East	0x16
South	0x18
South-West	0x1A
West	0x1C
North-West	0x1E
Rotate Clockwise	0x28
Rotate Anti-clockwise	0x29
Click	0x20
Double Click	0x22
Two Point Gesture	
North	0x30
North-East	0x32
East	0x34
South-East	0x36
South	0x38
South-West	0x3A
West	0x3C
North-West	0x3E
Click	0x40

Zoom In	**0x48**
Zoom Out	**0x49**

Note: The Terasic IP Multi-touch IP can also be found under the \IP folder in the system CD as well as the \IP folder in the reference designs.

Chapter 4

VEEK-MT Demonstrations

This chapter gives detailed description of the provided bundles of exclusive demonstrations implemented on VEEK-MT. These demonstrations are particularly designed (or ported) for VEEK-MT, with the goal of showing the potential capabilities of the kit and showcase the unique benefits of FPGA-based SOPC systems such as reducing BOM costs by integrating powerful graphics and video processing circuits within the FPGA.

4.1 System Requirements

To run and recompile the demonstrations, you should:

- Install Altera Quartus II 13.1 and NIOS II EDS 13.1 or later edition on the host computer. Users need to download Quartus II 13.1 from Altera's website.
- Install the USB-Blaster driver software. You can find instructions in the tutorial "Getting Started with Altera's DE2-115 Board" (tut_initialDE2-115.pdf) which is available on the DE2-115 system CD
- Copy the entire demonstrations folder from the VEEK-MT system CD to your host computer

4.2 Factory Configuration

The VEEK-MT development kit comes preconfigured with a default utility that boots up on power on and allows users to quickly select, load, and run different Ready-to-Run demonstrations stored on an SD Card using the VEEK-MT touch panel. **Figure 4-1** gives a snapshot of the default application selector interface (Note*). Every demonstration consists of an FPGA hardware image and an application software image. When you select a demonstration the application selector copies the hardware image to EPCS device and software image to flash memory and reconfigures the FPGA with your selection. For more comprehensive information of the application selector factory configuration, please refer to chapter 5.

Figure 4-1 Application Selector Interface

Note: Please insert the supplied SD card from this demonstration.

4.3 Painter Demonstration

This chapter shows how to control LCD and touch controller to establish a paint demo based on QSYS and Altera VIP Suite. The demonstration shows how multi-touch gestures and single-touch coordinates operate.

Figure 4-2 shows the hardware system block diagram of this demonstration. For LCD display processing, the reference design is developed based on the Altera Video and Image Processing Suite (VIP). The Frame Reader VIP is used for reading display content from the associated video memory, and VIP Video Out is used to display the display content. The display content is filled by NIOS II processor according to users' input.

For multi-touch processing, a Terasic Memory-Mapped IP is used to retrieve the user input, including multi-touch gesture and single-touch resolution. Note, the IP is encrypted, so the license should be installed before compiling the Quartus II project. For IP--usage details please refer to the section **3.7 Using Terasic Multi-touch IP** in this document.

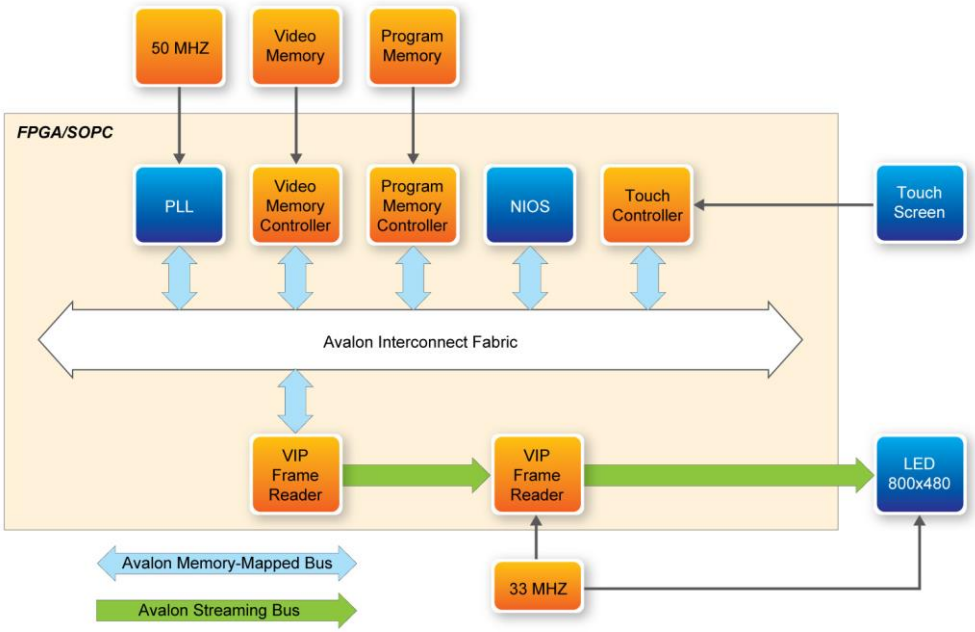

Figure 4-2 Block diagram of the Painter demonstration

■ Demonstration Source Code

- Project directory: Painter
- Bit stream used: Painter.sof
- Nios II Workspace: Painter \Software

■ Demonstration Batch File

Demo Batch File Folder: Painter \demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: Painter.sof
- Nios II Program: Painter.elf

■ Demonstration Setup

- Make sure Quartus II and Nios II are installed on your PC
- Power on the DE2-115 board
- Connect USB-Blaster to the DE2-115 board and install USB-Blaster driver if necessary
- Execute the demo batch file "test.bat" under the batch file folder, Painter \demo_batch
- After Nios II program is downloaded and executed successfully, you will see a painter GUI in the LCD. **Figure 4-3** shows the GUI of the Painter Demo.

- The GUI is classified into three areas: Palette, Canvas, and Gesture. Users can select pen color from the color palette and start painting in the Canvas area. If gesture is detected, the associated gesture symbol is shown in the gesture area. To clear canvas content, click the "Clear" button.
- **Figure 4-4** shows the photo when users paint in the canvas area. **Figure 4-5** shows the phone when counter-clockwise rotation gesture is detected. **Figure 4-6** shows the photo when zoom-in gesture is detected.

Figure 4-3 GUI of Painter Demo

Figure 4-4 Single Touch Painting

Figure 4-5 Counter-clockwise Rotation Gesture

Figure 4-6 Zoom-in Gesture

Note: execute the test.bat under Painter\demo_batch will automatically download the .sof and .elf file.

