Spartan-3A DSP FPGA Video Starter Kit

User Guide

UG456 (v2.1) March 15, 2010

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/29/07	1.0	Initial Xilinx release.
02/08/08	1.1	Various edits made throughout the text and modifications to Figure 1-1, Figure 1-2, Figure 1-4, Figure 1-6, Figure 3-3, Figure 3-5, and Figure 3-8 in coordination with the Spartan-3A DSP FPGA Video Starter Kit update release.
11/17/08	2.0	Updated for Video Starter Kit version 2.0 release.
03/15/10	2.1	Various edits made throughout the text; updated software version numbers; added “Downloading a Bitstream” to Appendix C, “Creating a Compact Flash for the Demo.”

Table of Contents

Appendix B: IP Repository

List of Figures

List of Tables

About This Guide

This user guide provides information and guidelines for using the Spartan®-3A DSP FPGA Video Starter Kit (VSK), a development platform consisting of the Spartan-3A DSP 3400A Development Platform, the FMC-Video daughter card, and a VGA camera.

Guide Contents

This manual contains the following chapters:

- Chapter 1, "Introduction," describes the contents of the Video Starter Kit and the system requirements.
- Chapter 2, "System Setup," contains instructions for setting up the hardware and software.
- Chapter 3, "Example Applications," describes the base platform, DVI pass-through, DVI frame buffer, Camera frame buffer, and S-Video frame buffer.
- Appendix A, "Development Environment," provides details about the hardware and software necessary to modify and rebuild the reference designs included in this kit.
- Appendix B, "IP Repository," describes in detail the IP blocks and provides an API Reference.
- Appendix C, "Creating a Compact Flash for the Demo," provides the step necessary to copy the demo onto a new Compact Flash card.
- Appendix D, "Troubleshooting," provides the diagnostic procedure to help isolate the likely cause if the demo fails to run as expected.

References

1. XtremeDSP Solution FMC-Video Daughter Board Reference Guide (UG458)
2. DVI Standard: www.ddwg.org/lib/dvi_10.pdf
3. MPMC Data Sheet (DS643)
4. Spartan-3A DSP 3400A Development Board Technical Reference Guide
5. Spartan-3A DSP FPGA Family: Data Sheet (DS610)
6. Spartan-3A DSP FPGA VSK Software Guide (UG514)

Acronyms

The following is a list of the acronyms used in this document.

BMP	Windows bitmap image file format
BSB	Base System Builder, part of the EDK
CD	Compact Disc
CF	Compact Flash card
CMOS	Complementary Metal Oxide Semiconductor
DDR2	Double Data Rate SDRAM
DSP	Digital Signal Processing or Digital Signal Processor
DVD	Digital Versatile Disc or Digital Video Disc
DVI	Digital Video Interface
ECC	Error Correcting Code
EDK	Xilinx Embedded Development Kit
FIR	Finite Impulse Response
FMC	FPGA Mezzanine Card Standard (VITA-57.1)
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input/Output
I^2C	Inter-Integrated Circuit
IIC	See I^2C
LED	Light Emitting Diode
MDM	MicroBlaze™ Debug Module
MPMC	The Xilinx Multi-Port Memory Controller IP core, part of the EDK. A memory controller that receives data from multiple sources and sends data to multiple destinations.
NPI	Native Port Interface indigenous to the MPMC
PCORE	EDK nomenclature for a MicroBlaze Peripheral Core
PIM	Port Interface Module indigenous to the MPMC
PM	Performance Monitor
SDRAM	Synchronous Dynamic Random Access Memory
SDTV	Standard Definition Television
SODIMM	Small Outline Dual In-Line Memory Module
UART	Universal Asynchronous Receiver Transmitter
VFBC	Video Frame Buffer Controller
VITA	VMEbus International Trade Association
VGA	Video Graphics Array
VSK	Spartan-3A DSP FPGA Video Starter Kit
XPS	Xilinx Platform Studio

Additional Resources

To find additional documentation, see the Xilinx website at:

www.xilinx.com/support/documentation/index.htm.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

www.xilinx.com/support/mysupport.htm.

Conventions

This document uses the following conventions. An example illustrates each convention.

Typographical

The following typographical conventions are used in this document:

Convention	Meaning or Use	Example
`Courier font`	Messages, prompts, and program files that the system displays	`speed grade: - 100`
`Courier bold`	Literal commands that you enter in a syntactical statement	**`ngdbuild`** *design_name*
Helvetica bold	Commands that you select from a menu	**File →Open**
	Keyboard shortcuts	**Ctrl+C**
Italic font	Variables in a syntax statement for which you must supply values	**`ngdbuild`** *design_name*
	References to other manuals	See the *User Guide* for more information.
	Emphasis in text	If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.
Dark Shading	Items that are not supported or reserved	This feature is not supported
Square brackets [ ]	An optional entry or parameter. However, in bus specifications, such as **`bus[7:0]`**, they are required.	**`ngdbuild`** `[`*`option_name`*`]` *`design_name`*
Braces { }	A list of items from which you must choose one or more	**`lowpwr`** `={`**`on`**`|`**`off`**`}`
Vertical bar |	Separates items in a list of choices	**`lowpwr`** `={`**`on`**`|`**`off`**`}`
Angle brackets < >	User-defined variable or in code samples	<directory name>

Convention	Meaning or Use	Example

Vertical ellipsis
.
.
.
	Repetitive material that has been omitted	`IOB #1: Name = QOUT'`
		`IOB #2: Name = CLKIN'`
		.
		.
		.

Horizontal ellipsis . . .	Repetitive material that has been omitted	**`allow block`** *block_name loc1 loc2 ... locn;*
Notations	The prefix '0x' or the suffix 'h' indicate hexadecimal notation	A read of address 0x00112975 returned 45524943h.
	An '_n' means the signal is active low	**`usr_teof_n`** is active low.

Online Document

The following conventions are used in this document:

Convention	Meaning or Use	Example
Blue text	Cross-reference link to a location in the current document	See the section "Additional Resources" for details. Refer to "Title Formats" in Chapter 1 for details.
Blue, underlined text	Hyperlink to a website (URL)	Go to www.xilinx.com for the latest speed files.

Chapter 1

Introduction

The Xilinx Spartan®-3A DSP FPGA Video Starter Kit (VSK) is a development platform consisting of the Spartan-3A DSP 3400A Development Platform, the FMC-Video daughter card, and a VGA camera. This platform provides a development environment that allows you to quickly begin to experiment with video processing using the Spartan-3A DSP family of FPGAs.

The Spartan-3A DSP 3400A Development Platform is built around a Spartan-3A DSP XC3SD3400A device that provides significant resources (for example, 126 embedded DSP blocks) for implementing high performance video processing systems and co-processors.

The FMC-Video daughter card is an add-on card that augments the video capabilities of the Spartan-3A DSP 3400A Development Platform. The FMC-Video daughter card supports the following video interfaces:

- DVI-I Input, both Digital and Analog
- Composite Input
- S-Video Input
- 2 Camera Inputs
- Composite Output
- S-Video Output

The VSK also includes a VGA camera based upon the Micron MT9V022 image sensor.

To illustrate some of the platform capabilities and to provide a convenient starting point to develop custom video processing systems, the VSK also includes several instructive demonstration and reference designs and fully functional evaluation software for programming the platform.

System Requirements

Hardware

- DVI video source
- S-Video source
- DVI display
- S-Video display
- Host computer

Note: DVI ports on the VSK support analog and digital connectivity. Analog VGA equipment can be converted to DVI by use of an adapter.

Software

- Terminal emulator

Kit Contents

The Spartan-3A DSP FPGA Video Starter Kit is comprised of a variety of hardware and software components. The design software, reference designs, and documentation are provided on CD/DVD media included with the kit. Each of these components is described in the following sections.

Hardware

The kit contains the following hardware:

- 1 – Spartan-3A DSP 3400A Development Platform
- 1 – FMC-Video Daughter Card
- 1 – VGA Camera Module
- 1 – Power Adapter with locale specific connector
- 1 – S-Video Cable
- 1 – RCA composite Cable
- 1 – CAT 5 Ethernet Crossover Cable
- 1 – CAT 6 Ethernet Patch Cable
- 1 – DVI Cable
- 1 – VGA Cable
- 1 – Compact Flash Card with boot image loaded
- 1 – Null Modem Cable
- 1 – Analog VGA to DVI Adaptor

Media

The kit contains the following media:

- Xilinx ISE® Design Suite 11.1 DVD (includes ISE Design Suite 60-day evaluation, as well as full versions of EDK and System Generator).

Note: Update 4 should be downloaded separately from the Xilinx Website.

• VSK Resources CD

The VSK Resources CD included with the kit contains the following:

• Documentation:

  - ♦ Spartan 3A-DSP FPGA VSK Quick Start Guide (UG455)
  - ♦ Spartan-3A DSP FPGA VSK User Guide (UG456)
  - ♦ Spartan-3A DSP FPGA VSK Software Guide (UG514)
  - ♦ Spartan-3A DSP FPGA Family: Data Sheet (DS610)

- Hardware:
  - ♦ Schematics/Gerbers/Specifications
  - ♦ Spartan-3A DSP 3400A Development Platform

- Spartan-3A DSP 3400A Development Board Technical Reference Guide
- ♦ FMC-Video daughter card
    - XtremeDSP™ Solution FMC-Video Daughter Board Technical Reference Guide (UG458)
- ♦ VGA Camera Module
- Reference Designs
    - ♦ Compact Flash Image
    - ♦ EDK
        - DVI Pass-Through
        - DVI Frame Buffer
        - Camera Frame Buffer
        - S-Video Frame Buffer
    - ♦ HDL
        - Pass-Through Test
        - Loopback Test

Hardware Overview

This section introduces the hardware included in VSK. The VSK hardware includes three separate boards (see Figure 1-1):

- Spartan-3A DSP 3400A Development Platform
- FMC-Video Daughter Card
- VGA Camera Module

Details about these boards are provided in the following sections.

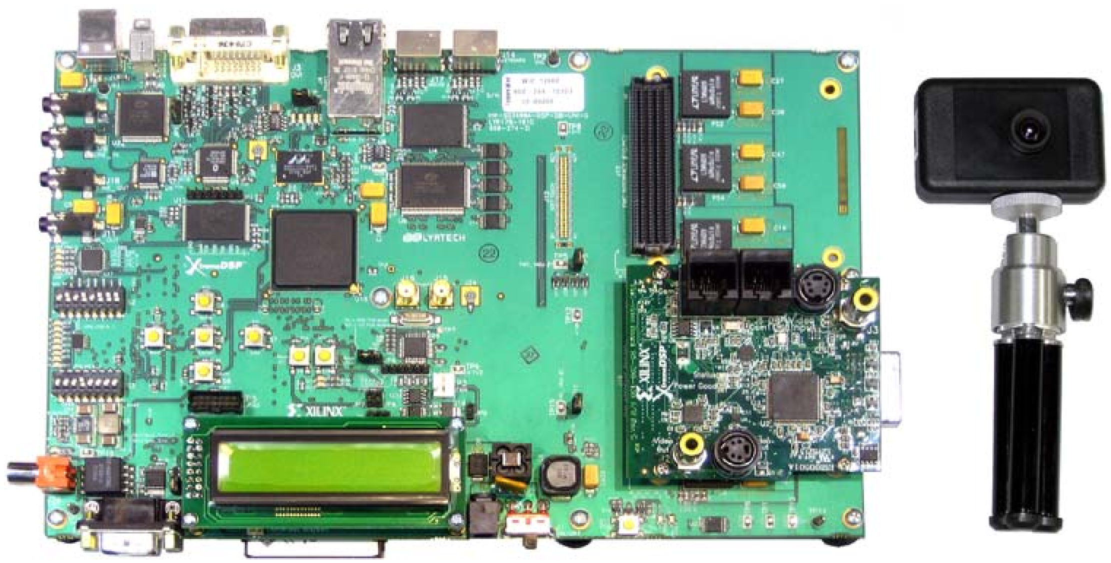


Figure 1-1: **Spartan-3A DSP FPGA 3400A Development Platform, FMC-Video, and Camera**

Spartan-3A DSP 3400A Development Platform

The carrier board is shown in Figure 1-2.

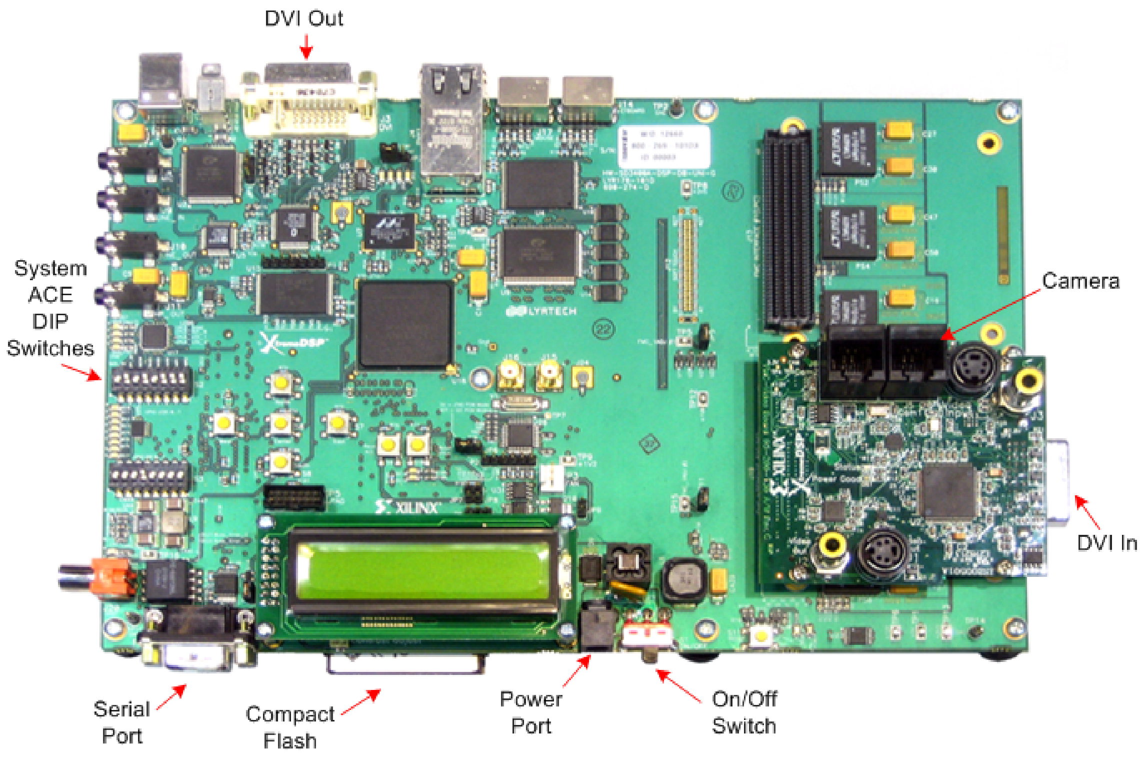


Figure 1-2: **Spartan-3A DSP 3400A Development Platform with FMC-Video (Lower Right)**

The components on this board that are of primary interest for VSK are as follows:

- Spartan-3A DSP FPGA
- DDR2 SDRAM (SODIMM)
- DB9 (RS232) connector
- Video encoder
- DVI connector
- System ACE™ controller
- Compact Flash
- FMC expansion module (see "FMC-Video")

Only the colored blocks in Figure 1-3 are used by the designs included in this kit.

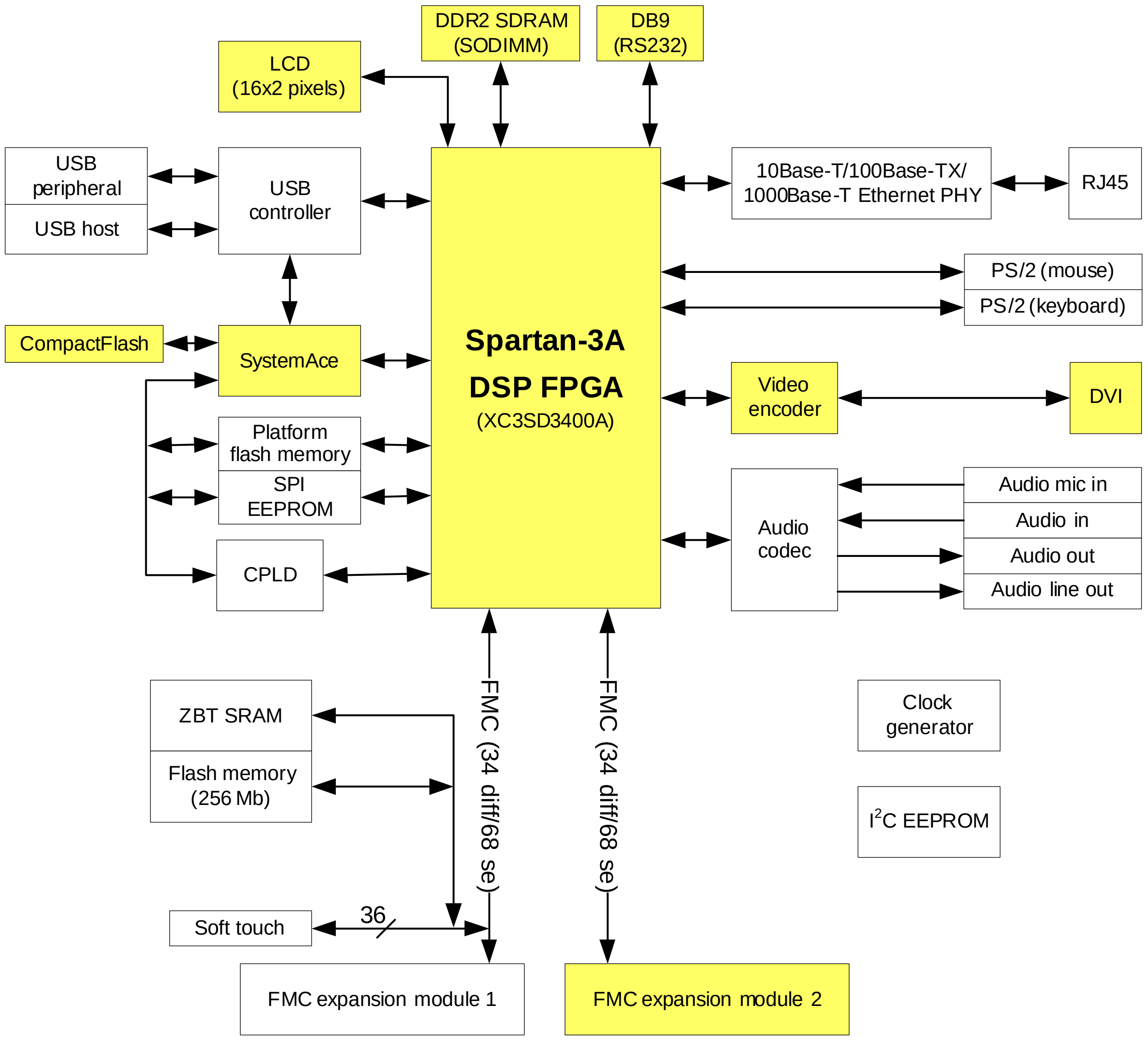


Figure 1-3: **Spartan-3A DSP 3400A Development Platform Block Diagram**

For more information about the carrier board, see the Spartan-3A DSP 3400A Development Board Technical Reference Guide [Ref 4].

