XAPP1205 (v1.0) March 28, 2014

Designing High-Performance Video Systems with the Zynq-7000 All Programmable SoC Using IP Integrator

Author: James Lucero and Bob Slous

Summary

With high-end processing platforms such as the Xilinx Zynq®-7000 All Programmable SoC, customers want to take full advantage of the processing system (PS) and custom peripherals available within the device. An example of this philosophy is a system containing multiple video pipelines in which live video streams are written into memory (input) and memory content is sent out to live video streams (output) while the processor is accessing memory. This application note covers design principles for obtaining high performance from the Zynq-7000 AP SoC memory interfaces, from AXI master interfaces implemented in the programmable logic (PL), and from the ARM® Cortex™-A9 processor(s).

With video streams, guaranteed worst-case latency is required to ensure that frames are not dropped or corrupted. To provide high-speed AXI interface masters in the PL with lower latency and direct access to the Zynq-7000 AP SoC memory interfaces, connections to the High Performance (HP) interfaces are required. The Zynq-7000 AP SoC contains four HP interfaces that are 64-bit or 32-bit AXI3 slave interfaces designed for high throughput.

This design uses four AXI Video Direct Memory Access (VDMA) cores to simultaneously move eight streams (four transmit video streams and four receive video streams), each in 1920 x 1080p format, 60 Hz refresh rate, and up to 24 data bits per pixel. Each AXI Video DMA core is driven from a video test pattern generator (TPG) with a Video Timing Controller (VTC) core to set up the necessary video timing signals. Data read by each AXI Video DMA core is sent to a common Video On-Screen Display (OSD) core capable of multiplexing or overlaying multiple video streams to a single output video stream. The onboard HDMI™ video display interface is driven by the output of the Video On-Screen Display core with additional IP cores.

An AXI Performance Monitor core is used to capture performance data. All four AXI Video DMA cores are connected to four separate HP interfaces using the AXI Interconnect and are controlled by the Cortex-A9 processor. This design uses 70% of the memory controller bandwidth.

The reference system design is targeted for the Zynq-7000 AP SoC ZC702 evaluation board.

Included Systems

The design is built using the Vivado® Design Suite, System Edition 2013.4 and the Vivado IP integrator feature. The IP integrator helps simplify the task of instantiating, configuring, and connecting IP cores together to form complex integrated systems. The design also includes software built using the Xilinx Software Development Kit (SDK). The software runs on the Zynq-7000 AP SoC PS and implements the control function. The complete IP integrator project and SDK tool project files are provided with this application note to allow the designer to examine and rebuild this design or use the files as a template for starting a new design.

Included with this application note is one reference system, `zc702_video_4x_ipi`, available in the ZIP file `xapp1205-high-performance-video-zynq.zip`. See the Reference Design section.

Introduction

High performance video systems can be created using available Xilinx LogiCORE™ IP AXI Interface cores. Using AXI interconnect, AXI3 ports on the Zynq-7000 AP SoC, and AXI Video DMA IP cores can form the basis of video systems capable of handling multiple video streams and multiple video frame buffers sharing a common DDR3 SDRAM. AXI is a standardized IP interface protocol based on the ARM AMBA4 and AMBA3 AXI specifications. The AXI interfaces used in this example design consists of AXI4, AXI3, AXI4-Lite, and AXI4-Stream interfaces as described in the AMBA4 and AMBA3 AXI specification [Ref 3]. These interfaces provide a common IP interface protocol framework for building the design.

AXI interconnect and AXI HP ports on the Zynq-7000 AP SoC implement a high-bandwidth multi-ported memory controller (MPMC) for use in applications where multiple devices share a common memory controller. This configuration is a requirement in many video, embedded, and communications applications where data from multiple sources move through a common memory device, typically DDR3 SDRAM.

The AXI Video DMA core implements a high-performance video optimized DMA engine with frame buffering, scatter gather (typically not used), and 2-Dimensional (2D) DMA features. The AXI Video DMA core transfers video data streams to and from memory and operates under dynamic software control or static configuration modes.

The Zynq-7000 AP SoC PS supplies clocks and resets throughout the system including the PL. High-level control of the system is provided in the Zynq-7000 AP SoC PS by the Cortex-A9 processor and through the I/O peripherals (IOP), on-chip memory (OCM), and processor support IP cores. To optimize the system to balance performance and area utilization, multiple AXI interface cores are used to implement segmented/hierarchical AXI interface networks with each AXI interface core individually tuned and optimized.

Hardware Requirements

The hardware requirements for this reference system are:

- Xilinx ZC702 Rev C evaluation board (in JTAG mode)
- Two USB Type-A to Mini-B 5-pin cables
- HDMI cable
- Display monitor supporting 1080P resolution (1920x1080 resolution at 60 frames/sec)

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

- Vivado Design Suite, System Edition 2013.4
- SDK 2013.4

Reference System Specifics

The reference system includes these LogiCORE IP cores:

- Processing System 7
- AXI Memory Interconnect (AXI_INTERCONNECT)
- Video Timing Controller (V_TC)
- Test Pattern Generator (V_TPG)
- AXI Video DMA (AXI_VDMA)
- AXI Performance Monitor (AXI_PERF_MON)
- Video On-Screen Display (V_OSD)
- AXI4-Stream to Video Out (AXI4S_VID_OUT)
- Chroma Resampler (V_CRESAMPLE)
- RGB to YCrCb Color-Space Converter (V_RGB2YCRCB)

The design also includes a clock generator and a custom core, ZYNQ_ADDR_SWITCH.

