Zynq-7000 All Programmable SoC ZC702 Base Targeted Reference Design (ISE Design Suite 14.4)

User Guide

UG925 (v3.0) January 31, 2013

XILINX®

Notice of Disclaimer

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
06/21/12	1.0	Initial Xilinx release.
10/08/12	2.0	Updated for ISE® Design Suite 14.2. Replaced AXI VTC with VTC throughout. Changed Sobel engine to Sobel filter. Chapter 1, Introduction: Added R, RW, COR, SoC, and SC to Table 1-1. Expanded descriptions of the TRD. Updated the block diagram in Figure 1-1. Chapter 2, Functional Description: In Figure 2-1, replaced the continuous line with a dotted line for Monitor and between VIDEO_MUX0 and Clk_detect and revised AXI Interconnect inputs. Added slice registers to Table 2-1. In Table 2-2, changed SOBEL_ENGINE to FILTER_VDMA and FILTER_ENGINE. Added FILTER_0_interrupt row in Table 2-3, and SOBEL_VDMA changed to FILTER_VDMA. In Table 2-4, GPIO bit number 7 became N/A. In PL Clocks, page 18, (FCLKCLK[3:0]) became (FCLK_CLK). In PL Reset, page 18, (FCLKRESETN[3:0]) became FCLK_RESET[3:0]_N and FCLKRESETN[0] changed to FCLK_RESET0_N. In Clocking, page 18, PS FCLKCLK[0] changed to PS FCLK_CLK0. In Table 2-5, clock signal sources and uses were updated. In Table 2-7, added xFilter, changed frequency and connection of s_axi_CONTROL_BUS_ACLK and deleted s_axi_SOBEL_CONTROL_ACLK. Changed frequency of INTERCONNECT_ACLK bus. Changed SOBEL_ENGINE to FILTER_ENGINE and SOBEL_VDMA to FILTER_VDMA. Added information to the end of the table for DVI2AXI_0, CRESAMPLE_0, and YUV2RGB_0 components. In Processor System Reset Module, page 21, input to the proc_sys_reset core is generated by PS Proc_sys_reset_1_N. In Clock Detect, page 23, FMC changed to FMC-IMAGEON. The v_cresample, v_ycrcb2rgb, vsrc_sel, sobel_filter_top, and dvi2axi sections were added. Starting in fmc imageon hdmi in, page 25, YCbCr was changed to YCrCb. In Figure 2-2, added xFilter to the Device Drivers. In the Boot Loader section, removed "HDMI in" chip. In xFilter, page 31, updated the introductory text for clarity and reformatted the IOTCL arguments. Modified the description of Figure 2-4 for clarity. Modified the descriptions of Figure 2-6 and Figure 2-7 for clarity and added description of cases 4, 5, and 6. Modified Software Sobel Filter Processing, page 37 for clarity. Added xFilter sections to Xilinx Linux Kernel, page 28. Figure 2-4, GUI for TRD Application was replaced and notes about the GUI were added. AXI was added to the Plot Graph descriptions. DDR3 was removed from Figure 2-6 and Figure 2-7. The Software Sobel Filter Processing, page 37 description and API changed. Appendix A, Register Description: New sections were added from Sobel Filter Registers, page 39 to the end of the appendix. Removed redundant instances of TPG. Appendix B, Directory Structure: Figure B-1, Directory Structure, was updated. Appendix D, Additional Resources: PG012, *LogiCORE IP Chroma Resampler Product Guide* and PG014, *LogiCORE IP YCrCb to RGB Color-Space Converter Product Guide* were added to references. Appendix E, Regulatory and Compliance Information was added. Appendix F, Warranty was added.

Date	Version	Revision
11/12/12	2.1	Updated for ISE® Design Suite 14.3. Chapter 1, Introduction: Document and web site references changed throughout the book. In Base TRD Key Features, page 10, *1 GB DDR3 running at 533 MHz* was removed. Chapter 2, Functional Description: Figure 2-1 line colors changed. Table 2-1 cell values changed. Table 2-2 Peripheral names changed. Under Clocking, page 18, RGB2YUV block was removed. In Table 2-6, the processor is ps7_0, and PERF_MON_HP0, PERF_MON_HP2, VTC_0, and DVI2AXI_0 sections changed. The *dvi2axi Converter* heading changed to *dvi2axi* on page 25. Figure 2-4 was replaced and a GUI explanation added. Figure 2-5 was added to show minimized GUI mode. In section Graphical User Interface, page 33, the File Browser and Command Line Shell paragraphs were removed. The directory structure changed in Figure B-1. Appendix D, Additional Resources: Appendix was reorganized. A link was added for the Master Answer Record for the ZC702 board. Appendix E, Regulatory and Compliance Information: A Declaration of Conformity link was added.
11/19/12	2.1.1	Made typographical edits.
01/31/13	3.0	Updated for ISE® Design Suite 14.4. Chapter 2, Functional Description: Figure 2-1 was changed in the Performance Monitor and TPG_0 areas. Table 2-1 changed Used 19,052 to 19,805, 42 to 43, 23,881 to 24,094, and % of Used 35 to 37. Table 2-2, deleted PERF_MON_HP2 row and renamed next row to PERF_MON_HP2_HP0_HP2. APU, page 16, end of Cortex-A9 Core section added "For this TRD, both ARM cores run at 667 MHz." Table 2-6, PERF_MON_HP0 changed to PERF_MON_HP0_HP2. In that same section added new row SLOT_1_AXI_ACLK 150 0 Yes clk_150mhz and removed PERF_MON_HP2 section. In TPG_0 section, changed clk to aclk, 148 to 150, video_clk_int to clk_150mhz, S_AXI_ACLK to s_axi_aclk, and DVI2AXI_0 section to VID_IN_AXI4S. *Reset Module* changed to Processor System Reset Module, page 21. *AXI VDMA Video Direct Memory Access*, page 21 changed to AXI Video Direct Memory Access. Heading VTC on page 22 changed to Video Timing Controller. In the same section, VTC IP in last paragraph changed to "Video Timing Controller IP." Added final sentence in that section "For detailed information on ...". The *TPG* section changed to Test Pattern Generator, page 24. Section logiCVC-ML, page 24 was added. In AXI Performance Monitor, page 25 added the final sentence, "For more information..." In Chroma Resampler, page 25 edited the last sentence and added "and licensing information ..." AXI Performance Monitor, page 25 changed Instance: PERF_MON_HP0_HP2 and dropped 's' at Instance. The last sentence in the same section changed from "Two AXI Performance Monitors are instantiated" to "Two slots of AXI Performance Monitors are used." Video Multiplexer, page 26 added the final sentence "For detailed information ..." The *dvi2axi* section on page 25 was removed. Figure 2-2 changed "Xilinx DMA" block to "Xilinx AXI VDMA." Table 2-7 changed "core DMA" to "Xilinx AXI VDMA." XVDMA Driver, page 30 changed "Xilinx DMA" to "Xilinx AXI VDMA." XFILTER_STOP, page 32 bullet switched "On demand mode" with "Continuous mode" in both sentences. Application, page 32 added a new sentence at end "The Linux kernel is in SMP mode and both ..." Figure 2-4 and Figure 2-5 were replaced. In Control and Decision Making, page 35 the bullet "Sobel Filter (using the xFilter driver)" was added. In User Space Device Control, page 36 deleted "Sobel filter" bullet.

Date	Version	Revision
01/31/13, *continued*	3.0	Appendix A, Register Description: Removed the *AXI TPG Registers* section. Modified Table A-4 and added a row for Bit Position 7. Appendix B, Directory Structure: The directory structure in Figure B-1 changed. Appendix D, Additional Resources: Added references for PG043 and PG103.

XILINX

Table of Contents

Appendix F: Warranty

Introduction

This user guide describes the Base Targeted Reference Design (TRD) based on Zynq™-7000 All Programmable SoC (AP SoC) architecture. The TRD is included with the ZC702 evaluation kit. To learn more about the ZC702 evaluation kit and how to evaluate different demonstrations based on Zynq-7000 AP SoC architecture, refer to UG926, *Zynq-7000 All Programmable SoC: ZC702 Evaluation Kit and Video and Imaging Kit Getting Started Guide*.

This document explains the functional architecture of the hardware and software components of the TRD. Also provided are register descriptions for IP/logic implemented in programmable logic (PL), Base TRD package directory structure, and pointers to enable the user to further develop embedded platforms based on Zynq-7000 AP SoC architecture.

To build hardware and software for the Base TRD, refer to the Xilinx Zynq-7000 Base Targeted Reference Design page wiki.xilinx.com/zc702-base-trd.

The Base Targeted AP SoC Reference Design

The Base TRD showcases various features and capabilities of the Zynq AP SoC Z-7020 device for the embedded domain in a single package using a Xilinx standard Zynq Linux-based video pipeline design.

The Base TRD consists of two processing elements: The Zynq-7000 AP SoC processing system (PS) and a video accelerator based on PL. The AP SoC allows the user to implement a specific functionality either as a software program running on the Zynq-7000 AP SoC PS or as a hardware design inside the PL. The Base TRD demonstrates an optimization of how the user can seamlessly switch between a software or a hardware implementation, contributing to ease of use. The TRD also demonstrates the value of offloading computation-intensive tasks onto PL, thereby freeing the CPU resources available for user-specific applications.

Software developers can leverage the Base TRD and start programming right away using the widely known Eclipse-based integrated development environment (IDE), GNU compiler tool chain, Linux operating system (OS), and libraries. Embedded hardware designers now have immediate access to the industry standard ARM® Cortex™-A9 core processor system and video IPs running on PL out of the box. In addition, the TRD also provides customers with access to the various PL-based video components such as video direct memory access

(VDMA), video timing controller (VTC), video test pattern generator (TPG), Sobel filter, and the Xylon logiCVC-ML display controller [Ref 1]. These IPs are part of the video pipeline implemented in the PL.

The Base TRD consists of a PS based on the embedded ARM Cortex-A9 core processor, a video processing pipeline implemented in PL, and a Linux-based software application that includes a Qt-based graphical user interface (GUI) [Ref 2] to provide user control and monitoring. The Linux-based software platform and software application run on the ARM Cortex-A9 cores. The software application works in tandem with hardware and provides the user the choice of offloading computation-intensive processing to the PL-based hardware subsystem.

