XAPP1291 (v1.0.1) July 22, 2016

Video Processing Subsystem Reference Design

Author: Bob Slous and Rohit Consul

Summary

This application note describes the Video Processing Subsystem core reference design, which showcases the capability and ease of use of these Xilinx® LogiCORE™ intellectual property (IP) cores:

- Video Processing Subsystem.
- Video Mixer.
- HDMI™ 1.4/2.0 Transmitter Subsystem.
- HDMI 1.4/2.0 Receiver Subsystem.
- Video PHY Controller.

The reference design uses the HDMI RX/TX connectivity IP cores to transfer video in and out of the FPGA device. The Video Processing Subsystem and Video Mixer IP cores form the processing chain that transforms the incoming video. The design targets the Xilinx Kintex®-7 FPGA KC705 evaluation board [Ref 1], which uses the Kintex-7 XC7K325T-2FFG900C FPGA and the inrevium TB-FMCH-HDMI4K [Ref 3] daughter card.

Download the Reference Design Files for this application note from the Xilinx website.

Reference Design

The reference design was created with the Vivado® Design Suite, System Edition 2016.2. The design also includes software built with the Xilinx Software Development Kit (SDK) 2016.2. The software runs on the MicroBlaze™ processor subsystem and implements control and status functions. The project files for the Vivado Design Suite and SDK are provided with this application note to allow for the examination and rebuilding of the design, or to use as a template for starting a new design.

Hardware

The reference design is built around the Video Processing Subsystem (V_PROC_SS), Video Mixer (V_MIXER), HDMI 1.4/2.0 Transmitter Subsystem (HDMI_TX_SS), HDMI 1.4/2.0 Receiver Subsystem (HDMI_RX_SS), and Video PHY (VPHY) Controller cores, and uses other Xilinx IP cores to form the complete system. The input and output of the system are HDMI video streams through an HDMI 2.0 daughter card that connects to the FMC HPC connector of the development board. See Figure 1.

