XAPP740 (v1.0) November 3, 2011

Designing High-Performance Video Systems with the AXI Interconnect

Author: James Lucero, Khang Dao, and Jose Alvarez

Summary

This application note covers the design considerations of a system using the performance features of the LogiCORE™ IP Advanced eXtensible Interface (AXI) Interconnect core. The design focuses on high system throughput through the AXI Interconnect core with F_{MAX} and area optimizations in certain portions of the design.

The design uses five AXI video direct memory access (VDMA) engines to simultaneously move 10 streams (five transmit video streams and five receive video streams), each in 1920 x 1080p format, 60 Hz refresh rate, and up to 32 data bits per pixel. Each VDMA is driven from a video test pattern generator (TPG) with a video timing controller (VTC) block to set up the necessary video timing signals. Data read by each AXI VDMA is sent to a common on-screen display (OSD) core capable of multiplexing or overlaying multiple video streams to a single output video stream. The output of the OSD core drives the DVI video display interface on the board. Performance monitor blocks are added to capture performance data. All 10 video streams moved by the AXI VDMA blocks are buffered through a shared DDR3 SDRAM memory and are controlled by a MicroBlaze™ processor.

The reference system is targeted for the Virtex-6 XC6VLX240TFF1156-1 FPGA on the Xilinx® ML605 Rev D evaluation board.

Included Systems

The reference design is created and built using version 13.2 of the Xilinx Platform Studio (XPS) tool, which is part of the ISE® Design Suite: System Edition. XPS helps simplify the task of instantiating, configuring, and connecting IP blocks together to form complex embedded systems. The design also includes software built using the Xilinx Software Development Kit (SDK). The software runs on the MicroBlaze processor subsystem and implements control, status, and monitoring functions. Complete XPS and SDK project files are provided with this application note to allow the user to examine and rebuild this design or use it as a template for starting a new design.

Introduction

High-performance video systems can be created using Xilinx AXI IP. The use of AXI Interconnect, Memory Interface Generator (MIG), and VDMA IP blocks can form the core of video systems capable of handling multiple video streams and frame buffers sharing a common DDR3 SDRAM memory. AXI is a standardized IP interface protocol based on the Advanced Microcontroller Bus Architecture (AMBA®) specification. The AXI interfaces used in the reference design consist of AXI4, AXI4-Lite, and AXI4-Stream interfaces as described in the AMBA AXI4 specifications [Ref 1]. These interfaces provide a common IP interface protocol framework to build the design around.

Together, the AXI interconnect and AXI MIG implement a high-bandwidth, multi-ported memory controller (MPMC) for use in applications where multiple devices share a common memory controller. This is a requirement in many video, embedded, and communications applications where data from multiple sources moves through a common memory device, typically DDR3 SDRAM.

AXI VDMA implements a high-performance, video-optimized DMA engine with frame buffering, scatter gather, and two-dimensional (2D) DMA features. AXI VDMA transfers video data streams to/from memory and operates under dynamic software control or static configuration modes.

A clock generator and processor system reset block supplies clocks and resets throughout the system. High-level control of the system is provided by an embedded MicroBlaze processor subsystem containing I/O peripherals, block RAMs, and processor support IP. To optimize the system to balance performance and area, multiple AXI Interconnect blocks are used to implement segmented/hierarchical AXI Interconnect networks with each AXI interconnect block individually tuned and optimized.

Hardware Requirements

The hardware requirements for this reference system are:

- Xilinx ML605 Rev D board
- Two USB Type-A to Mini-B 5-pin cables
- DVI cable
- Display monitor supporting 1080p resolution (1920 x 1080 resolution at 60 frames/sec)

The installed software tool requirements for building and downloading this reference system are:

- Xilinx Platform Studio 13.2
- ISE Design Suite 13.2
- SDK 13.2

Reference Design Specifics

The reference design includes the MicroBlaze processor, MDM, LMB block RAM, AXI_INTERCONNECT, AXI2AXI Connector, Clock Generator, MMCM_Module, PROC_SYS_RESET, AXI_UARTLITE, AXI IIC, AXI_INTC, AXI_V6_DDRX, AXI_BRAM, AXI_VTC AXI_TPG, AXI_VDMA, PERF_AXI_MM, AXI_OSD, and DVI_OUT IP cores.

See Figure 1 for the block diagram and Table 1 for the address map of the system.