FMC-Video

See Figure 1-4 for top and bottom views of the FMC-Video.

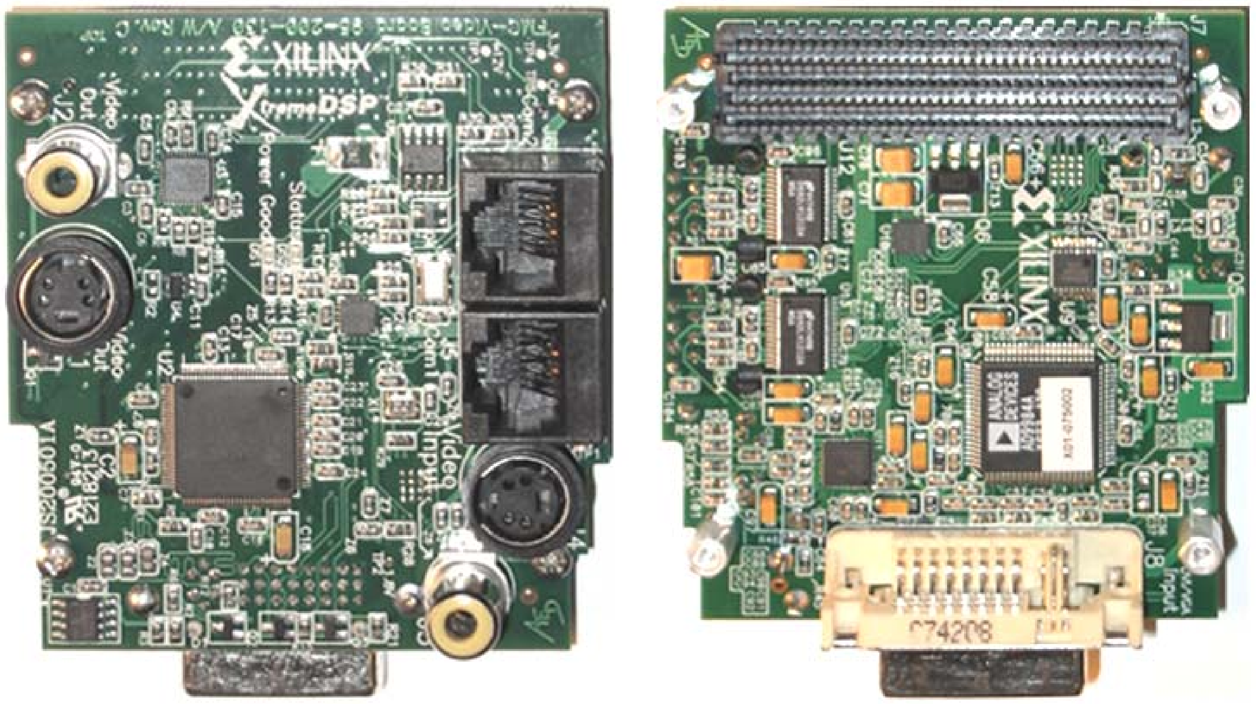


Figure 1-4: **FMC-Video Top (Left), Bottom (Right)**

The FMC-Video adds a number of video interfaces to the Spartan-3A DSP 3400A Development Platform:

- A DVI connector supports both analog and digital video data input
- Composite and S-Video SDTV input and output are supported via standard RCA and DIN-4 connectors, respectively
- Two 8-pin modular RJ45 connectors are included to interface to up to two cameras (see the XtremeDSP Solution FMC-Video Daughter Board Reference Guide (UG458) [Ref 1] for more information).

The board is designed to the VITA-57.1 FMC specification, which includes a connector for power, control, and data. A block diagram of this board can be seen in Figure 1-5.

> ***Warning!*** **The RJ45 connectors on FMC-Video are not Ethernet ports. Connecting anything other than the camera supplied with VSK to these ports could result in damage to your equipment.**

For more details about FMC-Video, see the XtremeDSP Solution FMC-Video Daughter Board Reference Guide (UG458) [Ref 1].

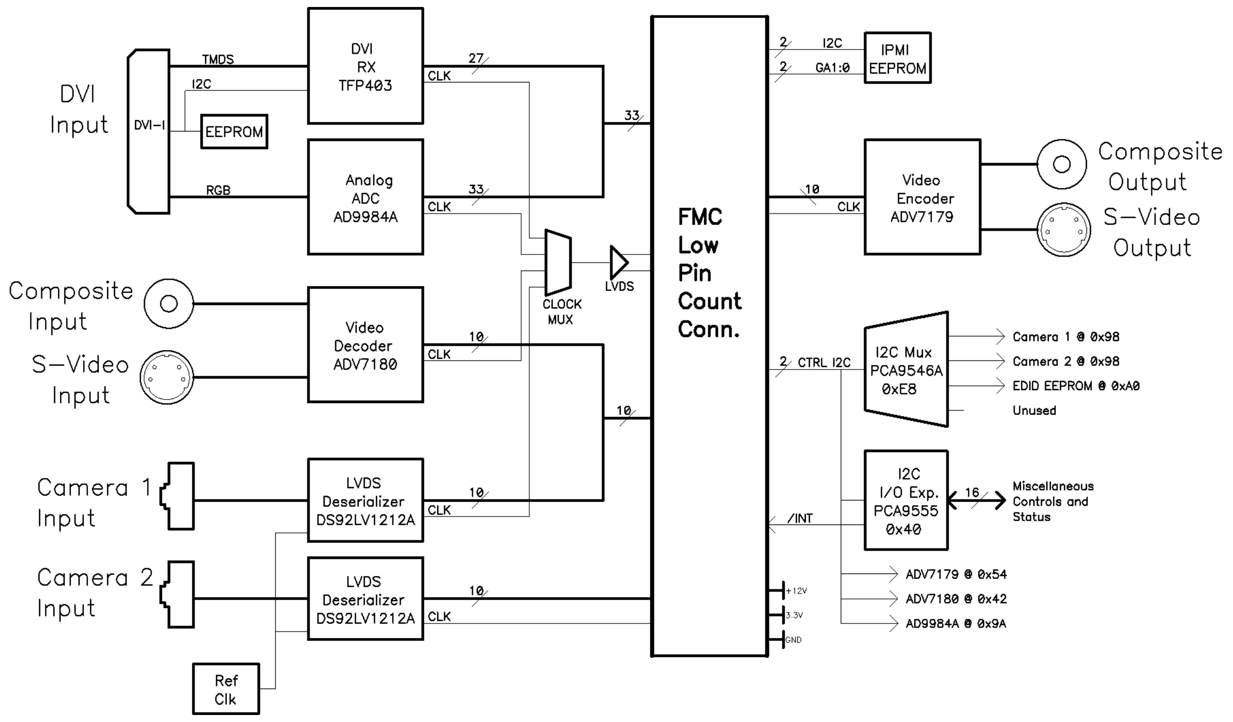


Figure 1-5: **FMC-Video Block Diagram**

See Figure 1-6 for the FMC-Video port diagram. The FMC-Video port descriptions are:

1. J2 – Composite Output
2. J1 – S-Video Output
3. DS1 & DS2 – Power Good and Status LEDs
4. J6 – Camera 2 Input (RJ45)
5. J5 – Camera 1 Input (RJ45)
6. J4 – S-Video Input
7. J3 – Composite Input
8. J7 – FMC Connector
9. J8 – DVI Input

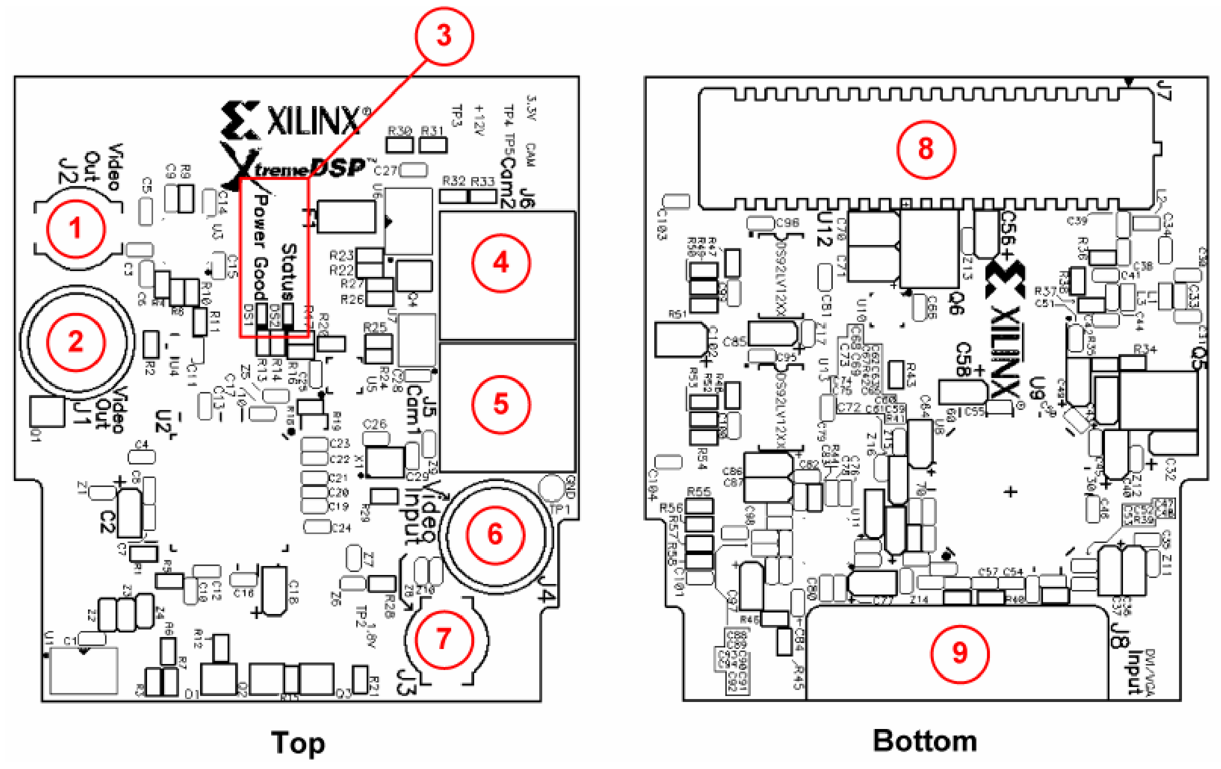


Figure 1-6: **FMC-Video Port Diagram**

VGA Camera Module

The camera provided with VSK is based on a Micron MT9V022 CMOS image sensor. See Figure 1-7.

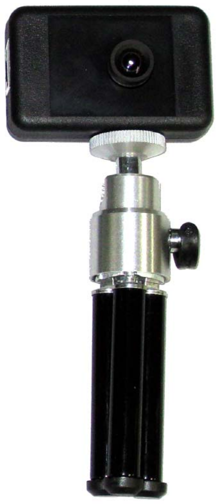


Figure 1-7: **VGA Camera Module**

The image sensor is interfaced to the FMC-Video via an RJ45 connector with a proprietary pin assignment. A standard CAT6 Ethernet cable can be used to connect the camera to FMC-Video. Figure 1-8 shows the basic connectivity of the camera components.

Warning! **The RJ45 connector on the camera is not an Ethernet port. Connecting the camera to anything other than FMC-Video could result in damage to your equipment.**

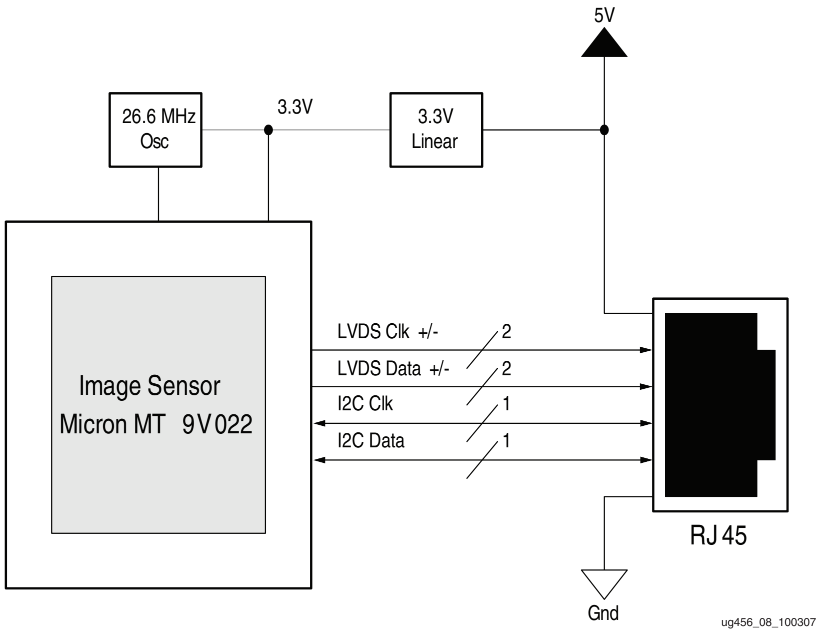


Figure 1-8: **VGA Camera Module Block Diagram**

More information about this camera can be found in the MT9V022 product brief available from Micron Technology Inc.

Note: A full data sheet requires a signed NDA with Micron.

Chapter 2

System Setup

Before Starting

1. Please read through the full procedure carefully before beginning.
2. Always use proper static handling precautions when working with hardware boards.
3. If your FMC-Video card did not come installed on your Spartan®-3A DSP 3400A Development Platform, refer to the documentation that came with it.

FPGA Configuration

1. Verify that the Spartan-3A DSP 3400A Development Platform boot mode DIP switch (S2) is set to 01010111 (arranged switch 1-8, 0=off, 1=on). This allows a bootloader program to be loaded from the Compact Flash card by the System Ace™ controller.
2. With the Spartan-3A DSP 3400A Development Platform power switch (S1) set to OFF, insert the Compact Flash memory card into the Compact Flash slot (P7) on the board.
3. Remove the jumper from JP6 on the Spartan-3A DSP FPGA Development Board. This allows the on-board PLL to be programmed by the I2C bus.

Connect the VGA Camera

1. Connect the VGA Camera to the FMC-Video card using the supplied CAT6 cable; plug one end into the RJ45 connector (J5) on the FMC-Video card.
2. Plug the other end of the CAT6 cable into the RJ45 connector on the VGA Camera.

   Warning! **The FMC-Video RJ45 connectors are not compatible with a standard Ethernet port, and are compatible ONLY with the VGA camera. Damage to your equipment could result if this port is connected to anything other than the provided camera with the provided CAT 6 cable.**

Connect the Video Source (DVI/VGA)]

1. Connect a DVI cable to the DVI output port of the video source.
2. Connect the DVI cable to the DVI input connector (J8) on the FMC-Video.

Note: A VGA source may also be used by connecting the VGA cable to a VGA-to-DVI adapter that is then plugged into the DVI input connector (J8) on the FMC-Video.

Connect the Video Source (S-Video)

1. Connect an S-Video cable to the S-Video output port of the video source.
2. Connect the S-Video cable to the S-Video input connector (J4) on the FMC-Video.

Connect the Video Display (DVI/VGA)

1. Connect a DVI cable to the DVI input port of the video display.
2. Connect the DVI cable to the DVI output connector (J3) on the Spartan-3A DSP 3400A Development Platform.

Note: A VGA monitor may also be used by connecting the VGA cable to a VGA-to-DVI adapter which is then plugged into the DVI output connector (J3) on the Spartan-3A DSP 3400A Development Platform.

Connect the Video Display (S-Video)

1. Connect an S-Video cable to the S-Video input port of the video display.
2. Connect the S-Video cable to the S-Video output connector (J1) on the FMC-Video.

Connect the PC

1. Connect a PC to the Spartan-3A DSP 3400A Development Platform through the PC's serial port using a 9-pin RS232 serial Null Modem cable.
2. Connect one end of the cable to the RS232 Port (J17) on the Spartan-3A DSP 3400A Development Platform.
3. Connect the other end of the cable to the serial port of the PC.
4. Configure the HyperTerminal (or equivalent terminal program) on the PC with the following settings:
   a. 9600 baud
   b. 8-bit data
   c. No parity
   d. 1 stop bit
   e. No flow control

Power Up the Hardware

1. With the power switch (S1) set to OFF, connect the provided power supply to the power connector (J7) on the Spartan-3A DSP 3400A Development Platform.
2. Plug the other end of the power supply into a wall outlet.
3. Turn the power switch (S1) to ON. The Power Good LED (DS1) on the Spartan-3A DSP 3400A Development Platform should be lit.