The Zynq-7000 AP SoC Base TRD reference design (Figure 1-1) consists of these components:

- A dual ARM Cortex-A9 core processor-based embedded Linux OS, board support package (BSP), and U-Boot boot loader
- PL-based hardware IPs that enable acceleration of computation-intensive video processing tasks
- Linux-based application software components to configure and control the PL-implemented hardware processing IPs and the data flow between PL-based video components and the PS

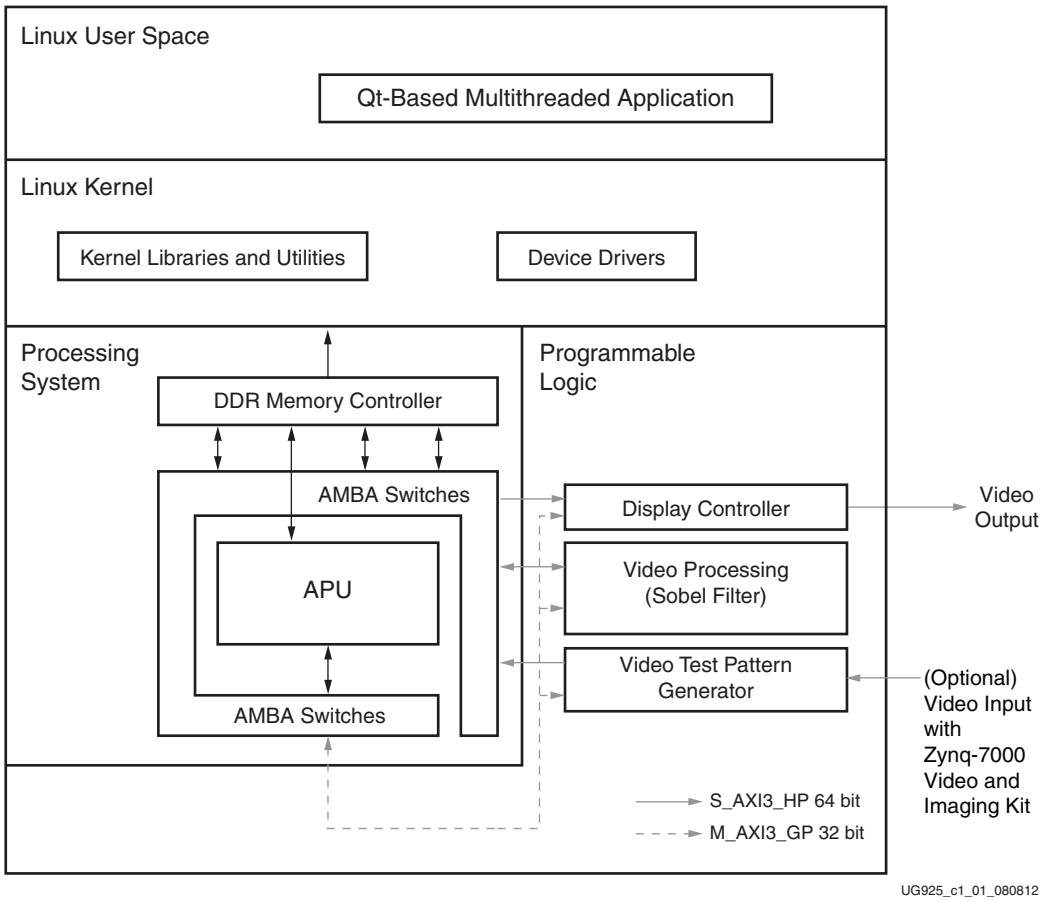

Figure 1-1: **Zynq-7000 AP SoC Base TRD System Block Diagram**

The TRD deliverables include source code for RTL design and software packages such as the Linux OS, device drivers, the application, and the GUI.

Note: The camera supplied with the video kit is not supported with the TRD.

Base TRD Key Features

Components in the Base TRD are further described in this section.

The Base TRD processing system (PS) includes:

- A dual ARM Cortex-A9 core
- ARM AMBA® AXI interconnect
- Multi-protocol, 32-bit DDR DRAM controller
- Standard peripheral interfaces including USB, Ethernet, UART, I2C, SD MMC, and GPIO

Base TRD programmable logic includes:

- Two AXI interconnects, 64-bit wide at 150 MHz
- One AXI interconnect, 32-bit wide at 75 MHz
- AXI VDMA(s)
- A full HD video input and output interface
- A Sobel accelerator
- Two AXI Performance Monitors

Base TRD software includes:

- Xilinx Zynq-7000 standard Linux kernel (based on Open Source Linux 3.x)
- Linux device drivers for TRD-specific IPs
- Qt-based Linux application demonstrating the video processing pipeline

List of Acronyms

Table 1-1 lists acronyms used in this document.

Table 1-1: **Acronyms**

Acronym	Definition
AFI	AXI FIFO interface
APU	Application processor unit
BSP	Board support package
COR	Clear on read
DTB	Device tree binary
DTS	Device tree source
EDK	Embedded Development Kit
AP SoC	All Programmable SoC
FMC	FPGA mezzanine card
FPS	Frames per second
FSBL	First-stage boot loader
GIC	General interrupt controller
GUI	Graphical user interface
HD	High definition
IDE	Integrated development environment
IOP	Input/output peripherals
IP	Intellectual property

Table 1-1: **Acronyms *(Cont'd)***

Acronym	Definition
KFLOPS	kilo floating-point operations per second
OCM	On-chip memory
OS	Operating system
PL	Programmable logic (inside the Zynq-7000 AP SoC)
PS	Processing system
R	Read only
RTL	Register transfer level
RW	Read/Write
SC	Self clear
SD	Secure Digital
SD MMC	Secure Digital Multimedia Card
SDK	Software Development Kit
SoC	System on Chip
TDP	Targeted Design Platform
TPG	(Video) Test Pattern Generator
TRD	Targeted Reference Design
TTC	Triple-timer counter
VDMA	Video direct memory access
VTC	Video timing controller
XPS	Xilinx Platform Studio
ZC702	Platform development board based on the Zynq AP SoC Z-7020 device
Zynq Z-7020	An implementation of the Zynq-7000 AP SoC with a fixed feature set and PL capabilities

Functional Description

This chapter describes the Zynq-7000 AP SoC ZC702 Base Targeted Reference Design (TRD) hardware design, software system, and video demonstration application components. It also describes how data flows through the various connected IPs and includes information about the flow of application control.

To build hardware and software for the Base TRD, refer to the Xilinx Zynq-7000 Base Targeted Reference Design wiki page wiki.xilinx.com/zc702-base-trd.

Hardware Architecture

The block diagram for the Base TRD is shown in Figure 2-1. This design has two parts:

- Processing system (PS)
- Video IPs and custom logic implemented in programmable logic (PL)

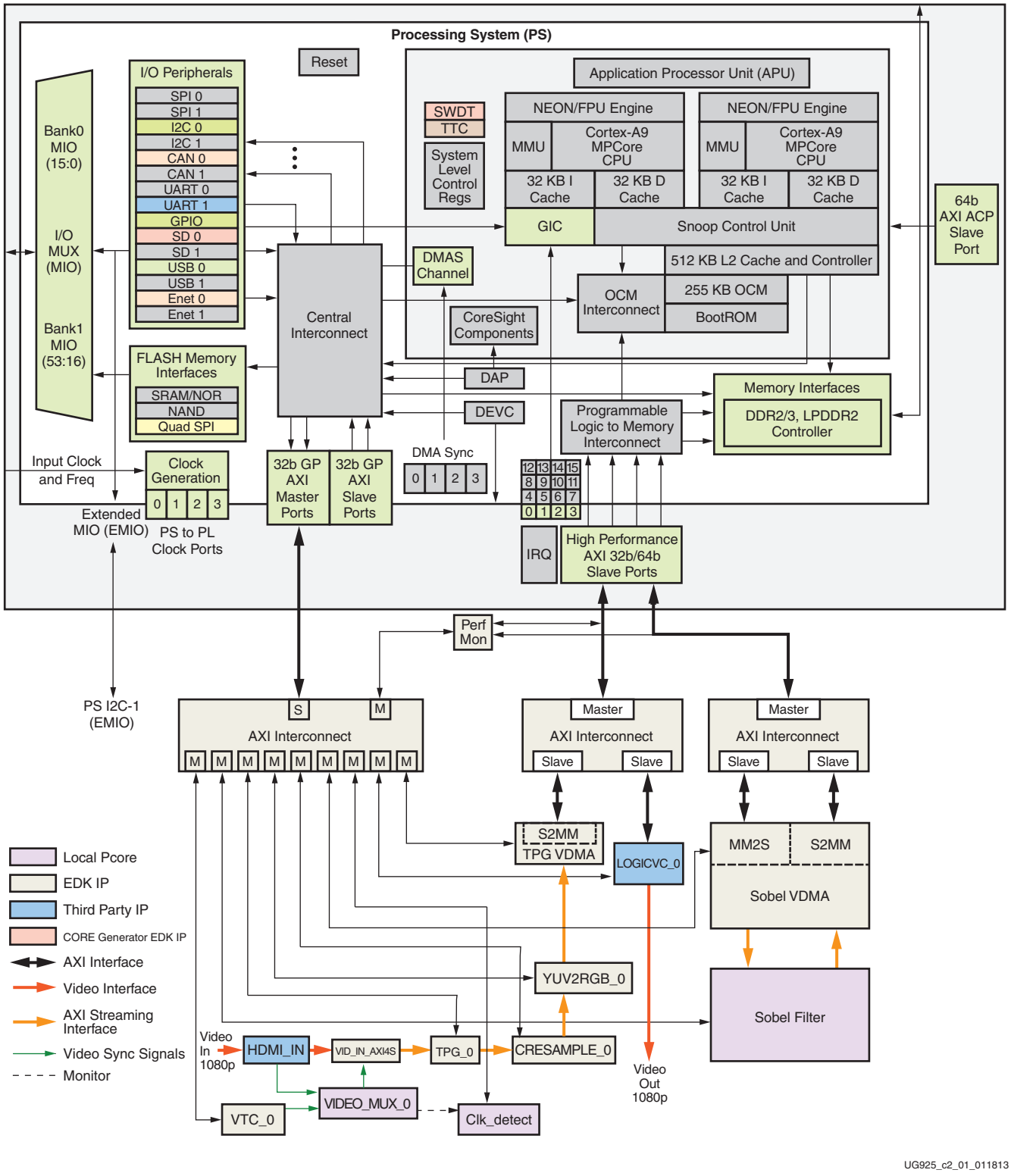

Figure 2-1: **Zynq-7000 AP SoC Base TRD Hardware Block Diagram**

This system is implemented in a Zynq-7000 AP SoC device (XC7Z020-CLG484-1) using ISE® Design Suite, version 14.3 tools.

The PL hardware utilization for the implemented design is shown in Table 2-1.

Table 2-1: **PL Hardware Utilization for Device XC7Z020-CLG484-1(1)**

FPGA Components	Total Available	Used	% of Used
LUTs	53,200	19,805	37
I/Os	200	43	21
FPGA Logic Memory			
RAMB36E1	140	21	15
RAMB18E1	280	15	5
Slice registers	106,400	24,094	22

Notes:
1. The figures provided here are only indicative of nature and can vary between different tool chain versions.

The PL-implemented video IP and custom logic address map is shown in Table 2-2.

Table 2-2: **FPGA Logic Address Map for the Zynq-7000 AP SoC ZC702 Base TRD**

Instance	Peripheral	Base Address	High Address
CLK_DETECT_0	clk_detect	0x40060000	0x4006FFFF
VTC_0	axi_vtc	0x40070000	0x4007FFFF
TPG_0	axi_tpg	0x40080000	0x4008FFFF
TPG_VDMA	axi_vdma	0x40090000	0x4009FFFF
FILTER_VDMA	axi_vdma	0x400B0000	0x400BFFFF
LOGICVC_0	LogiCVC	0x40030000	0x4003FFFF
PERF_MON_HP0_HP2	axi_perf_mon	0x400F0000	0x400FFFFF
CRESAMPLE_0	v_cresample	0x40040000	0x4004FFFF
YUV2RGB_0	v_ycrcb2rgb	0x40050000	0x4005FFFF
FILTER_ENGINE	sobel_filter_top	0x400D0000	0x400DFFFF

System Configuration

Processing System

This design makes full use of these four major components in the PS:

- Application processor unit (APU)
- Interconnect
- Input/output peripherals (IOP)

- Memory interfaces

This section describes some of the features of the PS used in this design. For detailed information about the complete feature set including a functional description, see UG585, *Zynq-7000 All Programmable SoC Technical Reference Manual*.