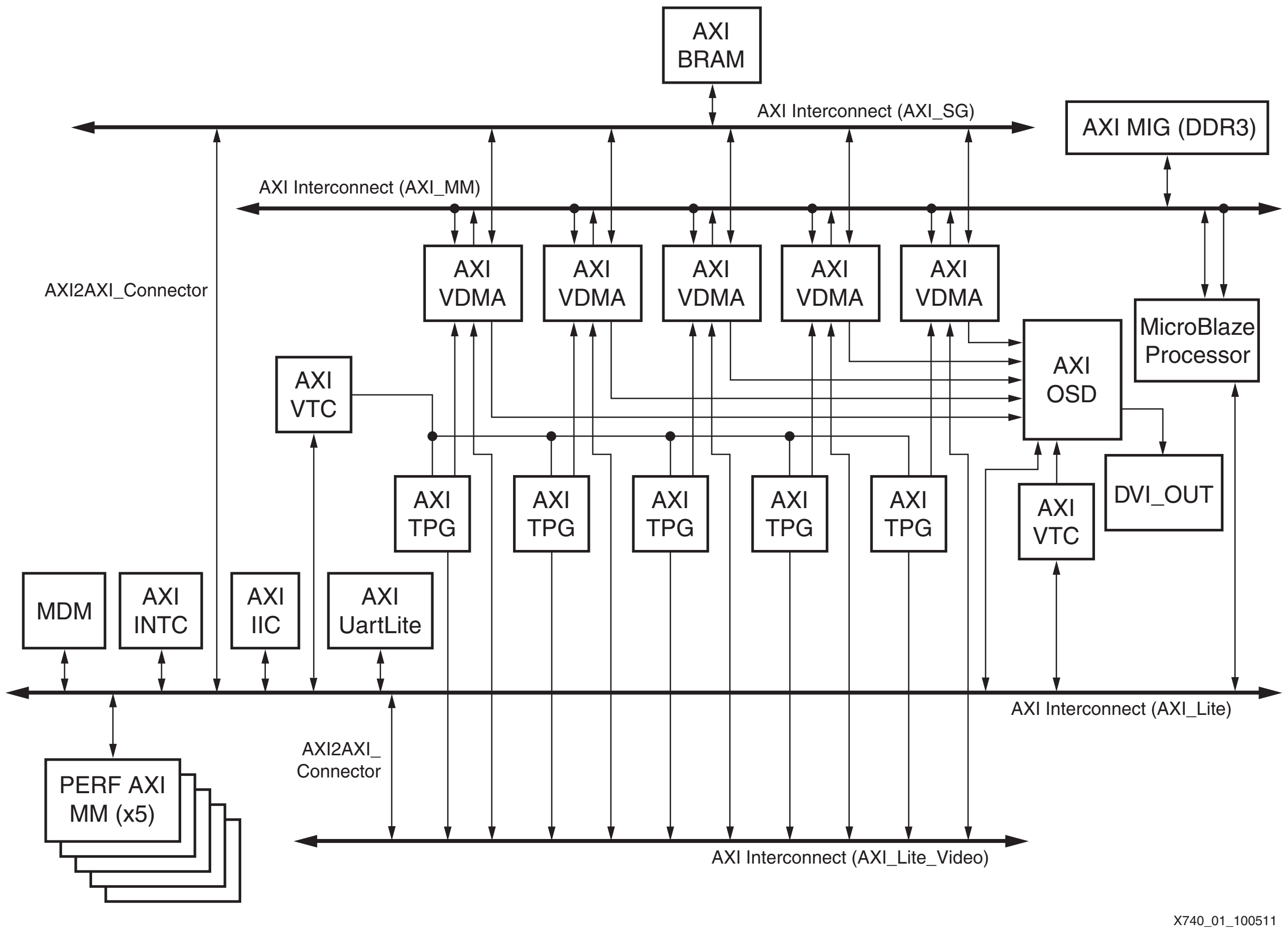


Figure 1: **Reference System Block Diagram**

Table 1: **Reference System Address Map**

Peripheral	Instance	Base Address	High Address
mdm	debug_module	0x74800000	0x7480FFFF
lmb_bram_if_cntlr	microblaze_0_ilmb	0x00000000	0x0000FFFF
lmb_bram_if_cntlr	microblaze_0_dlmb	0x00000000	0x0000FFFF
axi_uartlite	RS232_Uart_1	0x80600000	0x8060FFFF
axi_iic	IIC_DVI	0x80800000	0x8080FFFF
axi_intc	microblaze_0_intc	0x81200000	0x8120FFFF
axi_v6_ddrx	DDR3_SDRAM	0x40000000	0x5FFFFFFF
axi_vtc	axi_vtc_0	0x73820000	0x7382FFFF
axi_vtc	axi_vtc_1	0x73800000	0x7380FFFF
axi_osd	osd_0	0x73A00000	0x73A0FFFF

Table 1: **Reference System Address Map *(Cont'd)***

Peripheral	Instance	Base Address	High Address
axi_tpg	axi_tpg_0	0xBEE00000	0xBEE0FFFF
axi2axi_connector	axi2axi_connector_Video	0xB0000000	0xBFFFFFFF
axi2axi_connector	axi2axi_connector_SG	0xA0000000	0xA0007FFF
axi_vdma	axi_vdma_0	0xBE200000	0xBE20FFFF
axi_perf_mm	perf_axi_mm_0	0x90000000	0x9000FFFF
axi_tpg	axi_tpg_1	0xBEE10000	0xBEE1FFFF
axi_vdma	axi_vdma_1	0xBE210000	0xBE21FFFF
axi_perf_mm	perf_axi_mm_1	0x90010000	0x9001FFFF
axi_tpg	axi_tpg_2	0xBEE20000	0xBEE2FFFF
axi_vdma	axi_vdma_2	0xBE220000	0xBE22FFFF
axi_perf_mm	perf_axi_mm_2	0x90020000	0x9002FFFF
axi_tpg	axi_tpg_3	0xBEE30000	0xBEE3FFFF
axi_vdma	axi_vdma_3	0xBE230000	0xBE23FFFF
axi_perf_mm	perf_axi_mm_3	0x90030000	0x9003FFFF
axi_tpg	axi_tpg_4	0xBEE40000	0xBEE4FFFF
axi_vdma	axi_vdma_4	0xBE240000	0xBE24FFFF
axi_perf_mm	perf_axi_mm_4	0x90040000	0x9004FFFF

Hardware System Specifics

This section describes the high-level features of the reference design, including how to configure the main IP blocks. Information about useful IP features, performance/area trade-offs, and other configuration information are also provided. This information is applied to a video system, but the principles used to optimize the system performance are applicable to a wide range of high-performance AXI systems. For information about AXI system optimization and design trade-offs, see the *AXI Reference Guide* [Ref 2].

This application note assumes the user has some general knowledge of XPS. See *EDK Concepts, Tools, and Techniques: A Hands-On Guide to Effective Embedded System Design* [Ref 3] for more information about the XPS tools.

Video Related IP

The reference design implements five video pipelines running at 1080p60 (1920 x 1080 pictures at 60 frames/sec). Each picture consists of four bytes per pixel to represent an upper bound for high-quality video streams like RBGA or YUV 4:4:4 with alpha channel information (YUVA 4:4:4). Each video pipeline requires a bandwidth of 497.7 MB/s (~4 Gb/s).

The video traffic is generated by TPG IP cores and displayed by the OSD core. The total aggregate read/write bandwidth generated is equivalent to 10 video streams requiring nearly 5 GB/s (40 Gb/s).

This application note demonstrates AXI system performance using 10 high-definition video streams. At a minimum, video systems must include a source, some internal processing, and a destination. There can be multiple stages internally using a variety of IP modules. The canonical video system in Figure 2, page 5 shows that most video systems consist of input, pre-processing, main processing, post-processing, and output stages. Many of the video stages illustrated require memory access at video rates. Video data goes in/out of memory according to the requirements of internal processing stages. In this application note, a series of test pattern generators create the internal IP block memory traffic to simulate typical conditions.

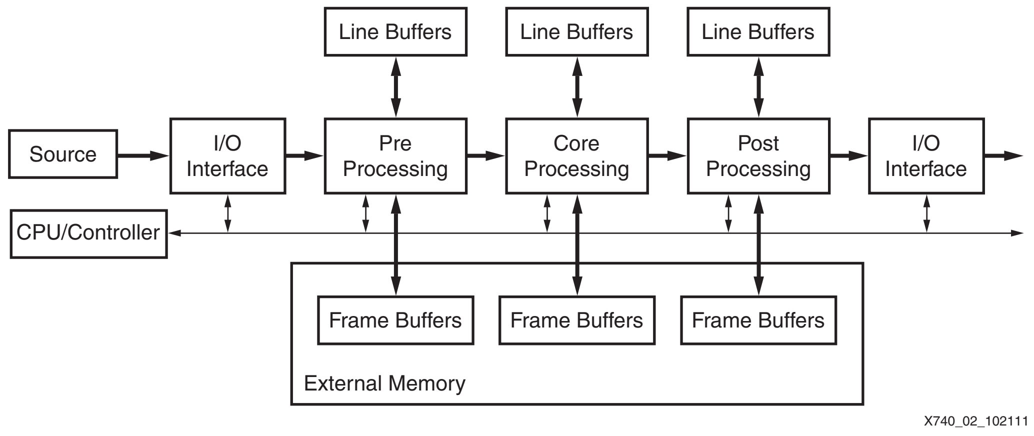


Figure 2: **Typical Video System**

AXI Interconnects

This design contains multiple AXI Interconnects each tuned to balance for throughput, area, and timing considerations (see *LogiCORE IP AXI Interconnect (1.03a)* data sheet [Ref 4]. The AXI_MM instance is used for high-speed masters and slaves that include high throughput and high F_{MAX} optimizations. The AXI_Lite, AXI_Lite_Video, and AXI_SG Interconnect instances are generally optimized for area and are used for the processor to access slave registers and to write VDMA buffer descriptors for control of AXI VDMA. In addition, AXI VDMA scatter gather (SG) engines are connected to fetch these buffer descriptors from the AXI_SG Interconnect portion of the design. The AXI VDMA scatter gather and buffer descriptor functions are described in detail in *LogiCORE IP AXI Video Direct Memory Access (axi_vdma) (v3.01.a)* data sheet [Ref 5].