Using the Bootloader

The Compact Flash (CF) card included with the VSK contains configuration bit files for a bootloader and for executing the video demos designs. When power is applied to the VSK, the bootloader hardware and software are automatically loaded from the CF card. The bootloader provides a menu system that allows you to select a video demo to run. After the bootloader is active, a menu is displayed in multiple places: the HyperTerminal, the DVI display, and the LCD character display on the board. You select a demo to run by providing input on the HyperTerminal or by using push buttons on the Spartan-3A DSP 3400A Development Platform.

1. Use the North (S4) and South (S8) push buttons on the Spartan-3A DSP 3400A Development Platform to scroll through the demo selections. You should see this displayed on both the video display and the LCD display (DS21) on the Spartan-3A DSP 3400A Development Platform.
2. Use the Center (S6) push button to load the demo. To reload the bootloader, press the Spartan-3A DSP 3400A Development Platform push button labeled Reset ACE (S9).

The bootloader is loaded by the System ACE chip on the Spartan-3A DSP 3400A Development Platform. Other designs can be automatically loaded by changing switches 6-8 on the boot mode DIP switch (S2). Table 2-1 defines the designs that can be directly loaded from the VSK Compact Flash card included with the VSK.

Table 2-1: **System ACE Configuration Setting**

System ACE CFG ADDR	DIP Switch Settings (S2-8, 7, 6)	Description
000	111	Bootloader (Application chooser)
001	110	DVI Pass-through demo
010	101	DVI Frame Buffer demo
011	100	Camera Frame Buffer demo
100	011	System Generator Bootloader
101	010	Reserved (for System Generator user design *after* System Generator is run)
110	001	S-Video Frame Buffer demo
111	000	Loopback test

Note: Loopback test is documented in the FMC-Video Technical Reference Manual (UG458) [Ref 1].

Running the Demonstrations

The demos are run by interacting with the VSK through a HyperTerminal program that has been configured as described previously in "Connect the PC." A series of menus allows you to exercise all of the capabilities of each demo design. Each menu is described in Chapter 3, "Example Applications," of this guide.)

Chapter 3

Example Applications

Base Platform

Overview

Each of the EDK reference designs included with the VSK is built from the base platform shown in Figure 3-1. The Base Platform is not a separate design that is delivered with this kit, but rather it is the starting point from which all the other designs were built. The information provided in this section can be applied to the designs discussed in the following sections.

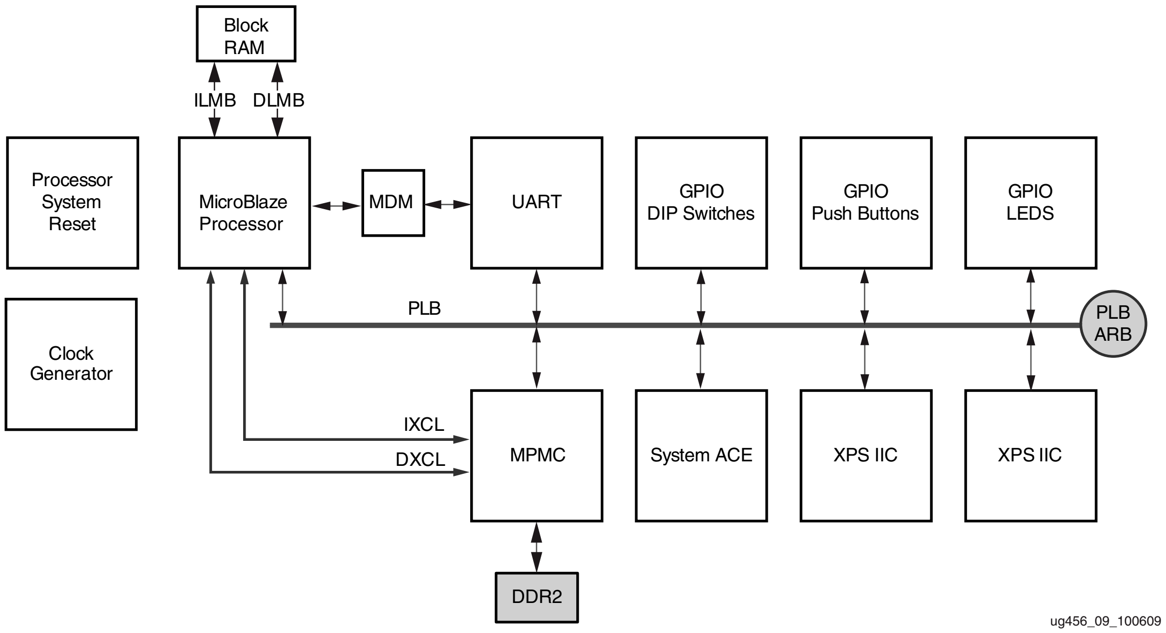


Figure 3-1: **Base Platform Block Diagram**

The base platform includes the following Processor IP blocks:

- MicroBlaze™ 32-bit Soft Microprocessor
- Local Memory Bus (LMB)
- LMB Block RAM Controller
- Block RAM Block Memory
- Processor Local Bus (PLB)
- XPS Uartlite

- Xilinx Platform Studio (XPS) General Purpose Input/Output (GPIO)
- XPS Inter-Integrated Circuit (IIC) Controller
- XPS System ACE™ Compact Flash Controller
- External Multi-port Memory Controller (MPMC)
- MDM MicroBlaze Debug Module
- Clock Generator
- Processor System Reset

The base platform is built around the MicroBlaze soft microprocessor. Instructions and data can be stored in the local block RAM or in the external DDR2 memory attached to the MPMC memory controller.

Table 3-1: **Base Platform: Memory Map**

IP Core	Version	Memory Address		Instances	Bus
		Low	High		
MicroBlaze Processor	7.20.d	-	-	microblaze_0	LMB, PLB, XCL
LMB BRAM IF Controller	2.10.b	0x00000000	0x0000FFFF	dlmb_cntlr, ilmb_cntlr	LMB
MPMC DDR2 Controller	5.04.a	0x10000000	0x1FFFFFFF	mpmc_0	XCL
xps_gpio	2.00.a	0x81400000	0x8140FFFF	DIP_Switches_8Bit	PLB
xps_gpio	2.00.a	0x81420000	0x8142FFFF	LEDs_8Bit	PLB
xps_gpio	2.00.a	0x81440000	0x8144FFFF	Push_Buttons_Position	PLB
xps_iic	2.02.a	0x81600000	0x8160FFFF	xps_iic_0	PLB
xps_iic	2.02.a	0x81620000	0x8162FFFF	xps_iic_1	PLB
xsp_sysace	1.01.a	0x83600000	0x8360FFFF	SysACE_CompactFlash	PLB
xps_uartlite	1.01.a	0x84000000	0x8400FFFF	RS232_Uart	PLB
mdm	1.00.g	0x84400000	0x8440FFFF	debug_module	PLB

Directory Structure

The files for the EDK-based demo designs can be found on the supplied CD in the Demonstrations\EDK directory. Since all of the demos use a shared design repository, all of the EDK demo designs have been included in a single ZIP file. Information about the HDL demos can be found in the *FMC-Video Technical Reference Guide* (UG458). Each of the EDK demos has its own directory with the following structure:

```
\data
\etc
\pcores
\<design_name>_Sw
system.mss
system.mhs
system.xmp
download.bit
```

A description of each directory follows.

`data`	Contains the user constraints file (UCF). For more information on this file and how to use it, see the ISE® UCF help topics at: www.xilinx.com/support/software_manuals.htm.
`executable.elf`	Software to run on the MicroBlaze™ in executable and linkable format.
`etc`	Contains files that specify the project EDK build properties.
`pcores`	Used for including custom hardware peripherals.
`<design_name>_Sw`	Contains the application software specific to the demo.
`system.xmp`	This is the top-level project design file. XPS reads this file and graphically displays its contents in the XPS user interface.
`system.mhs`	This is the system microprocessor hardware specification, or MHS file. It captures textually the system elements, their parameters, and connectivity. The MHS file is the hardware foundation for your project.
`system.mss`	This is the system microprocessor software specification, or MSS file. It captures the software portion of the design, describing textually the system elements and various software parameters associated with the peripheral. The MSS file is the software foundation for your project.
`download.bit`	This is the hardware configuration bit file. This file contains the configuration for the hardware design.

Note: More information about the EDK directory structure can be found in the EDK Documentation.

DVI Pass-Through Demo

Overview

This demo illustrates the following capabilities:

- Capturing a video stream from the input port
- Performing real-time image processing on the video stream
- Displaying the processed video

This demo illustrates one of the simplest forms of video processing systems, that is, processing the video data as it streams through the system. The video data is passed from one block to another along with the control signals, and is only stored in small line buffers made from on-chip memory. Because the data is processed in a streaming fashion, there is no need for frame buffers in DRAM. This demo can be used as a starting point and modified by users to suit their needs.

Figure 3-2 illustrates the video processing pipeline that is implemented in the DVI Pass-Through Demo. Descriptions of the dvi_in and dvi_out blocks can be found in Appendix B, "IP Repository."

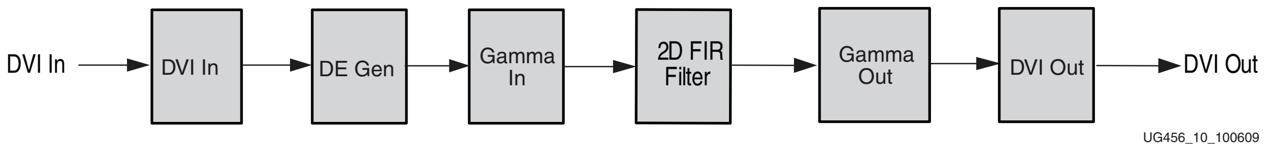


Figure 3-2: **DVI Pass-Through: Video Pipeline**

Gamma_In implements a Gamma transform which can be used to remove the Gamma correction done by the video source. Gamma_Out implements a Gamma Inverse transform which can be used to pre-distort the video stream to correct for the Gamma effect of the video display.

The 2-D FIR Filter implements a 5x5 2-D FIR filter that can be used to produce a variety of image processing effects, such as edge detection, image sharpening, or image smoothing. The coefficients used in the filter determine the operation of the filter. The MicroBlaze processor can be used to load new coefficients into the filter.

The DVI Pass-Through Demo only supports the following video resolutions shown in Table 3-2.

Table 3-2: **Video Resolutions Supported by the DVI Pass-Through Demo**

Digital	Analog
640x480P @ 60Hz	640x480P @ 60Hz
720x576P @ 50Hz	800x600P @ 60Hz
720x480P @ 60Hz	1024x768P @ 60Hz
800x600P @ 60Hz	720P @ 50Hz
1024x768 @ 60Hz	720P @ 60Hz
720P @ 50Hz	
720P @ 60Hz	

Table 3-2: **Video Resolutions Supported by the DVI Pass-Through Demo *(Cont'd)***

Digital	Analog
1080P @ 25Hz	
1080P @ 30Hz	
1080I @ 50Hz	
1080I @ 60Hz	

Using the Application

Follow the steps listed to set up and run the DVI Pass-Through Demo. If the demo fails to run, see Appendix D, "Troubleshooting."

1. Set the video source to one of the supported video resolutions.
2. Follow the steps listed in Chapter 2, "System Setup."

Note: The section on connecting the VGA Camera can be skipped.

Top Level Menu

The keys listed in Table 3-3may be used in the Top-Level menu to exercise the DVI Pass-Through Demo.

Note: : The VSK is initialized at Startup to support the video resolutions listed previously.

Table 3-3: **Key Descriptions for the DVI Pass-Through Demo Menu**

Key	Description
d	Initialize the VSK for digital video.
a	Initialize the VSK for analog video.
g	Enter the Gamma Processing menu.
f	Enter the 2-D FIR Filter menu.
x	Enter the DE_Gen menu.
i	Enter the IIC Diagnostics menu.
o	Read DVI_Out EDID PROM.
e	Read DVI_In EDID PROM.
E	Program the DVI_In EDID PROM.
?	Print the Top-Level menu Help screen.

Gamma Menu

The keys in Table 3-4 may be used in the Gamma menu to exercise the gamma processing portion of the demo.

Table 3-4: **Key Descriptions for the Gamma Menu**

Key	Description
i	Load Gamma_In with a new Gamma curve. (Toggles between Gamma(1) and Gamma(2.2))
o	Load Gamma_Out with a new Gamma curve. (Toggles between Inverse_Gamma(1) and Inverse_Gamma(2.2))
I	Print the contents of the Gamma_In function.
O	Print the contents of the Gamma_Out function.
s	Print the Status of the gamma functions.
?	Print the Gamma menu Help screen.
Esc	Exit the Gamma menu back to the Top-Level menu.

2-D FIR Filter Menu

The keys in Table 3-5 may be used in the 2-D FIR Filter menu to exercise the filtering portion of the demo.

Table 3-5: **Key Descriptions for the 2-D Filter Menu**

Key	Description
1-9	Load a new set of filter coefficients.
0	Create a new set of filter coefficients.
;	Decrement the filter gain.
'	Increment the filter gain.
c	Print the currently loaded filter coefficients.
g	Print the currently loaded filter gain.
s	Print the status of the filter.
?	Print the 2-D FIR Filter menu Help screen.
Esc	Exit the 2-D FIR Filter menu back to the Top-Level menu.

DE_Gen Menu

The keys in Table 3-6 may be used in the 2-D FIR Filter menu to exercise the filtering portion of the demo.

Table 3-6: **Key Descriptions for the DE_Gen Menu**

Key	Description
s	Print Frame Information.
b	Print Vertical & Horizontal Blanking.
c	Print DE_Gen Enable & Polarity Requests.
e	DE_Gen Enable (Toggles On/Off).
p	Polarity Request Enable (Toggles On/Off).
v	Vertical Polarity Request (Toggles Hi/Lo).
h	Horizontal Polarity Request (Toggles Hi/Lo).
u	Advance One Line Vertically.
d	Delay One Line Vertically.
l	Advance One Pixel Horizontally.
r	Delay One Pixel Horizontally.
[	Decrement the DE_Gen register selection.
]	Increment the DE_Gen register selection.
;	Decrement the data value.
'	Increment the data value.
,	Read the data value (data = register]).
.	Write the data value (register = data).
?	Print the DE_Gen menu Help Screen.

IIC Diagnostics Menu

The keys in Table 3-7 may be used in the IIC Diagnostics menu to explore and experiment with the IIC register settings of the various peripherals on the VSK.

Table 3-7: **Key Descriptions for the IIC Diagnostics Menu**

Key	Description
-	Decrement through the list of VSK peripherals.
=	Increment through the list of VSK peripherals.
[	Decrement the chip register address.
]	Increment the chip register address.
;	Decrement the data value by 0x1.
:	Decrement the data value by 0x100.
"	Increment the data value by 0x100.

Table 3-7: **Key Descriptions for the IIC Diagnostics Menu *(Cont'd)***

Key	Description
'	Increment the data value by 0x1.
,	Read the data value (data = chip[register]).
.	Write the data value (chip[register] = data).
?	Print the IIC Diagnostics menu Help screen.
Esc	Exit the IIC Diagnostics menu back to the Top-Level menu.

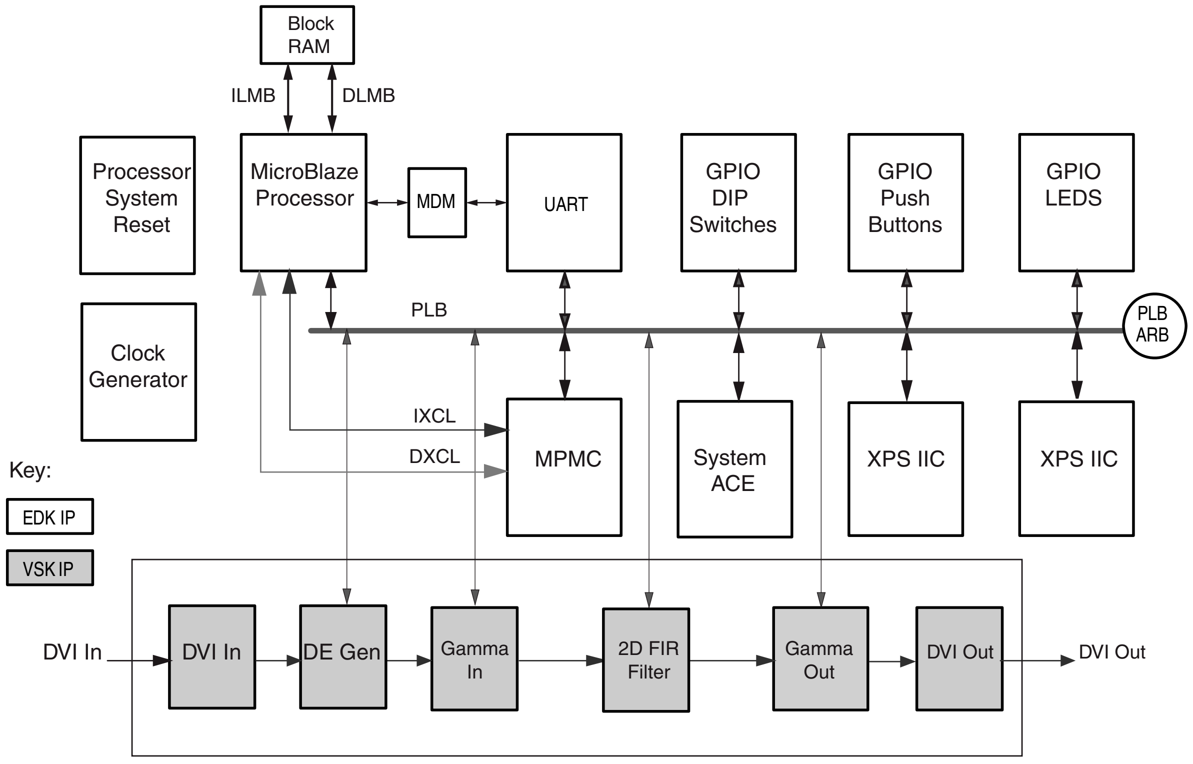
Understanding Source Files

Hardware

The system is controlled by a MicroBlaze processor that has the following responsibilities:

- Initializing the VSK peripherals
- Communicating with the PC HyperTerminal
- Controlling the Video Processing Pipeline by reading and writing control registers in the system.