The processor(s) or DMA controller (DMAC) in the Zynq-7000 AP SoC PS can access AXI slave interfaces in the PL using the AXI general purpose (GP) interfaces, which are 32-bit AXI3 master interfaces. The S_AXI_GPx interfaces are considered lower performance interfaces compared to S_AXI_HPx interfaces and are not used in this design. This design uses only one AXI GP interface (M_AXI_GP0) for slave registers in the design.

Figure 1 shows a block diagram of the reference system.

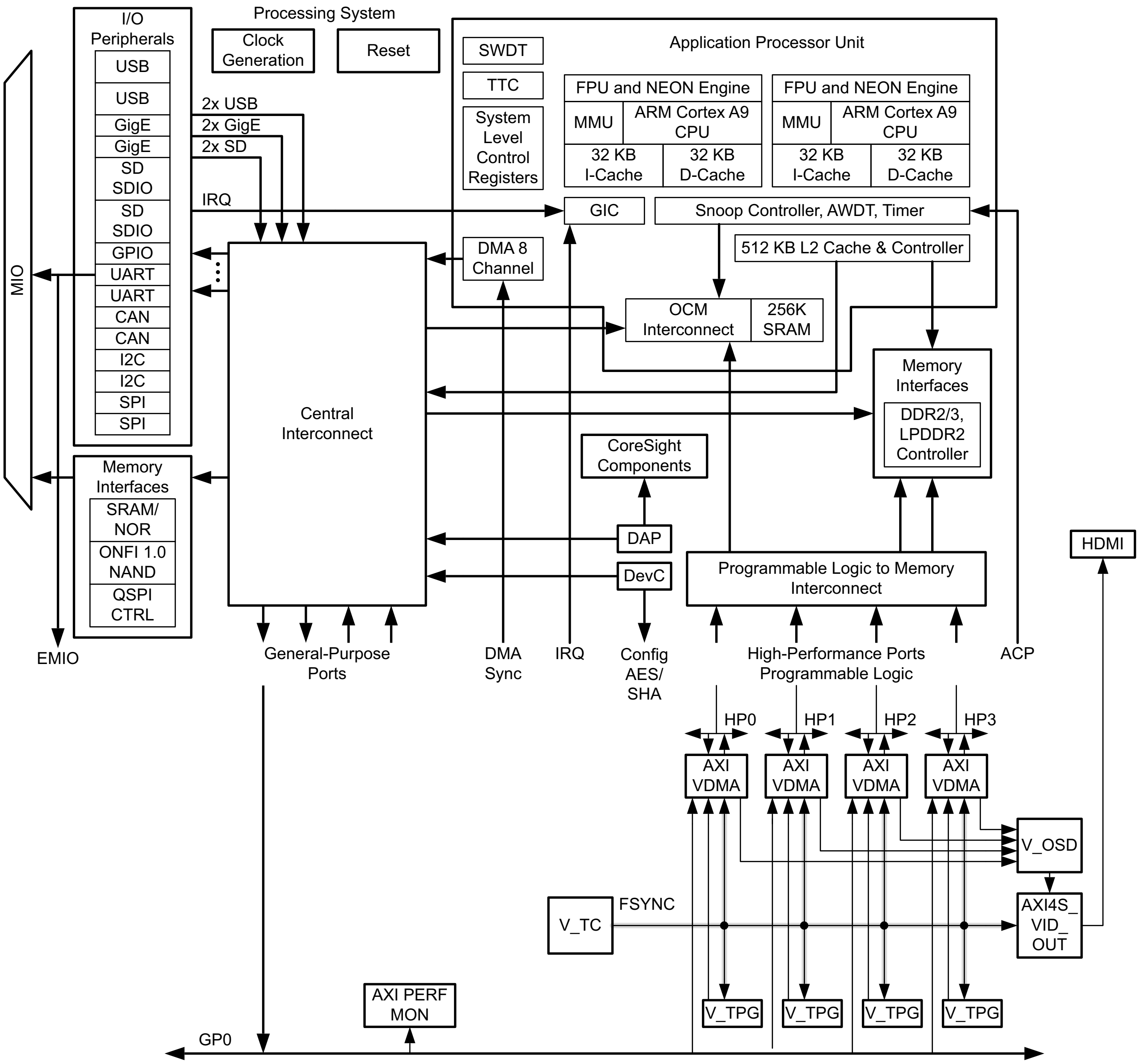


Figure 1: **Reference System Block Diagram**

Note: The block diagram in Figure 1 does not show the `V_RGB2YCRCB` (RGB to YCrCb Color-Space Converter) or `V_CRESAMPLE` (Chroma Resampler) IP blocks.

Table 1 shows the reference system address map.