AXI Interconnect (AXI_MM Instance)

This AXI Interconnect instance provides the highest F_{MAX} and throughput for the design by having a 256-bit core data width and running at 200 MHz.

The AXI Interconnect core data width and clock frequency match the capabilities of the attached AXI MIG so that width and clock converters between them are not needed. Sizing the AXI Interconnect core data width and clock frequency below the native width and clock frequency of the memory controller creates a system bandwidth bottleneck in the system. To help meet the timing requirements of a 256-bit AXI interface at 200 MHz, a rank of register slices are enabled between AXI_MM Interconnect and AXI MIG. Together, AXI Interconnect and AXI MIG form a 12-port AXI MPMC connected to a MicroBlaze processor ICache and DCache ports, five AXI VDMA MM2S ports, and five AXI VDMA S2MM ports. The configuration of this AXI Interconnect is consistent with the system performance optimization recommendations for an AXI MPMC based system as described in the Xilinx *AXI Reference Guide* [Ref 2].

AXI VDMA Instances

The AXI VDMA core is designed to provide video read/write transfer capabilities from the AXI4 domain to the AXI4-Stream domain, and vice versa. The AXI VDMA provides high-speed data movement between system memory and AXI4-Stream based target video IP. AXI4 interfaces are used for the high-speed data movement and buffer descriptor fetches across the AXI Interconnect.

The AXI VDMA core incorporates video-specific functionality, i.e., Gen-Lock and Frame Sync, for fully synchronized frame DMA operations and 2D DMA transfers. In addition to synchronization, frame store numbers and scatter gather or register direct mode operations are available for ease-of-control by the central processor.

In this design, the AXI VDMA scatter gather feature is used to demonstrate a system capable of complex DMA control and how such a system architecture would be implemented. However, many systems could be implemented sufficiently using the simpler register direct mode of AXI VDMA, which would remove the area cost of the scatter gather feature. Scatter gather should only be enabled if the system requires relatively complex software control of the AXI VDMA operations.

Initialization, status, and management registers in the AXI VDMA core are accessed through an AXI4-Lite slave interface.

This design uses five instances of AXI VDMA, each using the AXI4 SG engine interface, AXI4 MM2S interface, AXI4 S2MM interface, AXI4-Stream MM2S interface, and AXI4-Stream S2MM interfaces.

The 32-bit wide MM2S and S2MM interfaces from the AXI VDMA instances are connected to the AXI_MM instance of AXI Interconnect. The masters run at 148.5 MHz, which require asynchronous clock converters to the 200 MHz AXI Interconnect core frequency. Upsizers in the AXI Interconnect are used to convert 32-bit transactions from the AXI VDMA to 256-bit wide transactions to the AXI Interconnect core.

For maximum throughput for the AXI VDMA instances, the maximum burst length is set to 256. In addition, the master interfaces have a read and write issuance of 4 and a read and write FIFO depth of 512 to maximize throughput. These settings all follow performance recommendations for AXI endpoint masters as described in the Xilinx *AXI Reference Guide* [Ref 2].

In addition, line buffers inside the AXI VDMA for the read and write sides are set to 1K 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. See the *LogiCORE IP AXI Video Direct Memory Access (axi_vdma) (v3.01.a)* data sheet [Ref 5] for more information.

MicroBlaze Processor Instruction Cache and Data Cache

The MicroBlaze processor IC and DC masters are configured as 256-bit AXI4 masters. These masters run at 100 MHz because the MicroBlaze processor runs a software application from main memory that sets up and monitors the video pipelines. Running the MicroBlaze processor at this frequency helps timing and area. The 100 MHz clock setting ensures synchronous integer ratio clock converters in the AXI Interconnect can be used, which offers less latency and lower area than asynchronous converters.

AXI_V6_DDRX

The single slave connected to the Interconnect is the AXI_V6_DDRX memory controller (a block that integrates MIG into XPS). The memory controller's AXI Interface is 256 bits wide running at 200 MHz and disables narrow burst support for optimal throughput and timing. This configuration matches the native AXI interface clock and width corresponding to a 64-bit DDR3 DIMM at 400 MHz memory clock, which is the maximum performance of the memory controller for a Virtex-6 device in -1 speed grade.

The slave interface has a read/write issuance of eight. A 512-deep read FIFO is enabled for the port of the AXI Interconnect that is connected to the memory controller.

Register slices are enabled to ensure the interface meets timing at 200 MHz. These settings help ensure a high degree of transaction pipelining is active to improve system throughput. See the *Virtex-6 FPGA Memory Interface Solutions User Guide* [Ref 10] for more information about the memory controller.

AXI Interconnect (AXI_Lite, AXI_Lite_Video, AXI_SG)

The MicroBlaze processor data peripheral (DP) interface master writes and reads to all AXI4-Lite slave registers in the design for control and status information.

The AXI VDMA SG engines are 32 bits and do not require high F_{MAX} and throughput. Therefore, they are connected to a slower F_{MAX} portion of the design by a separate AXI Interconnect.

Because there are more than 16 AXI4-Lite slave interfaces in the design, AXI2AXI connectors and additional AXI Interconnect instances are required to allow the processor to access all the AXI4-Lite interfaces in the system.

The AXI_Lite, AXI_Lite_Video, AXI_SG AXI Interconnect blocks are configured for shared-access mode because high throughput is not required in this portion of the design. Therefore, area can be optimized over performance on these interconnect blocks. Also, these three interconnects are all clocked at 50 MHz to ensure that synchronous integer ratio clock converters in the AXI Interconnect can be used, which offer less latency and area than asynchronous clock converters.

AXI_Lite Interconnect

The slaves on the AXI_Lite Interconnect are for MDM, AXI_UARTLITE, AXI_IIC, AXI_INTC, AXI_VTC (two instances), PERF_AXI_MM (five instances), AXI OSD, and the slave AXI2AXI connectors to the AXI_Lite_Video and AXI_SG interconnects.

AXI_Lite_Video Interconnect

An AXI2AXI connector connects the AXI_Lite Interconnect to the AXI_Lite_Video Interconnect as a master. The slaves on this AXI Interconnect are AXI_TPG (five instances) and the AXI VDMA slave interface (five instances).

AXI_SG Interconnect

Another AXI2AXI connector connects the AXI_Lite Interconnect to the AXI_SG Interconnect as a master. In addition, the SG masters (five instances) are connected to this interconnect.

The slave connected to this AXI Interconnect is the AXI block RAM that acts as memory for holding AXI VDMA buffer descriptors. The AXI VDMA SG interfaces and the MicroBlaze processor are the main devices that communicate with AXI block RAM.

