

VEEK-MT2

VEEK with Multi-touch Capacitive Panel

User Manual



terasic
www.terasic.com

ALTERA

CONTENTS

Chapter 1	Introduction of the VEEK-MT2	1
	1.1 About the Kit	6
	1.2 Getting Help	6
Chapter 2	Architecture.....	7
	2.1 Layout and Components.....	7
	2.2 Block Diagram of the VEEK-MT2	8
	2.3 What's Difference Between VEEK-MT2 and VEEK-MT	9
Chapter 3	Using VEEK-MT2	10
	3.1 Configuring the Cyclone IV E FPGA	10
	3.2 Bus Controller	13
	3.3 Using the 7" LCD Capacitive Touch Screen	14
	3.4 Using 8-megapixel Digital Image Sensor.....	15
	3.5 Using the Gyroscope, Accelerometer and Magnetometer	19
	3.6 Using the Ambient Light Sensor	20
	3.7 Using Terasic Multi-touch IP	20
Chapter 4	VEEK-MT2 Demonstrations	23
	4.1 System Requirements	23
	4.2 Factory Configuration	23
	4.3 Painter Demonstration.....	24
	4.4 Picture Viewer	27
	4.5 Video and Image Processing	30

4.6 Camera Application.....	34
4.7 Video and Image Processing for Camera	38
4.8 Digital Accelerometer Demonstration.....	40
4.9 E-Compass Demonstration.....	43
Chapter 5 Application Selector.....	48
5.1 Ready to Run SD Card Demos.....	48
5.2 Running the Application Selector.....	49
5.3 Application Selector Details.....	49
5.4 Restoring the Factory Image	52
Chapter 6 Appendix	54
6.1 Revision History.....	54
6.2 Copyright Statement.....	54

Chapter 1

Introduction of the VEEK-MT2

The Video and Embedded Evaluation Kit - Multi-touch, Second Edition (VEEK-MT2) is a comprehensive design environment with everything embedded developers need to create processing-based systems. VEEK-MT2 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-MT2 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. An 8-megapixel digital image sensor, ambient light sensor, and 3-axis accelerometer make up the rich feature-set. The VEEK-MT2 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-MT2, in combination of the LCD touch panel and digital image module, provides embedded developers the ideal platform for multimedia applications with unparalleled 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-MT2.



Figure 1-1 The VEEK-MT2 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.
- Five-point touch support
- Gesture support

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

Table 1-1 General physical specifications of the LCD

<i>Item</i>	<i>Specification</i>	<i>Unit</i>
LCD size	7-inch (Diagonal)	-
Resolution	800 x3(RGB) x 480	dot
Dot pitch	0.0642(H) x0.1790 (V)	mm
Active area	154.08 (H) x 85.92 (V)	mm
Module size	179.4(H) x 117.4(V) x 7.58(D)	mm
Surface treatment	Anti-Glare	-
Color arrangement	RGB-stripe	-
Interface	Digital	-

■ 8-Megapixel Digital Image Sensor

- 8-Mega Pixels(3264x2448)
- Support Focus Control
- Automatic black level calibration (ABLC)
- Programmable controls for frame rate, mirror and flip, cropping, and windowing
- MIPI to Parallel Port Converter

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

Table 1-2 Key performance parameters of the CMOS sensor

<i>Parameter</i>	<i>Value</i>
Active Pixels	3264Hx2448V
Pixel size	1.4umx1.4um
Color filter array	RGB Bayer pattern
ADC resolution	10-bit
Pixel dynamic range	68.8dB
SNRMAX	36.7dB

■ Ambient Light Sensor

- Approximates human-eye response
- Precise luminance measurement under diverse lighting conditions
- Programmable interrupt function with user-defined upper and lower threshold setting
- 16-bit digital output with I2C fast-mode at 400 kHz
- Programmable analog gain and integration time
- 50/60-Hz lighting ripple rejection

■ Accelerometer Features

- Digital-output triple-axis accelerometer
- Programmable full scale range of $\pm 2g$, $\pm 4g$, $\pm 8g$ and $\pm 16g$ and integrated 16-bit
- Self-test

■ Gyroscope Features

- Digital-output X-, Y-, and Z-Axis angular rate sensors
- User-programmable full-scale range of ± 250 , ± 500 , ± 1000 , and $\pm 2000^\circ/\text{sec}$ and integrated 16-bit ADCs
- Self-test

■ Magnetometer

- 3-axis silicon monolithic Hall-effect magnetic sensor with magnetic concentrator
- Wide dynamic measurement range and high resolution with lower current consumption
- Output data resolution of 14 bit ($0.6\mu\text{T}/\text{LSB}$) or 16 bit ($15\mu\text{T}/\text{LSB}$)
- Full scale measurement range is $\pm 4800\mu\text{T}$
- Magnetometer normal operating current: $280\mu\text{A}$ at 8Hz repetition rate
- Self-test function with internal magnetic source to confirm magnetic sensor operation on end products



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-MT2 which includes component datasheets, demonstrations, schematic, and user manual.

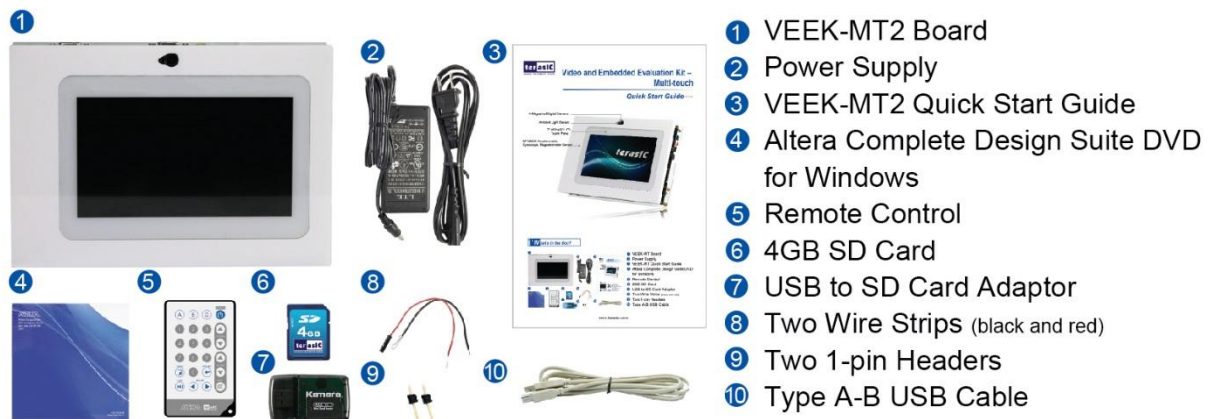


Figure 1-2 VEEK-MT2 kit package contents

1.2 Getting Help

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

- **Terasic Technologies**
- **9F., No.176, Sec.2, Gongdao 5th Rd, East Dist, Hsinchu City, 30070. Taiwan**
- **Tel: +886-3-5750-880**
- **Email: support@terasic.com**

Chapter 2

Architecture

This chapter describes the architecture of the VEEK-MT2 including block diagram and components.

2.1 Layout and Components

The picture of the VEEK-MT2 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-MT2 PCB and Component Diagram (top view)

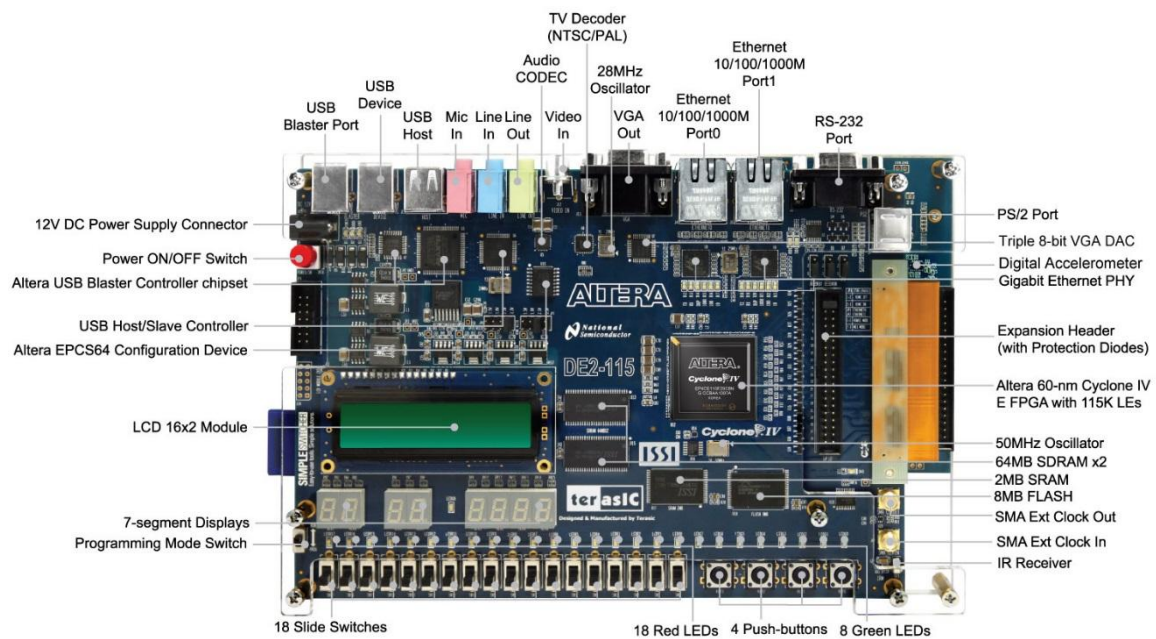


Figure 2-2 VEEK-MT2 PCB and Component Diagram (bottom view)

2.2 Block Diagram of the VEEK-MT2

Figure 2-3 gives the block diagram of the VEEK-MT2 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.