Table 1: **Reference System Address Map**

Peripheral	Instance	Base Address	High Address
processing_system7	processing_system7_1 (M_AXI_GP0)	0x40000000	0x7FFFFFFF
processing_system7	processing_system7_1 (S_AXI_HP0)	0x00000000	0x3FFFFFFF
processing_system7	processing_system7_1 (S_AXI_HP1)	0x00000000	0x3FFFFFFF
processing_system7	processing_system7_1 (S_AXI_HP2)	0x00000000	0x3FFFFFFF
processing_system7	processing_system7_1 (S_AXI_HP3)	0x00000000	0x3FFFFFFF
axi_vdma	axi_vdma_1	0x40000000	0x4000FFFF
axi_vdma	axi_vdma_2	0x40010000	0x4001FFFF
axi_vdma	axi_vdma_3	0x40020000	0x4002FFFF
axi_vdma	axi_vdma_4	0x40030000	0x4003FFFF
axi_perf_mon	perf_axi_mon_0	0x41000000	0x4100FFFF

Hardware System Specifics

This section describes the high-level features of the reference design, including how to configure the main IP blocks and the Zynq-7000 AP SoC PS. Information about useful IP features, performance and area trade-offs, and other configuration information is 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 interface-based designs. For information about AXI system optimization and design trade-offs, see *Xilinx AXI Reference Guide* (UG761) [Ref 2]. This application note assumes a general knowledge of the Zynq-7000 AP SoC family, AXI Protocol, Vivado design suite, and the Vivado IP integrator. See *Vivado Design Suite Tutorial: Embedded Processing Hardware Design* (UG940) [Ref 1] for more information about the Vivado IP integrator feature.

Zynq-7000 AP SoC PS

The Zynq-7000 AP SoC PS is configured to use one Cortex-A9 processor, one IIC (MIO Interface), one UART (MIO interface) and the Zynq-7000 AP SoC memory controller (MIO Interface). These features are enabled by the first stage boot loader (FSBL) created by the SDK tool.

The Zynq-7000 AP SoC PS provides clock and reset signals to both the PS and PL. See Table 2 for the clock signal frequencies used in this design.

Table 2: **Reference System Clock Frequencies**

Element	Clock Frequency (MHz)
Processor	666
32-bit DDR3 memory controller	533
High-speed AXI interfaces (S_AXI_HP0, S_AXI_HP1, S_AXI_HP2)	148.5
Low-speed AXI interfaces (M_AXI_GP0)	50
64-bit Memory Interconnect	355
32-bit Master and Slave Interconnect	222

Refer to *Zynq-7000 All Programmable SoC Technical Reference Manual* (UG585) [Ref 4] for more information about clock frequencies and Interconnects in the Zynq-7000 AP SoC PS.

Video Related IP Cores

The reference design implements four video pipelines running at 1080P60 (1920 x 1080 pictures at 60 frames/sec). Each frame consists of three bytes per pixel to represent an upper-bound, high-quality video stream such as RGB with alpha channel information. Each video pipeline requires a bandwidth of 373.248 MB/s, which is approximately 3 Gb/s.

The video traffic is generated by Test Pattern Generator IP cores (V_TPG) and displayed by the Video On-Screen Display core (V_OSD). The total aggregate read/write bandwidth generated is equivalent to eight video streams requiring nearly 3 GB/s (24 Gb/s).

This application note demonstrates AXI system performance using eight 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 several IP modules. The canonical video system is illustrated in Figure 2 to show that most video systems consists 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 is transferred into and out of memory according to the requirements of internal processing stages. In this application note, the internal block memory traffic is created by a series of test pattern generators to simulate typical conditions.

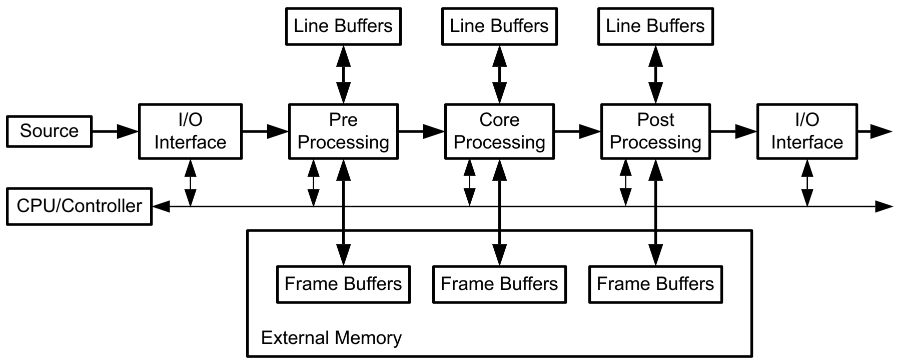


Figure 2: **Typical Video System**

AXI Interconnect (processing_system7_1_axi_periph) Instance

Because this portion of the design only connects to AXI4-Lite slave peripherals, the processing_system7_1_axi_periph instance is optimized for area by using the Minimize Area strategy to implement a shared bus topology. This strategy reduces system resources for the AXI interconnect and sets the AXI Master issuance and the AXI Slave acceptance values to 1. The Cortex-A9 processor writes and reads all AXI4-Lite slave registers in the design for control and status information. In addition, this portion of the design is clocked at 50 MHz which is a slower clock rate compared to the rest of the system.