APU

The APU includes the dual ARM Cortex-A9 core processor, snoop control unit (SCU), L2 cache controller, on-chip memory (OCM), 8-channel DMA, system watchdog timer (SWDT), and triple timer controller (TTC) blocks.

Cortex-A9 Core - The ARM Cortex-A9 core processor implements the ARMv7 architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java byte codes in the Jazelle state. The media processing engine implements ARM NEON coprocessor technology, a single instruction multiple data (SIMD) architecture that adds instructions targeted at audio, video, 3D graphics, image, and speech processing. For this TRD, both ARM cores run at 667 MHz.

General Interrupt Controller - The GIC collects interrupts from various sources and distributes these interrupts to each of the ARM cores. The interrupt distributor holds the list of pending interrupts for each ARM Cortex-A9 core processor and then selects the highest priority interrupt before issuing it to the Cortex-A9 processor interface. Interrupts of equal priority are resolved by selecting the lowest ID. A total of 64 shared peripheral interrupts (PL interrupts + PS I/O peripheral interrupts) are supported, starting from ID 32. Table 2-3 lists interrupt IDs for interrupts coming from PL.

Table 2-3: **Interrupt IDs for PL-Generated Interrupts**

Interrupt line	ID	Type	Instance
CVC_DISPLAY_interrupt	91	Level	CVC_DISPLAY
TPG_VDMA_s2mm_introut	90	Level	TPG_VDMA
FIILTER_VDMA_s2mm_introut	89	Level	FIILTER_VDMA
FIILTER_VDMA_mm2s_introut	88	Level	FIILTER_VDMA
FILTER_0_interrupt	87	Level	FILTER_ENGINE

Interconnect

The interconnect unit connects all PS and PL master and slave devices. There are a total of six Advanced eXtensible Interface (AXI) slave ports dedicated for AXI masters residing in the PL, and four of these ports contain deep FIFOs to improve data throughput. Two AXI master ports provide access to AXI slaves in the PL. In this design, masters in PL are connected through two AXI slave ports with deep FIFOs. One AXI master port is used to access registers in AXI slave IPs in PL.

An advanced peripheral bus (APB) master port is provided for accessing software programmable registers of all PS modules. The top level switch is AXI3-compliant, the soft

IPs provided by Xilinx are AXI4-compliant, and the soft AXI interconnect IP provides protocol bridging as needed.

S_AXI_HP - The high performance slave AXI interfaces (S_AXI_HP) connect the PL to AFI blocks in the PS. The PL has four AXI masters out of which two are connected to the S_AXI_HP0 port and two are connected to the S_AXI_HP2 port. The HP port enables a high throughput data path between AXI masters in the programmable logic and the processing system's DDR3 memory. The main aim of the AXI FIFO interface (AFI) units is to smooth out this variable latency, allowing the ability to stream data continuously from DDR to the PL masters and from the PL masters to DDR. The PL-side interface of AFI runs on the clock coming from the PL. In this design, a 150 MHz clock is connected from the PL side. The DDR-side clock is running on 2/3 of the DDR_CLK (533 MHz). The high performance AXI interface module provides several "hooks" to assist in bandwidth management of masters connected to different PL ports. Controlling issuance capability available from the PL port is one of the hooks exercised in this design to obtain a fair share of bandwidth between two masters, SOBEL VDMA, and the display controller.

M_AXI_GP - This AXI master port interfaces with AXI slave IPs in PL through an AXI Lite interconnect. The CPU manages initializing and controlling the video pipeline through this port.

IOP - The IOP unit includes communication peripherals. GPIO, Ethernet, USB, I2C, and SD controllers from the PS are used extensively in this design.

GPIO - The 64-bit general purpose input/outputs (GPIOs) are connected to the PL through the extendable multiplexed I/O (EMIO) interface. Sixty-four bits are divided into two banks, each of 32 bits. Because each GPIO bit can be dynamically configured as input or output, GPIO bits are used in this design for a variety of functions. Table 2-4 lists the GPIO bit and purpose in design.

Table 2-4: **GPIO Bits Functional Description**

GPIO Bit Number	Net Name	Purpose
0	ps7_0_GPIO_O[0]	Resets video receiving block dvi2axi bridge
1	ps7_0_GPIO_O[1]	Resets Sobel filter block
3	ps7_0_GPIO_O[3]	Selects a line for the video multiplexer—either the external video source or the internally generated test pattern
6	ps7_0_GPIO_O[6]	FMC-IMAGEON I2C multiplexer reset
2, 4, 5, 7	N/A	N/A

Memory Interfaces

The memory interfaces unit includes the DDR memory controller and nonvolatile memory controllers. The DDR memory controller includes a 4-port arbiter. One AXI port is dedicated for ARM CPU access and two ports are dedicated for high performance AXI interface master devices in the programmable logic. The remaining port is shared by all other AXI masters. In

this design, DDR3 is configured to run at 533 MHz, and the AXI interface is running at 355 MHz.

PL Clocks

The PS provides four (FCLK_CLK) fully programmable clocks to the PL. These clocks are routed directly to PL clock buffers to serve as a frequency source for the PL. The clock generator module in PL gets a 100 MHz clock from FCLK_CLK0.

PL Reset

The PS provides four FCLK_RESET[3:0]_N fully programmable reset signals to the PL. These signals are asynchronous to PS clocks. The PL logic reset block in this design receives input from FCLK_RESET0_N and generates necessary reset signals for the design implemented in PL.

Programmable Logic

Clocking

The FPGA logic design has three clock domains: AXI MM (memory-mapped) interconnect, AXI register interface, and video clock. These domains run at 150 MHz, 75 MHz, and 148.5 MHz, respectively.

The clock generator module receives a 100 MHz input clock from the PS FCLK_CLK0 and generates 75 MHz and 150 MHz. The AXI Lite interconnect works on 75 MHz. Apart from the AXI Lite interconnect, the register interface of AXI VDMA, AXI TPG, logiCVC-ML, VTC, perf_monitor, and clk_detect cores are driven by the 75 MHz clock.

Two instances of the AXI_MM interconnect connected to the HP port of the PS run on 150 MHz. The S2MM (stream to memory map) and MM2S (memory map to stream) channels of VDMAs are running at 150 MHz. The 150 MHz clock drives the logiCVC-ML memory read interface and also the AXI slave interface of the Sobel filter.

The video clock comes from the onboard clock synthesizer or from the FMC-IMAGEON card. BUFGMUX dynamically selects which video clock source drives logic running on the video clock domain. The AXI TPG, VTC, dvi2axi, and logiCVC-ML blocks run on the video clock.

Table 2-5 lists system clocks.

Table 2-5: **System Clocks**

Clock Signal	Source	Frequency	Use
FPGA_CLK	PS - FCLK_CLK0	100 MHz	Input clock to clock generator
clk_75mhz	Internal mixed-mode clock manager (MMCM)	75 MHz	Clock for AXI Lite interconnect, generated by clock generator
clk_150mhz	Internal MMCM	150 MHz	Clock for AXI MM interconnect, generated by clock generator
VIDEO_CLK_P, VIDEO_CLK_N	External differential video clock coming from clock synthesizer on board	148.5 MHz	Clock for display controller and video receiving blocks
fmc_imageon_in_0_clk_pin	External video clock coming from Imageon FMC	148.5 MHz	Clock for video receiving modules

Based on user clock configuration inputs, the clock generator determines the correct configuration of the PLLs.

Table 2-6 shows clock requirements of master and slave peripherals connected in system and their connection.

Table 2-6: **PL Clock Configuration**

Component	Frequency (MHz)	Phase	Buffered	Connection
clock_generator_0				
• CLKIN	100		Yes	FPGA_CLK
• CLKOUT0	75		Yes	clk_75mhz
• CLKOUT1	150		Yes	clk_150mhz
VIDEO_MUX_0				
• video_clk_1	148.5	0	Yes	VIDEO_CLK
• video_clk_2	148.5	0	Yes	fmc_imageon_hdmi_in_0_clk_pin
• video_clk	148.5	0	Yes	video_clk_int
Processor				
ps7_0				
• FCLK_CLK0	100	0	Yes	FPGA_CLK
• M_AXI_GP0_ACLK	75	0	Yes	clk_75mhz
• S_AXI_HP0_ACLK	150	0	Yes	clk_150mhz
• S_AXI_HP2_ACLK	150	0	Yes	clk_150mhz
Buses				
axi4_0				
• INTERCONNECT_ACLK	150	0	Yes	clk_150mhz

Table 2-6: **PL Clock Configuration *(Cont'd)***

Component	Frequency (MHz)	Phase	Buffered	Connection
axi4_1				
• INTERCONNECT_ACLK	150	0	Yes	clk_150mhz
axi4_lite				
• INTERCONNECT_ACLK	75	0	Yes	clk_75mhz
Peripherals				
proc_sys_reset_1				
• Slowest_sync_clk	75	0	Yes	clk_75mhz
LOGICVC_0				
• S_AXI_ACLK	75	0	Yes	clk_75mhz
• mclk	150	0	Yes	clk_150mhz
• vclk	148.5	0	Yes	VIDEO_CLK
FILTER_ENGINE				
• SYS_CLK	150	0	Yes	clk_150mhz
• s_axi_CONTROL_BUS_ACLK	150	0	Yes	clk_150mhz
SOBEL_SWRST_FF				
• Clk	75	0	Yes	clk_75mhz
FILTER_VDMA				
• m_axi_mm2s_aclk	150	0	Yes	clk_150mhz
• m_axi_s2mm_aclk	150	0	Yes	clk_150mhz
• m_axis_mm2s_aclk	150	0	Yes	clk_150mhz
• s_axi_lite_aclk	75	0	Yes	clk_75mhz
• s_axis_s2mm_aclk	150	0	Yes	clk_150mhz
TPG_SWRST_FF				
• Clk	75	0	Yes	clk_75mhz
TPG_VDMA				
• m_axi_s2mm_aclk	150	0	Yes	clk_150mhz
• s_axi_lite_aclk	75	0	Yes	clk_75mhz
• s_axis_s2mm_aclk	150	0	Yes	clk_150mhz
PERF_MON_HP0_HP2				
• SLOT_0_AXI_ACLK	150	0	Yes	clk_150mhz
• SLOT_1_AXI_ACLK	150	0	Yes	clk_150mhz
• S_AXI_ACLK	75	0	Yes	clk_75mhz
• CORE_ACLK	150	0	Yes	clk_150mhz
VTC_0				

Table 2-6: **PL Clock Configuration *(Cont'd)***

Component	Frequency (MHz)	Phase	Buffered	Connection
• clk	148.5	0	Yes	video_clk_int
• S_AXI_ACLK	75	0	Yes	clk_75mhz
TPG_0				
• aclk	150	0	Yes	clk_150mhz
• s_axi_aclk	75	0	Yes	clk_75mhz
CLK_DETECT_0				
• DUT_CLK	148.5	0	Yes	video_clk_int
• S_AXI_ACLK	75	0	Yes	clk_75mhz
HDMI_IN				
• clk	148.5	0	Yes	fmc_imageon_hdmi_in_0_clk_pin
VID_IN_AXI4S				
• vid_in_clk	148.5	0	Yes	video_clk_int
• aclk	150	0	Yes	clk_150mhz
CRESAMPLE_0				
• aclk	150	0	Yes	clk_150mhz
• s_axi_aclk	75	0	Yes	clk_75mhz
YUV2RGB_0				
• aclk	150	0	Yes	clk_150mhz
• s_axi_aclk	75	0	Yes	clk_75mhz

Processor System Reset Module

Instance: proc_sys_reset_1

The proc_sys_reset module implements a reset scheme. Input to the proc_sys_reset core is generated by PS Proc_sys_reset_1_N. The polarity of input reset to this block is indicated by parameter C_EXT_RESET_HIGH. In this design, C_EXT_RESET_HIGH is set to 0 as reset generated by PS is active-Low. This block generates various types of resets, such as reset for interconnect, peripheral reset, and so on. All the blocks in the PL are driven by interconnect reset, which is active-Low in polarity.