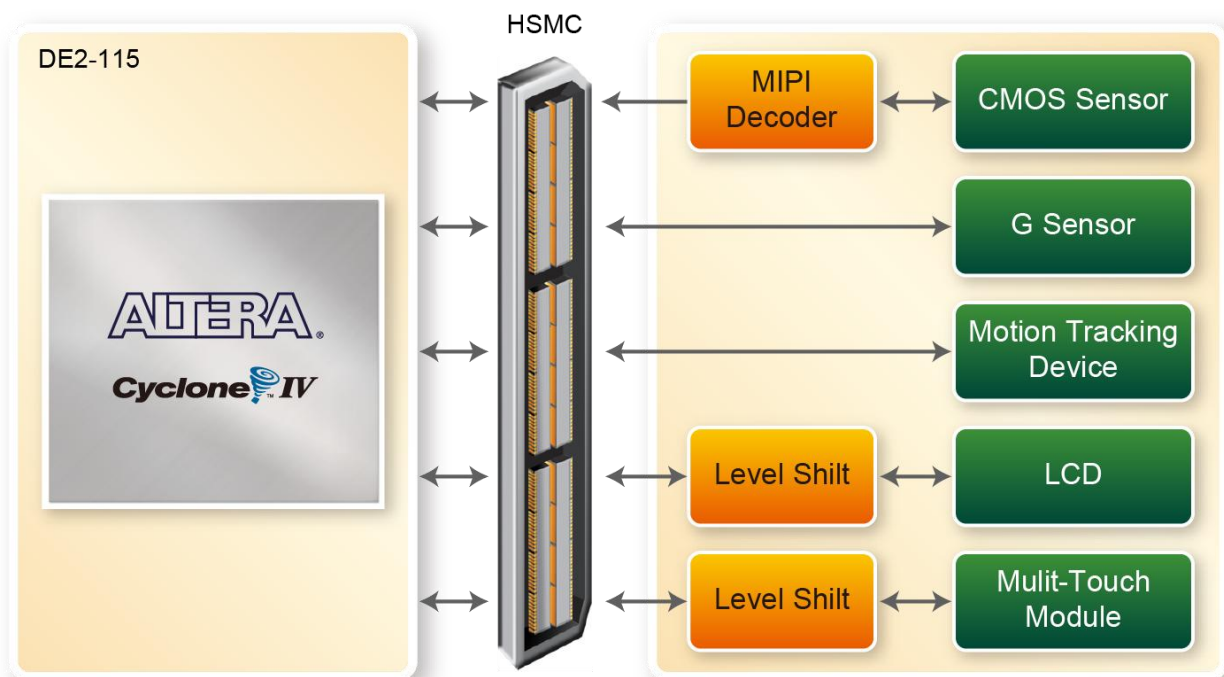


Figure 2-3 Block Diagram of VEEK-MT2

2.3 What's Difference Between VEEK-MT2 and VEEK-MT

Table 2-1 gives the difference between VEEK-MT2 and VEEK-MT.

Table 2-1 Difference between VEEK-MT2 and VEEK-MT

Signal Name	VEEK-MT2	VEEK-MT
LCD Touch Point	5 Point Touch	2 Point Touchs
Camera Module	<ul style="list-style-type: none">◦ 8-Mega Pixel◦ Auto Focus◦ MIPI Decoder	5-Mega Pixel
Motion Sensors	<ul style="list-style-type: none">◦ Gyroscope◦ Accelerometer◦ Magnetometer	Accelerometer

Chapter 3

Using VEEK-MT2

This section describes the detailed information of the components, connectors, and pin assignments of the VEEK-MT2.

3.1 Configuring the Cyclone IV E FPGA

The Video and Embedded Evaluation Kit (VEEK-MT2) 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-MT2 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-MT2

To use the JTAG interface for configuring FPGA device, the JTAG chain on the VEEK-MT2 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-MT2. 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.

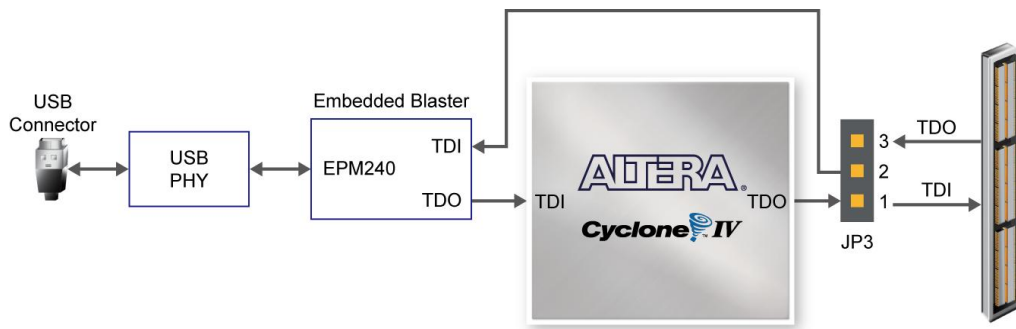


Figure 3-1 JTAG Chain

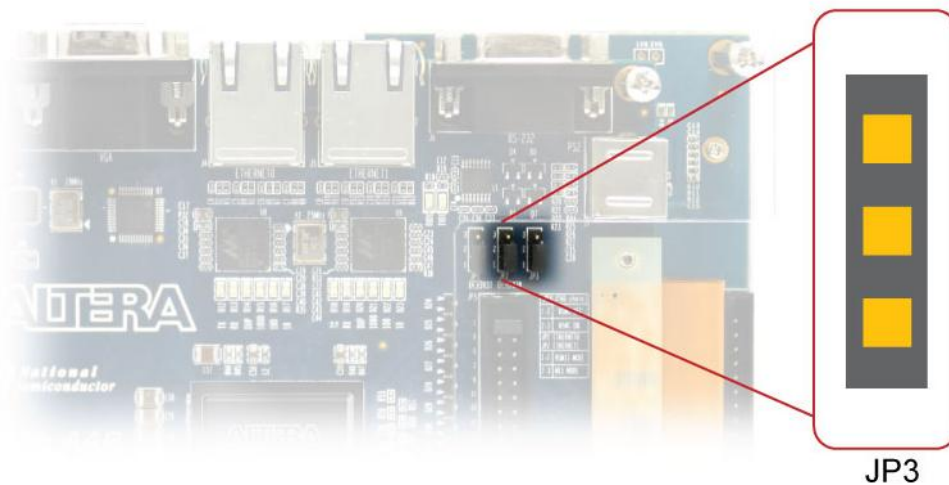


Figure 3-2 JTAG Chain Configuration Header

■ 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-MT2
- 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-MT2
- 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

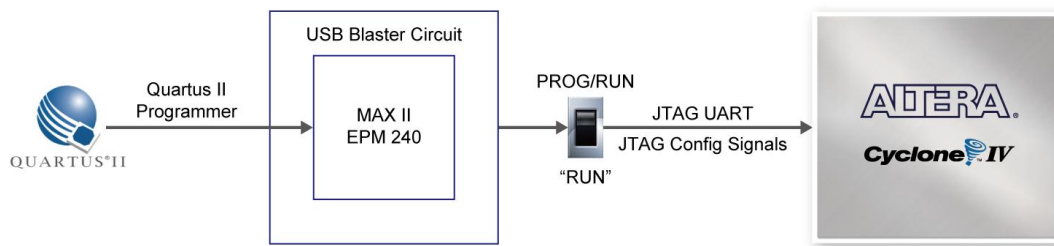


Figure 3-3 JTAG Chain Configuration Scheme

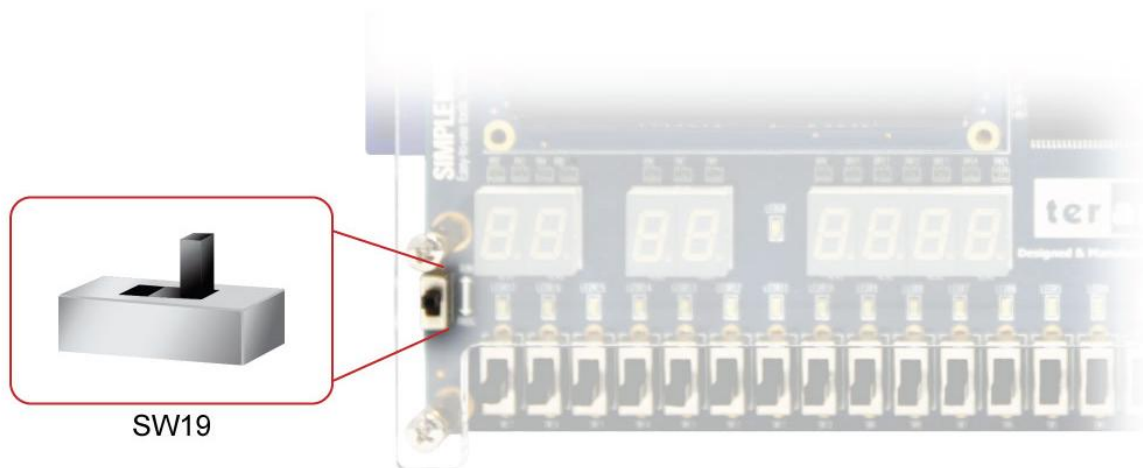


Figure 3-4 The RUN/PROG switch (SW19) is set to JTAG mode

■ 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-MT2
- Connect the supplied USB cable to the USB-Blaster port on the VEEK-MT2
- 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

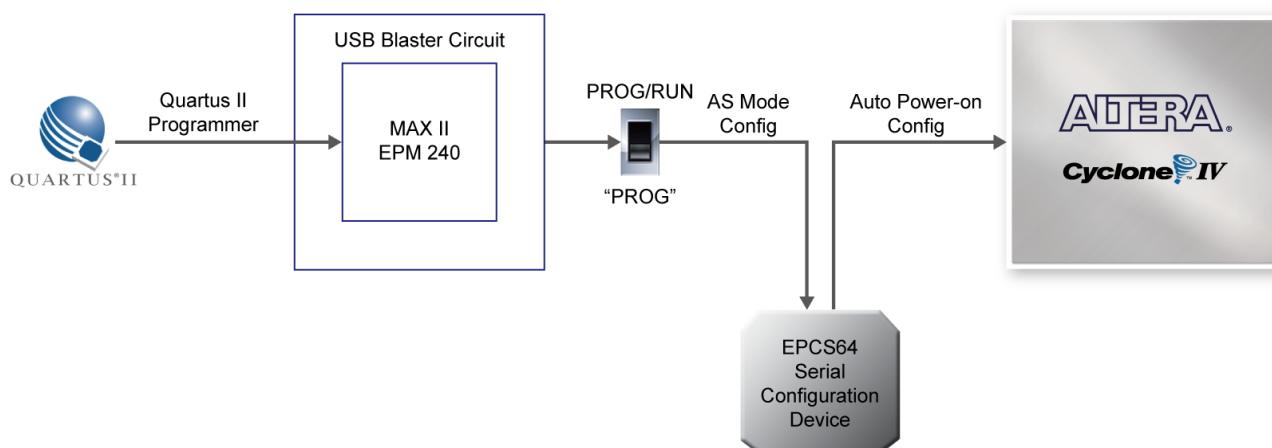


Figure 3-5 The AS Configuration Scheme

3.2 Bus Controller

The VEEK-MT2 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-MT2 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-MT2 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-MT2 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** and **Table 3-2**.