4.4 Picture Viewer

This demonstration shows a simple picture viewer implementation using Nios II based QSYS system. It reads JPEG images stored on SD Card and displays them on the LCD. The Nios II CPU decodes the images and fills the raw result data into frame buffers in SDRAM. The VEEK-MT will show the image the buffer being displayed points to. When users touch the LCD Touch Panel, it will proceed to display the next buffered image or last buffered image. **Figure 4-7** shows the block diagram of this demonstration.

The Nios II CPU here takes a key roll in the demonstration. It is responsible of decoding the JPEG images and coordinates the works of all the peripherals. The touch panel handling program uses the timer as a regular interrupter and periodically updates the pen state and sampled coordinates.

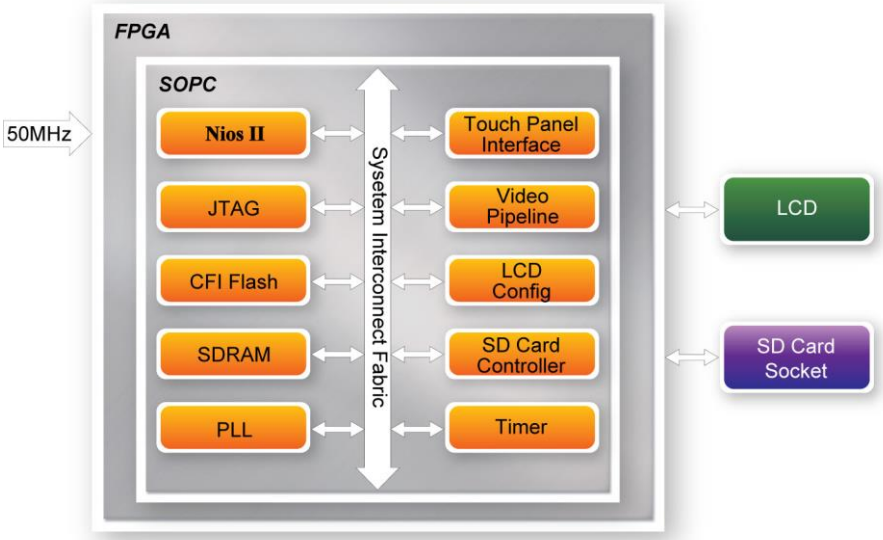

Figure 4-7 Block diagram of the picture viewer demonstration

■ Demonstration Source Code

- Project directory: Picture_Viewer
- Bit stream used: Picture_Viewer.sof
- Nios II Workspace: Picture_Viewer\Software

■ Demonstration Batch File

Demo Batch File Folder: Picture_Viewer\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: Picture_Viewer.sof
- Nios II Program: Picture_Viewer.elf

■ Demonstration Setup

1. Format your SD Card into FAT16 format
2. Place the jpg image files to the \jpg subdirectory of the SD Card. For best display result, the image should have a resolution of 800x480 or the multiple of that
3. Insert the SD card to the SD card slot on the VEEK-MT
4. Load the bit stream into the FPGA on the VEEK-MT
5. Run the Nios II Software under the workspace Picture_Viewer\Software (Note*)
6. After loading the application you will see a slide show of pictures on the SD card.
7. The next image will be displayed after the delay period.
8. You can control the slide show as follows :

- Press Forward **(**  **)** to advance, Reverse () to go back to previous image, Play/Stop ( **)** to play the slide or stop it.
- On the top corner you will see the Delay-period (seconds). You can increase or decrease the delay period by touching the **+ (**  **)** or **– (** **)** buttons.
- The max delay is 120 seconds, the min delay is 1 second, and the default delay is 10 seconds.
- You can hide the control buttons by clicking on the Hide button located at the top left corner of the touch screen. Touch anywhere on the screen to resume and to return to menu.

Figure 4-8 picture viewer demonstration

Note: execute the Picture_Viewer.bat under Picture_Viewer\demo_batch will automatically download the .sof and .elf file.

4.5 Video and Image Processing

The Video and Image Processing (VIP) Example Design demonstrates dynamic scaling and clipping of a standard definition video stream in either National Television System Committee (NTSC) or Phase Alternation Line (PAL) format and picture-in-picture mixing with a background layer. The video stream is output in high resolution (800×480) LCD touch panel.

The example design demonstrates a framework for rapid development of video and image processing systems using the parameterizable MegaCore® functions that are available in the Video and Image Processing Suite. Available functions are listed in **Table 4-1**. This demonstration needs the Quartus II license file includes the VIP suite feature.

Table 4-1 VIP IP cores functions

IP MegaCore Function	*Description*
Frame Reader	Reads video from external memory and outputs it as a stream.
Control Synchronizer	Synchronizes the changes made to the video stream in real time between two functions.
Switch	Allows video streams to be switched in real time.
Color Space Converter	Converts image data between a variety of different color spaces such as RGB to YCrCb.
Chroma Resampler	Changes the sampling rate of the chroma data for image frames, for example from 4:2:2 to 4:4:4 or 4:2:2 to 4:2:0.
2D FIR Filter	Implements a 3 x 3, 5 x 5, or 7 x 7 finite impulse response (FIR) filter on an image data stream to smooth or sharpen images.
Alpha Blending Mixer	Mixes and blends multiple image streams—useful for implementing text overlay and picture-in-picture mixing.
Scaler II	A sophisticated polyphase scaler that allows custom scaling and real-time updates of both the image sizes and the scaling coefficients.
Deinterlacer	Converts interlaced video formats to progressive video format using a motion adaptive deinterlacing algorithm. Also supports 'bob' and "weave" algorithms
Test Pattern Generator	Generates a video stream that contains still color bars for use as a test pattern.
Clipper II	Provides a way to clip video streams and can be configured at compile time or at run time.
Color Plane Sequencer	Changes how color plane samples are transmitted across the Avalon-ST interface. This function can be used to split and join video streams, giving control over the routing of color plane samples.
Frame Buffer	Buffers video frames into external RAM. This core supports double or triple-buffering with a range of options for frame dropping and repeating.
2D Median Filter	Provides a way to apply 3 x 3, 5 x 5, or 7 x 7 pixel median filters to video images.
Gamma Corrector	Allows video streams to be corrected for the physical properties of display

	devices.
Clocked Video Input/Output	**These two cores convert the industry-standard clocked video format (BT-656) to Avalon-ST video and vice versa.**

These functions allow you to fully integrate common video functions with video interfaces, processors, and external memory controllers. The example design uses an Altera Cyclone® IV E EP4CE115F29 featured VEEK-MT.