For detailed information about the complete feature set and a functional description of the proc_sys_reset IP, refer to DS406, *LogiCORE IP Processor System Reset Module Product Specification*.

AXI Interconnect

Instances: axi4_0, axi4_1, axi4_lite

FPGA logic design has two interconnects for AXI memory-mapped masters and one interconnect for the AXI register interface.

AXI memory-mapped interconnects are connected to masters like AXI_VDMA and logiCVC-ML. Slaves connected to these interconnects includes HP0 and HP2 ports of Zynq-7000 AP SoC PS. This interconnect operates at 150 MHz and the data width is 64-bit wide. The read/write acceptance and issuance are set to 8. The acceptance and issuance helps improve system performance. The PS HP port can accept a maximum burst length of 16. This imposes a limitation on getting minimum acceptable bandwidth for every master in a multi-master system. The optimum setting of issuance and acceptance reduces throttle on the bus and compensates for long latencies.

The AXI register interface is clocked at 75 MHz. The Zynq-7000 AP SoC PS GP0 port acts as master on this interconnect and connected slaves have register maps. AXI TPG and VTC are examples of slaves connected to this interconnect. The operations of the video pipeline are controlled by registers inside every IP. Depending upon data flow required in the video pipeline, the processor writes these registers through the AXI Lite interconnect. The AXI Lite interconnect accepts write or read transfers from the CPU, performs address decoding, selects a particular slave, and establishes a communication channel between the CPU and the slave device.

For detailed information about the complete feature set and a functional description of the AXI Interconnect IP, refer to DS768, *LogiCORE IP AXI Interconnect*.

AXI Video Direct Memory Access

Instances: TPG_VDMA, FILTER_VDMA

AXI VDMA has an AXI streaming interface on one side and an AXI memory-mapped interface on the other side. The VDMA has two channels: MM2S (memory-mapped to streaming) and S2MM (streaming to memory-mapped). The MM2S channel reads the number of data beats programmed through the C_MM2S_MAX_BURST_LENGTH parameter and presents it to the slave device connected through the streaming interface. The data width of the streaming interface can be different than the memory-mapped interface and controlled through C_M_AXIS_MM2S_TDATA_WIDTH. The data width of the S2MM memory-mapped interface is controlled by the C_M_AXI_MM2S_DATA_WIDTH parameter.

The S2MM channel receives data from the master device connected through the streaming interface. The C_S_AXIS_S2MM_TDATA_WIDTH parameter decides the width of the streaming interface. Data received on the streaming interface is then written into the system memory through the memory-mapped interface. The C_M_AXI_S2MM_DATA_WIDTH parameter decides the data width of the memory-mapped interface and C_S2MM_MAX_BURST_LENGTH governs the burst length of the write transaction.

In this design, the streaming interface data width is set to 32-bit wide and the memory-mapped interface is configured as 64-bit wide. The AXI VDMA is used in simple register direct mode, which removes the area cost of the scatter gather feature. Initialization, status, and management registers in the AXI VDMA core are accessed through an AXI4-Lite slave interface. To get the best possible throughput for AXI VDMA instances, the maximum burst length is set to 16. In addition, the master interfaces have a read and write issuance of 8 and a read and write FIFO depth of 512 to maximize throughput. The line buffers inside the AXI VDMA for the read and write sides are set to 4K deep and the store and forward feature of the AXI VDMA are enabled on both channels to improve system performance and reduce the risk of system throttling.

For detailed information on the complete feature set and a functional description of AXI VDMA IP, refer to PG020, *LogiCORE IP AXI Video Direct Memory Access Product Guide*.

Video Timing Controller

Instance: VTC_0

The VTC is a general purpose video timing generator and detector. The input side of this core automatically detects horizontal and vertical synchronization pulses, polarity, blanking timing, and active video pixels. This information can be used by application software to take various decisions and for configuration of the video pipeline. In the current design, application software measures resolution of external video and then decides whether to switch to the external video source or not. The same feature can be expanded in the future to configure the video pipeline based on input resolution.

The output side of the core generates the horizontal and vertical blanking and synchronization pulses. The width and interval of these pulses are configured through the AXI Lite interface. The AXI TPG block generates a video test pattern based on video timing pulses generated by VTC. In this design, VTC is used to generate video timing signals to match Full HD (1080p60) video format.

The video timing generator/detector block and AXI Lite interface of this core work on a single clock domain, that is, the video clock.

For detailed information on the complete feature set, a functional description, and licensing information for Video Timing Controller IP, refer to PG016, *LogiCORE IP Video Timing Controller Product Guide*.

Clock Detect

Instance: CLK_DETECT_0

The video pipeline in this design can take video input either from an external video source or from the internally generated video test pattern. There is a possibility that the user selects external video mode and the FMC-IMAGEON is not present. This scenario can lead to application failure and might drag the design into an unrecoverable state. This core gives

provision to the application to avoid such a scenario. The core determines the frequency of the incoming video signal, and based on this value, the application decides whether the incoming video rate is suitable for the application's functioning or not.

The management and configuration of core is controlled through the AXI register interface. The sampling duration register holds the number of cycles for which the internal cycle counter is On. This core has two counters: The sampling counter works on the AXI Lite clock and the cycle counter runs on the video clock. Both counters start counting after the EN bit in the control register is set. The sampling counter generates load pulse upon reaching the terminal count specified by the sampling duration register. The cycle counter state is then loaded into the cycle counter register. Both counters clear their states and start counting again until the EN bit in the control register is set to 0.

The register value of the cycle count can be used by the application to decide whether a video clock source is present or not, and if present, what the frequency of the video clock is. The value of cycle count read from the register also helps determine the input video standard, for example, if the cycle count is around 25 MHz, it is VGA mode. If it is 75 MHz, it is in HD ready (720p) mode, and if 148 MHz, it is Full HD (1080p).

Test Pattern Generator

Instance: TPG_0

The Test Pattern Generator contains an AXI register interface to access slave control registers from a processor. This IP can generate patterns like color bars, horizontal and vertical burst patterns, and zone plates. The generation of pattern is controlled through the pattern control register. It also enables the overlay of a box on a selected pattern. The motion control register controls the speed at which the box moves over a selected pattern. In this design, a zone plate pattern is used with a moving box. The size of the zone plate is controlled through zplate hdelta and zplate vdelta registers. The box size register controls the size of the box, and the color of the box is selected by the box color register. The width and height of the pattern is equal to 1920 x 1080, selected through the line length and frame height register.

For detailed information on the complete feature set and a functional description of Test Pattern Generator, refer to PG103, *LogiCORE IP Test Pattern Generator Product Guide*.

logiCVC-ML

Instance: LOGICVC_0

The logiCVC-ML is a multi-layer video display controller from Xylon [Ref 1]. The logiCVC-ML controller refreshes the display image by reading the video memory and converting the read data into a data stream acceptable for the display interface. It generates control signals for the display, and supports multiple layers with video processing functions such as alpha blending, transparency, and move around.

For detailed information about the complete feature set, a functional description, and license information for logiCVC-ML IP, refer to the Xylon data sheet [Ref 1].

AXI Performance Monitor

Instance: PERF_MON_HP0_HP2

The AXI Performance Monitor can monitor and analyze system behavior on the AXI interface. This core is used in the Base TRD to measure read and write throughput on AXI slave ports of the PS (HP0 and HP2), which are used to access DDR memory from PL. The core consists of the AXI4-Lite interface to configure and control the core.

This core is configured to measure the read and write throughput by counting the number of transactions per second. When the configured time interval expires, measured throughput in bytes is loaded into a register and read by the software application.

Two slots of AXI Performance Monitors are used to measure read and write throughput of HP0 and HP2 simultaneously.

For detailed information on the complete feature set and a functional description of AXI Performance Monitor, refer to PG037, *LogiCORE IP AXI Performance Monitor Product Guide*.

fmc imageon hdmi in

Instance: HDMI_IN

This IP core receives video from FMC-IMAGEON, in YCrCb 4:2:2 format, with embedded vblank and hblank signals, and extracts blanking information.

Chroma Resampler

Instance: CRESAMPLE_0

This IP is a chroma resampler, which is configured to convert the video from YCrCb 4:2:2 to YCrCb 4:4:4. This IP has an AXI4 slave interface to configure, control, and read status from the application software.

For detailed information on the complete feature set, a functional description, and license information for Chroma Resampler, refer to PG012, *LogiCORE IP Chroma Resampler Product Guide*.

YCrCb to RGB Color-Space Converter

Instance: YUV2RGB_0

This IP is a color space converter, which converts the video from YCrCb to RGB format as the rest of video IPs in the pipeline work on RGB video. This IP has an AXI4 slave interface to configure, control, and read status from the application software.

For detailed information on the complete feature set, a functional description, and license information for YCrCb to RGB Color-Space Converter IP, refer to PG014, *LogiCORE IP YCrCb to RGB Color-Space Converter Product Guide*.

Video Multiplexer

Instance: VIDEO_MUX_0

This IP multiplexes the sync signals from time base generator (VTC_0) and fmc_imageon_hdmi_in and feeds the Video In to AXI4-Stream (VID_IN_AXI4S). The selection is done by PS GPIO (ps7_0_GPIO_O[3]).

For detailed information on the complete feature set and a functional description of Video in to AXI4-Stream IP, refer to PG043, *Video in to AXI4-Stream IP Product Guide*.

Sobel Filter

Instance: FILTER_ENGINE

This IP has the AXI4-Lite interface through which the IP is configured and controlled.

The number of rows and columns are configured using AXI interface, and the filtering process starts when the Start register is written through the Sobel interface.

The Sobel filter detects the edge in the video frame and the processed frame is sent out on AXI stream interface.

This IP generates an interrupt after processing every frame, then the software initiates the process for the next frame by writing to the control register.

Video In to AXI4-Stream

Instance: VID_IN_AXI4S

This IP interfaces a video source to the AXI4-Stream interface. It also handles video data clock boundary crossing between the video clock domain and AXI4-Stream clock domain.

For detailed information on the complete feature set and a functional description of Video in to AXI4-Stream IP, refer to PG043, *Video in to AXI4-Stream IP Product Guide*.

Software Architecture

This section explains the software architecture for the Zynq-7000 AP SoC ZC702 Base TRD. Figure 2-2 illustrates a top level view of the software architecture.