AXI VTC

The AXI 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. The output side of the core generates the horizontal and vertical blanking and synchronization pulses used in a standard video system including support for programmable pulse polarity.

The AXI VTC contains an AXI4-Lite Interface to access slave registers from a processor. For more information about the AXI VTC, see the *LogiCORE IP Video Timing Controller v3.0* data sheet [Ref 9].

In this design, two AXI VTC instances are used without detection. The first instance is used for the video input portion of the video pipelines. The second instance is used for the AXI OSD, which is the read portion of the video pipelines.

The Video Timing Controller v3.0 core is provided under license and can be generated using the CORE Generator™ tool v13.2 or higher.

AXI TPG

The AXI TPG contains an AXI4-Lite Interface to access slave control registers from a processor.

In this reference design, the video traffic to DDR3 memory is generated by a series of TPGs. Each TPG block can generate several video test patterns that are commonly used in the video industry for verification and testing. In the reference design, the TPG is used as a replacement for other video IP because only the amount of traffic generated to demonstrate the performance of the system is of interest. The control software demonstrates generation of color bars, horizontal and vertical burst patterns, and the generation of zone plates. No matter which test pattern is chosen, the amount of data generated is the same, namely, 1080p60 HD video (1920 x 1080 pictures at 60 frames/second). For example, a YUVA 4:4:4 1080p60 pattern generates 497.7 MB/s, which is nearly 4 Gb/s data stream.

Several operating modes are accessible through software control. In this application note, the TPG always generates a test pattern which could be one of color bars, horizontal frequency sweep, vertical frequency sweep, or zone plate. These patterns are meant for testing purposes only and are not calibrated for broadcast industry standards.

AXI OSD

The OSD LogiCORE IP provides a flexible, video-processing block for alpha blending, compositing up to eight independent layers, and generating simple text and graphics capable of handling images up to 4K x 4K sizes in YUVA 4:4:4 or RGBA image formats in 8, 10, or 12 bits-per-color component. In this application note, the OSD blends the five video streams as separate display layers. As the video streams generated by the TPG cores are enabled through software control, the display shows the blended layers on top of each other. Figure 3 shows a three-level block diagram of the OSD core.

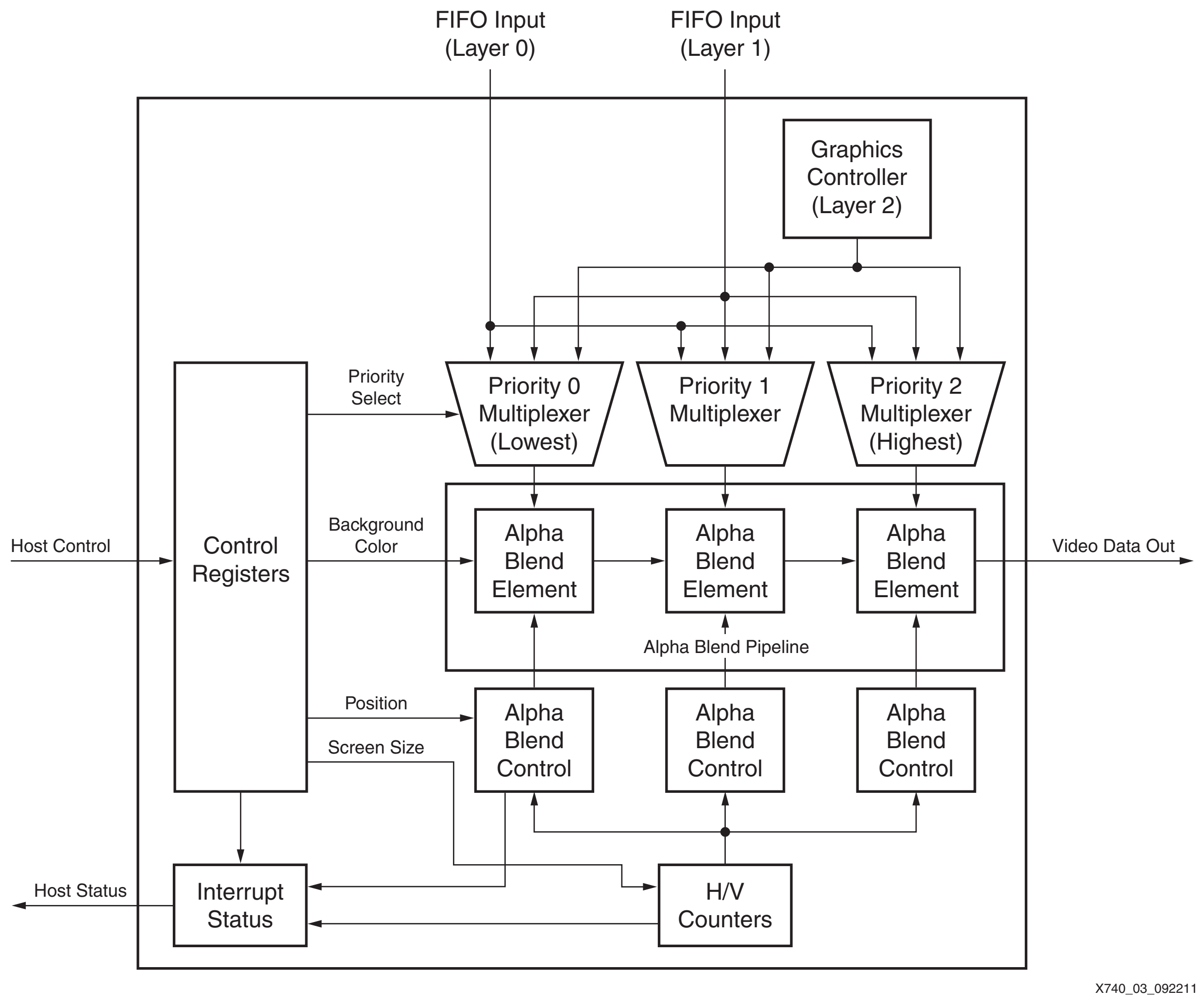


Figure 3: **Example 3-Layer OSD Block Diagram**

The AXI OSD contains an AXI4-Lite interface to access the slave registers from a processor. For more information about the AXI OSD, see the *LogiCORE IP Video On-Screen Display v2.0* data sheet [Ref 8].

The Video On-Screen Display core is provided under the SignOnce IP site license and can be generated using the CORE Generator tool, which is a part of the Xilinx ISE Design Suite software tool.

A simulation evaluation license for the core is shipped with the CORE Generator system. To access the full functionality of the core, including FPGA bitstream generation, a full license must be obtained from Xilinx.

PERF AXI MM

The PERF AXI MM core measures throughput for a master AXI read and a master AXI write channel connected to the AXI Interconnect. The core consists of a slave AXI4-Lite interface for the PERF AXI MM registers (see Table 2). The user needs to know the master ID width of the AXI Interconnect and the individual ID of the master to set the parameters of the core. The PERF AXI MM core only monitors the AXI read and write channels between the AXI master and the AXI Interconnect. The core does not modify or change any of the AXI transactions it is monitoring.