A video source is input through an analog composite port on VEEK-MT which generates a digital output in ITU BT656 format. A number of common video functions are performed on this input stream in the FPGA. These functions include clipping, chroma resampling, motion adaptive deinterlacing, color space conversion, picture-in-picture mixing, and polyphase scaling.

The input and output video interfaces on the VEEK-MT are configured and initialized by software running on a Nios® II processor. Nios II software demonstrates how to control the clocked video input, clocked video output, and mixer functions at run-time is also provided. The video system is implemented using the QSYS system level design tool. This abstracted design tool provides an easy path to system integration of the video processing data path with a NTSC or PAL video input, VGA output, Nios II processor for configuration and control. The Video and Image Processing Suite MegaCore functions have common open Avalon-ST data interfaces and Avalon Memory-Mapped (Avalon-MM) control interfaces to facilitate connection of a chain of video functions and video system modeling. In addition, video data is transmitted between the Video and Image Processing Suite functions using the Avalon-ST Video protocol, which facilitates building run-time controllable systems and error recovery.

Figure 4-9 shows the Video and Image Processing block diagram.

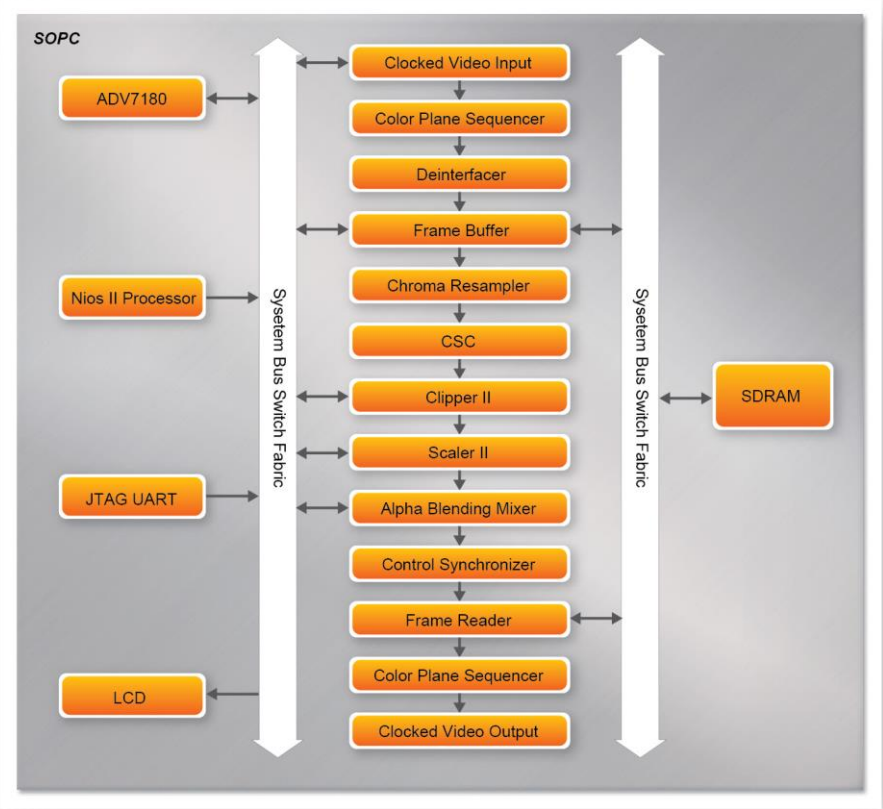

Figure 4-9 VIP Example SOPC Block Diagram (Key Components)

■ Demonstration Source Code

- Project directory: VIP
- Bit stream used: VIP.sof
- Nios II Workspace: VIP\Software

■ Demonstration Batch File

Demo Batch File Folder: VIP\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: VIP.sof
- Nios II Program: VIP.elf

■ Demonstration Setup

- Connect a DVD player's composite video output(yellow plug) to the Video-IN RCA jack(J12) of the VEEK-MT. The DVD player has to be configured to provide NTSC output or PAL output
- Connect the VGA output of the VEEK-MT to a VGA monitor (both LCD and CRT type of monitors should work)
- Load the bit stream into FPGA (note*)
- Run the Nios II and choose VIP\Software as the workspace. Click on the Run button (note *)
- Press and drag the video frame box will result in scaling the playing window to any size, as shown in **Figure 4-10**

Note:

(1).Executing VIP\demo_batch\ VIP.bat will download .sof and .elf files.

(2).You may need additional Altera VIP suite Megacore license features to recompile the project.

Figure 4-11 illustrates the setup for this demonstration.

Figure 4-10 The VIP demonstration running result

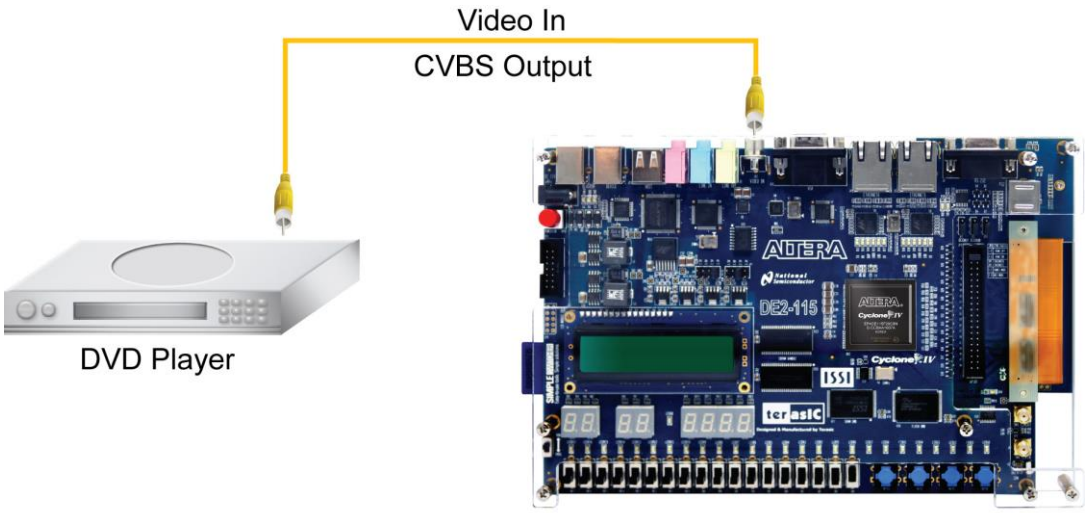

Figure 4-11 Setup for the VIP demonstration

4.6 Camera Application

This demonstration shows a digital camera reference design using the 5-Megapixel CMOS sensor and 8-inch LCD modules on the VEEK-MT. The CMOS sensor module sends the raw image data to FPGA on the DE2-115 board, the FPGA on the board handles image processing part and converts the data to RGB format to display on the LCD module. The I2C Sensor Configuration module is used to configure the CMOS sensor module. **Figure 4-12** shows the block diagram of the demonstration.