A multi-threaded Linux application is responsible for running the TRD demonstration. This application uses the Qt-based GUI, which is displayed through Display Monitor, to obtain user inputs. Depending on the inputs, it calls device drivers to configure the hardware and to enable a particular data path.

Three major software components are involved in the Base TRD:

- Boot loader
- Xilinx Linux kernel
- Application

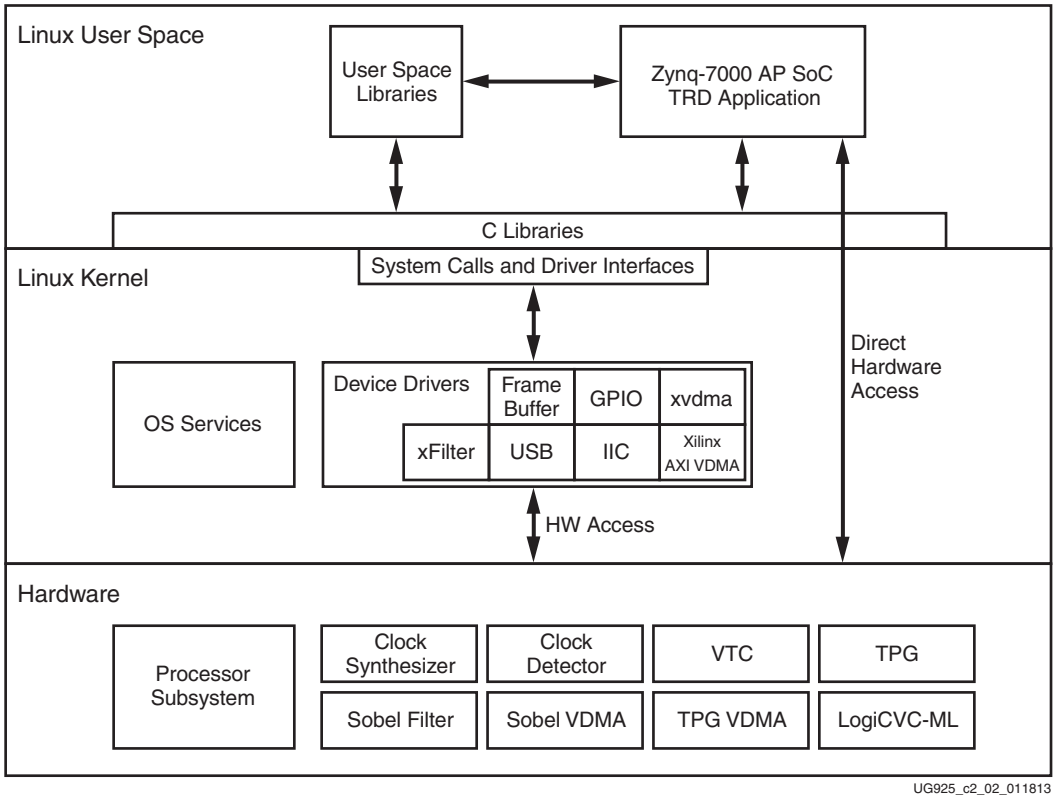

Figure 2-2: **Software Architecture: Top Level View**

Boot Loader

A two-stage boot loader is used for the Zynq-7000 AP SoC Linux boot-up. The FSBL is responsible for initializing required hardware and loads the second-stage boot loader, U-Boot, which is responsible for loading kernel image in the DDR memory.

The FSBL source code is generated through the Xilinx SDK tool, depending on the hardware design specification. That source code is modified for the Base TRD to initialize the HDMI in and HDMI out chips.

U-Boot is an open source universal boot loader used across various embedded platforms. The source code, customized for Zynq-7000 AP SoC Linux, is available on the Xilinx Open Source ARM git Repository: git.xilinx.com/.

Refer to the Zynq-7000 Base Targeted Reference Design wiki page (wiki.xilinx.com/zc702-base-trd) to build the FSBL and U-Boot.

Xilinx Linux Kernel

The Xilinx Linux kernel is based on the mainline open source kernel git tree, adding support for a variety of Xilinx IP core drivers and reference boards. The source code is available on the Xilinx Open Source ARM git Repository: git.xilinx.com/...

The Xilinx Linux kernel is extended (patched) to support IPs specific to this Base TRD. Patching and building the Linux kernel is explained in the Zynq-7000 Base Targeted Reference Design wiki page at wiki.xilinx.com/zc702-base-trd.

Table 2-7 lists kernel drivers used for the Base TRD.

Table 2-7: **Linux Kernel Drivers Used by the Base TRD**

Linux Driver	Function	Called By
Frame buffer	Drives the display controller (logiCVC-ML) to display the application UI and control data path	Application
Frame buffer	Frame buffer console display	Linux console
PS - GPIO	Provides Reset signals to VDMA IPs	Application
XVDMA	Controls VDMA IP for various data flows	Application
Xilinx AXI VDMA	Provides core AXI VDMA functionality and the actual hardware interface	XVDMA driver
xFilter	Control and configure filter IP (Sobel filter)	Application

All of the drivers in Table 2-7 except the XVDMA and xFilter drivers come with the standard Xilinx Linux kernel. The XVDMA and xFilter drivers are patched into the kernel.

Frame Buffer Driver

Linux provides a standard frame buffer, which is hardware-independent, and the application can use this buffer without knowing the underlying display controller. The Xylon frame buffer driver for CVC IP is registered with the standard frame buffer driver to provide support for the logiCVC-ML display controller.

The Xylon frame buffer driver is compiled with the kernel and probes for the hardware and resolution specification by scanning the `dtb` file at boot. If there is no entry for logiCVC-ML IP in the `dtb` file, then the driver does not load itself.

The application uses the standard Linux frame buffer driver, which has a character driver interface.

For this Base TRD, `/dev/fb0` is the node that is populated. To find the exact device node, iteratively match the **id** field of structure fb_fix_screeninfo acquired by the FBIOGET_FSCREENINFO IOCTL call for each device node populated. If the **id** field is *Xylon FB*, the frame buffer driver is for Xylon logiCVC-ML.

Note: An IOCTL (input/output control) is a general purpose Linux system call used for implementing the interface between a user application and a device driver.

More details on the frame buffer driver is available in kernel documentation at `<Linux Kernel source>/Documentation/fb/framebuffer.txt`.

PS-GPIO Driver

The GPIO SYSFS interface is used for GPIO configuration. The GPIO SYSFS interface allows the user to control I/O pins using files under the `/sys` directory.

When the system boots, all GPIO pins are owned by the kernel. The pins do not show up in the SYSFS system until they are exported. To export them, write the pin number (say 54) to the file `/sys/class/gpio/export`.

This results in the pseudo files for pin 54 showing up under `/sys/class/gpio/gpio54` as:

```
/sys/class/gpio/gpio54/direction
/sys/class/gpio/gpio54/value
```

The user can set the direction by writing **`in`** or **`out`** to the `direction` file. For the `out` direction, writing **`0`** or **`1`** on the corresponding `value` file resets or sets the pin, respectively. Similarly, the user can read the `value` file for the `in` direction.

XVDMA Driver

XVDMA is a character driver used for configuring and controlling video DMA transactions for both TPG and Sobel hardware. XVDMA internally calls the Xilinx AXI VDMA driver to complete the task and interrupt handling.

The device node that uses the XVDMA driver is `/dev/xvdma`. The following IOCTLs are defined for the XVDMA driver and contain the corresponding IOCTL arguments to be used:

- XVDMA_GET_NUM_DEVICES
  - This call obtains the number of VDMA probed and available for use.
  - Argument: Address of unsigned int [unsigned int *].
    - This gets filled up with the number of VDMA when the call returns.
- XVDMA_GET_DEV_INFO
  - This call gives device information like channel number, for the given VDMA ID.
  - Argument: Address of structure xvdma_dev [struct xvdma_dev *].
    - Before calling, the **device_id** field of this structure should be filled with VDMA ID. On return the rest of the structure is filled by the driver.
- XVDMA_DEVICE_CONTROL
  - This call sets the VDMA channel configuration.
  - Argument: Address of structure xvdma_chan_cfg [struct xvdma_chan_cfg *].
    - Before calling, this structure should be filled with required channel configurations.
    - To reset VDMA, only fill **`chan = <channel id>`** and **`config.reset = 1`** fields of structure.
- XVDMA_PREP_BUF
  - This call sets the buffer configurations.
  - Argument: Address of structure xvdma_buf_info [struct xvdma_buf_info *].
    - Before calling, this structure should be filled with required buffer configurations.
- XVDMA_START_TRANSFER
  - This call triggers the VDMA transfer.
  - Argument: Address of structure xvdma_transfer [struct xvdma_transfer *].
    - Before calling, this structure should be filled. The structure specifies the channel ID and whether the call is synchronous or asynchronous.
- XVDMA_STOP_TRANSFER

- This call stops the VDMA.
- Argument: Address of the unsigned int variable [unsigned int *].
  - Before calling, this int variable should be filled with the channel ID.

Structures used for the above IOCTL are defined in the `driver_include.h` file, which is part of the application source code provided with the Base TRD. The file contains comments explaining each field of the structure.

Xilinx AXI VDMA

This driver is not used by the application. It is internally called by XVDMA to perform operations on video DMA hardware.

xFilter

xFilter is a character driver used for configuring and controlling a filter IP, the

Sobel filter in this case. This driver takes input from the `dts` file for register map and interrupt ID.

xFilter works in two modes—Continuous mode and On demand mode. In Continuous mode, after the start ioctl call (XFILTER_START) is issued, xFilter runs until the stop ioctl call (XFILTER_STOP) is issued. In On demand mode, each start ioctl call represents the filtering of a single frame. The start ioctl call is always asynchronous. The user can confirm completion of the filter operation on the current frame by calling the wait ioctl call (XFILTER_WAIT_FOR_COMPLETION), which is a blocking call.

The device node that uses the xFilter driver is `/dev/xfilter`. The following IOCTLs are defined for the xFilter driver and contain the corresponding IOCTL arguments to be used:

- XFILTER_INIT
  - Initializes the resolution and other parameters for the filter.
  - Argument: Address of structure xFilterConfig [struct xFilterConfig *]
    - Before calling, this structure should be filled with resolution and filter mode information.
- XFILTER_START
  - Starts the filter engine
    - In Continuous mode, this call is automatically repeated on each frame done interrupt until the XFILTER_STOP call is made.
    - In On demand mode, this call starts the filter engine for one frame.
  - Argument: NULL

- XFILTER_STOP
  - Stops the filter engine for Continuous mode. This call has no effect for On-demand mode.
    - After this call, it is mandatory to have the XFILTER_INIT call before doing XFILTER_START
  - Argument: NULL
- XFILTER_WAIT_FOR_COMPLETION
  - In On demand mode, this blocking call returns after the completion of current frame processing.
  - Argument: NULL

Structures used for the above IOCTL are defined in the `driver_include.h` file, which is part of the application source code provided with the Base TRD. The file contains comments explaining each field of the structure.

Application

Figure 2-3 describes various components of the application.