Figure 3-3 shows the system-level block diagram of the DVI Pass-Through Demo. It extends the Base Platform described in the previous section by the addition of the video processing pipeline shown in the gray boxes.



UG456_11_100609

Figure 3-3: **DVI Pass-Through Demo: System-Level Block Diagram**

The following paragraphs discuss the portions of the DVI Pass-Through demo that are unique to this demo. Descriptions of the dvi_in and dvi_in blocks can be found in Appendix B, "IP Repository."

The Gamma_In and Gamma_Out components of the DVI Pass-Through demo are instances of the same gamma processing design that was created and verified using the Xilinx System Generator for DSP software tool. Once verified, each was exported to EDK as a peripheral core (PCORE) for system integration. System Generator software automatically generates the bus interface, memory map decoding logic, and a software driver for the peripheral. The memory map is defined symbolically in System Generator using shared memories that can be read or written by the MicroBlaze processor in real-time.

The System Generator model file can be found in the demo DVI Pass-Through project directory structure at:

```
VSK_CD: \Demonstrations\EDK\VSK_Repository\VSKProcessorIPLib\
pcores\gamma_plbw_v3_00_b\sysgen\gamma_8bit.mdl.
```

Note: A fully functional evaluation copy of Xilinx System Generator for DSP is included with the VSK and must be installed to view the System Generator model. See Appendix A, "Development Environment," for details.

Note: If the export of the PCORE fails, the issue may be due to a long target path. Using a target path such as `C:\temp\edk` may fix the issue.

The 2-D FIR Filter in the DVI Pass-Through demo was also implemented and verified using System Generator for DSP. After the design was completed, it was exported from the tool as a PCORE and then integrated into the overall design. The filter coefficients and the gain value used in the filter are stored in memories that can be read or written by the MicroBlaze processor in real-time. Again, System Generator software automatically generates the bus interface, memory map decode logic, and software driver for the custom peripheral.

The System Generator model file can be found in the demo DVI Pass-Through project directory structure at:

```
VSK_CD: \Demonstrations\EDK\DVI_Pass_Through_Demo\pcores\
rgb_2d_fir_plbw_v3_00_b\sysgen\fir_2d.mdl.
```

Software

This section describes the embedded software files in the DVI Pass-Through demo that are unique to this demo. The remaining files are all discussed in Appendix B, "IP Repository." All of the source code files can be found in the DVI Pass-Through project directory structure at:

```
VSK_CD:\Demonstrations\EDK\DVI_Pass_Through_Demo\

DVI_Pass_Through_Sw\src.
```

- **`vsk_top.c`** – The top-level C source code file for the DVI Pass-Through demo software application is the `vsk_top.c`. This file performs the initialization of the VSK at power-up and then creates the menu system that is used in the HyperTerminal.
- **`gamma.c`** – The software associated with the Gamma Processing PCORE can be found in the C source file `gamma.c`. This file implements the software routines that communicate with portions of the PCORE that can be read/written by the MicroBlaze processor.

- **`fir_2d.c`** – The software associated with the filter processing portion of the demo can be found in the C source file `fir_2d.c`. It implements the software routines that read and write the filter coefficients and gain value of the 2-D FIR filter. Using these routines, the coefficient values and gain can be adjusted in real-time by the MicroBlaze processor.

For more software details, see *Spartan-3A DSP FPGA Video Starter Kit Software User Guide, UG514* [Ref 6].

DVI Frame Buffer Demo

Overview

This demo illustrates the following capabilities:

- Capturing a video stream from a DVI source
- Buffering the video stream in external memory
- Displaying the buffered video
- Reporting memory bandwidth utilization data

This demo has two purposes. One is to illustrate a working video pipeline that uses external memory as a frame buffer. See Figure 3-4. The other is to provide a simple platform, using the EDK, that can be used as a starting point and modified by users to suit their needs.

In this design, data is passed in through the DVI In. This goes to the DE Gen, which generates the data valid information, for the Video to VFBC, so that you only send the active data into the MPMC. The default is a 3-frame buffer, and a simple sync signal that is connected between the Video to VFBC and the Display Controller to make sure that you read out one frame behind what is being written into the external memory. The display controller then reads data out of memory and passes it to the DVI out.

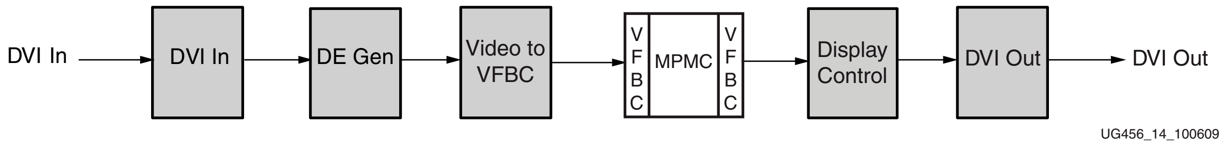


Figure 3-4: **DVI Frame Buffer: Video Pipeline**

The DVI Frame Buffer Demo only supports the following video resolutions:

Table 3-8: **Supported Video Resolutions by the DVI Frame Buffer Demo**

Digital	Analog
640x480P @ 60 Hz	640x480P @ 60 Hz
720x576P @ 50 Hz	800x600P @ 60 Hz
720x480P @ 60 Hz	1024x768P @ 60 Hz
800x600P @ 60 Hz	720P @ 50 Hz
1024x768 @ 60 Hz	720P @ 60 Hz
720P @ 50 Hz	

Table 3-8: **Supported Video Resolutions by the DVI Frame Buffer Demo**

Digital	Analog
720P @ 60 Hz	
1080P @ 25Hz	
1080P @ 30Hz	
1080I @ 50Hz	
1080I @ 60Hz	

Using the Application

1. Set the video source to one of the supported video resolutions.
2. Follow the steps listed in Chapter 2, "System Setup," to set up the VSK for running the DVI Frame Buffer Demo. If the demo fails to run, see Appendix D, "Troubleshooting."

Note: The section on connecting the VGA Camera can be skipped.

Top-Level Menu

The keys in Table 3-9 may be used in the Top-Level menu to exercise the DVI Frame Buffer Demo.

Note: The VSK is initialized at Startup to support the video resolutions listed previously.

Table 3-9: **Key Descriptions for the DVI Frame Buffer Demo Menu**

Key	Description
d	Initialize the VSK for digital video.
a	Initialize the VSK for analog video.
b	Report memory bandwidth data.
i	Enter the IIC Diagnostics menu.
o	Read DVI_Out EDID PROM.
e	Read DVI_In EDID PROM.
E	Program the DVI_In EDID PROM.
x	Enter the DE Generator Menu.
t	Test Pattern (toggle on/off).
?	Print the Top-Level menu Help Screen.

Digital Video Menu

The keys in Table 3-10 may be used to select between the supported digital video resolutions.

Table 3-10: **Key Descriptions for the Digital Video Menu**

Key	Description
1	Select 640x480P @ 60 Hz.
2	Select 640x480P @ 60 Hz.
3	Select 720x576P @ 60 Hz.
4	Select 800x600P @ 60 Hz.
5	Select 1024x768P @ 60 Hz.
6	Select 720P @ 50 Hz.
7	Select 720P @ 60 Hz.
8	Select 1080P @ 25 Hz.
9	Select 1080P @ 30 Hz.
a	Select 1080I @ 50 Hz.
b	Select 1080I @ 60 Hz.
?	Print the Digital Video menu help screen.
Esc	Exit the Digital Video menu back to the Top-Level menu.

Analog Video Menu

The keys in Table 3-11 may be used to select between the supported analog video resolutions.

Table 3-11: **Key Descriptions for the Analog Video Menu**

Key	Description
1	Select 640x480P @ 60Hz.
2	Select 800x600P @ 60Hz.
3	Select 1024x768P @ 60Hz.
4	Select 720P @ 50Hz.
5	Select 720P @ 60Hz.
?	Print the Analog Video menu help screen.
Esc	Exit the Analog Video menu back to the Top-Level menu.

IIC Diagnostics Menu

See the "DVI Pass-Through Demo" section.

DE Generator Menu

See the "DVI Pass-Through Demo" section.

Understanding Source Files

The files for this design can be found on the supplied CD in the following directory:

```
VSK_CD:\Demonstrations\DVI_Frame_Buffer_Demo
```

The directory structure for this project is the same as that described in the "Base Platform" section. More details are given about the files that are specific to this design in the following sections.

Hardware

The system is controlled by a MicroBlaze processor that has the following responsibilities:

- Initializing the VSK peripherals
- Communicating with the PC HyperTerminal
- Controlling the Video Processing Pipeline by reading and writing control registers in the system calculating memory bandwidth data

The embedded system was initially created using the EDK base system builder as described in the "Base Platform" section. IP that is specific to the video pipeline was then added to the Base Platform EDK project.

The following video IP was added to the project:

- dvi_in
- dvi_out
- de_gen
- video_to_vfbc
- display_controller

Figure 3-5 shows the system-level diagram of the DVI Frame Buffer demo. The video pipeline is shown as a gray box.

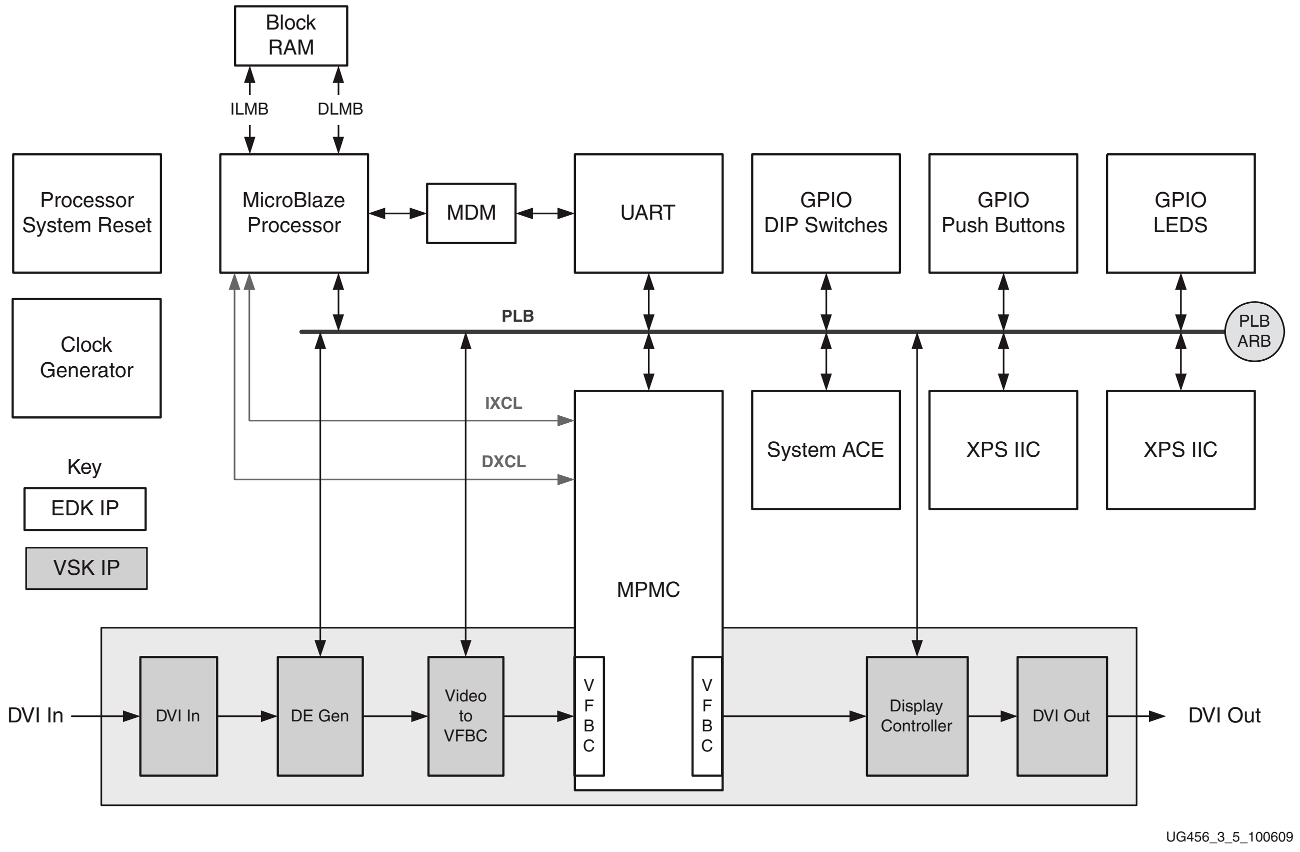


Figure 3-5: **DVI Frame Buffer Demo: System-Level Block Diagram**

DVI_IN and DVI_OUT blocks provide the connection to the DVI input and output on the VSK platform. Descriptions of these blocks can be found in Appendix B, "IP Repository."

The VFBC is a special interface for video frame data and is a part of the MPMC. It augments the capabilities of the MPMC by providing the ability for long burst control and rectangular region access that is well suited to video applications. For more information, see the MPMC data sheet (DS643) [Ref 3].

Each of the VSK IP blocks used in this demo were written in HDL and can be browsed for a more in-depth understanding of the blocks. All of the blocks reside in the VSK IP repository.

Software

This section discusses the software files in the DVI Frame Buffer demo that are unique to this demo. The remaining files are all discussed in Appendix B, "IP Repository." All of the source code files can be found in the EDK project directory structure at:

```
VSK_CD: \Demonstrations\EDK\DVI_Frame_Buffer_Demo\
DVI_Frame_Buffer_Sw\src
```

- **`vsk_top.c`** – The top-level C source code file for the DVI Frame Buffer demo software application is `vsk_top.c`. This file performs the initialization of the VSK at power-up and then creates the menu system that is used in the HyperTerminal.

- **`display_res.c`** – The software associated with setting registers for controlling the resolution input and output can be found in the C source file `display_res.c`. This file implements the software routines that communicate with portions of the related PCOREs that can be read/written by the MicroBlaze processor.
- **`vsk_bandwidth.c`** – The software associated with reporting memory bandwidth utilization data can be found in the C source file `vsk_bandwidth.c.` It implements the software routines that gather and display memory bandwidth for each port on the MPMC. These routines utilize the MPMC Performance Monitor (enabled in the ECC/Debug tab of the Configure IP menu of the MPMC in the EDK System Assembly View).

The MPMC Performance Monitor (PM) includes several registers that report information on the memory transactions for each port on the MPMC. The VSK bandwidth software utilizes the Global Cycle Count Register (PMGCC) and the Data Bin Registers (PMX_DATA_BINx). The global cycle counter register reports the total number of active clock cycles since the last PM initialization. The data bin registers report the total number of memory read/write transactions for each MPMC port since the last PM initialization. Each data bin is divided into sections for transaction type (read or write) and for transaction size. The transaction size can be a burst of length one byte to 64-words.

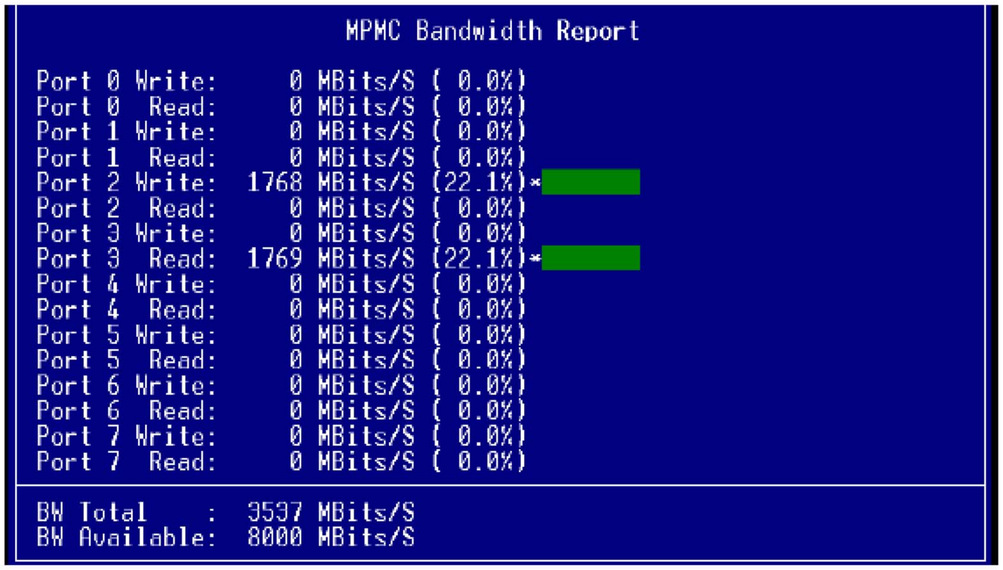
The VSK bandwidth software performs the following:

3. Initializing all MPMC Performance Monitor registers to zero
4. Enabling performance monitoring on all MPMC ports (0 through 7)[1]
5. Reporting the estimated bandwidth for each MPMC port for both read and write transactions

The bandwidth for each port is calculated as a percentage of the maximum allowable bandwidth. This can be found by summing the total number of transactions multiplied by the cycle count of each transaction. This sum is divided by the total number of active clock cycles (found in the PMGCC) for each port. The MPMC is clocked at 125 MHz from the IDT clock generator on the board. A 32-bit data bus is routed to the DDR2 memory. This provides a maximum theoretical bandwidth of 8000 Mbps.

1. Performance data is currently only reported for MPMC ports 2 and 3. These ports correspond to the DVI video input and output ports, respectively.