The AXI interface slaves are for four instances of AXI Video DMA slave interfaces and the AXI_PERF_MONITOR slave interface.

AXI Interconnect (axi_mem_intercon_x) Instances

The axi_mem_intercon_x instances are used for high speed masters and slaves, which include high-throughput and high F_{MAX} optimizations. These instances provide the highest F_{MAX} and throughput for the design by having a 64-bit core data width and running at 148.5 MHz. Each instance connects one AXI Video DMA (AXI MM2S and AXI S2MM Master interfaces) to one HP interface.

These AXI interconnect instances are optimized using the Custom strategy. With this setting, crossbar connectivity mode is used to accommodate high throughput. The AXI Master issuance value is set to 2 and the AXI Slave acceptance value is set to 4 so as not to overwhelm the design. Also, register slicing is enabled on each of the AXI Masters and AXI Slaves, and 512-element data packet FIFOs are added on each AXI Master interface for the highest F_{MAX} and buffering for the design. Protocol converters and upsizers are required to convert 32-bit AXI4 protocol (32-bit AXI4 Masters) to 64-bit AXI3 protocol (64-bit AXI3 Slave).

AXI Video DMA Instances

The AXI Video DMA core is designed to provide video read and write transfer capabilities from the AXI4 memory-mapped domain to AXI4-Stream, and vice versa. The AXI Video DMA core provides high-speed data movement between system memory and AXI4-Stream-based target Video IP cores. AXI4 memory-mapped interfaces are used for high-speed data movement and buffer description fetches across the AXI interface. With this design, register direct mode is used for the base addresses for the frame buffers.

The design incorporates video-specific functionality such as genlock and Frame Sync for fully synchronized frame DMA operations as well as two-dimensional DMA transfers. Scatter/gather or register direct mode operations are also available for ease of control by the central processor. This design uses the register direct mode operation.

In this design, four instances of AXI Video DMA cores are used with the AXI4 MM2S interface, AXI4 S2MM interface, AXI4-Stream MM2S interface, and AXI4-Stream S2MM interface. Also, with each instance the initialization, status, and management registers are accessed through an AXI4-Lite Slave interface.

The 32-bit wide AXI MM2S and AXI S2MM interfaces from the AXI Video DMA core instances are connected to the S_AXI_HPx instances (high-performance memory access AXI3 slaves).The AXI4-Stream interfaces are clocked at 148.5 MHz at 24-bits. The AXI Video DMA core handles data conversion/padding for 24-bits to the 32-bit AXI MM interface efficiently.

AXI Video DMA core instances are set for a maximum burst of 32. The AXI3 protocol only supports a maximum burst of 16. However, because the connected HP interface on the Zynq-7000 device is 64-bits, this interface supports a transfer size of 64 bits x 16 data beats (32 bits x 32 data beats). Also, read and write side line buffers inside the AXI Video DMA core are set to accommodate about one fourth of a line which is 1KB deep ((1920 x 3 bytes) x 1/4 = 1440 bytes deep).

Zynq-7000 AP SoC Memory Controller

Overview

The Zynq-7000 AP SoC memory controller on the ZC702 board is a 32-bit DDR3 controller clocked at 533 MHz. The addressing method used is row, bank, column.

The memory controller has four direct connections which are from the AXI_HP to DDR interface (two connections), from the central interconnect, and from the L2 cache interface in the Zynq-7000 device. Also, the Quality of Service (QoS) Priority module is positioned between these four sources and the connections to the memory controller, which enables regulation of traffic patterns by limiting requests. The HP0/HP1 and HP2/HP3 slave interfaces share direct connections to the memory controller using an AXI_HP to DDR interface. With this design, HP0/HP1/HP2/HP3 slave interfaces are used. For more information about these interfaces, refer to *Zynq-7000 All Programmable SoC Technical Reference Manual* (UG585) [Ref 4].

Achieving over 60% Utilization

To achieve optimal utilization of the memory controller, transactions from the master interfaces need to occur in different banks and must be aligned to KB/MB boundaries. With video designs, frame buffers need to be accessed in different banks, and the overlapping of banks must be minimized while the video design is running.

This design demonstrates 1080P60 frames (1920 x 1080) with three bytes per pixel. Each horizontal line is about 8 KB (1920 x 3 = 5760), so the AXI Video DMA line stride is set to 8 KB boundaries. The start of every new line is aligned on an 8 KB boundary. The vertical lines of each frame (1080 lines per frame) are aligned to a 2 KB boundary for each frame. Therefore, each frame buffer in this design is aligned on 16 MB boundaries (8 KB x 2 KB).