As soon as the configuration code is downloaded into the FPGA, the I2C Sensor Configuration block will initial the CMOS sensor via I2C interface. The CMOS sensor is configured as follow:

- Row and Column Size: 800 * 480
- Exposure time: Adjustable
- Pix clock: MCLK*2 = 25*2 = 50MHz
- Readout modes: Binning
- Mirror mode: Line mirrored

According to the settings, we can calculate the CMOS sensor output frame rate is about 44.4 **fps**.

After the configuration, The CMOS sensor starts to capture and output image data streams, the CMOS sensor Capture block extracts the valid pix data streams based on the synchronous signals from the CMOS sensor. The data streams are generated in Bayer Color Pattern format. So it's then converted to RGB data streams by the RAW2RGB block.

After that, the Multi-Port SDRAM Controller acquires and writes the RGB data streams to the SDRAM which performs as a frame buffer. The Multi-Port SDRAM Controller has two write ports

and read ports also with 16-bit data width each. The writing clock is the same as CMOS sensor pix clock, and the reading clock is provided by the LCD Controller, which is 33MHz.

Finally, the LCD controller fetches the RGB data from the buffer and displays it on the LCD panel continuously. Because the resolution and timing of the LCD is compatible with WVGA@800*480, the LCD controller generates the same timing and the frame rate can achieve about 25 fps.

For the objective of a better visual effect, the CMOS sensor is configured to enable the left right mirror mode. User could disable this functionality by modifying the related register value being written to CMOS controller chip.

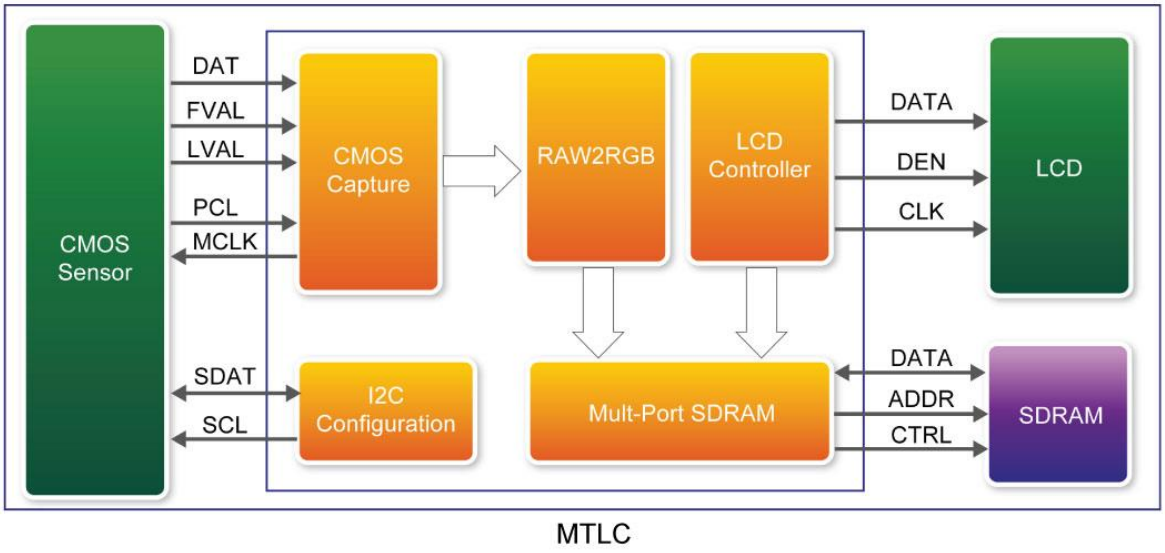

Figure 4-12 Block diagram of the digital camera design

■ Demonstration Source Code

- Project directory: Camera
- Bit stream used: Camera.sof

■ Demonstration Batch File

Demo Batch File Folder: Camera\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat
- FPGA Configure File: Camera.sof

■ Demonstration Setup

- Load the bit stream into FPGA by executing the batch file 'test.bat' under Camera\demo_batch\ folder
- The system enters the FREE RUN mode automatically. Press KEY[0] on the DE2-115 board to reset the circuit
- Press KEY[2] to take a shot of the photo; you can press KEY[3] again to switch back to FREE RUN mode and you should be able to see whatever the camera captures on the LCD display
- User can use the SW[0] and KEY[1] to set the exposure time for brightness adjustment of the image captured. When SW[0] is set to Off, the brightness of image will be increased as KEY[1] is pressed longer. If SW[0] is set to On, the brightness of image will be decreased as KEY[1] is pressed shorter
- User can use SW[17] to mirror image of the line. However, remember to press KEY[0] after toggle SW[17]

-  *Note: execute the test.bat under Camera\demo_batch will automatically download the .sof file.*

Table 4-2 summarizes the functional keys of the digital camera. **Figure 4-13** gives a run-time photograph of the demonstration.