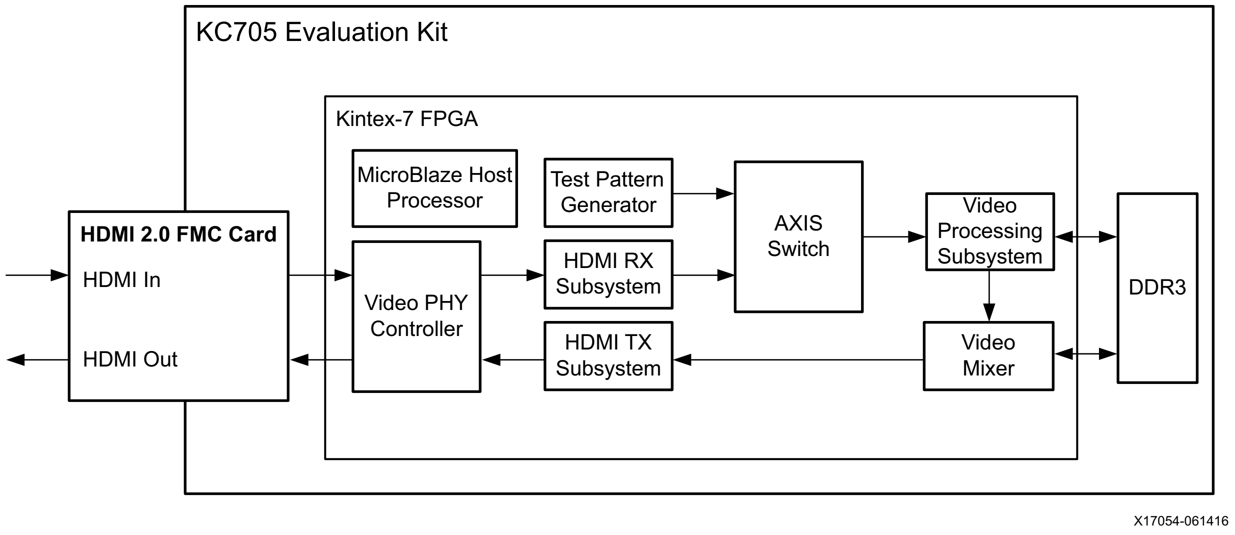


Figure 1: **KC705 Evaluation Kit Input and Output**

The Video Processing Subsystem is a hierarchical IP core that bundles video processing IP subcores in hardware and software and outputs them as a single IP core. The result is a video processing pipe as a black box. The Video Processing Subsystem core provides an out-of-the box ready-to-use video processing core, without requiring an understanding of the underlying complexities.

Note: All video processing IP subcores are developed with the Vivado High-Level Synthesis (HLS) tool.

For more information on the Video Processing Subsystem IP core, see the Video Processing Subsystem product page [Ref 2] and the *Video Processing Subsystem v1.0 Product Guide* (PG231) [Ref 4].

The reference design performs automatic video format conversion from the format received over the HDMI RX IP core to the format set up for the HDMI TX IP core. The HDMI TX IP core can be programmed independently from the HDMI RX IP core. For example, if the HDMI TX IP core transmits 3840x2160p at 30 Hz in the YUV 4:4:4 color space, and the HDMI RX IP core receives 1920x1080i at 60 Hz in the RGB color space, the Video Processing Subsystem IP core:

- Deinterlaces from 1080i to 1080p.
- Scales from 1080 to 2160p.
- Converts color space from RGB to YUV 4:4:4.
- Converts frame rate from 60 Hz to 30 Hz.

Features

The Video Processing Subsystem IP core has design time configurability for performance, quality, and functionality. It is device independent, and includes these features:

- One, two, or four pixel-wide video interface.
- Video resolution support up to UHD at 60 f/s.
- Run-time color space support for RGB, YUV 4:4:4, YUV 4:2:2, YUV 4:2:0.
- 8, 10, 12, and 16 bits per component support.
- Deinterlacing.
- Scaling.
- Color space conversion and correction.
- Chroma resampling between YUV4:4:4, YUV 4:2:2, YUV 4:2:0.
- Frame rate conversion through drop/repeat frames.

The Video Mixer IP core includes these features:

- Supports alpha blending of eight video/graphics layers.
- Optional logo (in block RAM) layer with color transparency support.
- Layers can either be memory mapped AXI4 interface or AXI4-Stream.
- Provides programmable background color.
- Provides programmable layer position and size.
- Provides scaling of layers by 1x, 2x, or 4x.
- Optional built-in color space conversion.
- Supports RGB, YUV 444, and YUV 422.
- Supports 8, 10, 12, and 16 bits per color component input and output on stream interface, 8-bit per color component on memory interface.
- Supports spatial resolutions from 64 × 64 up to 4,096 × 2,160.
- Supports 4K 60 f/s in all supported device families.

Reference Design Features

The reference design supports these features:

- Target device Kintex-7 FPGA KC705 evaluation board.
- Two pixel-wide interface.
- Color depth fixed to 8 bits.
- Full-fledged and scaler-only video processing subsystem configurations.
- Video resolutions up to UHD at @ 60 Hz (both at input and output interface), see Table 1.
- Video encoding RGB 4:4:4, YUV 4:4:4, and YUV 4:2:2.
- Interlaced input support: 1080i only.
- Supported use cases:
    - Color space (RGB/YUV) and format (YUV 4:4:4/4:2:2) conversion.
    - Scale up and down up to 4k2k at 60 Hz.
    - Zoom mode: crops a user-defined window within the input stream and scales it to panel resolution.
    - Picture-in-picture (PIP) mode: the input stream is scaled down to a user-defined window size and displayed at the user-defined coordinates on the panel.
    - Ability to paint the PIP background to a defined color.
    - Interlaced to progressive conversion.
    - Frame rate conversion:
        - Drop frames if input rate > output rate.
        - Repeat frames if output rate < input rate.