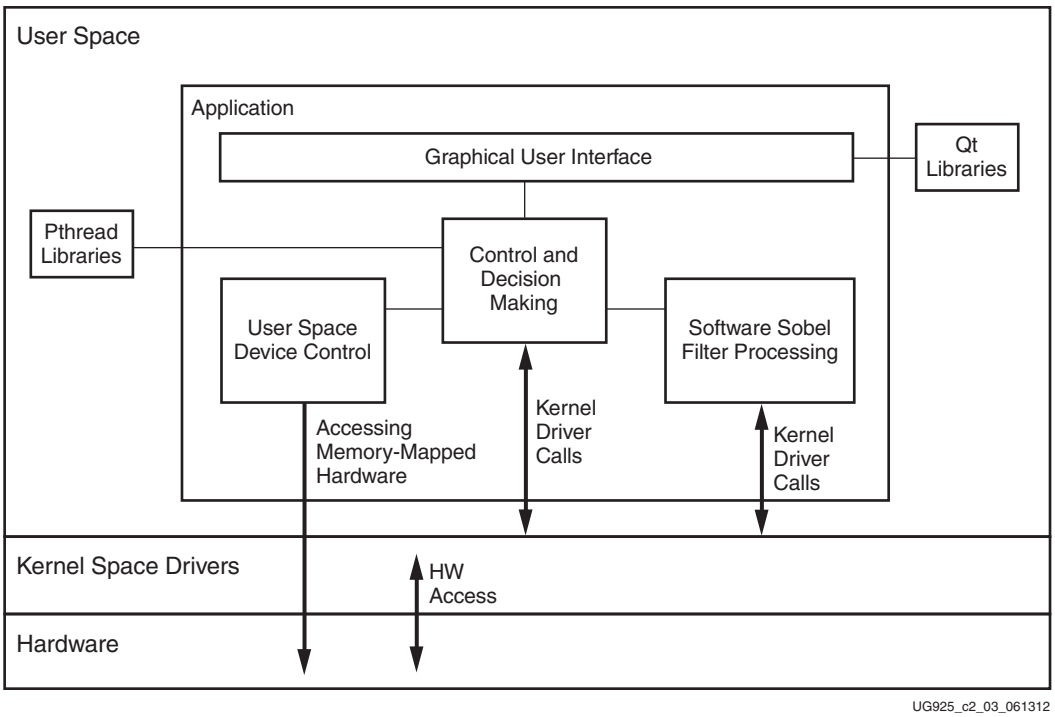

Figure 2-3: **Application Functional Blocks**

The application is divided into the following functional blocks:

- GUI
- Control and decision making
- User space device control
- Software Sobel filter processing

The first three components run in one thread of the application, while the software Sobel filter runs in a second separate thread. The Linux kernel is in SMP mode and both the cores are utilized for running different threads. At run time, the kernel assigns one of the cores for software Sobel filter processing and other core for running rest of the application and the operating system.

Graphical User Interface

The GUI for this Base TRD is designed using the Qt framework (see Figure 2-4).

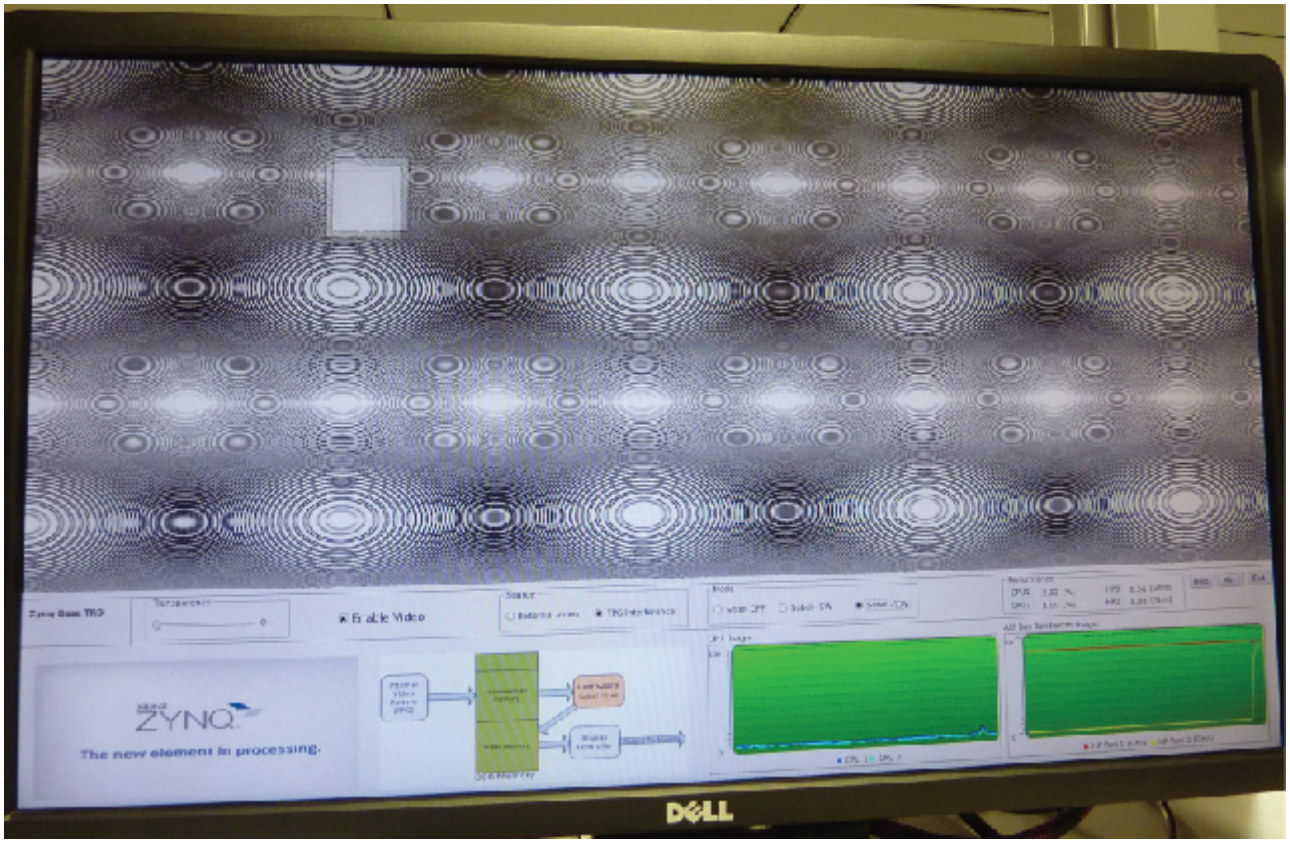

Figure 2-4: **GUI for TRD Application**

In Figure 2-4, the grey screen at the bottom is the GUI. The GUI can be minimized or maximized with the **MIN/MAX** button on the right side. The complete screen is the Video (or display) area, where the pattern or video is displayed. There is a Transparency slider that makes the GUI semi-transparent. The bottom portion of the GUI consists of the ZYNQ

banner, a pictorial view of Operation Mode and two graphs, which are only available in the MAX GUI mode. All other controls are available in both the MAX and MIN GUI mode.

Figure 2-5 shows the minimized (MIN) GUI mode.

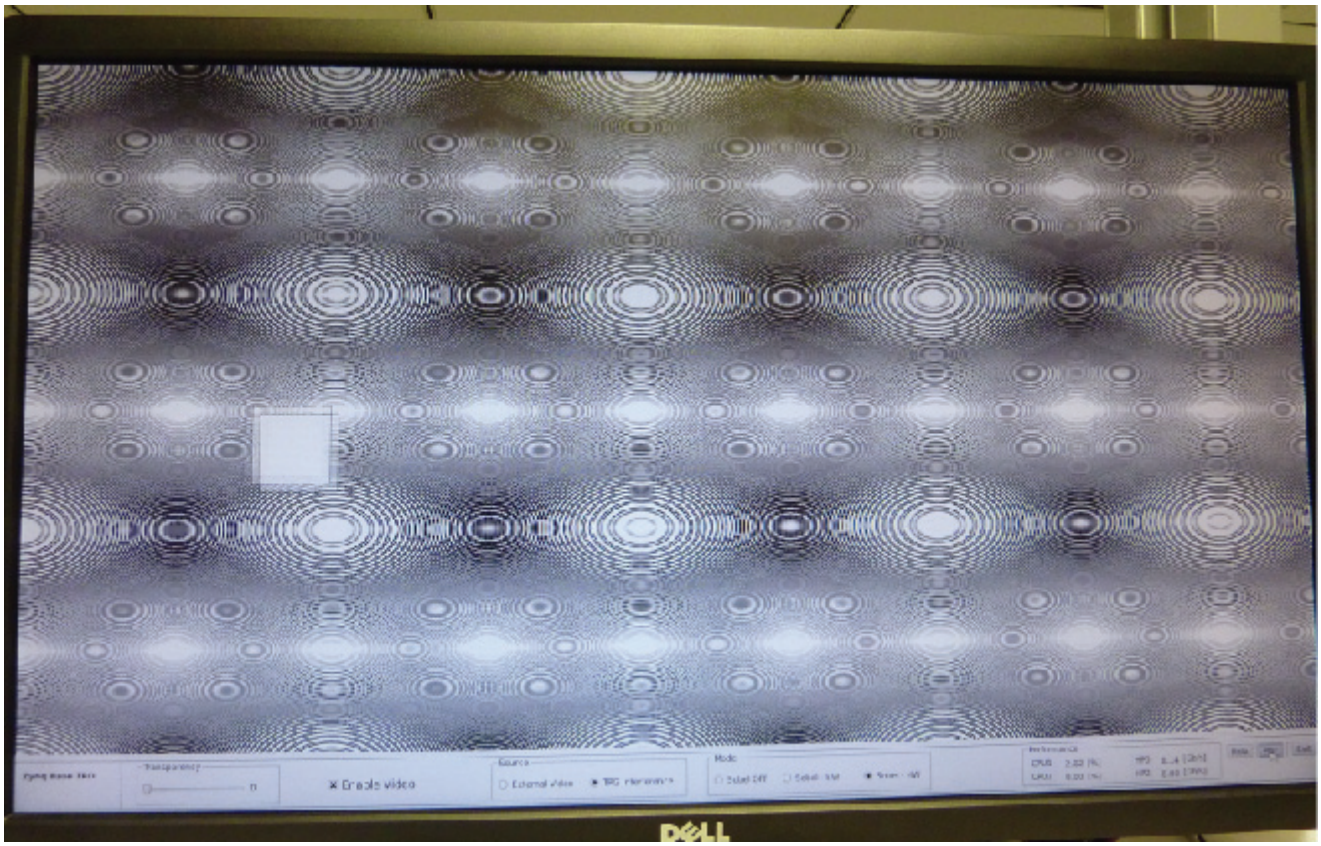

Figure 2-5: **Minimized GUI Mode**

The main functionality of the GUI includes:

- Getting user inputs
- Plotting graphs
- Displaying the video area

Get User Inputs

The Qt framework provides for having the mouse as input device (it internally uses Linux USB-HID class drivers). The input from the user includes **video Enable/Disable**, **Input Source Select**, and **Mode Select** for the video pipeline.

Plot Graph

Two graphs are plotted using the Qt framework. The first graph demonstrates CPU utilization for each ARM core, and the second demonstrates AXI memory bandwidth utilization on HP0 and HP2 ports

In the CPU utilization graph, the horizontal axis is for time and the vertical axis is for the percentage of CPU utilization.

In the memory bandwidth graph, the horizontal axis is for time and the vertical axis is for the Gb/s of read and write transactions on AXI.