The ZC702 board memory controller is configured to use this DDR address configuration:

DDR_ADDR[27:15]	Row
DDR_ADDR[14:12]	Bank
DDR_ADDR[11:0]	Column/Word

The AXI master address is reordered to ensure that each frame buffer is in its own bank. Because each frame buffer is aligned on 16 MB boundaries, it takes 24 bits to represent the address space [23:0] and three bits to represent the bank [26:24]. AXI address bits [26:24] are moved to [14:12] for the DDR address. The reordered AXI Master address looks like:

axi_addr[31:27] axi_addr[23:12] axi_addr[26:24] axi_addr[11:0]

An additional custom IP core, ZYNQ_ADDR_SWITCH, was created for this application note with Vivado IP packager to reorder the AXI master address to the AXI interface. With this reordering, the processor or software must handle the address reordering.

Caution! The ZYNQ_ADDR_SWITCH core is only an example for this application note.

Frame Buffers

With each AXI Video DMA core in normal genlock mode, two frame buffers are active at one time and are usually delayed by one frame buffer (i.e read/write operations are not occurring on the same frame buffer). To create a deterministic traffic pattern in this design, all AXI Video DMA core interfaces share the same FSYNC signal. This design uses a total of 12 frame buffers, but the memory controller only has eight banks. Because only eight frame buffers are active at one time in this design, the frame buffer locations must be carefully chosen to limit or eliminate bank overlaps.

Table 3 shows the frame buffer base addresses. The active frame buffers are arranged in a continuous pattern following this sequence:

1. FRAME0/FRAME2
2. FRAME1/FRAME0
3. FRAME2/FRAME1

Because all AXI Video DMA core interfaces share the same FSYNC signal, with the frame buffer layout of Table 3, note that one or two frame buffer overlaps can occur at the same time (see Table 4).

Table 3: **Frame Buffer Base Addresses**

Frame Buffer	Base Address	Bank Number
	AXI VDMA1	
FRAME0	0x08000000	BANK0
FRAME1	0x0A000000	BANK2
FRAME2	0x09000000	BANK1
	AXI VDMA2	
FRAME0	0x0B000000	BANK3
FRAME1	0x0D000000	BANK5

Table 3: **Frame Buffer Base Addresses *(Cont'd)***

Frame Buffer	Base Address	Bank Number
FRAME2	0x0C000000	BANK4
AXI VDMA3		
FRAME0	0x0E000000	BANK6
FRAME1	0x18000000	BANK0
FRAME2	0x0F000000	BANK7
AXI VDMA4		
FRAME0	0x19000000	BANK1
FRAME1	0x1B000000	BANK3
FRAME2	0x1A000000	BANK2

Table 4: **Frame Buffer Overlap**

Interface	Buffer Pattern		
	FRAME0/FRAME2	**FRAME1/FRAME0**	**FRAME2/FRAME1**
AXI VDMA1	BANK0/BANK1	BANK2/BANK0	BANK1/BANK2
AXI VDMA2	BANK3/BANK4	BANK5/BANK3	BANK4/BANK5
AXI VDMA3	BANK6/BANK7	BANK0/BANK6	BANK7/BANK0
AXI VDMA4	BANK1/BANK2	BANK3/BANK1	BANK2/BANK3
Overlaps	1	2	1

QoS

The QoS control mechanism allows the control of requests from interfaces to the memory controller by setting the maximum number of outstanding transactions, peak rates, average rate, and burst level. This is beneficial especially when video pipelines are used and cannot be throttled. This design does not use the QoS feature because the processor traffic does not throttle the video pipelines.

Video Timing Controller (V_TC)

The Video Timing Controller core 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. These functions are not enabled in this design because the Test Pattern Generator core is used for video input. 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 Video Timing Controller core is configured for 1080P frames through IP settings while configuring the core. This design does not contain an AXI4-Lite interface to access slave control registers from a processor.

The FSYNC signal from this core is connected to the AXI Video DMA S2MM FSYNC input signal and to the reset signal of the Test Pattern Generator cores to synchronize the core write operations with the video timing. The video timing interface is connected to the AXI4-Stream to video out which accepts the read data from the AXI Video DMA and Video On-Screen Display cores and converts the data for use with the HDMI configuration on the ZC702 board (using YUV 4:2:2 format).

Video Test Pattern Generator (V_TPG)

The Test Pattern Generator core is configured for 1080P frames, RGB (3 bytes per pixel), and is used to generate color patterns, zone plates, horizontal sweep, or checkerboard patterns through IP settings while configuring the core. This design does not contain an AXI4-Lite interface to access slave control registers from a processor.

In this design, the video traffic to DRAM is generated by the Test Pattern Generator core. The Test Pattern Generator core is capable of generating several video test patterns that are commonly used in the video industry for verification and testing. These patterns are intended for testing purposes only and are not calibrated to broadcast industry standards. Here, the TPG is used as a replacement for other video IP because the emphasis is on the amount of traffic generated to demonstrate the performance of the system. No matter which test pattern is chosen at IP configuration time, the amount of data generated is the same, namely, 1080P60 HD video (1920 x 1080 frames at 60 frames/second). For example, a RGB 1080P60 pattern generates 373.3 MB/s, which is nearly a 3 Gb/s data stream.

For the purposes of this application note, the Test Pattern Generator core always generates a test pattern which could be one of color bars, horizontal frequency sweep, vertical frequency sweep, or zoneplate patterns.