Several signals must be connected in the system to monitor the AXI protocol signals. These include the rd_clk, wr_clk, arvalid, awvalid, rvalid, rready, wvalid, wready, and S_AXI_ACLK. The read_start_cond and write_start_cond signals are connected to either net_vcc or signals in the design that should be included in the start condition of traffic monitoring.

For AXI read, the core measures between the first assertion of arvalid and read_start_cond until the value in the Number of Data Beats register is reached. A single data beat occurs when the rvalid and rready signals are asserted at the same time, meaning that one data beat has been transferred. The time between these two events are stored into the TX counter register.

For AXI write, the core measures between the first assertion of awvalid and write_start_cond until the value in the Number of Data Beats register is reached. The time between these two events is stored into the RX counter register.

In this application note, every video pipeline consists of the PERF AXI MM core to monitor performance on each AXI VDMA. All AXI VDMA instances are on the AXI_MM interconnect. The read_start_cond and write_start_cond signals come from the video pipeline's frame sync (fsync) signals from the AXI VTC instances. The other signals are connected from the AXI_MM interconnect.

Table 2: **PERF AXI MM Register**

Register Name	(offset from C_BASEADDR)	Access	Size in Bits	Description
Control	`0x0`	Read/Write	32	Bit0 = Look for transfer.
Number of Data Beats	`0x4`	Read/Write	32	Number of data beats to measure.

Status	`0x8`	Read	32	Bit0 = Busy RX
				Bit1 = Done RX
				Bit2 = Busy TX
				Bit3 = Done TX

TX Counter	`0xC`	Read	32	TX counter value.
RX Counter	`0x10`	Read	32	RX counter value.
Software Reset	`0x100`	Write	32	Reset core by writing `0xA`.

Driver Functions

Table 3 describes the PERF DMA driver functions to control the hardware.

Table 3: **PERF DMA Driver Functions**

Name	Purpose	Placement in Software
PERF_DMA_mReset	The PERF_DMA_mReset function's arguments are the base address for the PERF AXI MM core.	Use this function right before using the Setup_PERF_DMA function.
Setup_PERF_DMA	The Setup_PERF_DMA function's arguments are the base address for the PERF AXI MM and the number of data beats to measure. The control register is set to look for the transaction and the Number of Data Beats register is set.	Call this function before starting to monitor traffic.
Poll_Done_PERF_DMA	The Poll_Done_PERF_DMA function's argument is the base address for the PERF AXI MM core. The function polls the status register until the number of transactions are complete.	Call this function immediately after executing the Setup_PERF_DMA function.
TX_Transfer	The TX_Transfer function's arguments are the base address for the PERF AXI MM core, the period of the core in nanoseconds, and the total length of the transfer. This function reports the results of the TX transaction in MB/s.	Call this function after executing the Poll_Done_PERF_DMA function.
RX_Transfer	The RX_Transfer function's arguments are the base address for the PERF AXI MM core, the period of the core in nanoseconds, and the total length of the transfer. This function reports the results of the TX transaction in MB/s.	Call this function after executing the Poll_Done_PERF_DMA function.

Software Applications

AXI VDMA DISPLAY Application

The software application starts up the video pipelines and allows the user to examine bandwidth in real time and to display separate layers or alpha blend all layers on the LCD screen.

Application level software for controlling the system is written in C using the provided drivers for each IP. The programmer's model for each IP describes the particular API used by the drivers. Alternatively, application software can be written to use the IP control registers directly and handle the interrupts at the application level, but using the provided drivers and a layer of control at the application level is a far more convenient option.

The application software in the reference design performs the following actions:

- The software application first resets the DVI port on the ML605 board through the IIC interface.
- The TPG instances are set to write a default gray pattern that does not start until the AXI VTC instances are started.
- The AXI VDMA instances are started which consists of the processor writing the buffer descriptors to the AXI block RAM. The program then starts the read/write channels to begin the transfers for the VDMA instances.
- Then the AXI VTC instances are started with 1080p60 timing configuration.
- The AXI OSD is then configured for 1080p output.
- The DVI port on the ML605 is then configured for 1080p60 output.
- The five TPG instances in the design are then configured to write:
    - colorbars (layer 0)
    - zone plate patterns (layer 1)
    - vertical sweep (layer 2)
    - horizontal sweep (layer 3)
    - colorbars (layer 4)

After the initial setup sequence, the user can select to view a certain layer by selecting a number (option 0–4). When the number of a particular layer is selected, the OSD registers are modified to make the alpha blending on that particular layer be the highest value, while the others are at the smallest. In the case where option 5 is selected (alpha blending all layers), different values are given to the alpha blending register for each layer to show all layers on the LCD screen at the same time.

Options 6 and 7 set up the performance monitoring IP instances to measure throughput using these steps:

1. All performance monitors are reset.
2. All performance monitors's Number of Data Beat registers are set with values and then the control registers are set.
3. Measuring starts after fsync occurs.
4. All performance monitors poll to see when the transfer is complete.
5. Software functions are used to present the TX and RX throughput number on the HyperTerminal screen.

The only difference between options 6 and 7 is the amount of data being measured. Option 6 measures one *second* of video of data transfers. Option 7 measures only one *active frame* of video transfers.

Executing the Reference Design in Hardware

This section provides instructions to execute the reference design in hardware. This reference design runs on the ML605 board shown in Figure 4.

Note: The (xx) numbers in the instructions correspond to the callout numbers in Figure 4. Not all callout numbers are referenced.