The reference design can receive and transmit many different resolutions as listed in Table 1.

Table 1: **Supported Video Resolutions**

Resolution	TX	RX
Interlaced		
1080i50		√
1080i60		√
Progressive		
480p60	√	√
576p50	√	√
720p50	√	√
720p60	√	√
1080p24	√	√
1080p25	√	√

Table 1: **Supported Video Resolutions *(Cont'd)***

Resolution	TX	RX
1080p30	√	√
1080p50	√	√
1080p60	√	√
2160p24	√	√
2160p25	√	√
2160p30	√	√
2160p60	√	√
PC resolutions		
vgap60	√	√
svgap60	√	√
xgap60	√	√
sxgap60	√	√
wxgap60	√	√
wxga+p60	√	√
uxgap60	√	√
wuxgap60	√	√
wsxgap60	√	√

Video Path

A video Test Pattern Generator (TPG) IP core and HDMI RX IP core feed into a 2:1 programmable video switch (or multiplexer) that feeds into the Video Processing Subsystem IP core. This allows the software to dynamically switch between an active HDMI input or a test pattern input. The output of the Video Processing Subsystem IP core is connected to the Video Mixer IP core. In addition to this streaming input, the Video Mixer core includes memory mapped interfaces that enable it to mix three memory layers, a streaming layer, and a logo layer. The output of the Video Mixer IP core is connected to the HDMI TX IP core. If the HDMI RX IP core is the active input but no signal is detected, the software automatically sets the Video Mixer core to output a blue frame. The video path is shown in Figure 1.

Memory Subsystem

The memory subsystem consists of an AXI-MM interconnect 6:1 crossbar that feeds into the Xilinx Memory Interface Generator (MIG). The six ports feeding into the crossbar are the data and instruction cache ports from the MicroBlaze processor, the memory port of the Video Processing Subsystem IP core, and three layers from the Video Mixer IP. The MIG bus is configured to be 512 bits wide. The net maximum bandwidth requirements of the video data are 4.75 GB/s:

- 4 GB/s for 3840x2160x60 Hz write to memory and read from memory at 32 bits/pixel.
- 0.75 GB/s for 1920x1080ix60 Hz write to memory and 2x1920x1080ix60 Hz read from memory at 32-bits/pixel.

Processor Subsystem

A MicroBlaze processor is used to control the IP cores. The reference design uses the default configuration that is not optimized for performance because CPU load is not critical. The MicroBlaze processor has 8 KB of data and instruction caches. The MicroBlaze processor is run at a 100 MHz clock speed.

Clocking

The Video Processing Subsystem IP core uses three clocks that are derived through a clock generator from the 200 MHz system clock on the Kintex-7 FPGA KC705 board. The memory subsystem is running at this system clock at 200 MHz. This system clock feeds into the Clock Wizard IP, which derives a 300 MHz and a 100 MHz clock at the output. The 300 MHz clock drives the AXI4-Stream video interfaces. With the video processing IP subcores configured as two pixel per clock engines, a throughput of 600 megapixels is obtained. The 100 MHz clock drives the AXI4-Lite control interface and also drives the MicroBlaze processor.

The Video PHY core requires one GT reference clock for the HDMI TX core. The clock is generated by an external programmable clock generator. In the reference design, the SI5324 is used. This device is programmable through an I2C interface. The Video PHY core has two transceiver reference clocks for the HDMI RX core. The RX clock from the HDMI cable is connected to the first transceiver reference clock input. The second transceiver reference clock comes from an external free-running 125 MHz clock. This clock is used by the non-integer data recovery unit (NI-DRU) to recover low HDMI line rates that are below the minimum rate supported by the receive phase-locked loop (PLL).