Video Onscreen Display (V_OSD)

The Video On-Screen Display core provides flexible video processing for alpha blending and compositing of up to eight independent layers. The Video On-Screen Display core also provides simple text and graphics generation and is capable of handling images up to 4K by 4K pixels, in 8 possible image formats, and in 8, 10, or 12 bits per color component. In this application note, the Video On-Screen Display core is used to blend the four video streams as separate layers for display. As the video streams are generated by the Test Pattern Generator cores, the display shows the blended layers on top of each other. Figure 3 shows a top-level block diagram of the Video On-Screen Display core.

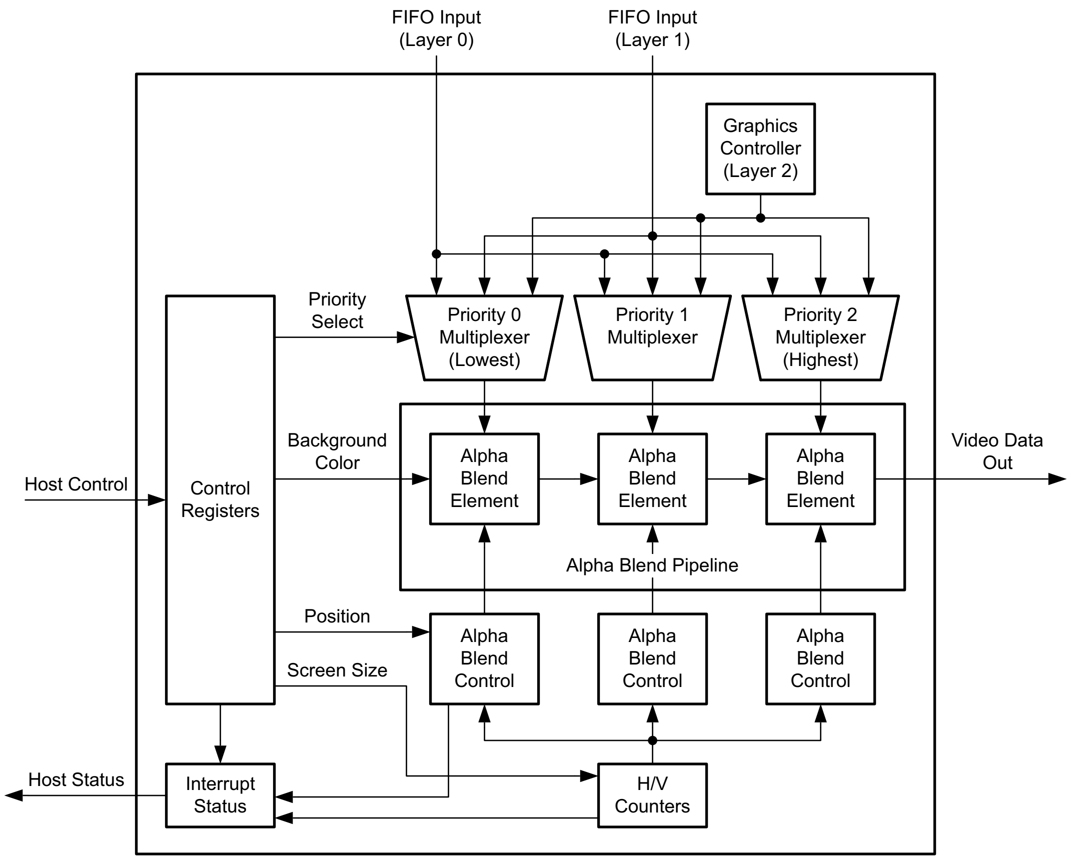


Figure 3: **Example 3-Layer OSD Block Diagram**

The Video On-Screen Display core is configured for 1080P RGB frames and four video layers using IP settings when configuring the core. This design does not contain an AXI4-Lite interface to access slave control registers from a processor.

AXI Performance Monitor (AXI_PERF_MON)

The AXI Performance Monitor core allows up to eight AXI interfaces to measure bus latency with different metrics. Four counters are used in this design, which provide read and write byte counts using the metric counter registers.

This design incorporates four bus monitor interfaces. The AXI_S_HP0, AXI_S_HP1, AXI_S_HP2, and AXI_S_HP3 slave interfaces are individual monitored. The Sample Interval register (number of clocks) is set to count for one second. During this time, the AXI Performance Monitor core monitors read and write bytes on the AXI interface during the one second interval and writes these values into the Metric Counter registers after the Sample Interval counter has overflowed. A software routine polls the Sample Interval Counter interrupt bit to determine when the one second interval is complete.

For more information about the AXI Performance Monitor core, refer to *LogiCORE IP AXI Performance Monitor Product Guide* (PG037) [Ref 10].

Converting the Video Stream for the ZC702 Board (YUV 4:2:2)

This design uses RGB pixels in the video stream (AXI4-Stream). Because the HDMI device connection on the ZC702 board requires a 16-bit YUV 4:2:2 signal format, conversion of the 36-bit RGB video stream is necessary.

The RGB to YCrCb Color-Space Converter core converts the RGB video stream to YUV 4:4:4. The Chroma Resampler core subsequently converts the YUV 4:4:4 stream to YUV 4:2:2. Finally, the output of the Chroma Resampler core is connected to the AXI4S_VIDEO_OUT core which drives the HDMI interface on the ZC702 board to properly display the video on a monitor.