Table 4-2 The functional keys of the digital camera demonstration

Component	*Function Description*
KEY[0]	Reset circuit
KEY[1]	Set the new exposure time (use with SW[0])
KEY[2]	Trigger the Image Capture (take a shot)
KEY[3]	Switch to Free Run mode
SW[0]	Off: Extend the exposure time
	On: Shorten the exposure time
HEX[7:0]	Frame counter (Display ONLY)

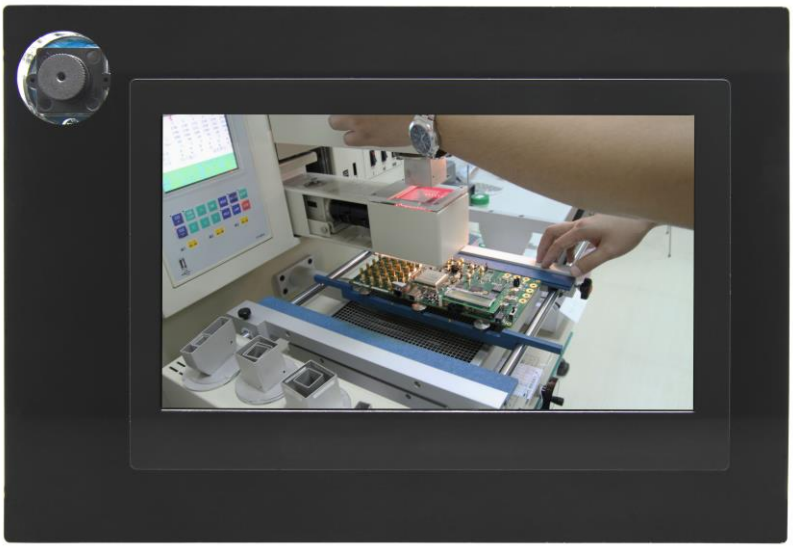

Figure 4-13 Screen shot of the VEEK-MT camera demonstration

4.7 Video and Image Processing for Camera

The Video and Image Processing (VIP) for Camera Example Design demonstrates dynamic scaling and clipping of a standard definition video stream in RGB format and picture-in-picture mixing with a background layer. The video stream is output in high resolution (800×480) on LCD touch panel.

The example design demonstrates a framework for rapid development of video and image processing systems using the parameterizable MegaCore® functions that are available in the Video and Image Processing Suite. Available functions are listed in Table 4 2. This demonstration needs the Quartus II license file includes the VIP suite feature.

These functions allow you to fully integrate common video functions with video interfaces, processors, and external memory controllers. The example design uses an Altera Cyclone® IV E EP4CE115F29 featured on the VEEK-MT.

A video source is input through the CMOS sensor on VEEK-MT which generates a digital output in RGB format. A number of common video functions are performed on this input stream in the FPGA. These functions include clipping, chroma resampling, motion adaptive deinterlacing, color space conversion, picture-in-picture mixing, and polyphase scaling.

The input and output video interfaces on the VEEK-MT are configured and initialized by software running on a Nios® II processor. Nios II software demonstrates how to control the clocked video input, clocked video output, and mixer functions at run-time is also provided. The video system is implemented using the QSYS system level design tool. This abstracted design tool provides an easy

path to system integration of the video processing data path with a NTSC or PAL video input, VGA output, Nios II processor for configuration and control. The Video and Image Processing Suite MegaCore functions have common open Avalon-ST data interfaces and Avalon Memory-Mapped (Avalon-MM) control interfaces to facilitate connection of a chain of video functions and video system modeling. In addition, video data is transmitted between the Video and Image Processing Suite functions using the Avalon-ST Video protocol, which facilitates building run-time controllable systems and error recovery.

For the objective of a better visual effect, the CMOS sensor is configured to enable the left right mirror mode. User could disable this functionality by modifying the related register value being written to CMOS controller chip.

Figure 4-14 shows the Video and Image Processing block diagram.

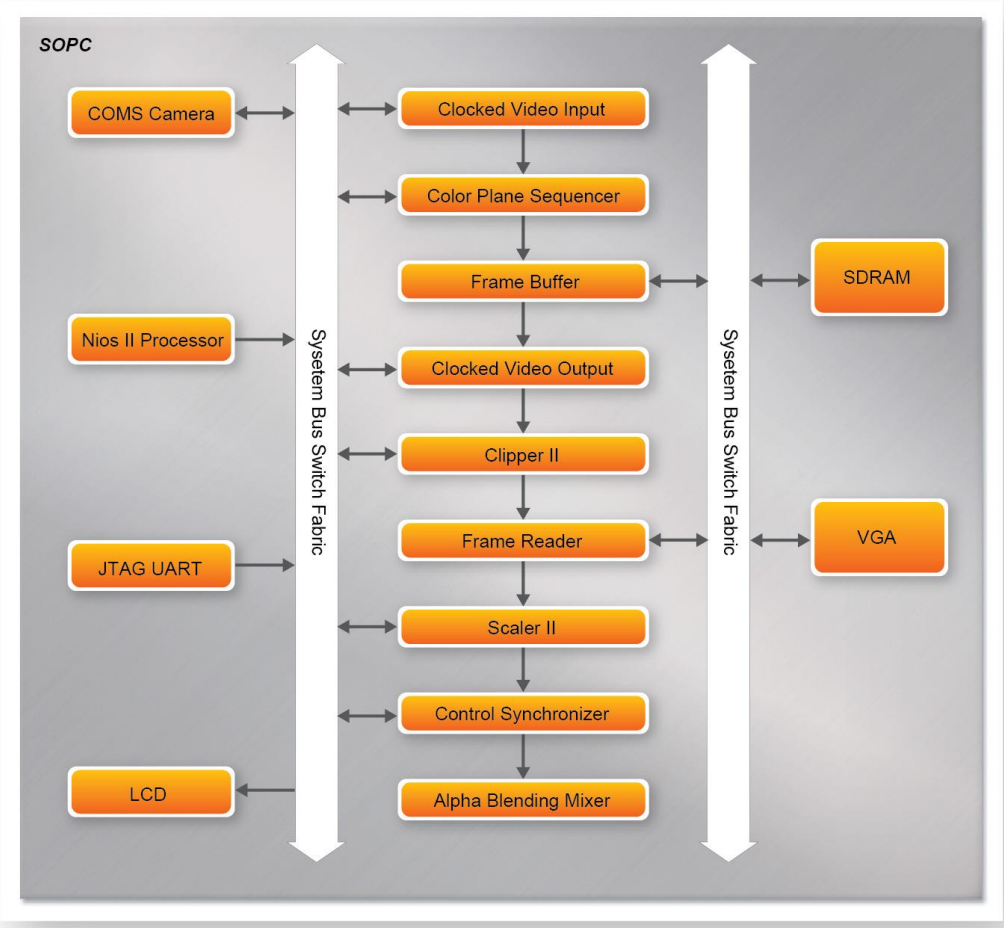

Figure 4-14 VIP Camera Example SOPC Block Diagram (Key Components)

■ Demonstration Source Code

- Project directory: VIP_Camera
- Bit stream used: VIP_Camera.sof
- Nios II Workspace: VIP_Camera \Software

■ Demonstration Batch File

Demo Batch File Folder: VIP_Camera\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: VIP_Camera.sof
- Nios II Program: VIP_Camera.elf

■ Demonstration Setup

- Connect the VGA output of the VEEK-MT to a VGA monitor (both LCD and CRT type of monitors should work)
- Load the bit stream into FPGA (note*)
- Run the Nios II and choose VIP_Camera\Software as the workspace. Click on the Run button (note *)
- The system enters the FREE RUN mode automatically. Press KEY[0] on the DE2-115 board to reset the circuit
- Press KEY[2] to stop run; you can press KEY[3] again to switch back to FREE RUN mode and you should be able to see whatever the camera captures on the VGA display
- User can use SW[17] to mirror image of the line. However, remember to press KEY[0] after toggle SW[17]
- Press and drag the video frame box will result in scaling the playing window to any size, as shown in **Figure 4-10**

Note:

(1).Execute VIP_Camera\demo_batch\VIP_CameraA.bat will download .sof and .elf files.