Along with the graphs, the utilization and bandwidth numbers are also displayed above the graphs. These numbers are available in both MIN and MAX GUI mode.

Display Video Area

This is the full screen area, where the output of the video pipeline is displayed.

Control and Decision Making

This block receives input from the GUI and maintains the state transition for the complete application. It communicates with all other blocks of the application and with the kernel drivers to change the state of hardware.

The following hardware is configured through this control block using the kernel drivers mentioned in Xilinx Linux Kernel:

- VDMA (using the XVDMA driver)
- VDMA reset and multiplexer switching for external video (using the GPIO driver)
- logiCVC-ML control (using the frame buffer driver)Sobel Filter (using the xFilter driver)
- Sobel Filter (using the xFilter driver)

Data Flow Use Cases

The video source can be either the video TPG (internal) or an external video source (HDMI IN). The six combinations of data flow use cases are listed below. Use cases 1, 2, and 3 use the TPG and use cases 4, 5, and 6 use external video as video source.

The Sobel filter can either be a software or hardware implementation or it is bypassed entirely (turned off). Use cases 1 and 4 have the filter turned off, use cases 2 and 5 refer to the software implementation, and use cases 3 and 6 refer to the hardware implementation. Figure 2-6 combines TPG/external video in one functional block. Figure 2-7 combines software and hardware implementation in one functional block, therefore, use cases 4, 5, and 6 refer to both Figure 2-6 and Figure 2-7.

There are six combinations of the data flow:

1. The TPG creates and writes the pattern in reserved video memory in DDR. The display controller displays video memory (TPG pattern) as shown in Figure 2-6.

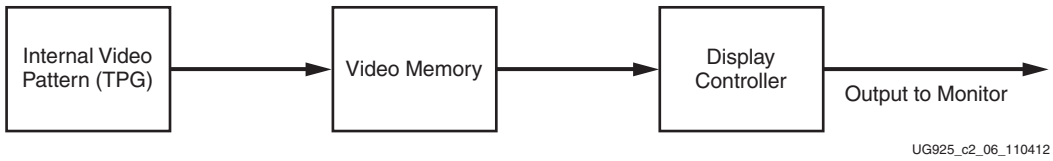

Figure 2-6: **Video Display Pipeline Data Flow**

2. Figure 2-7 shows the software Sobel filter configuration. The TPG creates and writes the pattern in intermediate DDR memory. The software Sobel filter reads the intermediate DDR memory, detects the edges, and writes the filtered image in the reserved video memory in DDR. The display controller displays video memory (filtered TPG pattern) as shown in Figure 2-7.

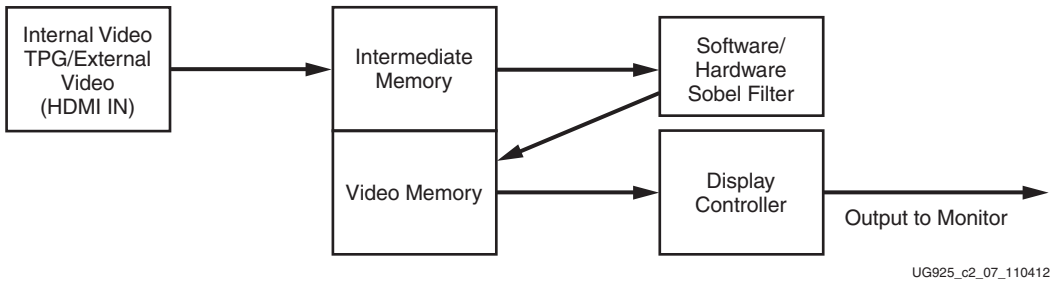

Figure 2-7: **Video Processing and Display Pipeline Data Flow**

3. Figure 2-7 shows the hardware Sobel filter configuration. The TPG creates and writes the pattern in intermediate DDR memory. The hardware Sobel filter reads the intermediate DDR memory (using VDMA), detects the edge, and writes the filtered image in the reserved video memory in DDR (using VDMA). The display controller displays video memory (filtered TPG pattern).

Similarly, there are three more cases (4, 5, and 6) where instead of an internally generated pattern, external video is used. See Figure 2-6 and Figure 2-7 with external video (HDMI IN) used instead of TPG for the video source. Cases 4, 5, and 6 are the same as cases 1, 2, and 3 with the only difference being the input source. In cases 1, 2, and 3, the input comes from TPG, whereas in cases 4, 5, and 6, the input comes from external video.

User Space Device Control

All the IPs need not have a kernel space driver, especially when they are memory-mapped and do not use interrupt. Such IPs can be configured and controlled from the user space by mapping the physical address range to the virtual address space in the user space.

These IPs are configured and controlled from the user space:

- TPG

- Clock detector
- Video timing controller

The APIs to configure these IPs are explained in the `udriver.h` file, which is part of the application source code provided with the Base TRD.

Software Sobel Filter Processing

This section describes the implementation of the software Sobel filter algorithm for edge detection. The algorithm runs in a separate thread and is turned On and Off by the controlling block. The application is compiled with the highest compiler optimization level (O3), so that the algorithm gives best performance. This algorithm takes the buffer address of the original frame as input and writes the filter image on the buffer address provided as output.

The following API is used for software Sobel filter processing:

```
void img_process(ZNQ_S32 *rgb_in_strm32, ZNQ_S32 *rgb_out_strm32, int height, int
width, int stride);
```

An object's edge detection filter algorithm for imaging/video looks for gradient magnitude changes in the image background and processing is performed in both the horizontal and vertical directions. The algorithm takes the RGB raw formatted image and detects the image object's edges based on the pixel luminance values.

```
Input : ZNQ_S32 *rgb_data_in
```

The implemented Sobel filter algorithm expects each pixel is a 32-bit ARGB integer with each color as 8-bit depth. A pointer to the ARGB pixel array is provided on a per frame basis.

```
Output : ZNQ_S32 *rgb_data_out
```

The implemented algorithm computes each output pixel's value of the frame using the Sobel filter edge detection algorithm. This function returns a 32-bit edge-detected ARGB array of the frame with each color as 8-bit depth.

Register Description

This appendix describes the details about configuration and control registers most commonly accessed by the Linux driver and application. The registers implemented in hardware are memory-mapped to the PS address range directly.

Clock Detect Registers

The clock detector's registers are used to detect the video clock.

Base address	0x40060000
Version	2.0.0a

Event Count Enable Register

The relative address of the clock detector's event count enable register is 0x00. Table A-1 describes this register's structure.

Relative address 0x00

Table A-1: **Event Count Enable Register**

Bit Position	Mode	Default Value	Description
31:1	-	–	Reserved
0	RW	0x0	Event count enable

Clock Count Register

The relative address of the clock detector's clock count register is 0x08. Table A-2 describes this register's structure.

Relative address 0x08

Table A-2: **Clock Count Register**

Bit Position	Mode	Default Value	Description
31:0	R	0x0	Clock counter value

Event Duration Register

The relative address of the clock detector's event duration register is 0x18. Table A-3 describes this register's structure.

Relative address 0x18

Table A-3: **Event Duration Register**

Bit Position	Mode	Default Value	Description
31:0	RW	0x0	After reaching this value, the event counter starts re-counting from zero. During the re-count, the clock counter is enabled.

Sobel Filter Registers

The Sobel filter registers are used to configure and control various internal features within the Sobel filter logic.

Base address 0x400D0000

Version N/A

Control and Status Register

The relative address of the Control and Status register is 0x00. Table A-4 describes this register's structure.

Relative address 0x00

Table A-4: **Control and Status Register**

Bit Position	Mode	Default Value	Description
31:8-6:3	-	-	Reserved.
7	RW	0x0	Auto Restart
2	R	0x1	IP is idle.

Table A-4: **Control and Status Register *(Cont'd)***

Bit Position	Mode	Default Value	Description
1	COR	0x0	Frame process is done.
0	RW/SC	0x0	Start processing the frame.

Global Interrupt Enable Register

The relative address of the Global Interrupt Enable register is 0x04. Table A-5 describes this register's structure.

Relative address 0x04

Table A-5: **Global Interrupt Enable Register**

Bit Position	Mode	Default Value	Description
31:1	-	-	Reserved.
0	RW	0x0	Global interrupt enable.

Interrupt Enable Register

The relative address of the Interrupt Enable register is 0x08. Table A-6 describes this register's structure.

Relative address 0x08

Table A-6: **Interrupt Enable Register**

Bit Position	Mode	Default Value	Description
31:1	-	-	Reserved.
0	RW	0x0	Frame processing done interrupt enable.

Interrupt Status Register

The relative address of the Interrupt Status register is 0x0C. Table A-7 describes this register's structure.

Relative address 0x0C

Table A-7: **Interrupt Status Register**

Bit Position	Mode	Default Value	Description
31:1	-	-	Reserved.
0	RW	0x0	Frame processing done interrupt status.

Number of Rows Register

The relative address of the Number of Rows register is 0x14. Table A-8 describes this register's structure.

Relative address 0x14

Table A-8: **Number of Rows Register**

Bit Position	Mode	Default Value	Description
31:0	RW	-	Number of rows in a frame.

Number of Columns Register

The relative address of the Number of Columns register is 0x1C. Table A-9 describes this register's structure.

Relative address 0x1C

Table A-9: **Number of Columns Register**

Bit Position	Mode	Default Value	Description
31:0	RW	-	Number of columns in a frame.

Directory Structure

This appendix describes the directory structure and organization of the files and folders delivered with the package.

Included Files and Systems

Figure B-1 gives a top level view of the directories and files included in the Zynq-7000 AP SoC ZC702 Base TRD package. Table B-1 summarizes how the directories provided relate to this user guide. This user guide uses the directories in the extracted Base TRD zipped package and assumes that the directories and files are placed or copied onto the user's local computer.