Software Applications

AXI Video DMA Display Application

This software application starts up the video pipelines and can alpha blend all layers on the LCD screen while allowing real time bandwidth examination. The software application is run from the OCM in the Zynq-7000 AP SoC PS.

Many of the Video IP cores are automatically configured at start-up to support the four Video On-Screen Display core video streams at 1080P60 based on parameter settings for each core. The cores do not contain AXI4-Lite slave interfaces for controlling slave registers. However, the AXI Video DMA core in this design contains an AXI4-Lite slave interface with which the software configures the core for 1080P where each horizontal line is set for 5760 bytes (1920 x 3 bytes per pixel) and each vertical line is set for 1080 pixels by the processor. There are twelve frame buffers in this design (3 frame buffers x 4 video pipelines).

Also, the AXI PERF MON core contains an AXI4-Lite slave interface allowing the processor to configure the core to produce throughput numbers for the video pipeline (the core monitors the HP0/HP1/HP2/HP3 read/write interfaces on the Zynq-7000 AP SoC device).

The software application performs these steps:

1. Configures the HDMI port for 1080P60 output on the ZC702 board using the IIC interface.
2. Starts the AXI Video DMA instances. This action consists of the processor writing the buffer descriptors to registers in the AXI Video DMA cores. The program then starts the read channel, and then the write channel which initiates the transfers for the AXI Video DMA instances.

After the initial setup sequence, two numbered options are available.

- Option 1 provides a 32-bit memory test to the DDR3 memory.
- Option 2 sets up the performance monitoring IP instance to measure read byte counts and write byte counts on the four monitored AXI interfaces (monitors are located on the AXI3 slaves on the Zynq-7000 AP SoC device). A software function is used to present the TX and RX throughput number for each video pipeline as well as the system throughput number on a host PC terminal display.

Executing the Reference Design in Hardware

Use these steps to prepare the ZC702 board:

1. Connect a USB cable from the host PC to the USB JTAG port of the ZC702 board. Ensure that the appropriate device drivers are installed.
2. Connect a second USB cable from the host PC to the USB UART port of the ZC702 board. Ensure that the USB-UART drivers have been installed.
3. Connect the ZC702 HDMI connector to a video monitor capable of displaying a 1920 x 1080p, 60 Hz video signal.
4. Connect the power supply cable to the ZC702 board.
5. Apply power to the ZC702 board.
6. Start a terminal program such as HyperTerminal on the host PC with these settings:
   - Baud Rate: **115200**
   - Data Bits: **8**
   - Parity: **None**
   - Stop Bits: **1**
   - 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 `<unzip_dir>/zc702_video_4x_ipi/ready_for_download` directory, follow these steps:

1. In the Vivado IDE Tcl console or a terminal window, change directories to the `ready_for_download` directory:

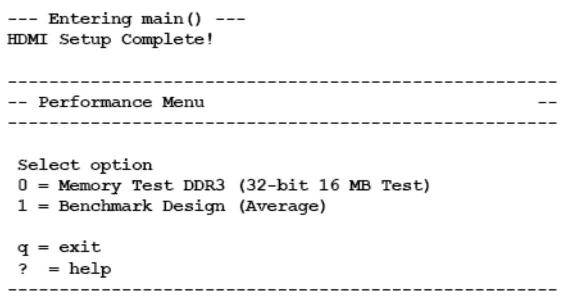
```
% cd <unzip dir>/zc702_video_4x_ipi/ready_for_download
```

2. Run the design with the following XMD command using the Tcl script that downloads the bitstream to the board, connects to the processor, downloads and runs the FSBL, and downloads and runs the software application:

```
% xmd -tcl xmd_top.tcl
```

Results from Running Hardware and Software

An alpha-blended display of all layers should now be visible on the LCD monitor connected to the ZC702 board, and the host PC terminal program should display the output shown in Figure 4.



```
--- Entering main() ---
HDMI Setup Complete!

-----------------------------------------------------
-- Performance Menu                                --
-----------------------------------------------------

 Select option
 0 = Memory Test DDR3 (32-bit 16 MB Test)
 1 = Benchmark Design (Average)

 q = exit
 ?  = help
-----------------------------------------------------
```

Figure 4: **Host PC Terminal Output**

The performance menu on the host PC terminal display has two options:

- **0**: 32-bit, 16 MB memory test executed in the DDR3 memory.
- **1**: Terminal shows real-time system performance parameters (one second of transfers)

Performance

The DDR3 PHY is set for 32 bits with a memory clock frequency of 533 MHz (1066 MHz data rate). The theoretical throughput on DDR3 is 4.2 GB/s which is the total bandwidth available in the design.