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

Table 3-1 LCD Horizontal Timing Specifications

Item	Symbol	Typical Value			Unit
		Min.	Typ.	Max.	
Horizontal Display Area	thd	-	800	-	DCLK
DCLK Frequency	fclk	26.4	33.3	46.8	MHz
One Horizontal Line	th	862	1056	1200	DCLK
HS pulse width	thpw	1		40	DCLK
HS Blanking	thb	46	46	46	DCLK
HS Front Porch	thfp	16	210	354	DCLK

Table 3-2 LCD Vertical Timing Specifications

Item	Symbol	Typical Value			Unit
		Min.	Typ.	Max.	
Vertical Display Area	tvd	-	480	-	TH
VS period time	tv	510	525	650	TH
VS pulse width	tvpw	1	-	20	TH
VS Blanking	tvb	23	23	23	TH
VS Front Porch	tvfp	7	22	147	TH

Table 3-3 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 8-megapixel Digital Image Sensor

Terasic VEEK-MT2 board equips with an 8M pixel MIPI camera module named OV8865 (See Figure 3-7). The OV8865 color image sensor is a high performance, 8 megapixel RAW image sensor that

delivers 3264x2448. It provides options for multiple resolutions while maintaining full field of view. Users can program image resolution, frame rate, and image quality parameters. Camera functions are controlled via I2C bus (CAMERA_I2C_SDA and CAMERA_I2C_SCL). The I2C device address is 0x6C. For more hardware description and register information about this camera module, please refer to the datasheet named OV8865 Data Sheet.pdf in the VEEK-MT2 System CD.

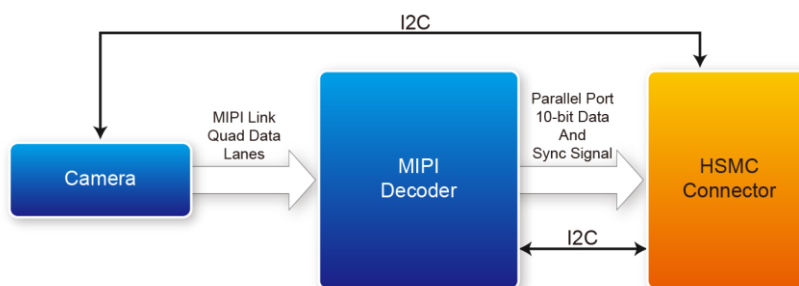


Figure 3-7 Block Diagram of the Bus Controller

■ Voice Coil Motor (VCM)

There is a Voice Coil Motor (VCM) driver chip named VCM149C on the MIPI camera module. Users can use the same I2C bus (I2C device address is 0x18) to modify the DAC value in the VCM driver chip that can allow the VCM to move its lens to the desired position for getting a sharp image and realizing the Auto Focus (AF) feature. Terasic also provides an AF demonstration and IP in the System CD, see section 4.6 for details. The datasheet of this VMC driver IC named VM149C VCM Driver IC.pdf also can be found in the System CD.

■ MIPI Decoder

The MIPI camera module output interface is MIPI interface, which can not directly connect to the Terasic FPGA board; therefore, a MIPI Decoder (TC358748XBG) is added to convert MIPI interface to a parallel port interface. Decoder users can quickly obtain the image data and process it. MIPI Decoder can convert MIPI Interface up to 24-bit data. The Camera module used on the D8M can only output 10 bit data, MIPI_PIXEL_D[9:0] the HSMC connector is the camera image output data bus.

FPGA also can read/write MIPI Decoder through a I2C bus (MIPI_I2C_SDA / MIPI_I2C_SCL ; I2C device address is 0x1C), which is different from the camera module I2C bus. On the VEEK-MT2 board, MIPI Decoder can output clocks to the MIPI camera and FPGA board. So in the demonstrations, most of them show how to control IC PLL parameters as well as others. Detailed clock functions are described in blow.

■ Clock Tree

Figure 3-8 is the VEEK-MT2 board's camera clock tree block diagram. MIPI Decoder PLL receives FPGA Reference Clock (MIPI_REFCLK) and outputs Clock to Camera sensor (MCLK), at the same time, MIPI Decoder PLL will also output a parallel port clock (MIPI_PIXEL_CLK) and feedback to the FPGA to deal with parallel data.

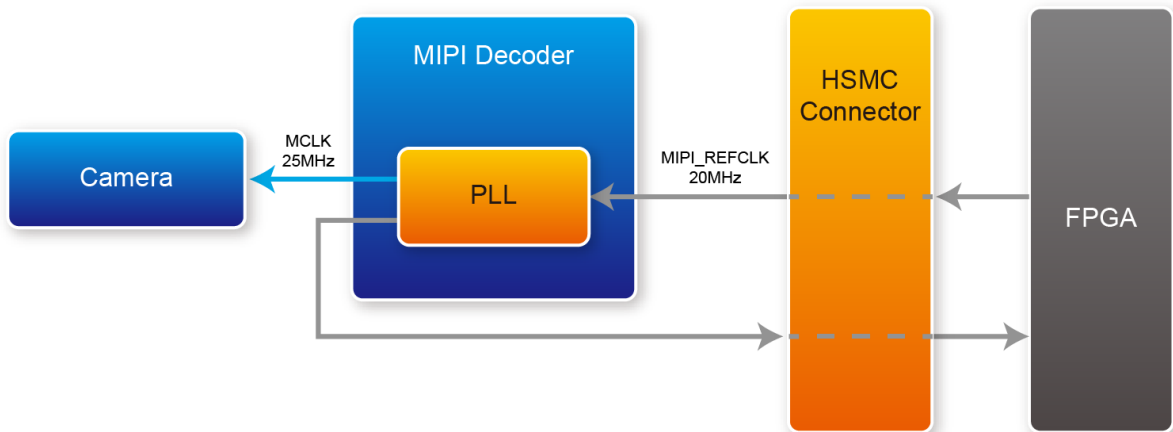


Figure 3-8 Block Diagram of the Bus Controller

In the provided demonstrations, MIPI_REFCLK is set to 20MHz, FPGA transmits this clock to the VEEK-MT2's MIPI Decoder PLL through the HSMC connector. No matter how much the camera resolution is, the MCLK fixed output is 25MHz. According to the output resolution, MIPI_PIXEL_CLK can be set as 25MHz for 640x480@60fps and 50MHz for 1920x1080@15fps.

For more MIPI Decoder PLL setting details, please refer to TC358746AXBG_748XBG_rev09.pdf "Chapter 5: Clock and System" or refer to Terasic demonstrations.

■ Pin Assignment

Table 3-4 Pin assignment of the Camera sensor

Signal Name	FPGA Pin No.	Description	I/O Standard
MIPI_PIXEL_D[0]	F24	Parallel Port Data	2.5V
MIPI_PIXEL_D[1]	F25	Parallel Port Data	2.5V
MIPI_PIXEL_D[2]	D26	Parallel Port Data	2.5V
MIPI_PIXEL_D[3]	C27	Parallel Port Data	2.5V
MIPI_PIXEL_D[4]	F26	Parallel Port Data	2.5V
MIPI_PIXEL_D[5]	E26	Parallel Port Data	2.5V
MIPI_PIXEL_D[6]	G25	Parallel Port Data	2.5V
MIPI_PIXEL_D[7]	G26	Parallel Port Data	2.5V
MIPI_PIXEL_D[8]	H25	Parallel Port Data	2.5V
MIPI_PIXEL_D[9]	H26	Parallel Port Data	2.5V
MIPI_PIXEL_D[10]	M26	Reserve	2.5V
MIPI_PIXEL_D[11]	M25	Reserve	2.5V
MIPI_PIXEL_D[12]	AF27	Reserve	2.5V
MIPI_PIXEL_D[13]	AE28	Reserve	2.5V
MIPI_RESET_n	D27	Master Reset signal for MIPI camera and bridge device	2.5V
MIPI_PIXEL_CLK	J27	Parallel Port Clock signal	2.5V
MIPI_PIXEL_HS	K26	Parallel Port Horizontal Synchronization signal	2.5V
MIPI_PIXEL_VS	K25	Parallel Port Vertical Synchronization signal	2.5V
MIPI_CS_n	F28	Chip Select	2.5V
MIPI_REFCLK	G23	Reference Clock Input of bridge device	2.5V
MIPI_I2C_SCL	AE26	I2C Clock for bridge device	2.5V
MIPI_I2C_SDA	AE27	I2C Data for bridge device	2.5V
CAMERA_PWDN_n	R22	Power Down signal of MIPI camera	2.5V
CAMERA_I2C_SCL	R21	I2C Clock for MIPI camera	2.5V
CAMERA_I2C_SDA	F27	I2C Data for MIPI camera	2.5V
MIPI_MCLK	G24	MIPI camera system clock (Reserve)	2.5V

3.5 Using the Gyroscope, Accelerometer and Magnetometer

The VEEK-MT2 is equipped with a Motion-Tracking device named MPU-9250. The MPU-9250 is a 9-axis Motion-Tracking device that combines a 3-axis gyroscope, 3-axis accelerometer and 3-axis magnetometer. Detail features of these sensors are listed below:

■ Gyroscope

The MPU-9250 consists of three independent vibratory MEMS rate gyroscopes, which detect rotation about the X-, Y-, and Z- Axes. When the gyros are rotated about any of the sense axes, the Coriolis Effect causes a vibration that is detected by a capacitive pickoff. The resulting signal is amplified, demodulated, and filtered to produce a voltage that is proportional to the angular rate. This voltage is digitized using individual on-chip 16-bit Analog-to-Digital Converters (ADCs) to sample each axis. The full-scale range of the gyro sensors may be digitally programmed to ± 250 , ± 500 , ± 1000 , or ± 2000 degrees per second (dps). The ADC sample rate is programmable from 8,000 samples per second, down to 3.9 samples per second, and user-selectable low-pass filters enable a wide range of cut-off frequencies.