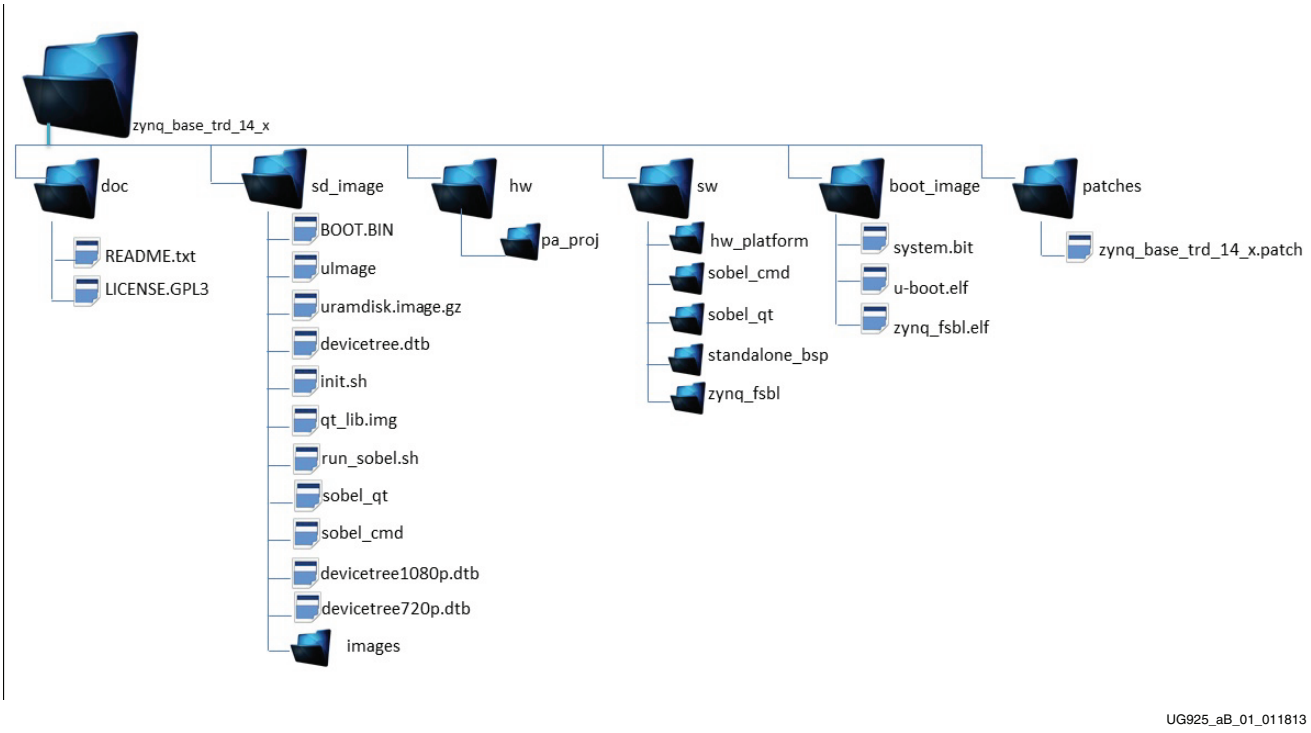

Figure B-1: **Directory Structure**

Table B-1: **Explanation of Directories in the Zynq-7000 AP SoC ZC702 Base TRD File System**

Directory	Purpose
doc	This directory contains the documents provided with the Zynq-7000 AP SoC ZC702 Base TRD, including this user guide.
sd_image	This directory contains a pre-built hardware design and software executables which provide a quick way to run the video demonstration. The application binary images are compatible to boot from the SD MMC and run the Linux application.
hw	This directory includes the Zynq-7000 AP SoC ZC702 Base TRD hardware design.
sw	This directory includes the Base TRD applications source code and hardware platform exported from the PlanAhead™ tool. The included ISE-SDK project can be used to build the application.
boot_image	This directory includes the hardware design bitfile and other Zynq-7000 system configuration executables in binary.
patches	This directory includes Base TRD Linux patches to the Xilinx standard Zynq-7000 AP SoC Linux kernel.

Install the Zynq-7000 AP SoC Design and Development Environment

Install the Xilinx ISE Design Suite

The user needs to install the Embedded Edition and System Edition of Xilinx ISE® Design Suite. Refer to UG798, *Xilinx Design Tools: Installation and Licensing Guide* to install and license ISE Design Suite.

Set Up Linux System Software Development Tools

To download and set up the ARM GNU tool chain for the Zynq-7000 AP SoC Linux system software and application development, refer to the Xilinx ARM GNU Tools wiki at wiki.xilinx.com/zynq-tools. The tool suite download requires a valid, registered Xilinx user login name and password.

Set Up git Tools

Git is a free software tool for managing distributed version control and the development of software project files. A git clone is a full-fledged repository with complete history and full revision tracking capabilities. Git gives the developer a local copy of the entire development project files and their existing (published) history on a git server within a repository organized as a set of directories. A user must clone a file or folder before using the same items from the project repository.

The user needs to have a git client installed on the Linux development PC. Refer to the Using Git web site wiki.xilinx.com/using-git for working with Xilinx git and to section 5.1 of UG821, *Zynq-7000 All Programmable SoC Software Developers Guide* for working with Xilinx git Linux repositories.

Appendix D

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support.

For continual updates, add the Answer Record to your myAlerts:

www.xilinx.com/support/myalerts.

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

Further Resources

The most up to date information related to the ZC702 board and its documentation is available on the following websites.

The Xilinx Zynq-7000 SoC ZC702 Evaluation Kit Product Page:

www.xilinx.com/zc702

The Zynq-7000 SoC ZC702 Evaluation Kit Master Answer Record:

www.xilinx.com/support/answers/47864.htm

These Xilinx documents provide supplemental material useful with this guide:

UG926, *Zynq-7000 All Programmable SoC: ZC702 Evaluation Kit and Video and Imaging Kit Getting Started Guide Zynq-7000 All Programmable SoC: ZC702 Evaluation Kit and Video and Imaging Kit Getting Started Guide*

UG585, *Zynq-7000 All Programmable SoC Technical Reference Manual*

DS406, *LogiCORE IP Processor System Reset Module Product Specification*

DS768, *LogiCORE IP AXI Interconnect*

PG020, *LogiCORE IP AXI Video Direct Memory Access Product Guide*

PG037, *LogiCORE IP AXI Performance Monitor Product Guide*

PG016, *LogiCORE IP Video Timing Controller Product Guide*

Xilinx Open Source ARM git Repository: git.xilinx.com/

UG798, *Xilinx Design Tools: Installation and Licensing Guide*

Xilinx ARM GNU Tools: wiki.xilinx.com/zynq-tools

Using Git: wiki.xilinx.com/using-git

git: the fast version control system home page: git-scm.com/

UG821, *Zynq-7000 All Programmable SoC Software Developers Guide*

Xilinx Zynq-7000 All Programmable SoC website
www.xilinx.com/products/silicon-devices/soc/zynq-7000/index.htm

Zynq-7000 All Programmable SoC product table
www.xilinx.com/publications/prod_mktg/zynq7000/Zynq-7000-combined-product-table.pdf

DS190, *Zynq-7000 All Programmable SoC Overview*

Zynq Linux: Downloading the Kernel Tree: xilinx.wikidot.com/zynq-linux#toc7

Zynq Linux: Configuring and Building the Linux Kernel:
xilinx.wikidot.com/zynq-linux#toc8

Xilinx Open Source Linux: wiki.xilinx.com/open-source-linux

Xilinx Device Tree Generator: xilinx.wikidot.com/device-tree-generator

Xilinx PlanAhead Design and Analysis Tool website
www.xilinx.com/tools/planahead.htm

UG873, *Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques Guide*

UG673, *Quick Front-to-Back Overview Tutorial: PlanAhead Design Tool*

PG012, *LogiCORE IP Chroma Resampler Product Guide*

PG014, *LogiCORE IP YCrCb to RGB Color-Space Converter Product Guide*

PG043 *LogiCORE IP Video In to AXI4-Stream v1.0*

PG103 *LogiCORE IP Test Pattern Generator*

More information on Zynq-7000 family boards, FMC extension cards, and other kits based on Zynq-7000 architecture is available here.

Xilinx Zynq-7000 All Programmable SoC Boards and Kits
www.xilinx.com/products/boards_kits/zynq-7000.htm

Xilinx Zynq-7000 Base Targeted Reference Design wiki page:
wiki.xilinx.com/zc702-base-trd

References

The following websites provide supplemental material useful with this guide:

1. Xylon IP Cores - logiCVC-ML Compact Multilayer Video Controller description
   www.logicbricks.com/Products/logiCVC-ML.aspx
2. Qt Online Reference Documentation. Qt is a toolkit for creating GUIs.
   doc.qt.nokia.com/
3. Device Tree general information
   devicetree.org/Main_Page
4. AMBA AXI4-Stream Protocol Specification
   infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0051a/index.html
5. PCI-SIG Documentation
   www.pcisig.com/specifications
6. logiCVC-ML Compact Multilayer Video Controller Data Sheet
   www.logicbricks.com/Documentation/Datasheets/IP/logiCVC-ML_hds.pdf
7. Silicon Labs CP210x USB to UART Bridge VCP Drivers
   www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

Regulatory and Compliance Information

This product is designed and tested to conform to the European Union directives and standards described in this section.

Declaration of Conformity

To view the Declaration of Conformity online, please visit:

http://www.xilinx.com/support/documentation/boards_and_kits/ce-declarations-of-conformity-xtp251.zip

Directives

2006/95/EC, *Low Voltage Directive (LVD)*

2004/108/EC, *Electromagnetic Compatibility (EMC) Directive*

Standards

EN standards are maintained by the European Committee for Electrotechnical Standardization (CENELEC). IEC standards are maintained by the International Electrotechnical Commission (IEC).

Electromagnetic Compatibility

EN 55022:2010, *Information Technology Equipment Radio Disturbance Characteristics – Limits and Methods of Measurement*

EN 55024:2010, *Information Technology Equipment Immunity Characteristics – Limits and Methods of Measurement*

This is a Class A product. In a domestic environment, this product can cause radio interference, in which case the user might be required to take adequate measures.

Safety

IEC 60950-1:2005, *Information technology equipment – Safety, Part 1: General requirements*

EN 60950-1:2006, *Information technology equipment – Safety, Part 1: General requirements*

Markings

This product complies with Directive 2002/96/EC on waste electrical and electronic equipment (WEEE). The affixed product label indicates that the user must not discard this electrical or electronic product in domestic household waste.

This product complies with Directive 2002/95/EC on the restriction of hazardous substances (RoHS) in electrical and electronic equipment.

This product complies with CE Directives 2006/95/EC, *Low Voltage Directive (LVD)* and 2004/108/EC, *Electromagnetic Compatibility (EMC) Directive*.

Warranty

THIS LIMITED WARRANTY applies solely to standard hardware development boards and standard hardware programming cables manufactured by or on behalf of Xilinx ("Development Systems"). Subject to the limitations herein, Xilinx warrants that Development Systems, when delivered by Xilinx or its authorized distributor, for ninety (90) days following the delivery date, will be free from defects in material and workmanship and will substantially conform to Xilinx publicly available specifications for such products in effect at the time of delivery. This limited warranty excludes: (i) engineering samples or beta versions of Development Systems (which are provided "AS IS" without warranty); (ii) design defects or errors known as "errata"; (iii) Development Systems procured through unauthorized third parties; and (iv) Development Systems that have been subject to misuse, mishandling, accident, alteration, neglect, unauthorized repair or installation. Furthermore, this limited warranty shall not apply to the use of covered products in an application or environment that is not within Xilinx specifications or in the event of any act, error, neglect or default of Customer. For any breach by Xilinx of this limited warranty, the exclusive remedy of Customer and the sole liability of Xilinx shall be, at the option of Xilinx, to replace or repair the affected products, or to refund to Customer the price of the affected products. The availability of replacement products is subject to product discontinuation policies at Xilinx. Customer may not return product without first obtaining a customer return material authorization (RMA) number from Xilinx.

THE WARRANTIES SET FORTH HEREIN ARE EXCLUSIVE. XILINX DISCLAIMS ALL OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, AND ANY WARRANTY THAT MAY ARISE FROM COURSE OF DEALING, COURSE OF PERFORMANCE, OR USAGE OF TRADE. (2008.10)

Do not throw Xilinx products marked with the "crossed out wheeled bin" in the trash. Directive 2002/96/EC on waste electrical and electronic equipment (WEEE) requires the separate collection of WEEE. Your cooperation is essential in ensuring the proper management of WEEE and the protection of the environment and human health from potential effects arising from the presence of hazardous substances in WEEE. Return the marked products to Xilinx for proper disposal. Further information and instructions for free-of-charge return available at: http:\\www.xilinx.com\ehs\weee.htm.