Selecting option 1 of the software application produces this output:

```
AXI_VDMA1 Tx = 373.248000
AXI_VDMA1 Rx = 373.248000
AXI_VDMA2 Tx = 373.248000
AXI_VDMA2 Rx = 373.248000
AXI_VDMA3 Tx = 373.248000
AXI_VDMA3 Rx = 373.248000
AXI_VDMA4 Tx = 373.248000
AXI_VDMA4 Rx = 373.248000

System Total Bandwidth = 2985.984000 MB/s
DDR3 Theoretical Bandwidth = 4266.666666 MB/s
Percent of DDR3 Theoretical Bandwidth = 69.984000%
```

Note: the values can vary slightly from those shown here.

Total bandwidth is approximately 2986 MB/s out of 4267 MB/s, which is approximately 70% of the total bandwidth of the main memory.

Building Hardware

Use these steps to rebuild the hardware design:

Note: The generated bitstream is located at `<unzip dir>/zc702_video_4x_ipi/HW/project_1.runs/impl_1/design_1_wrapper.bit`.

1. Open the `zc702_video_4x_ipi/HW/project_1.xpr` file with Vivado design suite.
2. In the Sources tab, expand `design_1_wrapper`.
3. Right-click `design_1_i` and select **Generate Output Products…**, then select **Generate**.
4. In the Flow Navigator pane, click **Project Manager**.
5. In the Design Runs tab, right-click `impl_1` and select **Launch Run**. Click **OK** for the next two dialogs to start synthesis followed by implementation.

   Note: This step might take more than one hour.
6. When the design implementation is complete, click **Generate Bitstream** in the Implementation Completed tab. Click **OK**.
7. In the Bitstream Generation Completed tab, click **Open Implemented Design** and click **OK**. This action allows the bitstream to be included when exporting the hardware design to SDK.
8. To export the hardware design to SDK, in the Sources/Hierarchy tab, right-click `design_1_i` and select **Export Hardware for SDK…**
9. Select **Export Hardware and Include bitstream**. Click **OK**.

Compiling Software and Running Design Through the SDK Tool

Compiling Software with the SDK Tool

1. Start the SDK tool. In Linux, type **`xsdk`**.
2. In the Workspace Launcher, select this Workspace:
   `<unzip dir>/zc702_video_4x_ipi/SW/SDK_Workspace`
3. Click **OK**.
4. The Board Support Package (BSP), hardware platform, and software applications must be imported. Click **File > Import > General > Existing Projects into Workspace**.
5. Click **Next** and browse to the `<unzip dir>/zc702_video_4x_ipi/SW` directory. Click **OK**.

6. Make sure that all check boxes are selected, including **FSBL**, **hw_platform_0**, **standalone_bsp_0**, and **zynq_axi_vdma_display**. Click **Finish**.

   Note: The BSP and software applications are compiled. This should take 2–5 minutes. If the zynq_axi_vdma_display software application contains build errors, right-click zynq_axi_vdma_display in the Project Explorer tab and select **Change Referenced BSP**, then select **standalone_bsp_0**.
7. Existing software applications can now be modified, and new software applications can be created with the SDK tool.

Running the Hardware and Software Through the SDK Tool

1. Click **Xilinx Tools->Program FPGA**.
2. Click **Program**.
3. In the Project Explorer Window, right-click **zynq_axi_vdma_display > Run As > Launch on Hardware**. By default, the SDK tool runs the FSBL using a Tcl script.

Reference Design

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

https://secure.xilinx.com/webreg/clickthrough.do?cid=356501, registration required.

Table 5 shows the reference design checklist.

Table 5: **Reference Design Checklist**

Parameter	Description
General	
Developer name	James Lucero
Target devices (stepping level, ES, production, speed grades)	Zynq-7000 AP SoC XC7Z020-1CLG484C
Source code provided	Yes
Source code format	VHDL, Verilog
Design uses code/IP from existing Xilinx application note/reference designs, CORE Generator™ software, or third party	Yes
Simulation	
Functional simulation performed	N/A
Timing simulation performed	N/A
Test bench used for functional and timing simulations	N/A
Test bench format	N/A
Simulator software/version used	N/A
SPICE/IBIS simulations	N/A
Implementation	
Synthesis software tools/version used	Vivado Design Suite 2013.4
Implementation software tools/versions used	Vivado Design Suite 2013.4
Static timing analysis performed	Yes
Hardware Verification	

Table 5: **Reference Design Checklist *(Cont'd)***

Parameter	Description
Hardware verified	Yes
Hardware platform used for verification	Zynq-7000 AP SoC ZC702 evaluation board

References

1. *Vivado Design Suite Tutorial: Embedded Processor Hardware Design* (UG940)
2. *AXI Reference Guide* (UG761)
3. AMBA AXI4 Stream Protocol Specification
4. *Zynq-7000 All Programmable SoC Technical Reference Manual* (UG585)
5. *LogiCORE IP AXI Interconnect Product Guide* (PG059)
6. *LogiCORE IP AXI Video Direct Memory Access Product Guide* (PG020)
7. *LogiCORE IP Video On-Screen Display Product Guide* (PG010)
8. *LogiCORE IP Test Pattern Generator Product Guide* (PG103)
9. *LogiCORE IP Video Timing Controller Product Guide* (PG016)
10. *LogiCORE IP AXI Performance Monitor Product Guide* (PG037)

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
03/28/2014	1.0	Initial Xilinx release.

Notice of Disclaimer