(2).You may need additional Altera VIP suite Megacore license features to recompile the project.

Figure 4-15 illustrates the setup for this demonstration.

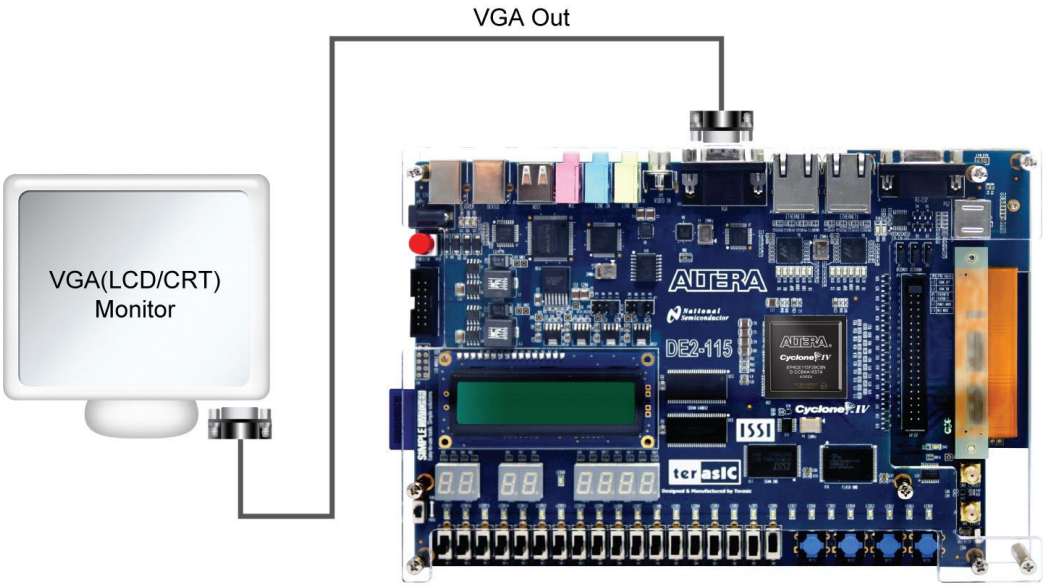

Figure 4-15 Setup for the VIP_Camera demonstration

4.8 Digital Accelerometer Demonstration

This demonstration shows a bubble level implementation based on a digital accelerometer. We use I^2C protocol to control the ADXL345 digital accelerometer, and the APDS-9300 Miniature Ambient Light Photo Sensor. The LCD displays the interface of our game. When tilting the VEEK-MT, the ADXL345 measures the static acceleration of gravity. In our Nios II software, we compute the change of angle in the x-axis and y-axis, and show angle data in the LCD display. The value of light sensor will change as the brightness changes around the light-sensor.

Figure 4-16 shows the hardware system block diagram of this demonstration. The system is clocked by an external 50MHz Oscillator. Through the internal PLL module, the generated 150MHz clock is used for Nios II processor and other components, and there is also 10MHz for low-speed peripherals.

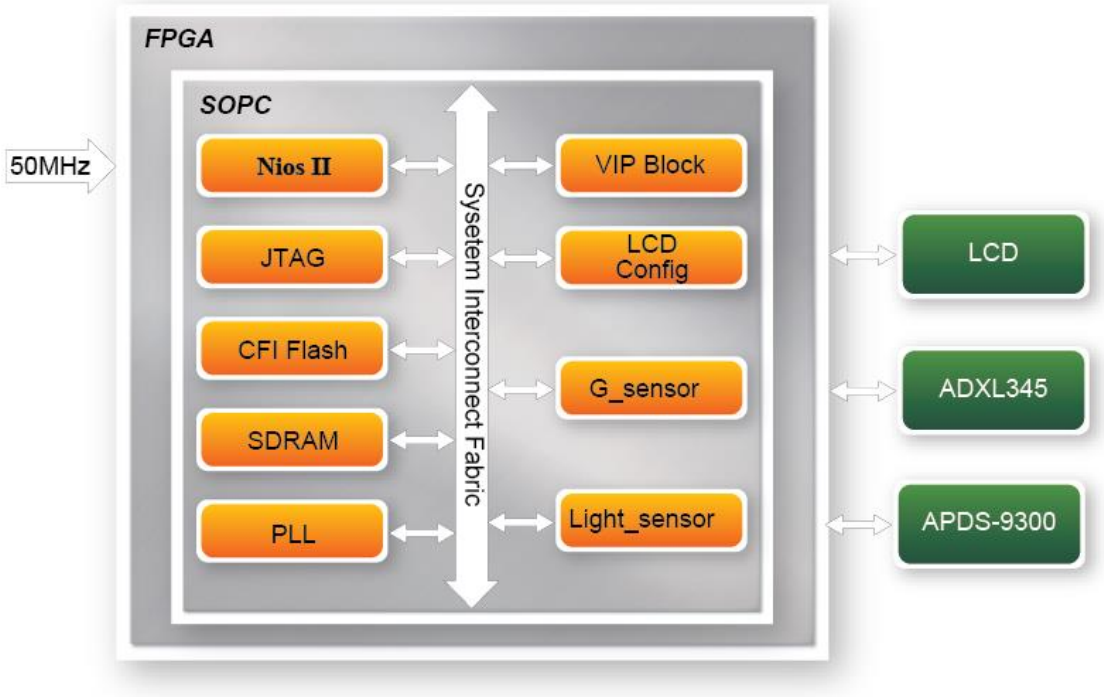

Figure 4-16 Block diagram of the digital accelerometer demonstration

■ Demonstration Source Code

- Project directory: G_sensor
- Bit stream used: G_sensor.sof
- Nios II Workspace: G_sensor\Software

■ Demonstration Batch File

Demo Batch File Folder: G_sensor\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: G_sensor.sof
- Nios II Program: G_sensor.elf

■ Demonstration Setup

- Load the bit stream into the FPGA on the VEEK-MT.
- Run the Nios II Software under the workspace G_sensor\Software (Note*).
- After the Nios II program is downloaded and executed successfully, a prompt message will be

displayed in nios2-terminal: “its ADXL345’s ID = e5”.