■ Accelerometer

The MPU-9250's 3-Axis accelerometer uses separate proof masses for each axis. Acceleration along a particular axis induces displacement on the corresponding proof mass, and capacitive sensors detect the displacement differentially. The MPU-9250's architecture reduces the accelerometers' susceptibility to fabrication variations as well as to thermal drift. When the device is placed on a flat surface, it will measure 0g on the X- and Y-axes and +1g on the Z-axis. The accelerometers' scale factor is calibrated at the factory and is nominally independent of supply voltage. Each sensor has a dedicated sigma-delta ADC for providing digital outputs. The full scale range of the digital output can be adjusted to $\pm 2g$, $\pm 4g$, $\pm 8g$, or $\pm 16g$.

■ Magnetometer

The 3-axis magnetometer uses highly sensitive Hall sensor technology. The magnetometer portion of the IC incorporates magnetic sensors for detecting terrestrial magnetism in the X-, Y-, and Z- Axes, a sensor driving circuit, a signal amplifier chain, and an arithmetic circuit for processing the signal from each sensor. Each ADC has a 16-bit resolution and a full scale range of $\pm 4800 \mu T$.

Communication with all registers of the device is performed using either I2C at 400kHz or SPI at 1MHz. For applications requiring faster communications, the sensor and interrupt registers may be read using SPI at 20MHz. 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** gives the pin assignment information of the LCD touch panel. 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 MPU-9250.

<i>Signal Name</i>	<i>FPGA Pin No.</i>	<i>Description</i>	<i>I/O Standard</i>
MPU_AD0_SDO	K27	I2C Slave Address LSB (AD0); SPI serial data output (SDO)	2.5V
MPU_CS_n	F28	Chip select (SPI mode only)	2.5V
MPU_FSYNC	G28	Frame synchronization digital input	2.5V
MPU_INT	G27	Interrupt digital output	2.5V
MPU_SCL_SCLK	M27	I2C serial clock (SCL); SPI serial clock (SCLK)	2.5V
MPU_SDA_SDI	K28	I2C serial data (SDA); SPI serial data input (SDI)	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 3-6 contains the pin names and descriptions of the ambient light sensor module.

<i>Signal Name</i>	<i>FPGA Pin No.</i>	<i>Description</i>	<i>I/O Standard</i>
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.v`, which is located in System CD \IP folder.

The IP decodes I2C information and outputs coordinates and gesture information. The inputs and outputs of this IP module are shown below:

```

module i2c_touch_config(
    //Host Side
    iCLK,
    iRSTN,
    oREADY,
    INT_n,
    oREG_X1,
    oREG_Y1,
    oREG_X2,
    oREG_Y2,
    oREG_X3,
    oREG_Y3,
    oREG_X4,
    oREG_Y4,
    oREG_X5,
    oREG_Y5,
    oREG_GESTURE,
    oREG_TOUCH_COUNT,
    //I2C Side
    I2C_SDAT,
    I2C_SCLK
);

```

The purpose of signals for this IP is described in

Table 3-8. The IP requires a 50 MHz signal as a reference clock to the iCLK pin and system reset signal to the iRSTN. INT_n, The signals of I2C_SCLK, and I2C_SDAT pins should be connected to the MTL_TOUCH_INT_n, MTL_TOUCH_I2C_SCL, and MTL_TOUCH_I2C_SDA signals in the 2x20 GPIO header, respectively.

When touch activity occurs, the control application should check whether the value of oREG_GESTURE matches a pre-defined gesture ID defined in Table 3 4 and the relative X/Y coordinates can be derived from oREG_X and oREG_Y. Figure 3 1 shows the signaltap II waveform of the IP. When the oREADY rises, it indicates touch activity, and the associated information can be collected from the oREG_X1~ oREG_X5, oREG_Y1~ oREG_Y5, oREG_TOUCH_COUNT, and oREG_GESTURE pins.

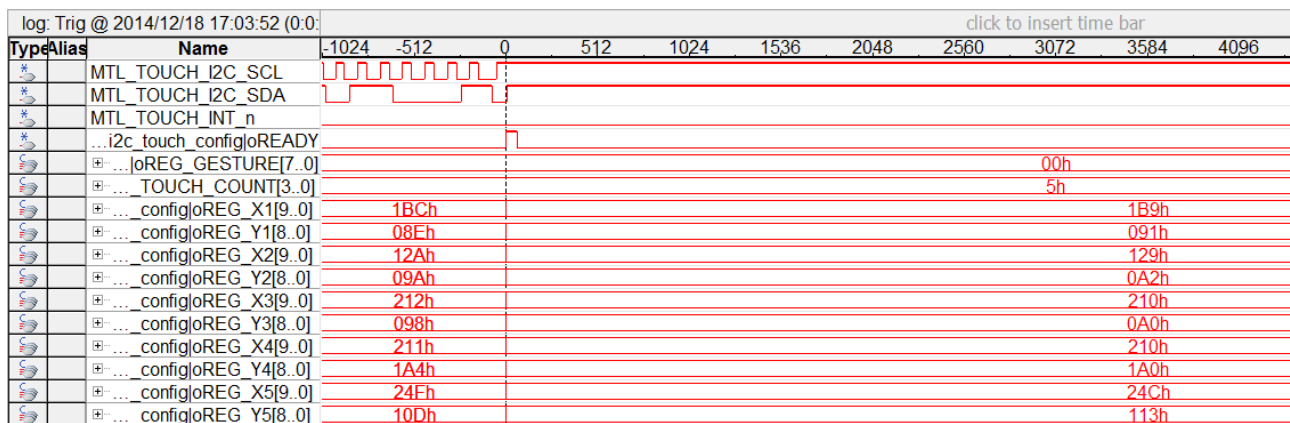


Figure 3-9 Signaltap II Waveform for Multi-Touch IP

Table 3-7 Definition of Terasic Multi-touch IP Signals

<i>Pin Name</i>	<i>Direction</i>	<i>Description</i>
iCLK	Input	Connect to 50MHz clock
iRSTN	Input	Connect to system reset signal
INT_n	Input	Connect to interrupt pin of touch IC
oREADY	Output	Triggered when the data of following six outputs are valid
oREG_X1	Output	10-bit X coordinate of first touch point
oREG_Y1	Output	9-bit Y coordinate of first touch point
oREG_X2	Output	10-bit X coordinate of second touch point
oREG_Y2	Output	9-bit Y coordinate of second touch point
oREG_X3	Output	10-bit X coordinate of first touch point
oREG_Y3	Output	9-bit Y coordinate of second touch point
oREG_X4	Output	10-bit X coordinate of first touch point
oREG_Y4	Output	9-bit Y coordinate of second touch point
oREG_X5	Output	10-bit X coordinate of first touch point
oREG_Y5	Output	9-bit Y coordinate of second touch point
oREG_TOUCH_COUNT	Output	2-bit touch count. Valid value is 0, 1, or 2.
oREG_GESTURE	Output	8-bit gesture ID (See Table 3-8)
I2C_SCLK	Output	Connect to I2C clock pin of touch IC
I2C_SDAT	Inout	Connect to I2C data pin of touch IC

The gestures and IDs supported are shown in

Table 3-8.

Table 3-8 Gestures and Its IDs

<i>Gesture</i>	<i>ID (hex)</i>
Move Up	0x10
Move Left	0x14
Move Down	0x18
Move Right	0x1C
Zoom In	0x48
Zoom Out	0x49
No Gesture	0x00



Note:

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

Chapter 4

VEEK-MT2 Demonstrations

This chapter gives detailed description of the provided bundles of exclusive demonstrations implemented on VEEK-MT2. These demonstrations are particularly designed (or ported) for VEEK-MT2, 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:

- For demonstration of VEEK-MT2, please install Altera Quartus II 15.0 and NIOS II EDS 15.0 or later edition on the host computer. Users need to download Quartus II 15.0 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-MT2 system CD to your host computer.

4.2 Factory Configuration

The VEEK-MT2 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-MT2 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 for 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 the 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 I2C interface IP is used for accessing touch resolution and gestures, additionally, a PIO IP used for accessing interrupt signal.

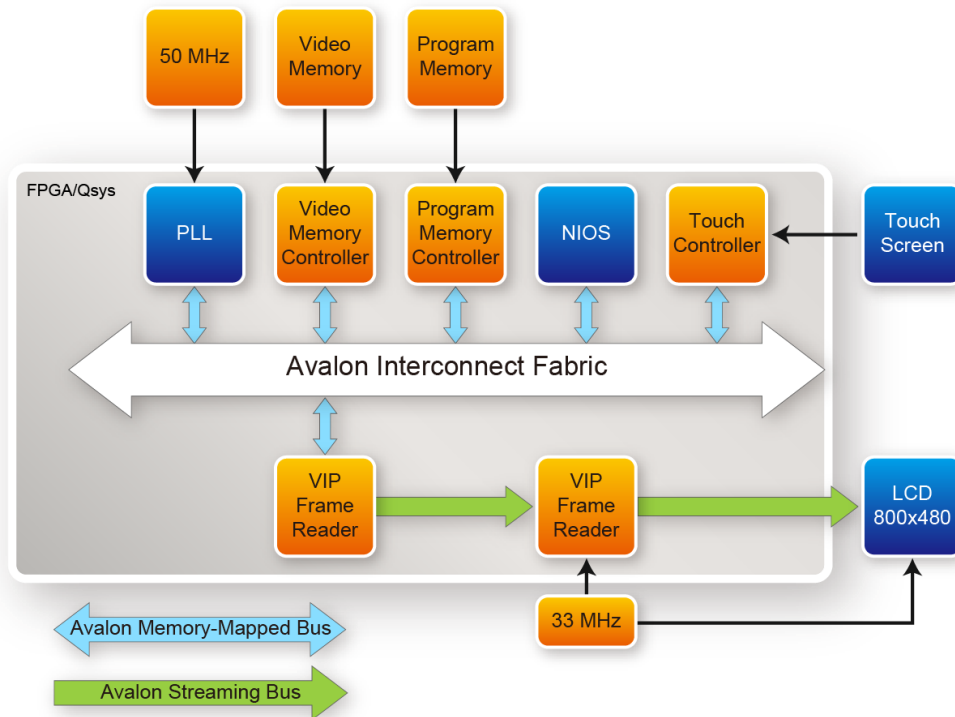


Figure 4-2 Block diagram of the Painter demonstration

■ 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 photo when zoom-in gesture is detected.