A final bandwidth report for all MPMC ports is printed to the UART. An example report is shown in Figure 3-6:


```
                MPMC Bandwidth Report

Port 0 Write:     0 MBits/S ( 0.0%)
Port 0  Read:     0 MBits/S ( 0.0%)
Port 1 Write:     0 MBits/S ( 0.0%)
Port 1  Read:     0 MBits/S ( 0.0%)
Port 2 Write:  1768 MBits/S (22.1%)*
Port 2  Read:     0 MBits/S ( 0.0%)
Port 3 Write:     0 MBits/S ( 0.0%)
Port 3  Read:  1769 MBits/S (22.1%)*
Port 4 Write:     0 MBits/S ( 0.0%)
Port 4  Read:     0 MBits/S ( 0.0%)
Port 5 Write:     0 MBits/S ( 0.0%)
Port 5  Read:     0 MBits/S ( 0.0%)
Port 6 Write:     0 MBits/S ( 0.0%)
Port 6  Read:     0 MBits/S ( 0.0%)
Port 7 Write:     0 MBits/S ( 0.0%)
Port 7  Read:     0 MBits/S ( 0.0%)

BW Total     :  3537 MBits/S
BW Available:  8000 MBits/S
```

Figure 3-6: **MPMC External Memory Bandwidth Report**

Performance data is currently only reported for MPMC ports 2 and 3. These ports correspond to the DVI video input and output ports, respectively.

See the MPMC Data Sheet [Ref 3] for more information on the MPMC and the use of the Performance Monitor. For more software details, see *Spartan-3A DSP FPGA Video Starter Kit Software User Guide, UG514* [Ref 6].

Camera Frame Buffer Demo

Overview

This demo illustrates the following capabilities:

1. Capturing a video stream from a camera
2. Performing processing on the video stream
3. Buffering the video stream in external memory
4. Displaying the processed video at a different frame rate
5. Using a microprocessor to configure various aspects of the video pipeline.

The basic video pipeline demonstrated by this design is shown in Figure 3-7.

In this design, data is generated by the camera and sent through a camera processing block which does some basic processing like a bayer filter. This is then sent through a Gamma block for data correction, and then on to the Video to VFBC, so that you only send the active data into the MPMC. The default is a 3-frame buffer, and a simple sync signal that is connected between the Video to VFBC and the Display Controller to make sure that you read out one frame behind what is being written into the external memory. The display controller then reads data out of memory and passes it to the DVI out.

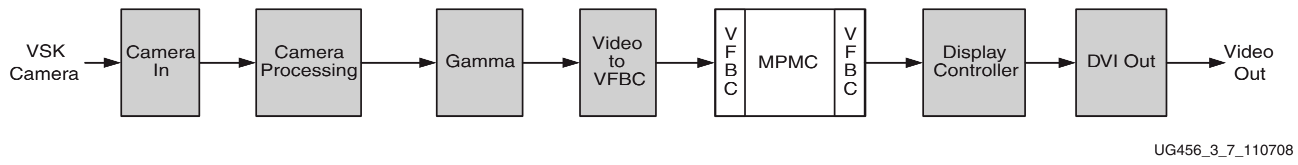


Figure 3-7: **Camera Frame Buffer Video Pipeline**

This design supports video capture and display at 640x480P @ 60Hz.

Using the Application

This section assumes the hardware has been correctly set up (as detailed in Chapter 2, "System Setup"). Make sure the steps under "Connect the VGA Camera" and "Connect the Video Display (DVI/VGA)" have been followed. If the demo fails to run, see Appendix D, "Troubleshooting."

Note: The lens on the camera can be screwed in and out to adjust the focus.

Basic Menu Features

When the design starts up, a menu of options appears on the serial terminal. The commands available at the main menu are shown in Table 3-12.

Table 3-12: **Main Menu Commands**

Key	Description
c	Enter Camera configuration menu.
P	Enter Processing menu.
s	Enter Storage menu.
i	Enter the IIC Diagnostics menu.
?	Print the Top-Level menu Help screen.

Camera Commands

The Camera menu gives access to configuration of the physical camera itself.

The camera commands are as follows:

Table 3-13: **Camera Menu Commands**

Key	Description
t	Test pattern generation select.
x	Auto exposure control on/off.
g	Auto gain control on/off.
n	Noise control on/off.
m	Configure dynamic range mode.
r	Reset camera defaults.
Esc	Exit back to the Top-Level menu.
?	Print menu Help screen.

Processing Commands

The Processing menu gives access to configuration of the Camera Processing block. Most functions in the block can be enabled and configured independently. For more details about this block, see "Camera Processing."

The Processing commands are as follows:

Table 3-14: **Processing Menu Commands**

Key	Description
P	Stuck pixel correction on/off.
,	Stuck pixel threshold decrease.
.	Stuck pixel threshold increase.
h	Brightness control on/off.
-	Brightness setting decrease.
=	Brightness setting increase.
n	Contrast control on/off.
[	Contrast setting decrease.
]	Contrast setting increase.
c	Color balance control on/off.
r	Color balance select red.
g	Color balance select green.
b	Color balance select blue.
;	Color balance gain decrease (selected color).
'	Color balance gain increase (selected color).
o	Gamma control on/off.
s	Show image statistics (red, green, blue min/max).
i	Initialization processing settings to defaults.
Esc	Exit back to the Top-Level menu.
?	Print menu Help screen.

Storage Commands

Settings made in the processing menu can be stored to Compact Flash and reloaded using commands in this menu. Additionally, still images or video clips (sequence of stills) from the camera can be stored in BMP format. The stored still images can be found on the Compact Flash at `\STILLXX.BMP` (where XX = 00 to 49) while the stored video sequences can be found in directories named `\VIDEOXX` (where XX = 00 to 04). All that is necessary to view the images on a PC is a Compact Flash reader. After the maximum number of stills or clips has been reached, files must be removed from the Compact Flash card before more can be recorded.

Note: Still images captured using video capture commands can be stitched together into AVI files for playback using a utility such as EASYBMPtoAVI. Take note of the frame rate at which the video sequences were captured so that the video can be played back correctly.

Table 3-15: **Storage Menu Commands**

Key	Description
s	Store processing settings to flash.
r	Read processing settings from flash.
w	Write image to flash.
v	Write video sequence to flash.
Esc	Exit back to the Top-Level menu.
?	Print menu Help screen.

IIC Diagnostics

See the "DVI Pass-Through Demo" section.

Understanding Source Files

The files for this design can be found on the supplied CD in the following directory:

```
VSK_CD:Demonstrations\EDK\Camera_Frame_Buffer_Demo
```

The directory structure for this project is the same as that described in the "Base Platform" section. More details are given about the files that are specific to this design in the following sections.

Hardware

Overview

See Figure 3-8. This demo makes use of the same basic hardware described for the "DVI Frame Buffer Demo" demo with a couple modifications:

- Camera Input instead of DVI_IN
- Addition of Camera Processing and Gamma

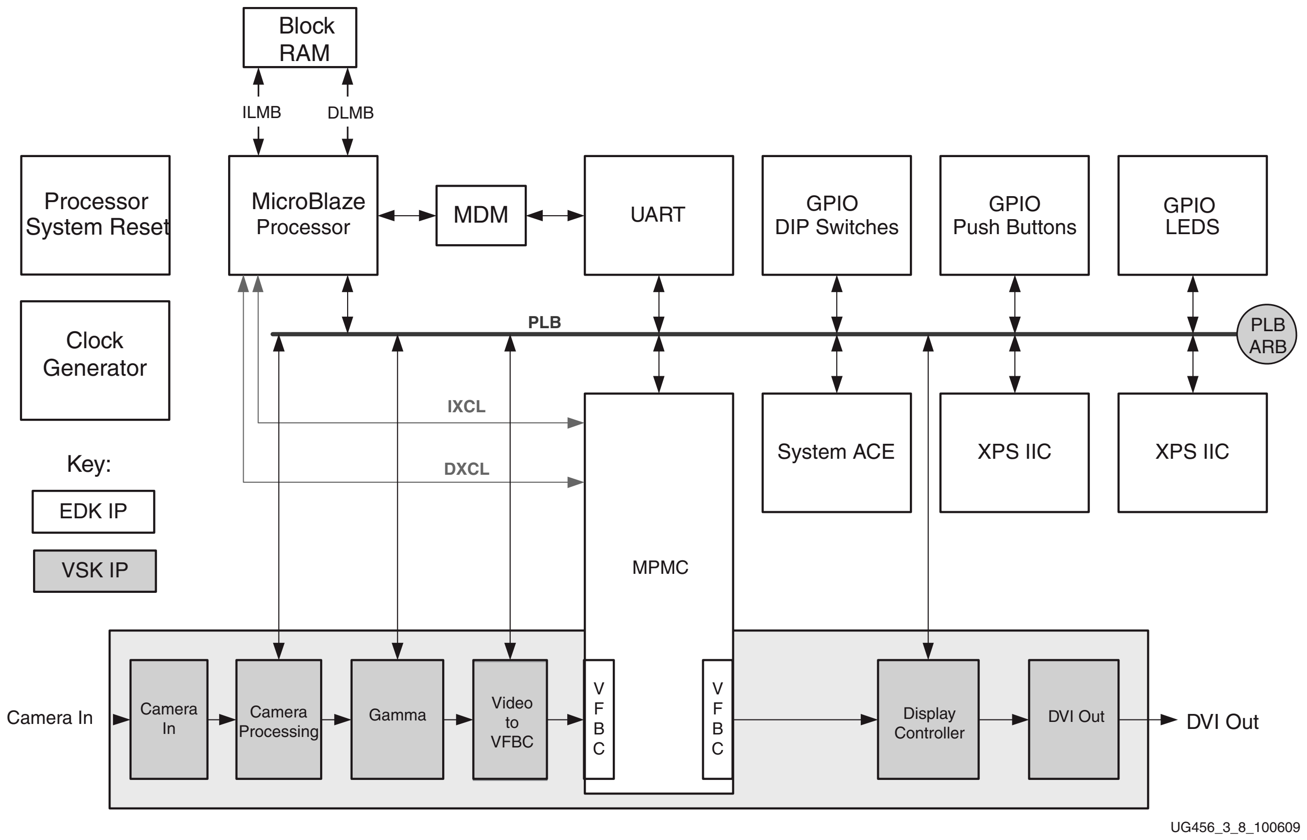


Figure 3-8: **Camera Frame Buffer Hardware**

Details about all of the blocks in video pipeline can be found in Appendix B, "IP Repository," except the Camera Processing block. The Camera Processing block is described in the following section.

Camera Processing

The Camera Processing block is an EDK PCORE developed specifically for this design. This block was designed in System Generator and exported as an EDK PCORE.

This block contains the following functions:

- Wide Dynamic Range Expansion – Increases resolution from 8 bits to 12 bits per pixel using methods described in Micron MT9V022 data sheet.
- Stuck Pixel Correction – Adaptive median filtering of pixels with a configurable threshold.
- Bayer Conversion – Linear interpolation of RGB color components from Bayer pattern color filter array (CFA).
- Brightness Control – Global offset control by way of adder/subtractor.
- Contrast Control – Global digital gain stage using a multiplier.
- Color Balance Control – Individual color gains for red, green, and blue components.
- Image Statistics – Calculates global maximums and minimums for each color component.

The diagram for the Camera Processing block can be seen in Figure 3-9.

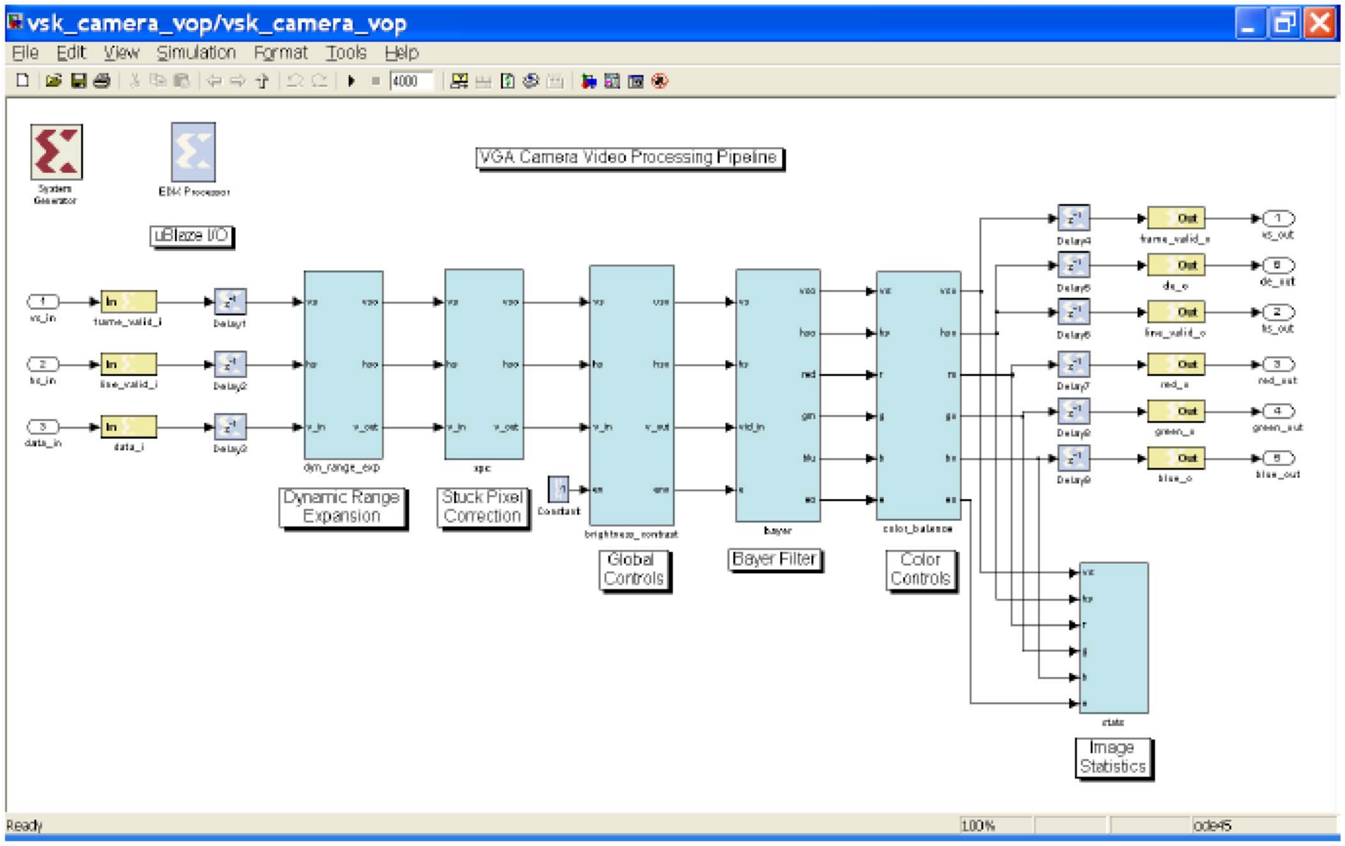


Figure 3-9: **Camera Processing Block Diagram**

As part of the EDK PCORE export function, System Generator software automatically generates the bus interface, the memory map decode logic, and device driver for the custom peripheral. The source files for this block can be found in:

```
\pcores\vsk_camera_vop_sm_v3_00_b
```

The System Generator model used to generate the PCORE can be found at:

```
\pcores\vsk_camera_vop_plbw_v3_00_b\sysgen\vsk_camera_vop.mdl
```

For more information about the functions in this block, open the model in MATLAB® software to view the implementation details.

Note: Xilinx System Generator must be installed to view the System Generator model. See Appendix A, "Development Environment," for details.

Note: If the export of the PCORE fails, the issue may be due to a long target path. Using a target path such as `C:\temp\edk` may fix the issue.

Software

The `\Camera_Frame_Buffer_Sw` directory for the project contains the following:

- \src
- executable.elf

The following files in `\src` are specific to this design:

- `vsk_camera_menu.c` – Implementation of Camera menu
- `vsk_camera_menu.h` – Header file for above
- `vsk_camera_menu_l.h` – Header file for above
- `vsk_processing_menu.c` – Implementation of Processing menu
- `vsk_processing_menu.h` – Header file for above
- `vsk_processing_menu_l.h` – Header file for above
- `sk_storage_menu.c` – Implementation of Storage menu
- `vsk_storage_menu.h` – Header file for above

For more software details, see *Spartan-3A DSP FPGA Video Starter Kit Software User Guide, UG514* [Ref 6].

S-Video Frame Buffer Demo

Overview

This demo illustrates the following capabilities:

- Capturing a video stream from an S-Video source
- Buffering the video stream in external memory
- Displaying the buffered video

This demo has two purposes. One is to illustrate a working video pipeline that uses external memory as a frame buffer. See Figure 3-10. The other is to provide a simple platform, using the EDK, that can be used as a starting point and modified by users to suit their needs.

In this design, data is passed in through the SDTV In. This goes to the DE Gen, which generates the data valid information for the following blocks. It then goes to a Gamma correction block followed by a 2D FIR Filter. The output of the 2D FIR Filter is read by the Video to VFBC and then written into the external memory using the VFBC port on the MPMC. The default is a 3-frame buffer, and a simple sync signal that is connected between the Video to VFBC and the Display Controller to make sure that you read out one frame behind what is being written into the external memory. The display controller then reads data out of memory and passes it to the DVI out.

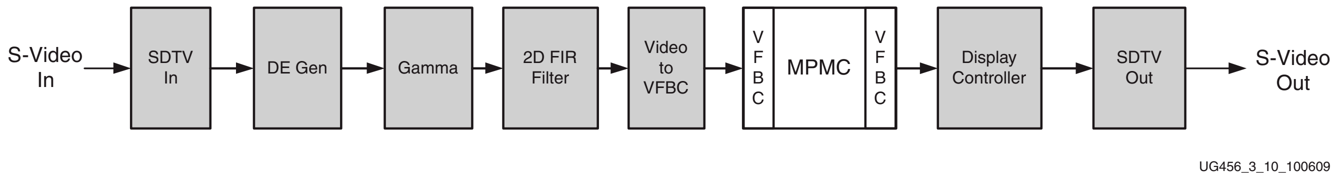


Figure 3-10: **S-Video Frame Buffer: Video Pipeline**

The S-Video Frame Buffer Demo supports the following video resolutions:

- NTSC
- PAL

Using the Application

Perform the following steps to set up and run the S-Video Frame Buffer Demo. If the demo fails to run, see Appendix D, "Troubleshooting."