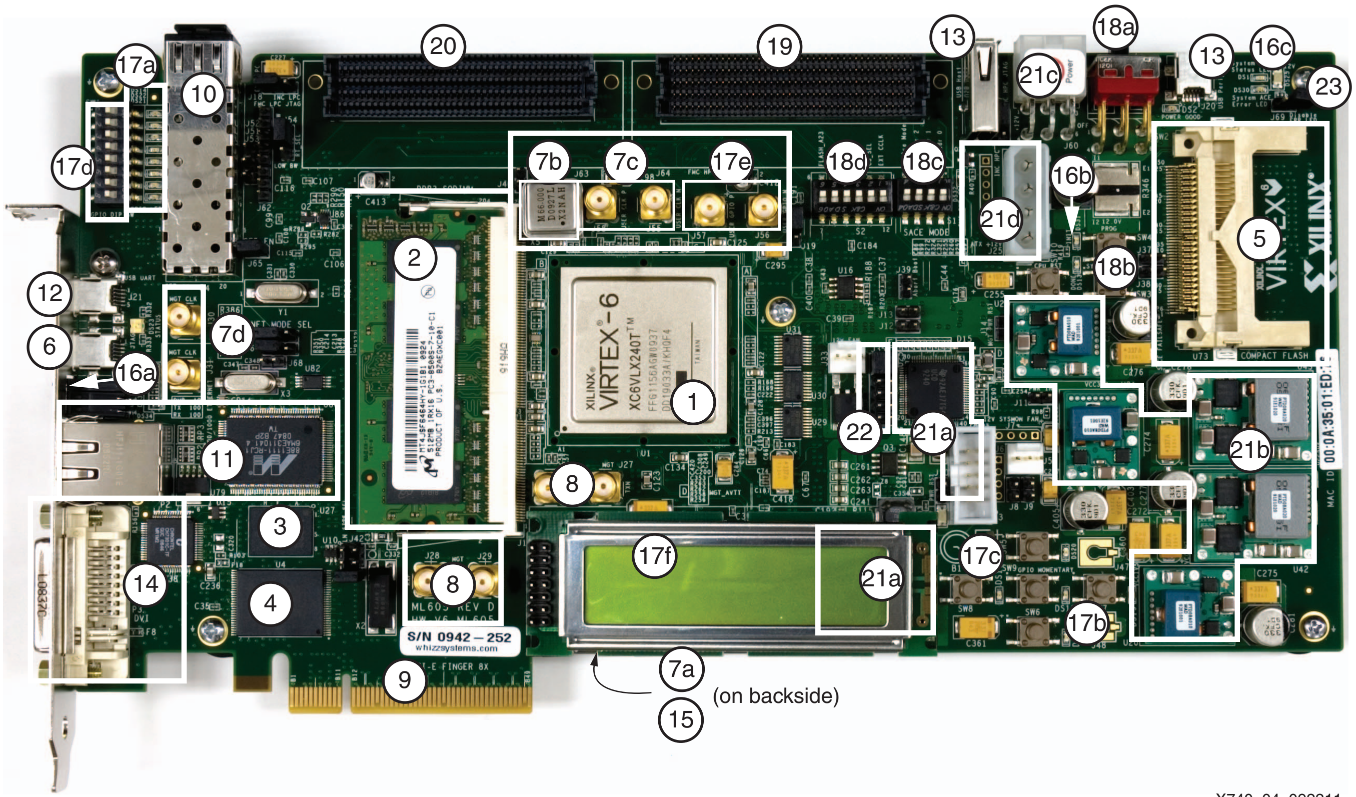


Figure 4: **ML605 Board**

6. Connect a USB cable from the host PC to the USB JTAG port (6). Ensure the appropriate device drivers are installed.
7. Connect a second USB cable from the host PC to the USB UART port (12). Ensure that the USB-UART drivers described above have been installed.
8. Connect the ML605 DVI connector (14) to a video monitor capable of displaying a 1920 x 1080p, 60 Hz video signal.
9. Connect power supply cable (21C).
10. Set power ON (18a).
11. Start a terminal program (e.g., HyperTerminal) on the host PC set for:
    a. Baud Rate: 9600
    b. Data Bits: 8
    c. Parity: None
    d. Stop Bits: 1
    e. Flow Control: None

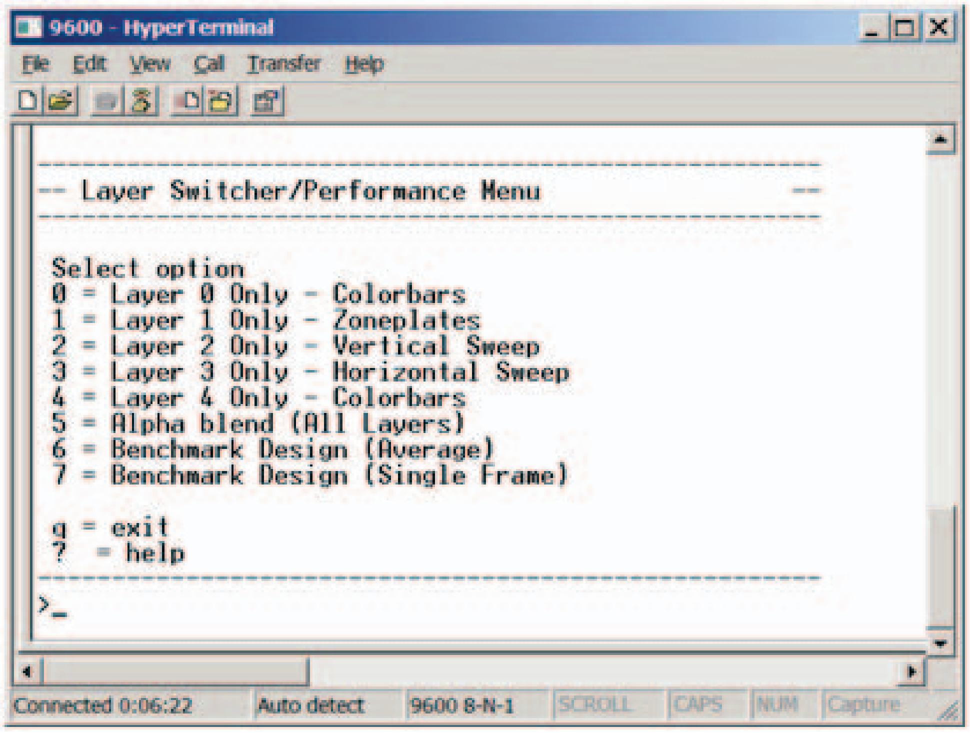
Executing the Reference System using the Pre-Built Bitstream and the Compiled Software Application

To execute the system using files in the `ready_for_download` directory in the `<unzip_dir>/ml605_video_5x_pipeline/` directory, follow these steps:

1. In a command shell or terminal window change directories into the `ready_for_download` directory.
   ```
   % cd <unzip dir>/ml605_video_5x_pipeline/ready_for_download
   ```
2. Invoke `xmd` by typing in the following command:
   ```
   % xmd
   ```
3. Download the bitstream inside XMD by typing in the following command:
   ```
   XMD% fpga -f download.bit
   ```
4. Connect to the processor inside XMD by typing in the following command:
   ```
   XMD% connect mb mdm
   ```
5. Download the processor code (ELF) file by typing in the following command:
   ```
   XMD% dow axi_vdma_display.elf
   ```
6. Run the software by typing in the following command:
   ```
   XMD% run
   ```

Results from Running Hardware and Software

The LCD monitor connected to the ML605 board displays a colorbar pattern and the HyperTerminal screen displays the output shown in Figure 5.



```
-------------------------------------------------------------
-- Layer Switcher/Performance Menu                         --
-------------------------------------------------------------

 Select option
 0 = Layer 0 Only - Colorbars
 1 = Layer 1 Only - Zoneplates
 2 = Layer 2 Only - Vertical Sweep
 3 = Layer 3 Only - Horizontal Sweep
 4 = Layer 4 Only - Colorbars
 5 = Alpha blend (All Layers)
 6 = Benchmark Design (Average)
 7 = Benchmark Design (Single Frame)

 q = exit
 ?  = help
-------------------------------------------------------------
>_
```

X740_05_092111

Figure 5: **HyperTerminal Output**

The user can choose from eight options displayed on the HyperTerminal screen:

- 0 — Shows the colorbar layer on LCD
- 1 — Shows zone plate layer on LCD
- 2 — Shows vertical sweep layer on LCD
- 3 — Shows horizontal sweep layer on LCD
- 4 — Shows another colorbar layer on LCD
- 5 — Shows an alpha blend of all layers at the same time (0–4) on LCD
- 6 — HyperTerminal shows real time system performance (one second of transfers)
- 7 — HyperTerminal shows real time system performance of one frame

Performance

The AXI_MM interconnect is 256 bits running at 200 MHz. The theoretical maximum bandwidth on each channel is 6.4 GB/s.