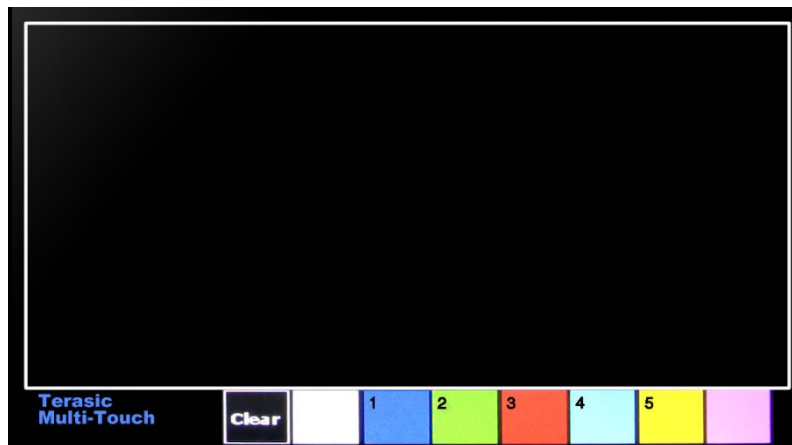


Figure 4-3 GUI of Painter Demo

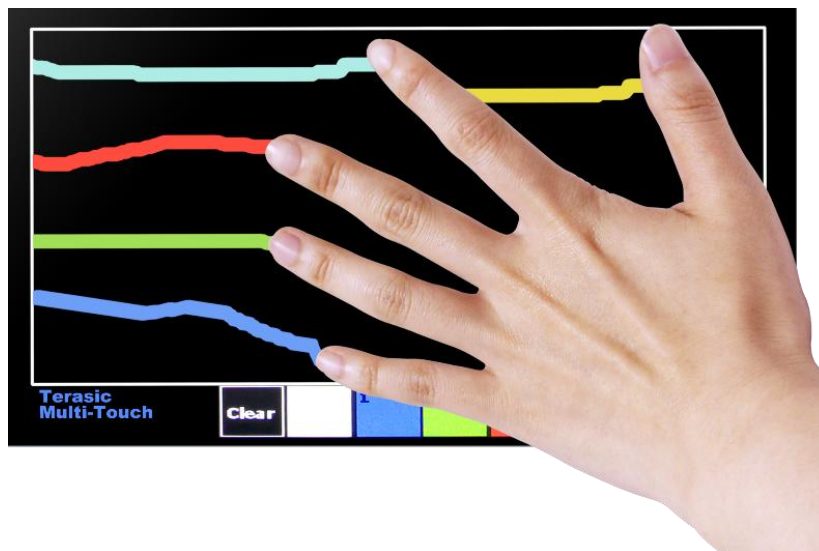


Figure 4-4 Five Point Touch Painting

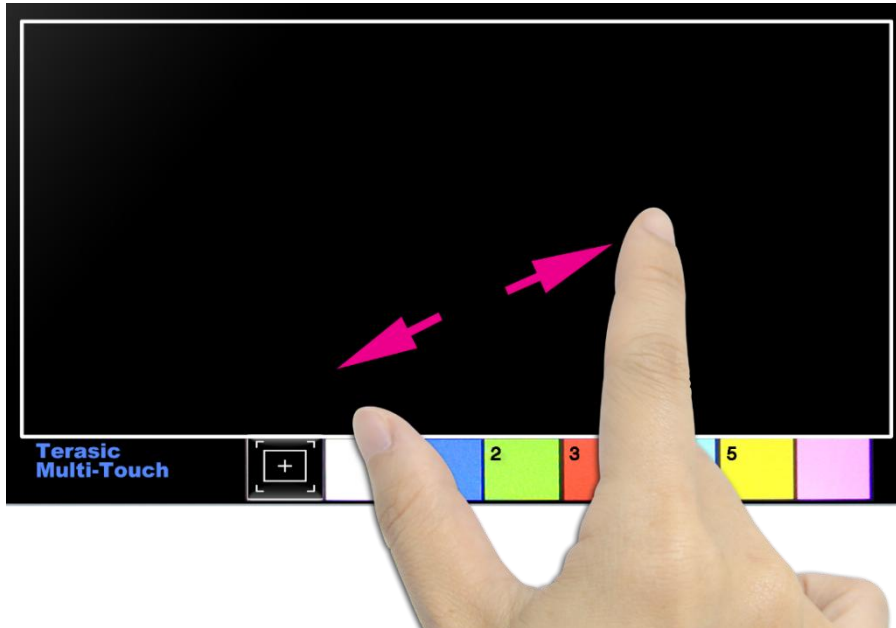


Figure 4-5 Zoom-in Gesture



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-MT2 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 4-6** shows the block diagram of this demonstration.

The Nios II CPU here takes a key role 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.

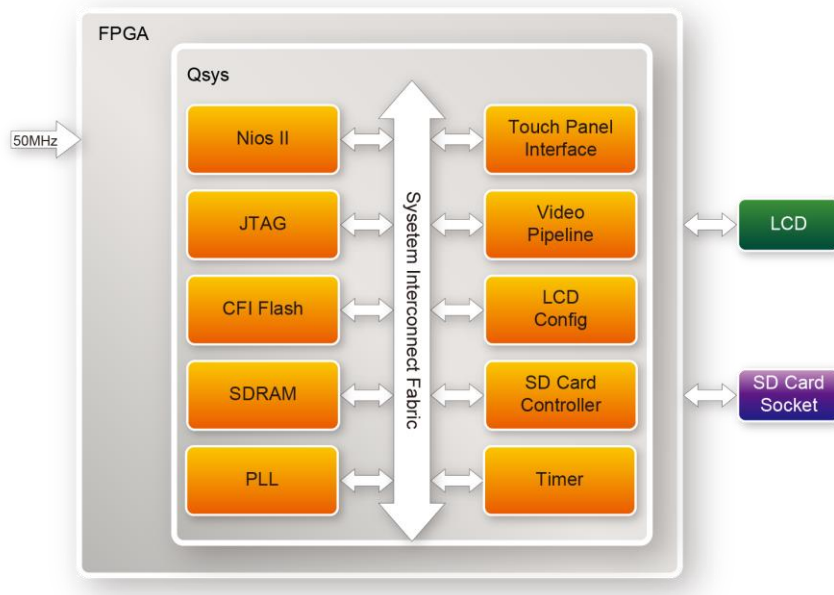


Figure 4-6 Block diagram of the picture viewer demonstration

■ 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 FAT32 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-MT2.
4. Load the bit stream into the FPGA on the VEEK-MT2.
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 4-7 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 Function	MegaCore 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-MT2.

A video source is input through an analog composite port on VEEK-MT2 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-MT2 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 4-8 shows the Video and Image Processing block diagram.

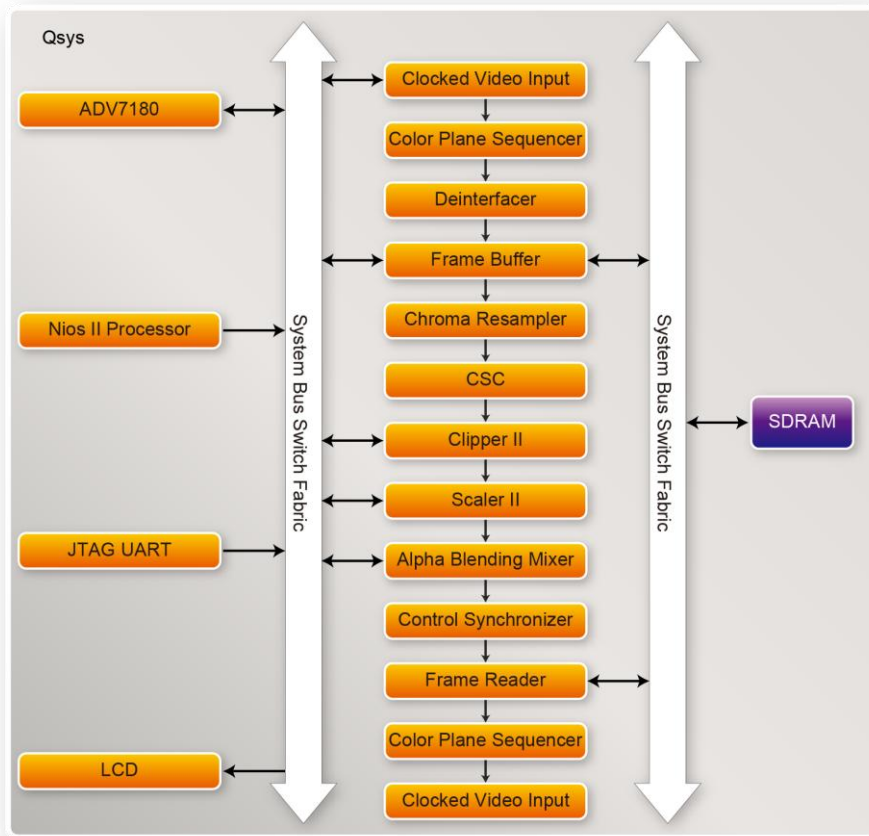


Figure 4-8 VIP Example SOPC Block Diagram (Key Components)

■ 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 the VGA output of the VEEK-MT2 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 *)
- Connect a DVD player's composite video output(yellow plug) to the Video-IN RCA jack(J12) of the VEEK-MT2. The DVD player has to be configured to provide NTSC output or PAL output
- Press and drag the video frame box will result in scaling the playing window to any size, as shown in [Figure 4-9](#)



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 4-10 illustrates the setup for this demonstration.