1. Set the video source to one of the supported video resolutions.
2. Follow the steps listed in Chapter 2, "System Setup."

Note: The section on connecting the VGA Camera can be skipped.

Top-Level Menu

The keys in Table 3-16 may be used in the Top-Level menu to exercise the S-Video Frame Buffer Demo.

Note: The VSK is initialized at Startup to support the video resolutions listed previously.

Table 3-16: **Keys Used to Exercise the S-Video Frame Buffer Demo**

Key	Description
s	Enter S-Video resolution menu (NTSC, PAL).
x	Enter the DE Gen menu.
g	Enter the Gamma Processing menu.
f	Enter the 2-D FIR Filter menu.
i	Enter the IIC Diagnostics menu.
t	Test Pattern (toggle on/off).
?	Print the Top-Level menu Help screen.

S-Video Menu

The keys in Table 3-17 may be used to select between the supported S-video resolutions.

Table 3-17: **Keys for Selecting the Supported S-Video Resolutions**

Key	Description
1	Select NTSC S-Video (720x480i60).
2	Select PAL S-Video (720x576i50).
3	Toggle Color Bars (generated by ADV7179).
?	Print the S-Video menu Help screen.
Esc	Exit the S-Video menu back to the Top-Level menu.

DE Gen Menu

See the "DVI Pass-Through Demo" section.

Gamma Processing Menu

See the "DVI Pass-Through Demo" section.

2-D FIR Filter Menu

See the "DVI Pass-Through Demo" section.

IIC Diagnostics Menu

See the "DVI Pass-Through Demo" section.

Understanding Source Files

The files for this design can be found on the supplied CD in the following directory:

```
VSK_CD: \Demos\S-Video_Frame_Buffer_Demo
```

The directory structure for this project is the same as that described in the "Base Platform" section. More details about the files that are specific to this design are given in the following sections.

Hardware

The system is controlled by a MicroBlaze processor that has the following responsibilities:

- Initializing the VSK peripherals
- Communicating with the PC HyperTerminal
- Controlling the Video Processing Pipeline by reading and writing control registers in the system

The embedded system was initially created using the EDK base system builder as described in the "Base Platform" section. IP specific to the video pipeline was then added to the Base Platform EDK project.

The following video IP was added to the project:

- sdtv_in
- sdtv_out
- de_gen
- sdtv_gamma_plbw
- ycbcr_2d_fir_plbw
- video_to_vfbc
- display_controller

Figure 3-11 shows the system-level diagram of the DVI Frame Buffer demo. The video pipeline is shown inside the gray box.

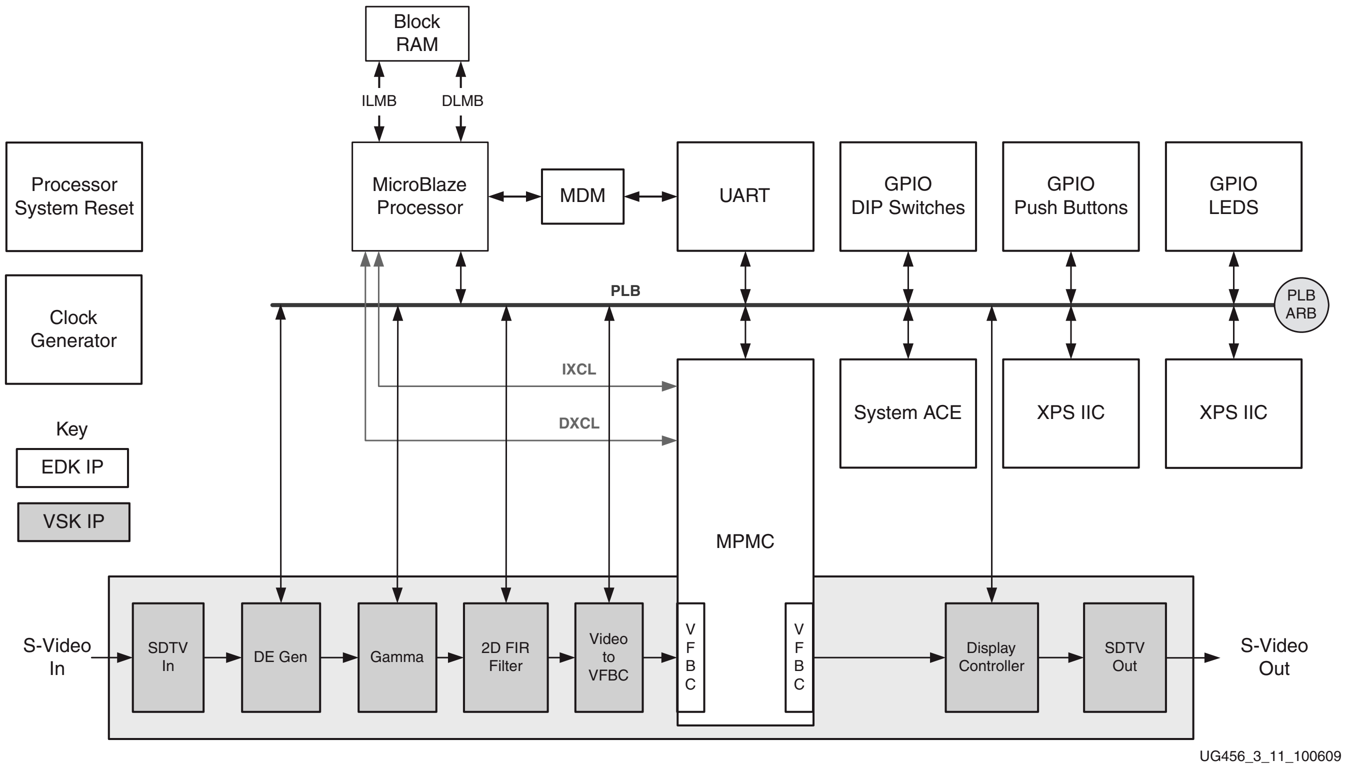


Figure 3-11: **S-Video Frame Buffer Demo**

SDTV In and SDTV out blocks provide the connection to the S-Video input and output on the VSK platform. Descriptions of these blocks can be found in Appendix B, "IP Repository."

The VFBC is a special interface for video frame data and is a part of the MPMC. It augments the capabilities of the MPMC by providing the ability for long burst control and rectangular region access that is well suited to video applications. For more information, see the MPMC data sheet (DS643)[Ref 3].

Each of the VSK IP blocks used in this demo were written in HDL and can be browsed for a more in-depth understanding of the blocks. Some of the blocks reside in the VSK IP repository, while others are located within the `\pcores` directory of this project.

Software

This section discusses the software files in the S-Video Frame Buffer demo that are unique to this demo. The remaining files are discussed in Appendix B, "IP Repository." All of the source code files can be found in the EDK project directory structure at:

```
VSK_CD: \Demonstrations\EDK\S-Video_Frame_Buffer_Demo\
S-Video_Frame_Buffer_Sw\src
```

- **`vsk_top.c`** – The top-level C source code file for the S-Video frame buffer demo software application is the `vsk_top.c`. This file performs the initialization of the VSK at power-up and then creates the menu system that is used in the HyperTerminal.

- **`sdtv_gamma.c`** – The software associated with the Gamma Processing PCORE can be found in the C source file `sdtv_gamma.c`. This file implements the software routines that communicate with portions of the PCORE that can be read/written by the MicroBlaze processor.
- **`ycbcr_fir_2d.c`** – The software associated with the filter processing portion of the demo can be found in the C source file `ycbcr_fir_2d.c`. It implements the software routines that read and write the filter coefficients and gain value of the 2-D FIR filter. Using these routines, the coefficient values and gain can be adjusted in real time by the MicroBlaze processor.

For more software details, see *Spartan-3A DSP FPGA Video Starter Kit Software User Guide, UG514* [Ref 6].

Appendix A

Development Environment

This appendix provides details about the hardware and software necessary to modify and rebuild the reference designs included in this kit. For details on how to install and use these products, refer to the documentation supplied with them.

Software Requirements

The following products must be installed to rebuild the example applications for this kit:

1. Xilinx ISE® 11.4
2. Xilinx EDK 11.4

The following are required to modify components of the DVI Pass-Through, Camera Frame Buffer, and S-Video Frame Buffer applications that were developed in Xilinx System Generator:

1. MathWorks MATLAB® 2008a, 2008b, or 2009a with Simulink®
2. Xilinx System Generator 11.4

Note: Evaluation versions of all the Xilinx tools previously mentioned are included on the CD/DVD media that comes with VSK. Evaluation versions of MATLAB/Simulink can be requested from MathWorks via their website at: www.mathworks.com.

Hardware Requirements

To download new configuration bitstreams to the board, a Xilinx JTAG cable such as one of the following is recommended:

- Parallel Cable IV
- Platform Cable USB

Note: A Xilinx Platform Cable USB is included with VSK.

Project Requirements

Before building any of the projects in EDK, the following steps should be followed:

1. Copy the EDK_Demonstrations.zip file from the VSK CDROM to the hard drive of the computer.
2. Unzip the entire EDK_Demonstrations.zip file to the hard drive of the computer.
3. Make sure that all project directories have read/write privileges.

Appendix B

IP Repository

IP Blocks

The IP repository is located at VSK_CD: `\Demonstrations\EDK\VSK_Repository`.

camera (v2.00a)

Introduction

The camera PCORE provides a connection to the camera that is included with the Xilinx Video Starter Kit. The camera is based on the Micron MT9V022 Digital Image Sensor and transports the video stream to the FMC-Video card in a serial format via a CAT 6 cable. The serial video is de-serialized on the FMC-Video card by the DS92LV1212A deserializer chip. The resulting parallel data stream is the input to the camera block.

Table Table B-1 provides the camera I/O signals and their descriptions. See Figure B-1.

Table B-1: **camera I/O Signals**

Signal	Direction	Bus Interface	Description
frame_valid	I	–	Frame Valid input
line_valid	I	–	Line Valid input
data_in(7:0)	I	–	Bayer pixel input
clk	I	–	Clock input (Pixel Rate)
ce	I	–	Clock enable (active High)
frame_valid	O	CAMERA_VIDEO_OUT	
line_valid	O	CAMERA_VIDEO_OUT	
data_in(7:0)	O	CAMERA_VIDEO_OUT	

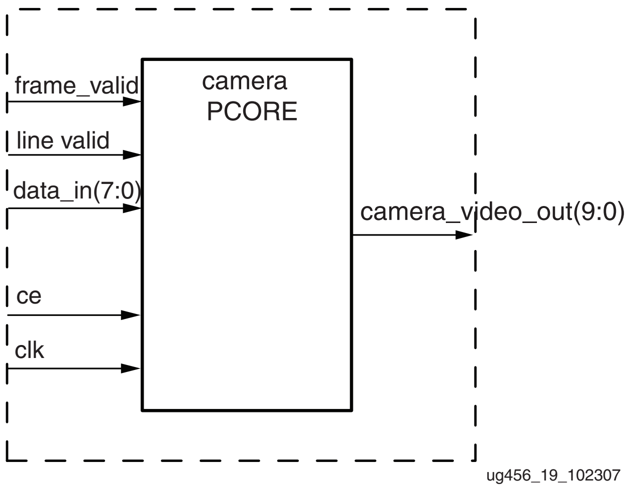


Figure B-1: **camera PCORE**

Functional Description

The camera PCORE brings in the input signals from the input chip, registers the signals, and groups the video signals into a unified bus that can be connected to other EDK PCOREs for processing. The camera PCORE receives video inputs from the MT9V022 Digital Image Sensor.

A bus interface called CAMERA_VIDEO_OUT has been defined for the camera PCORE outputs. In EDK, these outputs can be accessed either individually by their signal names or as a group by their bus interface name.

Modes

The camera PCORE supports only configuration of input signals.

Resolutions

The camera PCORE supports the following resolutions:

- 640x480P @ 60 Hz (27 MHz)
- 720x480P @ 60 Hz (27 MHz)

IIC Programming for the camera PCORE

The MT9V022 Digital Image Sensor that is associated with the camera PCORE must be properly initialized to function correctly. It also has a number of image processing capabilities that can be controlled by you. The image sensor can be controlled by an IIC programming. The IIC programming is not handled by the camera PCORE. The MicroBlaze™ processor performs the IIC processing by way of the XPS_IIC PCORE. You can take advantage of the functions provided in "Libraries and Drivers," but ultimately you will be responsible for verifying that the proper IIC programming is performed for your application.

de_gen (v2.00a)

Introduction

The de_gen PCORE provides the ability to generate a Data Enable signal as well as Field signals for video streams that do not have them.

de_gen I/O Signals

Table B-2 lists and describes the de_gen I/O signals.

Table B-2: **de_gen I/O Signals**

Signal	Direction	Bus Interface	Description
de_i	I	DVI_VIDEO_IN	Data Enable input
vsync_i	I	DVI_VIDEO_IN	Vertical Sync input
hsync_i	I	DVI_VIDEO_IN	Horizontal Sync input
red_i(7:0)	I	DVI_VIDEO_IN	Red input
green_i(7:0)	I	DVI_VIDEO_IN	Green input
blue_i(7:0)	I	DVI_VIDEO_IN	Blue input
enable	I	–	Clock Enable (active High)
clk	I	–	Clock input. (Pixel Rate)
reset	I	–	Reset (active High)
field_o	O	–	Field output (even=0, odd=1)
de_o	O	DVI_VIDEO_OUT	Data Enable output
vsync_o	O	DVI_VIDEO_OUT	Vertical Sync output
hsync_o	O	DVI_VIDEO_OUT	Horizontal Sync output
red_o(7:0)	O	DVI_VIDEO_OUT	Red output
green_o(7:0)	O	DVI_VIDEO_OUT	Green output
blue_o(7:0)	O	DVI_VIDEO_OUT	Blue output

Note: This block additionally has a PLB interface for connection to a MicroBlaze processor. This interface is used for software configuration of registers and status reporting.

Functional Description

The de_gen PCORE generates a Data Enable signal when the video source is analog. The Data Enable signal indicates when active video is present and needs to be written to external memory. It does this by analyzing the input HSYNC and VSYNC signals combined with front porch and back porch values. The MicroBlaze processor writes the porch values to the block over the PLB interface. The de_gen PCORE is inactive when the video source is DVI, because the Data Enable signal is already generated by the source.

The Vertical Back Porch value includes all of the clock cycles between the active edge of HSYNC and the first active video pixel. This includes the Vertical Back Porch, the HSYNC pulse width, and the border preceding the first active video pixel. The Vertical Front Porch value includes all of the clock cycles between the last active video pixel and the active edge of HSYNC. This includes the Vertical Front Porch and the border following the last active video pixel. The Horizontal Back Porch value includes all of the lines of data between the active edge of VSYNC and the first line of active video. This includes the Horizontal Back Porch, the VSYNC pulse width, and the border preceding the first line of active video. The Horizontal Front Porch value includes all of the lines of data between the last line of active video and the active edge of VSYNC. This includes the Horizontal Front Porch as well as the border following the last line of active video.

The de_gen PCORE is capable of handling interlaced video formats and generates a Field signal that denotes whether the current output is the even field or the odd field. For progressive video formats, the Field signal remains Low. The vsync, hsync, red, green, and blue signals pass through de_gen unchanged, but are delayed to match DE.

Bus interfaces called DVI_VIDEO_IN and DVI_VIDEO_OUT have been defined for the de_gen PCORE. In EDK, these signals can either be accessed individually by their signal names or as a group by their bus interface name.

Resolutions

The de_gen PCORE supports the following resolutions:

- 640x480P @ 60 Hz
- 20x480P @ 60 Hz
- 720x480I @ 60 Hz
- 720x576P @ 50 Hz
- 720x576I @ 50 Hz
- 800x600P @ 60 Hz
- 1024x768P @ 60 Hz
- 1280x720P @ 60 Hz
- 1280x720P @ 50 Hz
- 1920x1080I @ 50 Hz
- 1920x1080I @ 60 Hz
- 1920x1080P @ 25 Hz
- 1920x1080P @ 30 Hz

display_controller (v3.00a)

Introduction

The display controller PCORE reads video frames out of memory and displays them to a DVI/VGA or S-Video display at any resolution requiring up to a 74.25 MHz pixel clock.

display_controller I/O Signals

Table B-3 lists and describes the display_controller I/O signals.

Table B-3: **display_controller I/O Signals**

Signal	Direction	Bus Interface	Description
ce	I	–	Clock enable input
display_clk	I	–	IP clock input – pixel rate
gray_frame_count(2:0)	I	–	Gray coded frame count input
rd0_almost_empty	I	XIL_VFBC	Almost empty on VFBC read FIFO
rd0_data(31:0)	I	XIL_VFBC	Data input from VFBC read port
rd0_clk	O	XIL_VFBC	Clock output to VFBC read port
rd0_reset	O	XIL_VFBC	Reset output to VFBC read port
rd0_read	O	XIL_VFBC	Read output to VFBC read port
rd0_end_burst	O	XIL_VFBC	End burst output to VFBC read port
cmd0_clk	O	XIL_VFBC	Clock output to VFBC command port
cmd0_data(31:0)	O	XIL_VFBC	Data output to VFBC command port
cmd0_end	O	XIL_VFBC	End output to VFBC command port
cmd0_reset	O	XIL_VFBC	Reset output to VFBC command port
cmd0_write	O	XIL_VFBC	Write output to VFBC command port
pixel_clk	O	–	Clock output to the dvi_out block
de_o	O	DVI_VIDEO_OUT	Data Enable output
vsync_o	O	DVI_VIDEO_OUT	Vsync output
hsync_o	O	DVI_VIDEO_OUT	Hsync output
red_o(7:0)	O	DVI_VIDEO_OUT	Red output
green_o(7:0)	O	DVI_VIDEO_OUT	Green output
blue_o(7:0)	O	DVI_VIDEO_OUT	Blue output

Note: This block additionally has a PLB interface for connection to a MicroBlaze processor. This interface is used for software configuration of registers and status reporting.