The DDR3 PHY is set for 64 bits with a memory clock frequency of 400 MHz (800 MHz data rate). The theoretical throughput on DDR3 is 6.4 GB/s, which is the total bandwidth available in the design.

Using option 6 of the software application should show the following output (the numbers might vary slightly from the values shown below):

```
DMA0 TX 497.699333
DMA0 RX 497.997408

DMA1 TX 497.699521
DMA1 RX 497.997408
```

```
DMA2 TX 497.700490
DMA2 RX 497.997408

DMA3 TX 497.699685
DMA3 RX 497.997408

DMA4 TX 497.699605
DMA4 RX 497.997408
```

Total bandwidth is ~4,976 MB/s out of 6,400 MB/s, which is around 78% of the total theoretical bandwidth of the main memory.

Using option 7 of the software application should show the following output (the numbers might vary slightly from the values shown below):

```
DMA0 TX 499.790069
DMA0 RX 518.459871

DMA1 TX 499.781349
DMA1 RX 518.459871

DMA2 TX 499.757218
DMA2 RX 518.459871

DMA3 TX 499.780335
DMA3 RX 518.459871

DMA4 TX 499.785405
DMA4 RX 518.459871
```

Total bandwidth is ~5,090 MB/s out of 6,400 MB/s, which is around 80% of the total bandwidth of the main memory. Throughput with option 7 is higher than with option 6 because only one active frame's performance is analyzed, which does not include blank periods between frames.

Building Hardware

This section covers rebuilding the hardware design. Before rebuilding the project, ensure that the licenses for AXI OSD and AXI VTC are installed. To obtain evaluation licenses for the AXI VTC or AXI OSD, see the following Web pages:

- http://www.xilinx.com/products/intellectual-property/EF-DI-VID-TIMING.htm
- http://www.xilinx.com/products/intellectual-property/EF-DI-OSD.htm

Note: The generated bitstream is at:

```
<unzip
dir>/ml605_video_5x_pipeline/HW/V6_MB_video_pipelines/implementation/down
load.bit
```

1. Open `ml605_video_5x_pipeline/HW/V6_MB_video_pipelines/system.xmp` in XPS.
2. Select `Hardware > Generate Bitstream` to generate a bitstream for the system.
3. Run `Device Configuration > Update Bitstream` to initialize the block RAM with a bootloop program to ensure the processor boots up with a stable program in memory.

Compiling Software and Running Design through SDK

Compiling Software in SDK

1. Start SDK. In Linux, type in `xsdk` to start SDK.
2. In the Workspace Launcher, select the following Workspace:

   `<unzip dir>/ml605_video_5x_pipeline/SW/SDK_Workspace`
3. Click **OK**.
4. Set the repository, by selecting **Xilinx Tools > Repositories**
   a. For Local Repositories click on **New...**
   b. Change directories to

      `<unzip dir>/ml605_video_5x_pipeline/SW/repository`
   c. Click **OK**.
5. Import the BSP, hardware platform, and software applications by selecting `File > Import > General > Existing Projects into Workspace`.
6. Click **Next**, then browse to `<unzip dir>/ml605_video_5x_pipeline/SW`.
7. Click **OK**.
8. Ensure all check boxes are selected (including **axi_vdma_display** and **hw_platform_0**)
9. Ensure the associated software applications are selected.
10. Click **Finish**.

    Note: The BSP and software applications compile at this step. The process takes 2–5 minutes.
11. At this point, the user can modify existing software applications and create new software applications in SDK.

Running the Hardware and Software through SDK

1. Select **Xilinx Tools > Program FPGA**.

   Note: Ensure bootloop is used for microblaze_0.
2. Click **Program**.
3. In the Project Explorer Window, right click `axi_vdma_display > Run As > Launch on Hardware`.

Design Characteristics

This reference design is implemented in a Virtex-6 FPGA (XC6VLX240TFFG1156-1) using the ISE Design Suite: Embedded Edition 13.2.

The resources used are:

- Total LUTs used: 53,727 out of 150,720 (35%)
- Total I/Os used: 138 out of 600 (23%)
- Total internal memory used:
  - RAMB36E1s: 121 out of 416 (29%)
  - RAMB18E1s: 30 out of 832 (3%)

Note: Device resource utilization results are dependent on the implementation tool versions. Exact results can vary. These numbers should be used as a guideline.

Reference Design

The reference design has been fully verified and tested on hardware. The design includes details on the various functions of the different modules. The interface has been successfully placed and routed at 200 MHz on the main AXI Interfaces to the memory controller using the Xilinx ISE Design Suite 13.2 tools.

The reference design files for this application note can be downloaded at:

https://secure.xilinx.com/webreg/clickthrough.do?cid=177271

The reference design checklist is shown in Table 4.

Table 4: **Reference Design Checklist**

Parameter	Description
General	
Developer name	James Lucero, Khang Dao, and Jose Alvarez
Target devices (stepping level, ES, production, speed grades)	Virtex-6 FPGA
Source code provided	Yes
Source code format	VHDL/Verilog (some sources encrypted)
Design uses code/IP from existing Xilinx application note/reference designs, CORE Generator software, or 3rd-party	Reference designs provided for EDK and video cores generated from the CORE Generator tool
Simulation	
Functional simulation performed	N/A—Simulation not supported
Timing simulation performed	N/A—Simulation not supported
Testbench used for functional and timing simulations	N/A—Simulation not supported
Testbench format	N/A—Simulation not supported
Simulator software/version used	N/A—Simulation not supported
SPICE/IBIS simulations	N/A—Simulation not supported
Implementation	
Synthesis software tools/version used	XST 13.2
Implementation software tools/versions used	ISE Design Suite 13.2: System Edition
Static timing analysis performed	Y (passing timing in PAR/TRCE)
Hardware Verification	
Hardware verified	Y
Hardware platform used for verification	ML605 board

Utilization and Performance

Table 5 shows the device and utilization information.

Table 5: **Device and Utilization**

Device	Speed Grade	Package	Slice Registers	Occupied Slices	Slice LUTs	I/Os	RAMB36E1s	RAMB18E1s
XC6VLX240T	-1	FFG1156	73,574 (24%)	24,766 (65%)	53,727 (35%)	138 (23%)	121 (29%)	30 (3%)

Device resource utilization is detailed in Table 6 for IP cores shown in Figure 1, page 3. The information in Table 6 is taken from the Design Summary tab in XPS under the `Design Overview > Module Level Utilization` report selection. The utilization information is approximate due to cross-boundary logic optimizations and logic sharing between modules.