Additional Cores

In addition to the V_PROC_SS, V_MIXER, VPHY, HDMI_TX_SS, and HDMI_RX_SS cores, the reference design includes these cores (see Figure 2):

- MicroBlaze.
- MicroBlaze Debug Module.
- AXI Interconnect.
- Local Memory Bus.
- LMB Block RAM Controller.
- Block Memory Generator.
- Clocking Wizard.
- Processor System Reset.
- AXI UARTLite.
- AXI Interrupt Controller.
- AXI IIC.
- AXI GPIO.
- Test Pattern Generator.
- Concat.
- AXI4-Stream Register Slice.
- Utility Buffer.
- Utility Vector Logic.
- AXI Stream Switch.
- AXI Timer.
- MIG.

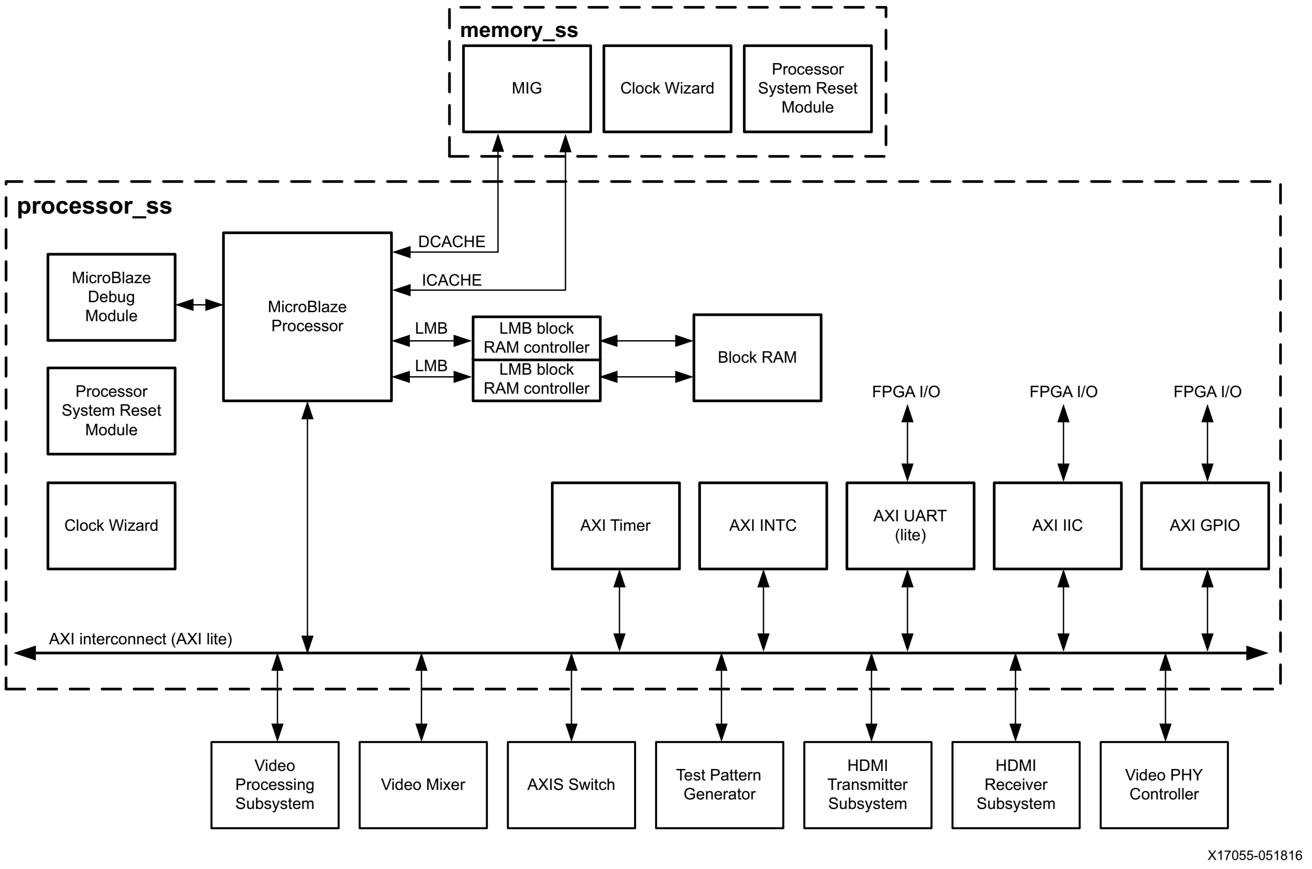


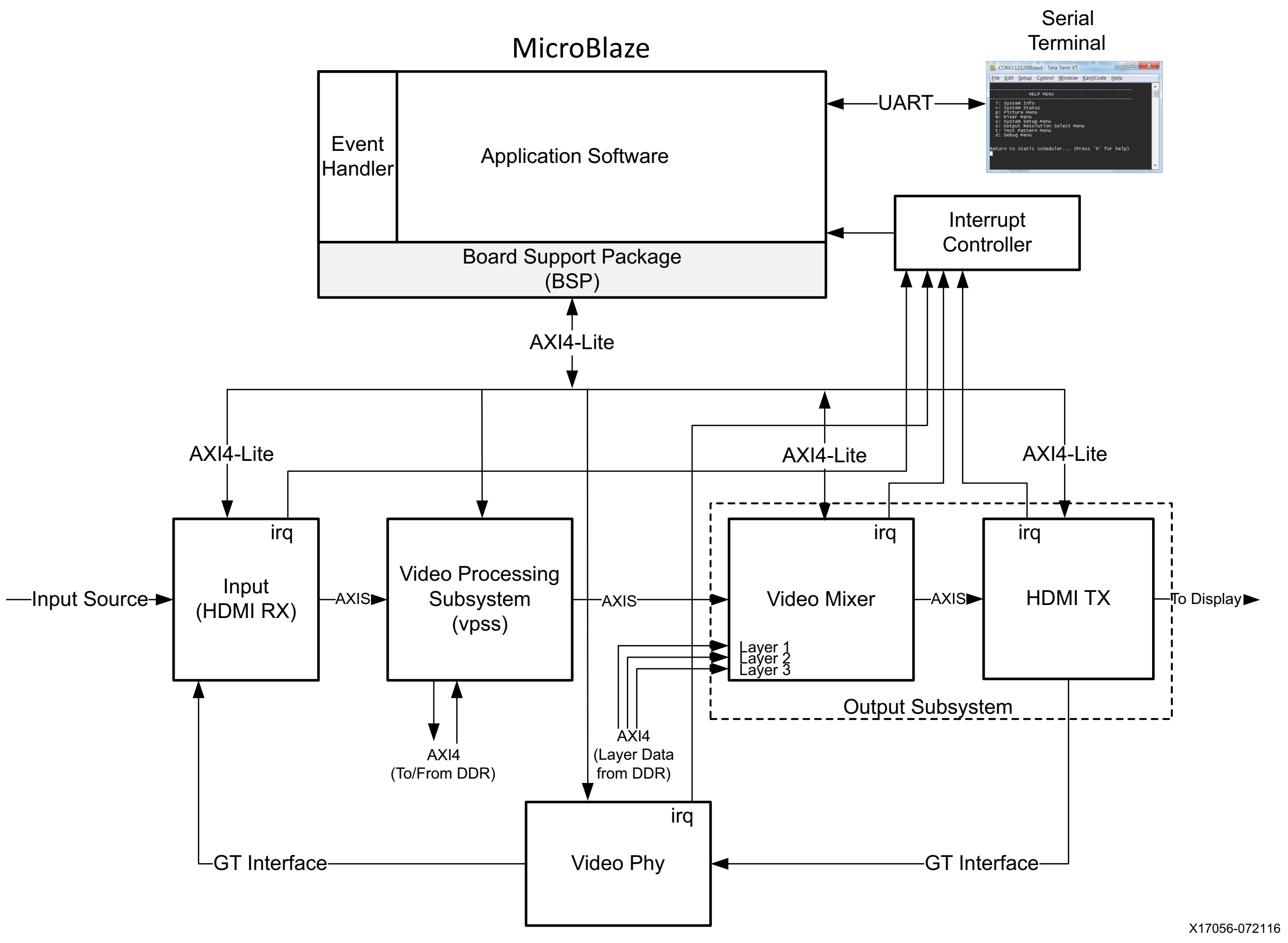
Figure 2: **Reference Design**

Table 2: **Processor Subsystem Address Map**

Peripheral	Instance	Base Address	High Address
lmb_bram_if_cntlr	ilmb_bram_if_cntlr	0x00000000	0x00001FFF
lmb_bram_if_cntlr	dlmb_bram_if_cntlr	0x00000000	0x00001FFF
axi_gpio	mixer_reset	0x40000000	0x4000FFFF
axi_gpio	axi_gpio_0	0x40010000	0x4001FFFF
axi_uartlite	axi_uartlite_0	0x40600000	0x4060FFFF
axi_iic	axi_iic_0	0x40800000	0x4080FFFF
axi_intc	axi_intc	0x41200000	0x4120FFFF
axi_timer	axi_timer_0	0x41C00000	0x41C0FFFF
v_hdmi_rx_ss	v_hdmi_rx_ss_0	0x44A00000	0x44A0FFFF
axis_switch	axis_switch_0	0x44A10000	0x44A1FFFF
v_hdmi_tx_ss	v_hdmi_tx_ss_0	0x44A20000	0x44A3FFFF