- Tilt the VEEK-MT to all directions, and you will find that the angle of the g-sensor and value of light sensor will change. When turning the board from -80° to -10° and from 10° **to 80°** in Y-axis, or from 10° to 80° and from -80° to -10° in Y-axis, the image will invert **Figure 4-17** shows the demonstration in action.

Figure 4-17 Digital Accelerometer demonstration

Note:

Execute G_sensor \demo_batch\test.bat to download .sof and .elf files.

Chapter 5

Application Selector

The application selector utility is the default code that powers on the FPGA and offers a graphical interface on the LCD, allowing users to select and run different demonstrations that reside on an SD card.

5.1 Ready to Run SD Card Demos

You can find several ready-to-run SD card demos in your SD card root directory as well as in the System CD under **Factory_Recovery\Application_Selector** folder. **Figure 5-1** shows the photograph of the application selector main interface.

Figure 5-1 Application selector main interface

Also, you can easily convert your own applications to be loadable by the application selector. For more information see "**Creating Your Own Loadable Applications**" in section 5.3. If you have lost the contained files in the SD card, you could find them on the VEEK-MT System CD under the **Factory_Recovery** folder.

5.2 Running the Application Selector

- Connect power to the VEEK-MT
- Insert the SD card with applications into the SD card socket of VEEK-MT
- Switch on the power (SW18) (1*)
- Scroll to select the demonstration to load using the side-bar
- Tap on the Load button to load and run a demonstration (2*)

Note:
(1)If the board is already powered, the application selector will boot from EPCS, and a splash screen will appear while the application selector searches for applications on the SD Card.
(2)The application will begin loading, and a window will be displayed showing the progress. Loading will take between 2 and 30 seconds, depending on the size of the application.

5.3 Application Selector Details

This section describes some details about the operation of the application selector utility.

■ SD Card

The Application Selector uses an SD card for storing applications. The SD card must be formatted with the FAT 16 file system, and can be any capacity up to 2GB. Long file names are supported. The Nios II CPU access the SD card through an SD card SPI controller.

■ Application Files

Each loadable application consists of two binary files, all stored on the SD card. The first binary file represents the software portion of the example and must be derived from an .ELF file as described in the section of this document titled “Creating Your Own Loadable Applications”. This binary file can be named anything supported by the FAT16 file system, the only restriction being that the name must end with ***_SW***.bin. The second binary file represents the hardware portion of the example and must be derived from a .SOF file as described in the section of this document titled “Creating Your Own Loadable Applications”. This file can be named anything supported by the FAT 16 file system, the only restriction being that the name must end with ***_HW***.bin.

■ SD Card Directory Structure

All loadable applications on the SD card must be located in a top-level directory named Application_Selector. Under the Application_Selector directory, each application is located in its own subdirectory. The name of that subdirectory is important because the application selector utility uses that name as the title of the application when displaying it in the main menu. The subdirectory names can be anything so long as they adhere to the FAT file system long file name rules. Spaces

are permitted.

■ CFI Flash

CFI flash is used to store the software binary files of applications. All software binary files used by the application selector contain a boot copier which is pre-ended by the Nios2-elf-objcopy utility during file conversion process described in the "Creating Your Own Loadable Applications" section. The boot copier copies the software code to program memory before running it. The Application software binary file is stored in flash at load time to an offset 0x0.

■ EPCS Device

EPCS is used to store both the binary file of the Application Selector (both hardware and software image) itself, as well as hardware binary files of applications which are being loaded. The Application Selector binary file is permanently stored in EPCS device at offset 0x0. Hardware binary files for the applications being loaded get written to EPCS at load time to an offset 0x400000.

■ Creating Your Own Loadable Applications

It is easy to convert your own Nios II design into an application which is loadable by the Application Selector utility. All you need is a hardware image (a .SOF file) and a software image which runs on that hardware (a Nios II .ELF file).

The only restrictions are:

- The hardware designs must contain a CFI Flash controller (1*)
- If the .SOF file contains a Nios II CPU, then its reset address should be set to CFI Flash at offset 0x0
- Before compiling the software, make sure you have set your software's program memory (.text section) in Flash memory under the System Library Properties (Nios II IDE) page or through BSP Editor (Nios II SBT for Eclipse) utility (2*)

Once you have your working .SOF and .ELF file pair, perform the following steps to convert them to a loadable application selector compatible application.

- Copy both the .SOF and .ELF files into a common directory relying on your choice. This directory is where you will convert the files
- On your host PC, launch a Nios II Command Shell from Start ->Programs -> Altera -> Nios II <version #> EDS -> Nios II Command Shell
- From the command shell navigate to where your SOF file is located and create your hardware binary file using the following commands
- Convert .sof file into .flash file
- sof2flash --epcs --input="your example.sof" --output= "your example_*HW*.flash"(3*)
- Convert .flash file into .binary file

- nios2-elf-objcopy –I srec –O binary “your example_*HW*.flash” “your example_*HW*.bin”
- From the command shell navigate to where your ELF file is located and create your software binary file using the following command
- nios2-elf-objcopy –O binary “your example.elf” “your example_*SW*.bin ”(4,5*)
- Create a new subdirectory and name it what you would like the title of your application to be shown as in the application selector
- Using an SD card reader, copy the directory onto an SD Card into a directory named “Application_Selector”. The directory structure on the SD Card should look like this:
- Application_Selector\<Name of Application>\{<elf_name>_SW.bin;<sof_name>_HW.bin}
- Place the SD card in the VEEK-MT, and switch on the power. The Application Selector will start up, and you will now see your application appear as one of the selections

Note:
(1).You may not need a CFI Flash controller when your design does not contain a Nios II processor or you store your software code within the on-chip memory and use the .hex initialization file.
(2).If you would like to use other memories such as SRAM or SDRAM as the program memory, you may need to perform two steps to convert your .elf file into .bin file to make the software properly run on VEEK-MT. The commands seem to look like this:
elf2flash --base=flash_base_address --end=flash_end_address --reset=flash_base_address --input="<your software name>.elf" --output="<your software name>.flash" --boot="$SOPC_KIT_NIOS2/components/altera_nios2/boot_loader_cfi.srec"
nios2-elf-objcopy –I srec –O binary <your software name>.flash <your software name>_SW.bin
(3).You may pad a --compress option for saving binary image space because the Cyclone IV E series support the decompress feature while loading hardware image from EPCS device.
(4).The command will use the default HAL boot loader and link it to the .text section.
(5).You can also use the tool ‘bin_demo_batch’ to convert your sof and elf to bin. Copy ”your example.sof” and “your example.elf” to the bin_demo_batch folder, rename them to test.sof, test.elf, execute the test.bat, then the final test_HW.bin and test_SW.bin are your target files.