Table 6: **Module Level Resource Utilization**

IP Core	Instance Name	Slices(1)	Slice Regs	LUTs	LUTRAM	BRAM/FIFO	DSP48E1	BUFG	BUFR	MMCM_ADV
AXI V6 DDR3 Controller	DDR3_SDRAM	3,979	7,693	6,643	1500	000	0	0	2	0
AXI Interconnect	AXI_Lite	811	1,456	985	136	0	0	0	0	0
	AXI_Lite_Video	181	75	255	0	0	0	0	0	0
	AXI_MM	9,125	24,206	15,890	2,570	49	0	0	0	0
	AXI_SG	220	80	264	0	0	0	0	0	0
PERF AXI MM	perf_axi_mm_0	171	375	412	0	0	0	0	0	0
	perf_axi_mm_1	179	375	386	0	0	0	0	0	0
	perf_axi_mm_2	180	375	418	0	0	0	0	0	0
	perf_axi_mm_3	182	375	376	0	0	0	0	0	0
	perf_axi_mm_4	181	375	384	0	0	0	0	0	0
AXI VTC	axi_vtc_0	458	719	695	2	0	0	0	0	0
	axi_vtc_1	470	720	685	2	0	0	0	0	0
AXI TPG (Includes Associated Glue Logic)	axi_tpg_0	387	1,090	895	14	1	3	0	0	0
	xsvi2axi_0	10	27	8	0	1	0	0	0	0
	axi_tpg_1	378	1,090	879	14	1	3	0	0	0
	xsvi2axi_1	7	27	12	0	1	0	0	0	0
	axi_tpg_2	375	1,090	880	14	1	3	0	0	0
	xsvi2axi_2	10	27	8	0	1	0	0	0	0
	axi_tpg_3	394	1,090	891	14	1	3	0	0	0
	xsvi2axi_3	9	27	8	0	1	0	0	0	0
	axi_tpg_4	388	1,090	897	14	1	3	0	0	0
	xsvi2axi_4	8	27	12	0	1	0	0	0	0
AXI VDMA	axi_vdma_0	2,004	4,709	3,190	233	10	0	0	0	0
	axi_vdma_1	2,038	4,709	3,067	233	10	0	0	0	0
	axi_vdma_2	2,008	4,709	3,094	233	10	0	0	0	0
	axi_vdma_3	2,050	4,709	3,085	233	10	0	0	0	0
	axi_vdma_4	2,033	4,709	3,100	233	10	0	0	0	0

Table 6: **Module Level Resource Utilization *(Cont'd)***

IP Core	Instance Name	Slices[(1)]	Slice Regs	LUTs	LUTRAM	BRAM/FIFO	DSP48E1	BUFG	BUFR	MMCM_ADV
AXI OSD (Includes Associated Glue Logic and Display Driver)	osd_0	1,885	4,650	2,149	139	0	15	0	0	0
	dvi_out_0	0	0	0	0	0	0	0	0	0
	vfbc2axi_0	6	32	8	0	0	0	0	0	0
	vfbc2axi_1	6	32	8	0	0	0	0	0	0
	vfbc2axi_2	6	32	12	0	0	0	0	0	0
	vfbc2axi_3	7	32	4	0	0	0	0	0	0
	vfbc2axi_4	6	32	9	0	0	0	0	0	0
Clock, Reset, and Miscellaneous System Logic	clock_generator_0	0	0	0	0	0	0	4	0	1
	mmcm_module_0	1	0	1	0	0	0	1	0	1
	proc_sys_reset_0	18	31	21	2	0	0	0	0	0
	system	18	8	13	0	0	0	0	0	0
MicroBlaze Processor Subsystem (Includes Local Memory and Debug Module for JTAG Based Debug)	microblaze_0	1,817	1,622	2,879	475	18	3	0	0	0
	debug_module	95	130	124	23	0	0	1	0	0
	microblaze_0_bram_block	0	0	0	0	16	0	0	0	0
	microblaze_0_d_bram_ctrl	5	2	3	0	0	0	0	0	0
	microblaze_0_dlmb	1	1	0	0	0	0	0	0	0
	microblaze_0_i_bram_ctrl	3	2	1	0	0	0	0	0	0
	microblaze_0_ilmb	1	1	0	0	0	0	0	0	0
	microblaze_0_intc	116	141	179	0	0	0	0	0	0
AXI IIC	IIC_DVI	209	248	334	20	0	0	0	0	0
AXI BRAM Controller	Internal_BRAM	254	385	466	0	0	0	0	0	0
	axi_bram_if_cntlr_1_bram	0	0	0	0	8	0	0	0	0
AXI UartLite	RS232_Uart_1	69	83	97	18	0	0	0	0	0
Total		**N/A**[(1)]	**73,418**	**53,727**	**6,122**	**151**	**33**	**6**	**2**	**2**

Notes:

1. Slices can be packed with basic elements from multiple IP cores and hierarchies. Therefore, a slice is counted in every hierarchical module that each of its packed basic elements belong to. This results in some double counting of slice counts when adding up the slice counts across modules.

Table 7 summarizes the bandwidth calculations for the physical memory interface.

Table 7: **Physical DDR3 Memory Interface Maximum Theoretical Bandwidth**

Data Width	Data Rate	Maximum Theoretical Bandwidth
64 bits (SODIMM)	800 Mb/s	6.40 GB/s (51.2 Gb/s)

Table 8 summarizes the total bandwidth of video data moved through memory.

Table 8: **Average Bandwidth Used for Video Traffic**

Frame Resolution	Refresh Rate	Bits per Pixel	Number of Video Streams	Total Aggregate Video Bandwidth
1920x1080	60 Hz	32 bits	10	4.98 GB/s (39.8 Gb/s)

Table 9 summarizes the percentage of the maximum theoretical bandwidth used by the video streams.

Table 9: **Percentage of the Maximum Theoretical Bandwidth Used**

Total Aggregate Video Bandwidth	Maximum Theoretical Bandwidth	Percentage of the Maximum Theoretical Bandwidth Used(1)
4.98 GB/s (39.8 Gb/s)	6.40 GB/s (51.2 Gb/s)	77.8%

Notes:

1. It is not possible to utilize 100% of the theoretical bandwidth of the DDR3 memory because some memory clock cycles must be used for refresh, write leveling, periodic reads, bank/row changes, and read/write turnaround.

References

This document uses the following references:

1. AMBA AXI4 specifications
http://infocenter.arm.com/help/topic/com.arm.doc.set.amba/index.html#specs
2. UG761, *AXI Reference Guide*
3. UG683, *EDK Concepts, Tools, and Techniques: A Hands-On Guide to Effective Embedded System Design*
4. DS768, *LogiCORE IP AXI Interconnect (1.03a)*
5. DS799, *LogiCORE IP AXI Video Direct Memory Access (axi_vdma) (v3.01.a)*
6. UG081, *MicroBlaze Processor Reference Guide: Embedded Development Kit EDK 13.2*
7. UG111, *Embedded System Tools Reference Manual: EDK 13.2*
8. DS837, *LogiCORE IP Video On-Screen Display v2.0*
9. DS857, *LogiCORE IP Video Timing Controller v3.0*
10. UG406, *Virtex-6 FPGA Memory Interface Solutions User Guide*
11. UG668, *Getting Started with the Virtex-6 FPGA ML605 Embedded Kit*

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
11/03/11	1.0	Initial Xilinx release.

Notice of Disclaimer