Figure 4-9 The VIP demonstration running result

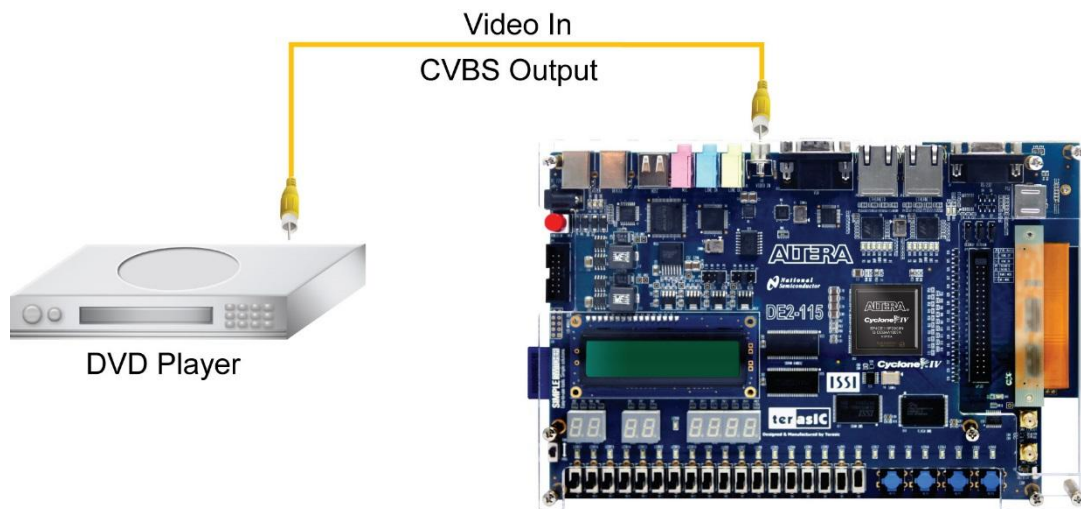


Figure 4-10 Setup for the VIP demonstration

4.6 Camera Application

This demonstration shows a digital camera reference design using the 8-Megapixel CMOS sensor and LCD modules on the VEEK-MT2. 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.

Figure 4-11 shows the block diagram of the demonstration. The CMOS sensor's raw data will be written in SDRAM first. After finishing writing a frame, Sdram_Control module will read out the data from SDRAM to RAW2RGB_J module to convert RAW data to RGB data. The RGB data will output along with the signal timing generated by VGA_Controller to the LCD. In this block, it also provides an "auto-focusing" module, "monitor CMOS sensor frame rate" module, as well as a "pixel clock" module. All module functions are described below:

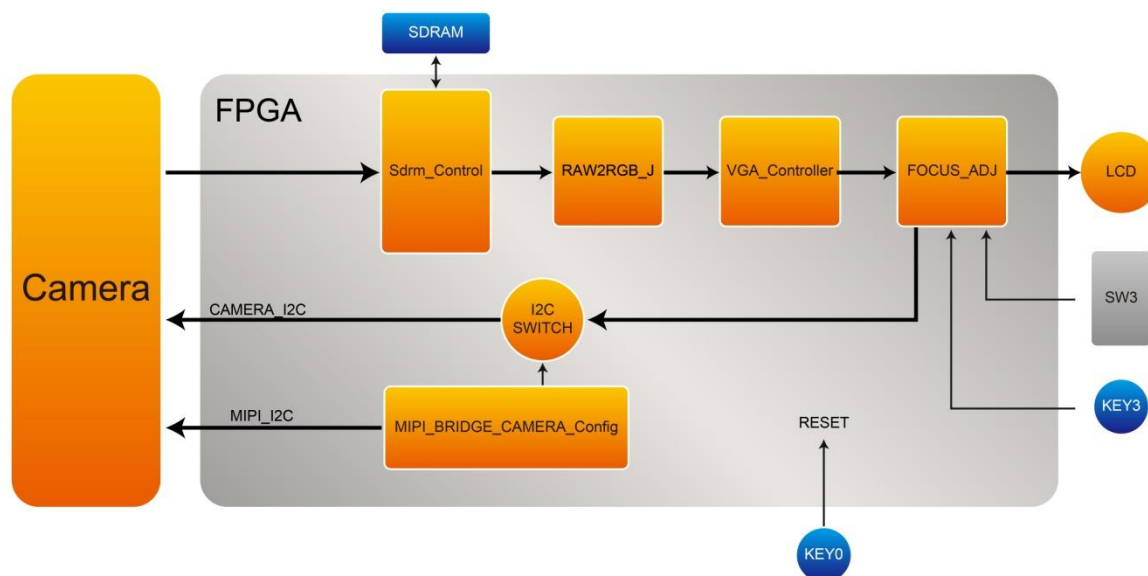


Figure 4-11 Block diagram of the digital camera design

- **MIPI_BRIDGE_CAMERA_Config:** the MIPI BRIDGE I2C and Camera I2C setting controller, such as set to output 800X480@60Hz timing. It mainly writes I2C corresponding parameters to MIPI-BRIDGE IC register and Camera Sensor IC register respectively through their own I2C buses. MIPI_I2C bus is used to write MIPI BRIDGE I2C (I2C Slave Address = 0x1c.), CAMERA_I2C bus is used to write Camera Sensor (IC Slave Address = 0x6c).
- **Sdram_Control:** This module is one Sdram_Controller can control general external SDRAM Memory and read/write image data.
- **RAW2RGB_J:** This module is to convert RAW data to RGB data.
- **VGA_Controller:** the LCD signal timing generator, can generate signal timing which the resolution is 800x480.
- **FOCUS_ADJ:** This module provides two main functions.

■ The Default Camera Settings

In this demonstration, the default camera settings are:

- Resolution: 800x480
- Pixel Data: RAW10
- Frame Rate: 60 fps
- Bin Mode: 1

The demo batch file includes the following files:

■ 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

- Connect the DE2-115 board (J9) to the host PC with a USB cable and install the USB-Blaster II driver if necessary
- Plug the 12V adapter to DE2-115 Board.
- Power on the DE2-115 board.
- 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 (See [Figure 4-12](#)).
- LED0~1 light up, stand the settings of MIPI-BRIDGE I2C and Camera I2C are completed.
- LED7~9 blink in 1Hz, stand MIPI-PIXEL-CLOCK, MIPI-REF-CLOCK and MT2-PIXEL-CLOCK are generated correctly.
- HEX1~0 decimal number “60” stands Camera capturing frame rate is 60Hz
- Camera capturing image displays on LCD, if the LCD image is fuzzy, please touch the LCD once that will perform the focus operation. Users can also use “ZOOM IN” gesture on the LCD screen (there will be a yellow box on image, see [Figure 4-13](#)), then, touching the LCD screen once, the middle area focus operation will be performed. Using “ZOOM OUT” gesture on the LCD screen will clear the yellow box.
- [Table 4-2](#) summarizes the functional keys and details of each LED status.

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

Component	Function Description
LED[0]	Lights up when MIPI-BRIDGE I2C setting is successful
LED[1]	Lights up when CAMERA I2C setting is successful
LED[7]	Blink in 1Hz (MIPI-PIXEL-CLOCK/50M)
LED[8]	Blink in 1Hz (MIPI-REF-CLOCK/20M)
LED[9]	Blink in 1HZ (LCD-PIXEL-CLOCK/33M)
KEY[0]	SYSTEM RESET
ZOOM IN Gesture	Area Focus Mode
ZOOM Out Gesture	Whole Area Focus Mode
HEX[1:0]	Frames Per Second(FPS)



Figure 4-12 Screen shot of the VEEK-MT2 camera demonstration



Figure 4-13 Area Focus Mode of the VEEK-MT2 camera demonstration



Note: Executing the test.bat under Camera\demo_batch will automatically download the .sof file.

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 4-1**. 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-MT2.

A video source is input through the CMOS sensor on VEEK-MT2 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-MT2 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 with autofocus function. The autofocus function works at first when the demonstration starting up.

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

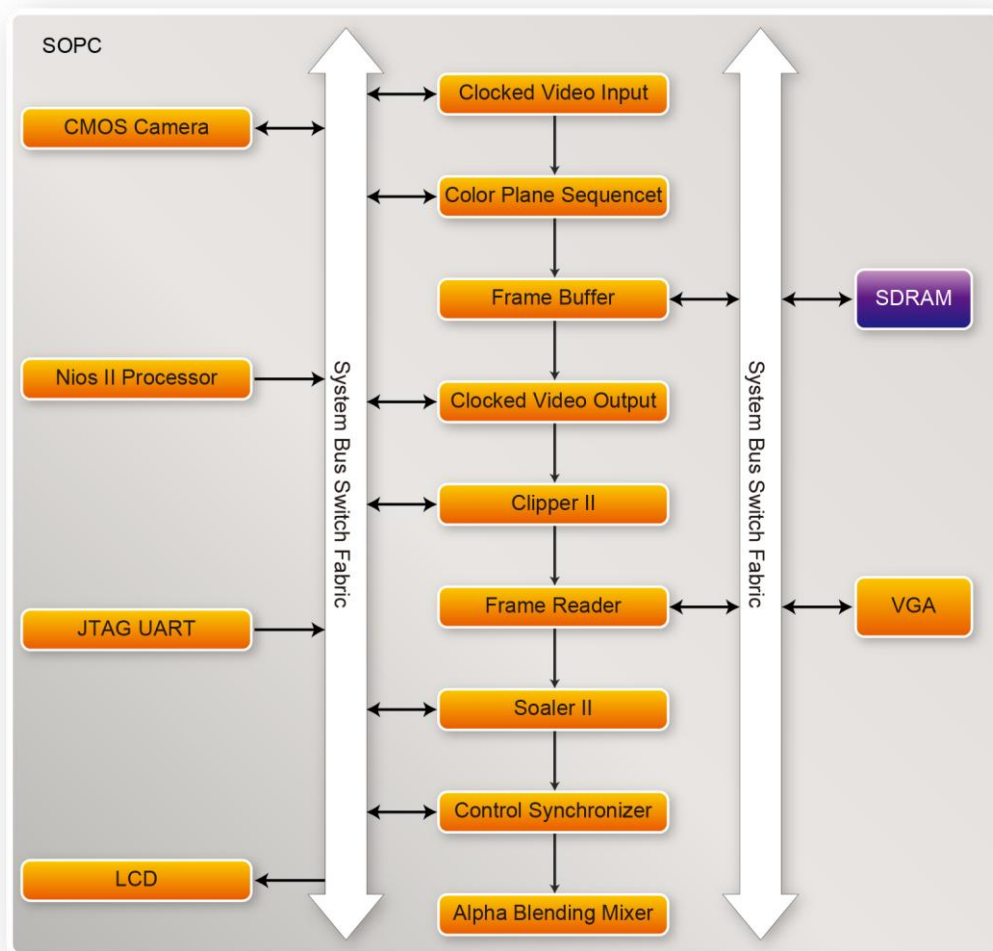


Figure 4-14 VIP Camera Example SOPC Block Diagram (Key Components)

■ 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

- 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 (See Figure 4-15).
- Press and drag the video frame box will result in scaling the playing window to any size.