Functional Description

The display_controller PCORE generates properly timed video synchronization signals for any of the given resolutions based on software configuration registers and the frequency of the display clock input. The display clock is generated by an external clock generation chip. It also reads video data from frame buffers via the VFBC. When the PCORE generates a VSYNC signal, the block looks at the gc_frame_count signal to determine which frame buffer it needs to read from next. It then sends a command to the VFBC to initiate a read of the frame. When the block asserts the DE signal, it reads video data from the VFBC read port FIFO. This process is continued until the next VSYNC is issued indicating the current frame is finished.

Modes

Output Modes

- Progressive
- Interlaced

Pixel Formats

- RGB 4:4:4
- YCbCr 4:2:2

Resolutions

Software configurable registers are used for the locations of the frame buffers, for holding VFBC commands, and for video sync timing information. This gives the display_controller the flexibility to support a wide range of resolutions. This is limited by the frequency of the display clock (currently limited to 74.25 MHz) and the width of the timing registers. The highest resolution supported is 1920x1080P.

Only the following resolutions are tested:

- 640x480P @ 60 Hz
- 720x480P @ 60 Hz
- 720x480I @ 60 Hz
- 720x576P@ 50 Hz
- 720x576I @ 50 Hz
- 800x600P @ 60 Hz
- 1024x768P @ 60 Hz
- 1280x720P @ 60 Hz
- 1280x720P @ 50 Hz
- 1920x1080I @ 50 Hz
- 1920x1080I @ 60 Hz
- 1920x1080P @ 25 Hz
- 1920x1080P @ 30 Hz

dvi_in (v2.00a)

Introduction

The dvi_in PCORE provides a connection to the TFP403 DVI Receiver chip, as well as to the AD9984A Video DAC chip on the FMC-Video card. Table B-4 provides the dvi_in I/O signals and their descriptions.

Table B-4: **dvi_in I/O Signals**

Signal	Direction	Bus Interface	Description
de	I	–	Data Enable input. (Active video)
vsync	I	–	Vertical Sync input
hsync	I	–	Horizontal Sync input
red(7:0	I	–	Red input
green(7:0)	I	–	Green input
blue(7:0)	I	–	Blue input
mode	I	–	Output Mode (1=digital mode, 0=analog mode)
clk	I	–	Clock input. (Pixel Rate)
ce	I	–	Clock enable (active High)
de_o	O	DVI_VIDEO_OUT	Data Enable output. (Active video)
vsync_o	O	DVI_VIDEO_OUT	Vertical Sync output
hsync_o	O	DVI_VIDEO_OUT	Horizontal Sync output
red_o (7:0)	O	DVI_VIDEO_OUT	Red output
green_o (7:0)	O	DVI_VIDEO_OUT	Green output
blue_o (7:0)	O	DVI_VIDEO_OUT	Blue output

Functional Description

The dvi_in PCORE brings in the input signals from the input chip, registers the signals, and groups the video signals into a unified bus that can be connected to other EDK PCOREs for processing. See Figure B-2. The dvi_in PCORE receives video inputs from either the TFP403 or the AD9984A. The TFP403 is used if the input signal is digital. The AD9984A is used if the input signal is analog.

A bus interface called DVI_VIDEO_OUT has been defined for the dvi_in PCORE outputs. In EDK, these outputs can be accessed either individually by their signal names or as a group by their bus interface name.

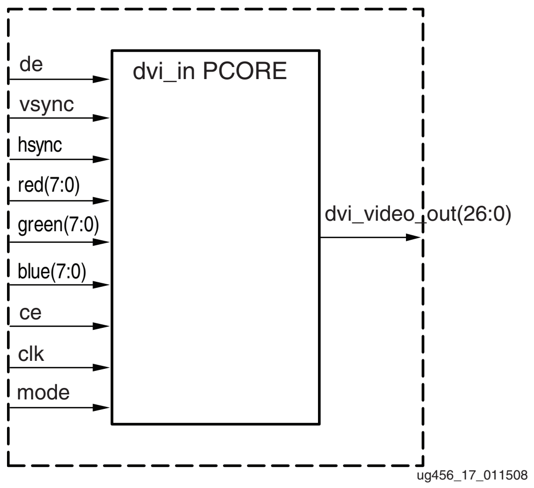


Figure B-2: **dvi_in PCORE**

Modes

The dvi_in PCORE supports one digital video format and one analog video format. The selected mode determines the usage of the DVI_VIDEO_OUT bus as shown in Table B-5. The mode is selected by using the IIC bus to initialize the VSK.

Table B-5: **dvi_in Modes**

DVI_VIDEO_OUT(26:0)	4:4:4 RGB DVI	4:4:4 RGB VGA
26	de	N/A
25	Vsync	Vsync
24	Hsync	Hsync
23:16	Red(7:0)	Red(9:2)
15:8	Green(7:0)	Green(9:2)
7:0	Blue(7:0)	Blue(9:2)

Resolutions

The dvi_in PCORE supports the following resolutions.

Digital Inputs

- 800x600P @ 60 Hz (40.0 MHz)
- 1024x768 @ 60 Hz (65.0 MHz)
- 720P @ 50 Hz (74.25 MHz)
- 720P @ 60 Hz (74.25 MHz)
- 1080P @ 25 Hz (74.25 MHz)
- 1080P @ 30 Hz (74.25 MHz)
- 1080I @ 50 Hz (74.25 MHz)
- 1080I @ 60 Hz (74.25 MHz)

Analog Inputs

- 640x480P @ 60 Hz (25.175 MHz)
- 800x600P @ 60 Hz (40 MHz)
- 1024x768P @ 60 Hz (65 MHz)
- 720P @ 50 Hz (74.25 MHz)
- 720P @ 60 Hz (74.25 MHz)

IIC Programming for the dvi_in PCORE

The dvi_in PCORE supports multiple modes and video resolutions. The IIC programming is used to set the VSK into the desired mode and resolution. The IIC programming is not handled by the dvi_in PCORE. The MicroBlaze processor performs the IIC processing by way of the XPS_IIC PCORE. You can take advantage of the functions provided in the API Reference, but ultimately you will be responsible for verifying that the proper IIC programming is performed for your application.

dvi_out (v3.00a)

Introduction

The dvi_out PCORE provides a connection to the CH7301C DVI Transmitter Device on the Xilinx Video Starter Kit. Table B-6 provides the dvi_out I/O signals and their descriptions. See Figure B-3.

Table B-6: **dvi_out I/O Signals**

Signal	Direction	Bus Interface	Description
de_i	I	DVI_VIDEO_IN	Data Enable input. (Active video)
vsync_i	I	DVI_VIDEO_IN	Vertical Sync input
hsync_i	I	DVI_VIDEO_IN	Horizontal Sync input
red_i (7:0)	I	DVI_VIDEO_IN	Red input
green_i (7:0)	I	DVI_VIDEO_IN	Green input
blue_i (7:0)	I	DVI_VIDEO_IN	Blue input
clk	I	–	Clock input. (Pixel Rate)
ce	I	–	Clock enable (active High)
de	O	–	Data Enable (Active Video)
vsync	O	–	Vertical Sync
hsync	O	–	Horizontal Sync
dvi_data(11:0)	O	–	Data output
dvi_clk_p	O	–	DVI Clock (positive phase)
dvi_clk_n	O	–	DVI Clock (negative phase)
reset_n	O	–	DVI Reset (active Low)

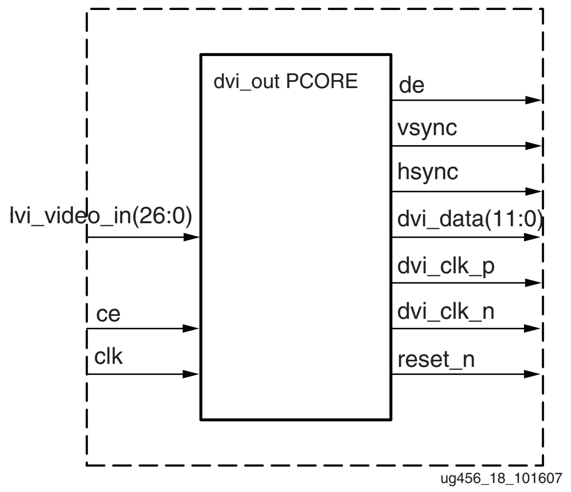


Figure B-3: **dvi_out PCORE**

Functional Description

The dvi_out PCORE brings in the DVI_VIDEO_IN bus and formats the video data to the format required by the CH7301C. The CH7301 is capable of driving either digital DVI displays or analog VGA displays. When driving digital displays, the de, vsync, and hsync signals are required. For analog displays, the vsync and hsync signals are required.

A bus interface called DVI_VIDEO_IN has been defined for some of the dvi_out PCORE inputs. In EDK, these inputs can be accessed either individually by their signal names or as a group by their bus interface name.

Modes

The dvi_out PCORE supports one video format. The DVI_VIDEO_IN bus is used as shown in Table B-7.

Table B-7: **dvi_out Modes**

DVI_VIDEO_IN(26:0))	4:4:4 RGB
26	de
25	Vsync
24	Hsync
23:16	Red(7:0)
15:8	Green(7:0)
7:0	Blue(7:0)

Resolutions

The dvi_out PCORE supports the following resolutions:

- 640x480P @ 60 Hz (25.175 MHz)
- 720x576P @ 50 Hz (27 MHz)
- 720x480P @ 60 Hz (27 MHz)
- 800x600P @ 60 Hz (40.0 MHz)
- 1024x768 @ 60 Hz (65.0 MHz)
- 720P @ 50 Hz (74.25 MHz)
- 720P @ 60 Hz (74.25 MHz)
- 1080P @ 25 Hz (74.25 MHz)
- 1080P @ 30 Hz (74.25 MHz)
- 1080I @ 50 Hz (74.25 MHz)
- 1080I @ 60 Hz (74.25 MHz)

IIC Programming for the dvi_out PCORE

The dvi_out PCORE supports multiple modes and video resolutions. The IIC programming is used to set the VSK into the desired mode and resolution. The IIC programming is not handled by the dvi_out PCORE. The MicroBlaze processor performs the IIC processing by way of the XPS_IIC PCORE. You can take advantage of the functions provided in the API Reference, but ultimately you will be responsible for verifying that the proper IIC programming is performed for you application.

gamma_plbw (v3.00b)

Introduction

The gamma_plbw PCORE can be used to compensate for Gamma effects associated with video displays.

gamma_plbw I/O Signals

Table B-8 lists and describes the gamma_plbw I/O signals.

Table B-8: **gamma_plbw I/O Signals**

Signal	Direction	Bus Interface	Description
de_i	I	DVI_VIDEO_IN	Data Enable input
vsync_i	I	DVI_VIDEO_IN	Vertical Sync input
hsync_i	I	DVI_VIDEO_IN	Horizontal Sync input
red_i(7:0)	I	DVI_VIDEO_IN	Red input
green_i(7:0)	I	DVI_VIDEO_IN	Green input
blue_i(7:0)	I	DVI_VIDEO_IN	Blue input
clk	I	–	
de_o	O	DVI_VIDEO_OUT	Clock input (Pixel Rate)

Table B-8: **gamma_plbw I/O Signals *(Cont'd)***

Signal	Direction	Bus Interface	Description
vsync_o	O	DVI_VIDEO_OUT	Vertical Sync output
hsync_o	O	DVI_VIDEO_OUT	Horizontal Sync output
red_o(7:0)	O	DVI_VIDEO_OUT	Red output
green_o(7:0)	O	DVI_VIDEO_OUT	Green output
blue_o(7:0)	O	DVI_VIDEO_OUT	Blue output

Note: This PCORE additionally has a PLB interface for connection to a MicroBlaze processor. This interface is used for software configuration of registers and status reporting.

Functional Description

The gamma_plbw PCORE provides the ability to adjust streaming video by any pre-computed formula. The red, green, and blue channels can be adjusted independently. The gamma_plbw PCORE uses memory based look-up tables that can be read or written by the MicroBlaze processor in real-time.

Bus interfaces called DVI_VIDEO_IN and DVI_VIDEO_OUT have been defined for the gamma_plbw PCORE. In EDK, these signals can be accessed either individually by their signal names or as a group by their bus interface name.

This PCORE buffers the gamma settings and updates only the values currently being used during blanking intervals.

Resolutions

The gamma_plbw PCORE supports the following resolutions:

- 40x480P @ 60 Hz
- 20x480P @ 60 Hz
- 720x480I @ 60 Hz
- 720x576P @ 50 Hz
- 720x576I @ 50 Hz
- 800x600P @ 60 Hz
- 1024x768P @ 60 Hz
- 1280x720P @ 60 Hz
- 1280x720P @ 50 Hz
- 1920x1080I @ 50 Hz
- 1920x1080I@ 60 Hz
- 1920x1080P @ 25 Hz
- 1920x1080P @ 30 Hz

sdtv_in (v2.00a)

Introduction

The sdtv_in PCORE receives an 8-bit ITU-R BT.656 YCrCb 4:2:2 video stream by providing a connection to the AD7180 chip on the FMC-Video card.

sdtv_in I/O Signals

Table B-9 lists and describes the sdtv_in I/O signals.

Table B-9: **sdtv_in I/O Signals**

Signal	Direction	Bus Interface	Description
vsync_in	I	–	Vertical Sync input
hsync_in	I	–	Horizontal Sync input
sdtv_data(7:0)	I	–	Red pixel input
clk	I	–	Clock input. (Pixel Rate)
ce	I	–	Clock enable (active high)
vsync_out	O	DVI_VIDEO_OUT	Vertical Sync output
hsync_out	O	DVI_VIDEO_OUT	Horizontal Sync output
video_out(7:0)	O	DVI_VIDEO_OUT	Video output

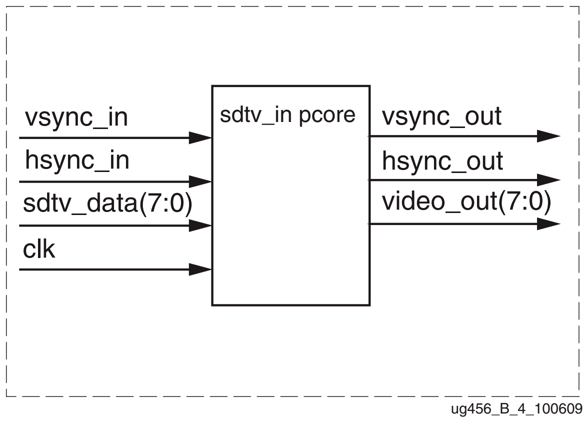


Figure B-4: **sdtv_in I/O Signals**

Functional Description

The sdtv_in PCORE brings in the input signals from the input chip, registers the signals, and then groups the video signals into a unified bus that can be connected to other EDK PCOREs for processing. The sdtv_in PCORE receives video inputs from ADV7180.

The clock driving this PCORE is 2x the pixel rate of the video format. The 8-bit data bus contains both the Y (Luma) and C (Chroma) data in the following multiplexed format:

```
Cb1, Y1, Cr1, Y2, Cb3, Y3, Cr3, Y4, Cb5, Y5, Cr5, Y6, …
```

Modes

The sdtv_in PCORE supports the YCrCb 4:2:2 video format for NTSC and PAL standards. See Table B-10.

Table B-10: **sdtv_in Modes**

video_out (9:0)	4:2:2 YCbCr
9	Vsync
8	Hsync
7:0	Ycbcr(7:0)

Resolutions

The sdtv_in PCORE supports the following standards:

- NTSC
- PAL

IIC Programming for the sdtv_in PCORE

The sdtv_in PCORE supports multiple modes and video resolutions. The IIC programming is used to set the VSK into the desired mode and resolution. The IIC programming is not handled by the sdtv_in PCORE. The MicroBlaze processor performs the IIC processing by way of the XPS_IIC PCORE and software driver. You can reuse functions provided in the API Reference, but ultimately you will be responsible for verifying that the proper IIC programming is performed for your application.

sdtv_out (v3.00a)

Introduction

The sdtv_out PCORE transmits an 8-bit ITU-R BT.656 YCrCb 4:2:2 video stream to the ADV7179 on the FMC-Video.

sdtv_out I/O Signals

Table B-11 lists and describes the sdtv_out I/O signals.

Table B-11: **sdtv_out I/O Signals**

Signal	Direction	Bus Interface	Description
video_in(7:0)	I	DVI_VIDEO_IN	Video input bus
vsync_in	I	DVI_VIDEO_IN	Vertical Sync
hsync_in	I	DVI_VIDEO_IN	Horizontal Sync
clk	I	–	Clock input. (Pixel Rate)
ce	I	–	Clock enable (active High)
Vsync_out	O	–	Vertical Sync
Hsync_out	O	–	Horizontal Sync
sdtv_data(7:0)	O	–	Data output
sdtv_clk	O	–	DVI Clock

Functional Description

The sdtv_out PCORE brings in the video_in bus and formats the video data to the format required by the ADV7179.

The clock driving this PCORE is 2x the pixel rate of the video format. The 8-bit data bus contains both the Y (Luma) and C (Chroma) data in the following multiplexed format.

```
Cb1, Y1, Cr1, Y2, Cb3, Y3, Cr3, Y4, Cb5, Y5, Cr5, Y6, …
```

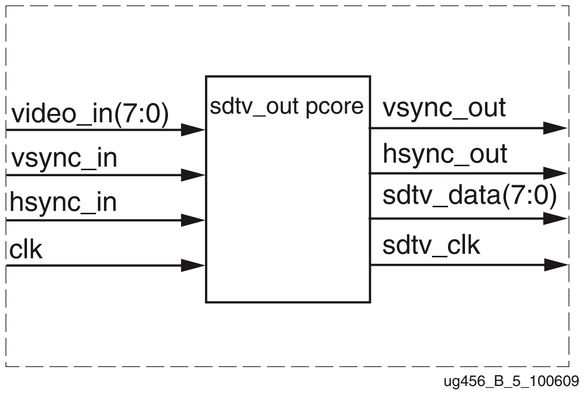


Figure B-5: **sdtv_out I/O Signals**

Modes

The sdtv_out PCORE supports the YCrCb 4:2:2 video format.