5.4 Restoring the Factory Image

This section describes some details about the operation of restoring the Application Selector factory image.

■ Combining factory recovery binary files

In the factory settings, you need to program Application Selector binary files to EPCS. Before programming, you should combine application selector software binary file and hardware binary file together by executing the instructions below:

- Copy both the Selector.sof and Selector.elf files into a common directory relying on your choice. This directory is where you will convert the files

- On your host PC, launch a Nios II Command Shell from Start ->Programs -> Altera -> Nios II <version #> EDS -> Nios II Command Shell
- From the command shell navigate to where your SOF file is located and create your hardware binary file using the following command commands listed below
- Convert Selector.sof file into Selector_*HW*.flash file
- sof2flash --epcs –input= Selector.sof --output= Selector_HW.flash
- Convert .flash file into .bin file
- nios2-elf-objcopy –I srec –O binary Selector_*HW*.flash Selector_*HW*.bin
- From the command shell navigate to where your ELF file is located and create your software bin image using the following command commands listed below
- Convert Selector.elf into Selector_*SW*.flash
- elf2flash –epcs --after=Selector_HW.flash --input=Selector.elf --output= Selector_*SW*.flash
- Convert Selector_*SW*.flash into Selector_*SW*.bin
- nios2-elf-objcopy –I srec –O binary Selector_*SW*.flash Selector_SW.bin
- Combine Selector_*HW*.bin and Selector_SW.bin using the following command
- cat Selector_*HW*.bin Selector_*SW*.bin > Selector.bin
- The generated Selector.bin is our target binary file

■ Restoring the original binary file

- To restore the original contents of the Application Selector, perform the following steps:
- Copy Selector project into a local directory of your choice. The Selector project is placed in Demonstrations\Selector
- Power on the VEEK-MT, with the USB cable connected to the USB Blaster port
- Download the Selector.sof to the board by using either JTAG or AS programming
- Run the Nios II and choose Selector\Software as the workspace
- Choose Nios II > Flash Programmer to open the flash programmer
- Choose Program a file into memory, choose your Selector.bin file. See **Figure 5-2**
- Click Program Flash to start program Selector.bin to EPCS in the board
- When program finish, power on again

Note: You can also use 'Selector_batch' to generate selector.bin and restore the original binary file by executing the Selector.bat under the Factory_Recovery\Selector_batch folder.

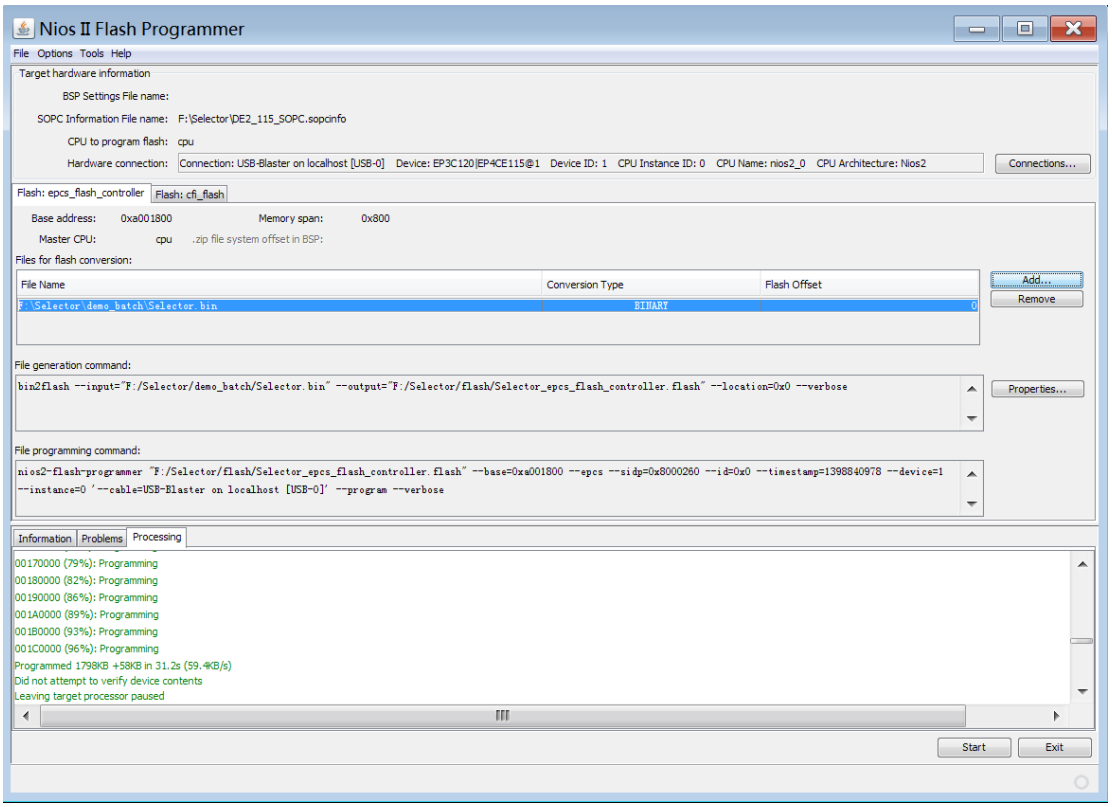

Figure 5-2 Programming Flash settings

Chapter 6

Appendix

6.1 Revision History

Version	*Change Log*
V1.0	**Initial Version (Preliminary)**
V1.1	**Modify for Q13.1**
V1.2	**Modify What's in box**

6.2 Copyright Statement