Figure 4-15 The VIP_Camera demonstration running result



Note: (1).Executing VIP_Camera\demo_batch\VIP_CameraA.bat will download .sof and .elf files.

4.8 Digital Accelerometer Demonstration

This demonstration shows a bubble level implementation based on a digital accelerometer. There are two I2C controllers used in this demonstration. One is used to communicate with MPU9250 to retrieve the gravity information, and another is used to communicate with the APDS-9300 Miniature Ambient Light Photo Sensor. The demonstration using the gravity information retrieved from the MPU9250 to implement a bubble level on the LCD panel. When tilting the VEEK-MT2, the NIOS II program reads the acceleration of gravity from the MPU9250. Based on the gravity information, the NIOS II program can compute the change of angle in the x-axis and y-axis, and shows the angle data in the LCD display. The demonstration also displays two ambient light levels, measured by APDS-9300, on the LCD panel.

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 100MHz clock is used for Nios II processor, SDRAM, SSRAM, and other high speed controllers. The PLL also generate a

20MHz for low-speed peripherals, such as LEDs and buttons. The NIOS II program is running on the SRAM; the SDRAM is used as video frame buffers in the Altera VIP block. The I2C_OpenCore controllers are used to communicate with the MPU9250 chip and the APDS-9300 chips. The VIP block is used to implement video streaming for displaying graphic on the LCD panel. The video steaming is:

Nios II Processor → SDRAM → VIP: Frame Reader → VIP: Clocked Video Output → LCD Panel..

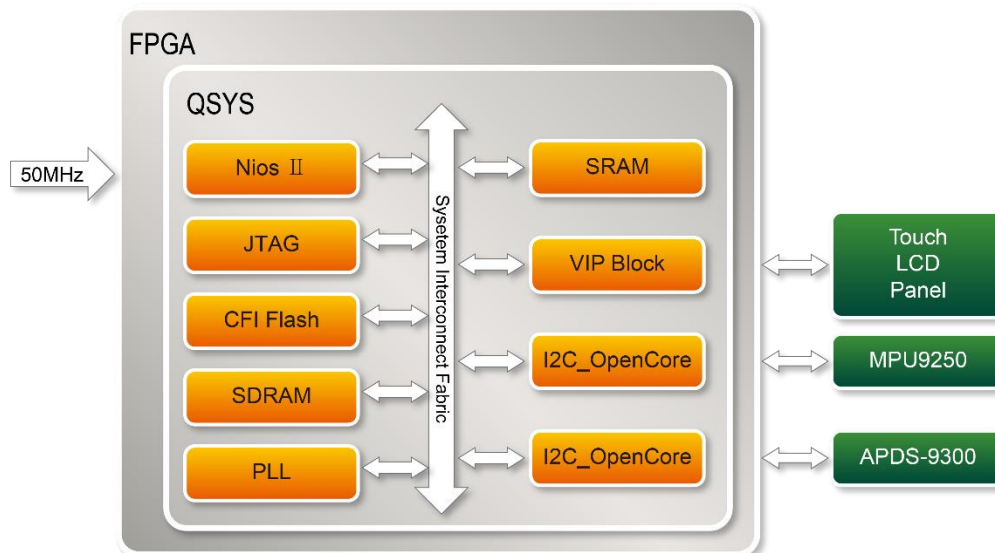


Figure 4-16 Block diagram of the digital accelerometer demonstration

In the NIOS II program, the GUI (graphic user interface) is implemented in the `gui_gsensor.cpp`. The MPU9250 class, implemented in the `MPU9250.cpp/h` is used to retrieve accelerometer information from the MPU9250. The member function **initialize** of the MPU9250 class should be called first before calling any other member function. In this demonstration, the member function **getMotion9** is called to get the gravity value of X/Y/Z. The `light_sensor.cpp/h` includes the library which allows it to communicate with the APDS-9300 Miniature Ambient Light Photo Sensor. Before reading the light level information, call function `Light_Init` first to initialize the sensor APDS-9300. The function **Light_GetID** is designed to get the chip ID of APDS-9300. Function **Light_Get_ADCData0** and **Light_Get_ADCData1** are designed to get the two ADC values in APDS-9300

■ Demonstration Source Code

- Project directory: `G_sensor`
- Bit stream used: `VEEK_MT2.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: VEEK_MT2.sof
- Nios II Program: GUI_APP.elf

■ Demonstration Setup

- Execute the test.bat to configure FPGA and launch the NIOS II program (Note*).
- Tilt the VEEK-MT2 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.

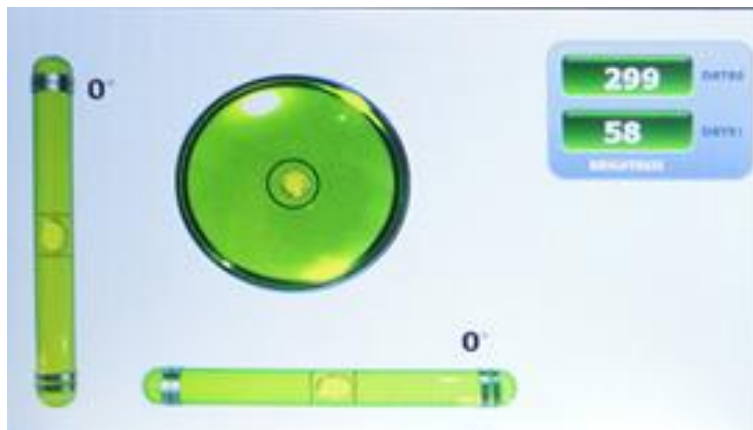


Figure 4-17 Digital Accelerometer demonstration



Note: Executing G_sensor\demo_batch\test.bat to download .sof and .elf files.

4.9 E-Compass Demonstration

This demonstration shows an e-compass on the LCD panel based on information measured by accelerometer and magnetometer in the MPU9250. There are two I²C controllers are used in this demonstration. One is use to communicate with touchscreen chip in the LCD module, and another is used to communicate with the MPU9250 to retrieve the measured gravity and magnetic field information. The compass angel is calculated based on the gravity and magnetic field measured by the MPU9250. The calculated compass angle will be displayed on the LCD panel. A bubble level is also displayed on the LCD panel to indicate the level surface. Note, the magnetometer should be calibrated before the e-compass will work accurately. The calibration process is also included in this demonstration.

Figure 4-18 shows the graphic interface of the e-compass demonstration. The left area show the gravity information measured by accelerometer in the MPU9250. Also, a bubble level is shown on the LCD. The center of LCD panel shows the e-compass. The north direction and magnetic field information are shown on the LCD panel. The Calibration button is used to start the calibration process for the e-compass.



Figure 4-18 Graphic Interface of e-compass demonstration

Figure 4-19 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 100MHz clock is used for Nios II processor, SDRAM, SSRAM, and other high speed controllers. The PLL also generate a 20MHz for low-speed peripherals, such as LEDs and buttons. The NIOS II program is running on the SRAM and the SDRAM is used as video frame buffers in Altera VIP block. The I2C_OpenCore controllers are used to communicate with the touch screen chip and the MPU9250 chip. The VIP block is used to implement video streaming for displaying graphic on the LCD panel. The video steaming is:

Nios II Processor → SDRAM → VIP: Frame Reader → VIP: Clocked Video Output → LCD Panel.

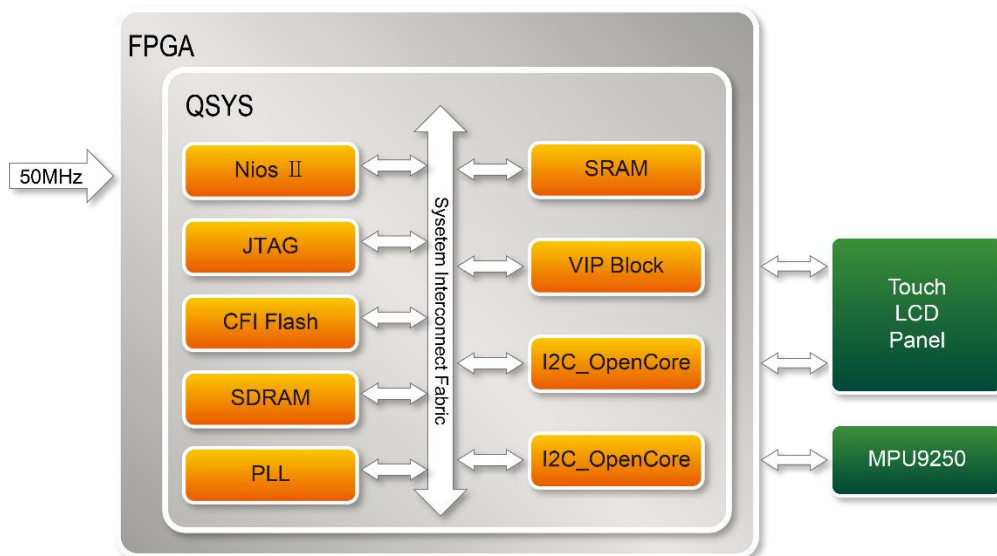


Figure 4-19 Block diagram of the e-compass demonstration

In the NIOS II program, the graphic interface is implemented in the `gui_gsensor.cpp`. The MPU9250 class (implemented in `MPU9250.cpp.h`) is used to retrieve accelerometer information from the MPU9250. The member function `initialize` should be called before calling any other member functions. In this demonstration, the member function `getMotion9` is called to get the gravity value of X/Y/Z. The function `Light_Init` is designed to initialize the sensor APDS-9300. The function `Light_GetID` is designed to get the chip ID of APDS-9300. The Function `Light_Get_ADCData0` and `Light_Get_ADCData1` are designed to get the two ADC values in APDS-9300.