Table B-12: **sdtv_out Modes**

video_in (9:0)	4:2:2 YCbCr
9	vsync
8	hsync
7:0	ycbcr(7:0)

Resolutions

The sdtv_out PCORE supports the following standards:

- NTSC
- PAL

IIC Programming for the sdtv_out PCORE

The sdtv_out PCORE supports multiple modes and video resolutions. The IIC programming is used to set the VSK into the desired mode and resolution. The IIC programming is not handled by the sdtv_out PCORE. The MicroBlaze processor performs the IIC processing by way of the XPS_IIC PCORE with software driver. You can reuse functions provided in the API Reference, but ultimately you will be responsible for verifying that the proper IIC programming is performed for your application.

video_to_vfbc (v2.00a)

Introduction

The VIDEO_TO_VFBC PCORE manages the storing of DVI formatted video into frame buffers. It writes the video data to the VFBC interface on the MPMC memory controller. Using the sync and DE signals, the VIDEO_TO_VFBC generates properly timed command words to the VFBC.

video_to_vfbc I/O Signals

Table B-13 lists and describes the video_to_vfbc I/O signals.

Table B-13: **video_to_vfbc I/O Signals**

Signal	Direction	Bus Interface	Description
ip_clk	I	–	IP clock input
reset	I	–	Reset input
ddr_rdy	I	–	Indicates when the memory controller has finished initialization
dcm_locked	I	–	Indicates when the DCM that supplies the ip_clk has reached lock
freeze	I	–	Stops the PCORE from incrementing its frame counter

Table B-13: **video_to_vfbc I/O Signals *(Cont'd)***

Signal	Direction	Bus Interface	Description
field	I	–	Field input (even=0, odd=1)
de_i	I	DVI_VIDEO_IN	Data Enable input
vsync_i	I	DVI_VIDEO_IN	Vsync input
hsync_i	I	DVI_VIDEO_IN	Hsync input
red_i(7:0)	I	DVI_VIDEO_IN	Red input
green_i(7:0)	I	DVI_VIDEO_IN	Green input
blue_i(7:0)	I	DVI_VIDEO_IN	Blue input
wd0_clk	O	XIL_VFBC	Clock output to VFBC write port
wd0_reset	O	XIL_VFBC	Reset output to VFBC write port
wd0_write	O	XIL_VFBC	Write output to VFBC write port
wd0_data(31:0)	O	XIL_VFBC	Data output to VFBC write port
wd0_data_be(3:0)	O	XIL_VFBC	Byte enable output to VFBC write port
cmd0_clk	O	XIL_VFBC	Clock output to VFBC command port
cmd0_reset	O	XIL_VFBC	Reset output to VFBC command port
cmd0_write	O	XIL_VFBC	Write output to VFBC command port
cmd0_data(31:0)	O	XIL_VFBC	Data output to VFBC command port
gc_frame_count(2:0)	O	–	Gray coded frame count output
end_of_frame	O	–	End of frame output
last_frame_done	O	–	Output indicates frame capture has completed

Note: This block additionally has a PLB interface for connection to a MicroBlaze processor. This interface is used for software configuration of registers and status reporting.

Functional Description

When a VSYNC signal is received a new video frame is about to arrive. The PCORE responds by incrementing to the next frame buffer to be written to and issues a command to the VFBC to start writing to the new buffer. It then waits until it receives the DE (Data Enable) signal, indicating active video, at which time it writes the data to a VFBC data FIFO. The PCORE continues the process of writing data to the VFBC until it receives the next VSYNC. The PCORE also sends a gray coded frame index to the display controller to facilitate frame synchronization.

Modes

Input Modes

- Progressive
- Interlaced (weaved)

Pixel Formats

- RGB 4:4:4
- YCbCr 4:2:2

Resolutions

Software configurable registers are used for the locations of the frame buffers and for holding VFBC commands. This gives the video_to_vfbc the flexibility to support a wide range of resolutions.

Only the following resolutions are tested:

- 640x480P @ 60 Hz
- 720x480P @ 60 Hz
- 720x480I @ 60 Hz
- 720x576P @ 50 Hz
- 720x576I @ 50 Hz
- 800x600P @ 60 Hz
- 1024x768P @ 60 Hz
- 1280x720P @ 60 Hz
- 1280x720P @ 50 Hz
- 1920x1080I @ 50 Hz
- 1920x1080I @ 60 Hz
- 1920x1080P @ 25 Hz
- 1920x1080P @ 30 Hz

Libraries and Drivers

For details on software libraries and drivers, see *Spartan-3A DSP FPGA Video Starter Kit Software Guide, UG514* [Ref 6].

Creating a Compact Flash for the Demo

Downloading a Bitstream

Before creating a Compact Flash image, it is recommended that you download and test the design first. The following steps will walk you through the process of downloading and running your design, using the Camera Frame Buffer as an example.

1. Launch EDK XPS program:
   All Program →Xilinx ISE Design Suite →EDK →Xilinx Platform Studio
2. Open the Camera Frame Buffer design:
   EDK_Demonstrations\Camera_Frame_Buffer_Demo\system.xps
3. Launch HyperTerminal program:
   All Program →Accessories →Communications →HyperTerminal
4. From within the EDK XPS software, execute the command:
   Device Configuration →Download Bitstream

   This will download the existing bitstream to the board and program the FPGA.
5. Execute the command:
   Software →Build All User Applications

   This will re-compile the software and create an .exe file that can be download to the board using the debugger XMD window.
6. In EDK, execute the pull-down menu command:
   Debug →Launch XMD

   The embedded debugger environment will be used to download the new bitstream to the VSK.

   Note: If this is the first time XMD has been run, a dialog box will appear asking you to set XMD Debug options. Click **OK**.
7. If the "XMD Debug Options" dialog box appears (Figure C-1), make sure the default options are set and click **OK**.

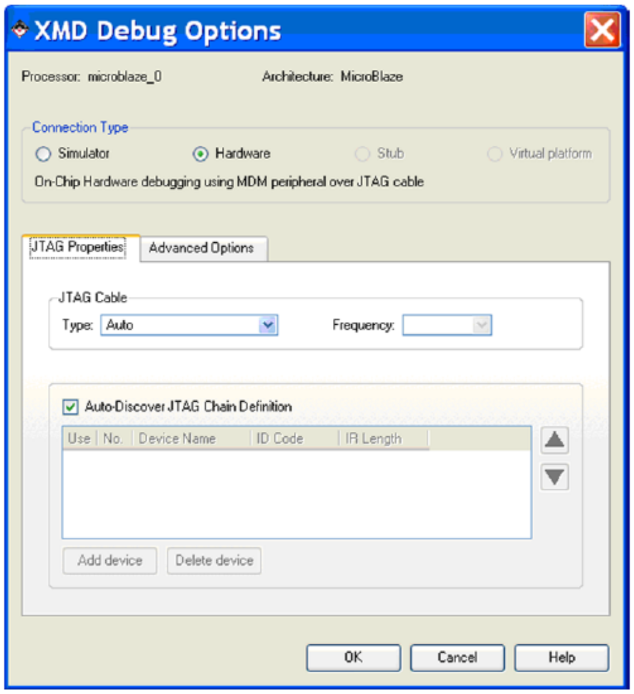
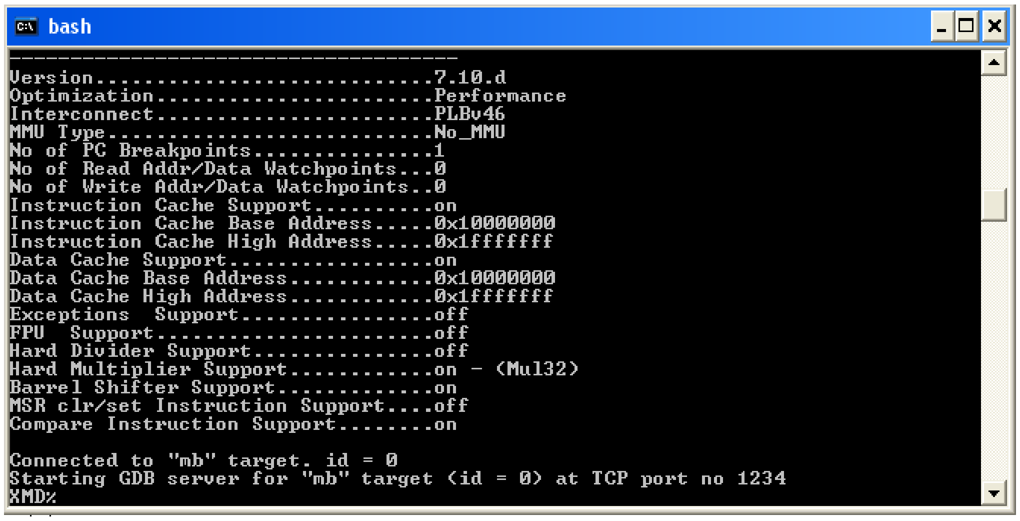


Figure C-1: **Programming FPGA with Bitstream**

After some time, the XMD prompt appears; you should see feedback shown in Figure C-2.



```
--------------------------------------
Version............................7.10.d
Optimization.......................Performance
Interconnect.......................PLBv46
MMU Type...........................No_MMU
No of PC Breakpoints...............1
No of Read Addr/Data Watchpoints...0
No of Write Addr/Data Watchpoints..0
Instruction Cache Support..........on
Instruction Cache Base Address.....0x10000000
Instruction Cache High Address.....0x1fffffff
Data Cache Support.................on
Data Cache Base Address............0x10000000
Data Cache High Address............0x1fffffff
Exceptions  Support................off
FPU  Support.......................off
Hard Divider Support...............off
Hard Multiplier Support............on - (Mul32)
Barrel Shifter Support.............on
MSR clr/set Instruction Support....off
Compare Instruction Support........on

Connected to "mb" target. id = 0
Starting GDB server for "mb" target (id = 0) at TCP port no 1234
XMD%
```

Figure C-2: **XMD Command Prompt Window**

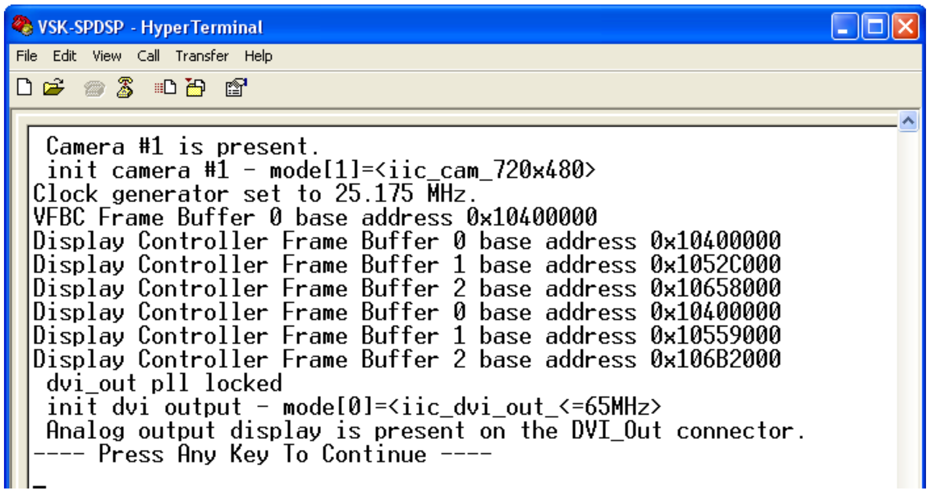
8. To load the new bitstream into the VSK Spartan®-3A DSP FPGA, type the following commands at the XMD prompt:

```
XMD% stop

XMD% dow Camera_Frame_Buffer_Sw/executable.elf

XMD% run
```

After typing "run" at the HyperTerminal, you should see the following messages:



```
 Camera #1 is present.
 init camera #1 - mode[1]=<iic_cam_720x480>
Clock generator set to 25.175 MHz.
VFBC Frame Buffer 0 base address 0x10400000
Display Controller Frame Buffer 0 base address 0x10400000
Display Controller Frame Buffer 1 base address 0x1052C000
Display Controller Frame Buffer 2 base address 0x10658000
Display Controller Frame Buffer 0 base address 0x10400000
Display Controller Frame Buffer 1 base address 0x10559000
Display Controller Frame Buffer 2 base address 0x106B2000
 dvi_out pll locked
 init dvi output - mode[0]=<iic_dvi_out_<=65MHz>
 Analog output display is present on the DVI_Out connector.
---- Press Any Key To Continue ----
_
```

9. Hit any key to being up the software interface menu:

```
-------------------------------------------------------
--Xilinx Video Starter Kit Camera Frame Buffer Demo --
-------------------------------------------------------

 c   Enter Camera configuration menu
 p   Enter Processing menu
 s   Enter Storage menu
 i   Enter the IIC Diagnostics menu
 ?   Print the Top-Level menu Help Screen
-------------------------------------------------------
>
```

You have now verified that your design is working and you are ready to create a Compact Flash demo.

Compact Flash

The Compact Flash card provided comes preloaded with the FPGA configuration bit streams and the bootloader design in the form of ACE files. In the event that it is accidentally erased or a duplicate is required, the following step copies the demo onto a new Compact Flash card.

Note: To work with the Spartan-3A DSP 3400A Development Platform, the Compact Flash card must be formatted with the FAT12 or FAT16 file system. For more information, see AR #14456.

Using a Compact Flash card reader/writer, copy the contents from the `VSK_CD:\Demonstrations\SystemACE` folder of the demo distribution CD to a blank FAT12 or FAT16 formatted Compact Flash card.

Troubleshooting

If the demo fails to run as expected, the following diagnostic procedures can help isolate the likely cause. These procedures assume that all the steps in Chapter 2, "System Setup," have been followed and the system has been powered on in the order given. See Table 2-1 for correct DIP switch settings.

Test 1

1. Confirm that the FPGA on the Spartan®-3A 3400A DSP Development Platform has been correctly configured with the bit stream.
2. Make sure the DONE LED (DS6) is lit on the Spartan-3A DSP 3400A Development Platform (above the bank of configuration DIP switches). If this is not the case, the problem could be:
   a. The CF card has not been programmed with a bit stream.
   b. The CF card was formatted with FAT32 rather than FAT12 or FAT16.

Test 2

1. Confirm that the FMC Video card installed on the Spartan-3A DSP 3400A Development Platform is functioning properly.
2. Check that there are two green LEDs (Power Good and Status) lit on the FMC Video card after the system has been powered up. If this is not the case, the problem could be:
   a. The FMC Video card is not being powered properly and may be loose.
   b. The FMC Video card has not been initialized by the FPGA. Try resetting the board and reloading the demo.

Test 3

1. Confirm serial connectivity between the PC and the FPGA on the Spartan-3A DSP 3400A Development Platform.
2. If you do not see characters in the serial terminal upon powering up the board, then it is likely one of the following is the problem:
   a. The serial cable is loose.
   b. The terminal settings (see Chapter 2, "System Setup") on the computer have not been configured correctly.

Test 4

1. Confirm that the display is connected and powered up.
2. When the demo is started up, you should see video on the display as well as text in the terminal window stating that the display has been detected. If video is not shown or the display is not detected, then it is likely that one of the following is the problem:
   a. The power connection to the display is loose or disconnected.
   b. DVI cable connection is loose or disconnected.
   c. The resolution and frame rate being displayed is not supported by the display (consult the documentation for the display to verify that it is supported).

Test 5

1. Confirm that DVI video source is connected and transmitting video (applies only to "DVI Pass-Through Demo" and "DVI Frame Buffer Demo" designs).
2. When the demo is started up, you should see text in the terminal window stating the video source has been detected. If video source is not detected, then it is likely that one of the following is the problem:
   a. The video source is no longer transmitting. Try connecting the video source directly to a display to confirm the video signal is still active.
   b. DVI cable connection is loose or disconnected.
   c. The resolution and frame rate being transmitted is not supported by the demo. Confirm the settings are for one of the supported resolution/frame rates for the given demo.

Test 6

1. Confirm that demo software has detected the camera (applies only to "Camera Frame Buffer Demo" design).
2. When the demo is started up, you should see in the text displayed in the terminal window that the camera has been detected and configured. The default is to use the camera 1 input. If this is not the case, then it is likely one of the following is the problem:
   a. The CAT6 cable is loose.
   b. The CAT6 cable is not straight-through, but rather crossed over.
   c. The FMC Video card is malfunctioning (see "Test 2").
   d. The camera is damaged.

Test 7

1. Confirm that S-Video source is connected and transmitting video (applies only to the "S-Video Frame Buffer Demo" design).
2. When the demo is started up, you should see in the text displayed in the terminal window that the video source has been detected. If video source is not detected, then it is likely that one of the following is the problem:
   a. The video source is no longer transmitting. Try connecting the video source directly to a display to confirm the video signal is still active.
   b. S-Video cable connection is loose or disconnected.

Test 8

1. Confirm that the EDID of the VSK has been properly programmed applies only to "DVI Pass-Through Demo" and "DVI Frame Buffer Demo" designs).
2. When the VSK DVI input is connected to the DVI/VGA output port of your PC and you are unable to select one or more of the supported resolutions, then it is possible that the VSK EDID is improperly programmed. Use the following steps to update it:
   a. Follow the steps in "Connect the PC" in Chapter 2, "System Setup."
   b. If a cable is connected to the VSK DVI input, disconnect it.
   c. Load either the "DVI Pass-Through Demo" or "DVI Frame Buffer Demo" design using the Bootloader menu.
   d. Using a terminal program, type "E" at the main menu.
   e. Choose option "0"to program the default EDID.
   f. Reconnect the cable from your PC to the VSK DVI input.
   g. It may be necessary to reboot the PC with VSK connected so that the EDID can be properly recognized.