The section introduces the method to calculate the heading angle based on the values measured by accelerometer and magnetometer in this demonstration. If the compass is on a flat surface, then heading angle can be calculated with the following formula:

$$\arctan(X_h/Y_h)$$

where X_h and Y_h are the magnetic field in X and Y directions. If the compass is not on a flat surface, the X_h and Y_h can be calculated with the following formula:

$$X_h = m_x * \cos(\text{pitch}) + m_y * \sin(\text{roll}) * \sin(\text{pitch}) - m_z * \sin(\text{pitch}) * \cos(\text{roll})$$

$$Y_h = m_y * \cos(\text{roll}) + m_z * \sin(\text{roll})$$

Where $m_x/m_y/m_z$ are the measured values by the magnetometer in the MPU9250. The pitch and roll can be calculated by the gravity values measured by the accelerometer in the MPU9250.

The measurement of magnetic field will be subjected to distortion. There are two categories of these distortions: the hard iron distortions and the soft iron distortions. The hard iron errors refer to the presence of magnetic fields around the sensor (magnets, power supply wires) and are related to the measurement offset errors, while the soft iron errors refer to the presence of ferromagnetic materials around the sensor, which skew the density of the Earth's magnetic field locally and are related to scaling offset errors. In other words, to get the correct magnetometer data you should calibrate it first. In this demonstration, we corrected the magnetic field with the following formula:

$$X_{\text{calibrated}} = X - X_{\text{offset}}$$

$$Y_{\text{calibrated}} = Y - Y_{\text{offset}}$$

$$Z_{\text{calibrated}} = Z - Z_{\text{offset}}$$

where X_{offset} , Y_{offset} and Z_{offset} are the average value of X/Y/Z magnetic field, defined as:

$$X_{\text{offset}} = (X_{\text{max}} - X_{\text{min}})/2$$

$$Y_{\text{offset}} = (Y_{\text{max}} - Y_{\text{min}})/2$$

$$Z_{\text{offset}} = (Z_{\text{max}} - Z_{\text{min}})/2$$

where $X_{\text{max}}/X_{\text{min}}/Y_{\text{max}}/Y_{\text{min}}/Z_{\text{max}}/Z_{\text{min}}$ are the maximal and minimal value of X/Y/Z values reported by magnetometer in the MPU9250 while the VEEK-MT2 is rotated in its X/Y/Z axes as shown in Figure 4-20.

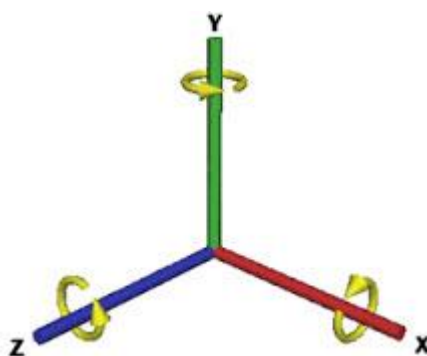


Figure 4-20 Rotate VEEK-MT2 around each of its 3 axes

■ Demonstration Source Code

- Project directory: E_Compass
- Bit stream used: VEEK_MT2.sof
- Nios II Workspace: E_Compass\software

■ Demonstration Batch File

Demo Batch File Folder: E_Compass\demo_batch

The demo batch file includes the following files:

- Batch File: test.bat, test.sh
- FPGA Configure File: VEEK_MT2.sof
- Nios II Program: GUI_APP.elf

■ Demonstration Setup

- Execute the test.bat to configure FPGA and launch the NIOS II program (Note*).
- The E-Compass will be displayed on the LCD panel.
- Touch the Calibration button to start the calibration process. When the button is touched, the LCD content will be changed as shown in **Figure 4-21**.
- Rotate VEEK-MT2 in X/Y/Z axis individually. Touch the Finished button when rotation is done.
- When the Finished button is touched, the LCD content is changed as shown in **Figure 4-22**. The minimal, maximal, and average values are displayed in the LCD. If the value is valid, please touch Yes button to apply the calibration result.
- Keep the VEEK-MT2 on as flat a surface as possible by watching the bubble level on the LCD. The e-compass will show the accuracy North direction on the LCD.

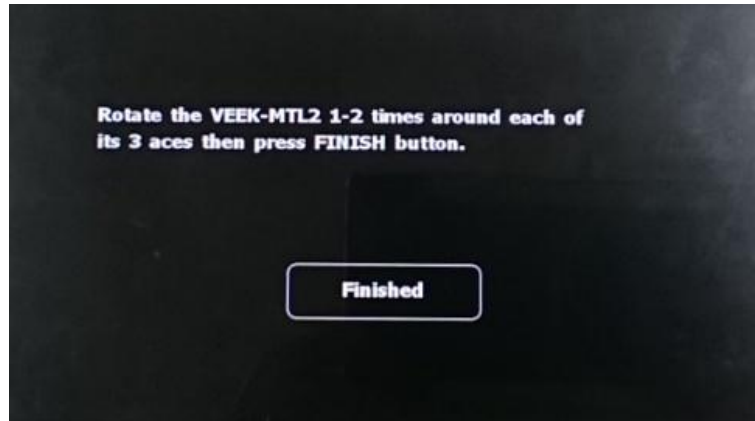


Figure 4-21 Hint for rotate VEEK-MT2

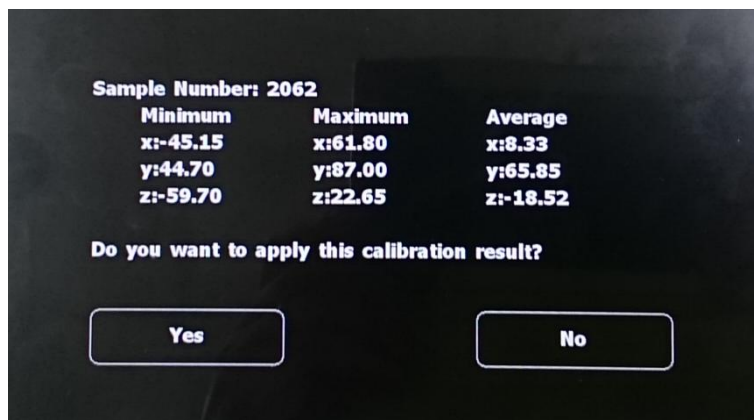


Figure 4-22 Magnetic Calibration Information



Note:

Execute `E_Compass \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-MT2 System CD under the **Factory_Recovery** folder.

5.2 Running the Application Selector

- Connect power to the VEEK-MT2
- Insert the SD card with applications into the SD card socket of VEEK-MT2
- Switch on the power (SW18)
- Scroll to select the demonstration to load using the side-bar
- Tap on the Load button to load and run a demonstration



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 4GB. 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 --input="your example.sof" --output="your example_HW.flash" --epcs --verbose`
- Convert .elf file into .flash file
- `elf2flash --base=0x08800000 --end=0x08ffffff --reset=0x08800000 --input=" your example.elf" --output=" your example SW.flash" --boot=boot_loader_cfi.srec`
- 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 "`
- 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\- Place the SD card in the VEEK-MT2, 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-MT2. 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-MT2, 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.

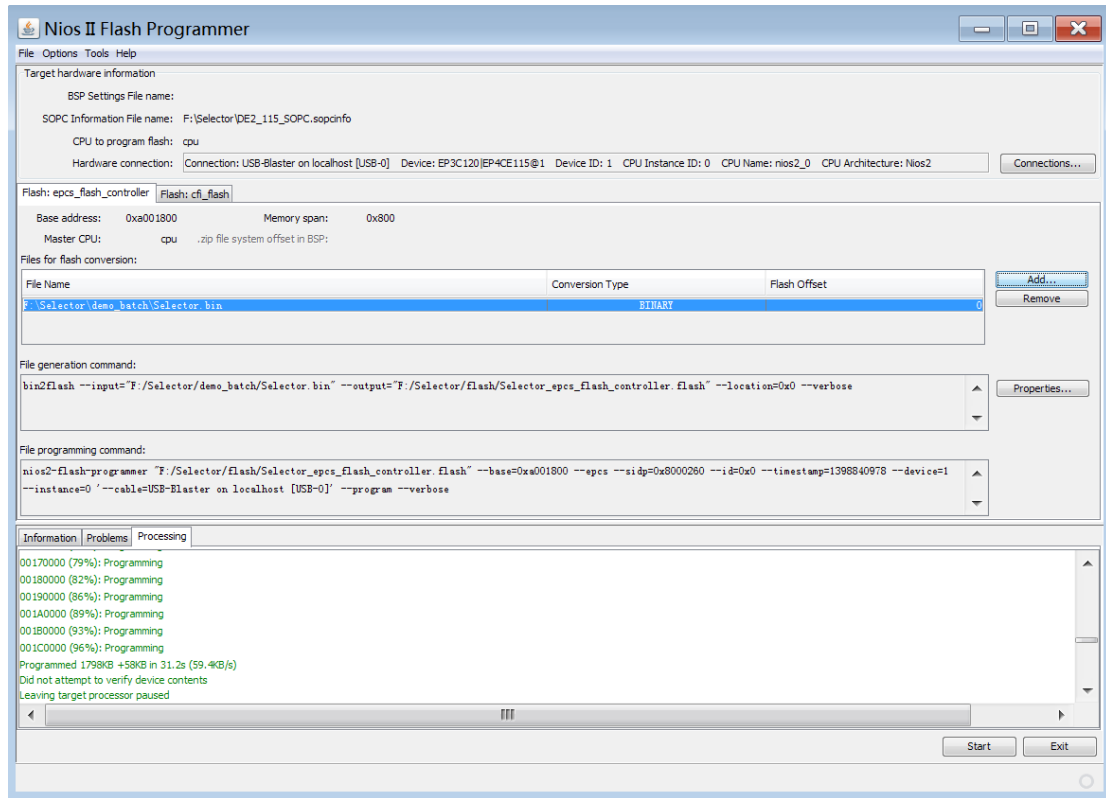


Figure 5-2 Programming Flash settings