v_mix	v_mix_0	0x44A40000	0x44A7FFFF
v_tpg	v_tpg_0	0x44A80000	0x44A8FFFF
vid_phy_controller	vid_phy_controller_0	0x44A90000	0x44A9FFFF
v_proc_ss	v_proc_ss_0	0x44B00000	0x44BFFFFF
mig_7series	mig_7series	0x80000000	0xBFFFFFFF

Software Application

The reference design incorporates the HDMI RX/TX connectivity IP cores to transmit native video in and out of the FPGA. The Video Processing Subsystem and Mixer IP cores form a processing chain that performs transformations on the incoming video. The MicroBlaze processor is the host CPU used in the design along with support peripherals, such as UART, I2C, and interrupt controller. The software view of the pipeline is shown in Figure 3.



```
------------------------------------------------------
               HELP MENU
------------------------------------------------------
  i: System Info
  v: System Status
  p: Picture Menu
  m: Mixer Menu
  s: System Setup Menu
  o: Output Resolution Select Menu
  t: Test Pattern Menu
  d: Debug Menu

Return to static scheduler... (Press 'h' for help)
```

Figure 3: **Software Pipeline Overview**

The application software architecture is modular and provides the required separation between input, output, and the processing pipeline. The application software can be scaled to include additional input/output connectivity IP cores, such as DP or SDI.

The application software is comprised of these sub-blocks:

- Main

  This block is the starting point for the application. It connects all of the block drivers according to the defined hardware pipeline and, subsequently, initializes and starts the pipeline.

  After starting the processing chain, control transfers to the static scheduler (while(1) loop) that continuously monitors the system for user interaction, either from a serial terminal or from an external stimuli (interrupts triggered by input/output frequency change).

  Source Files: `main.c`, `platform.c/h`, `platform_config.h`, `evenhandler.c/h`, `system.c/h`

- Input Subsystem

  This block manages the input domain of the design and is responsible for initializing and bringing up the connectivity IP cores (for example, HDMI RX IP), including any board peripheral required by the input domain. Application-level callback functions required by the included IP core drivers are also implemented with this block.

  This block exposes the system input interface structure so the user can define the connectivity IP core to be used in the design. Also, the user can define the associated APIs for the system to perform basic operations, such as initialize, start, or stop the input domain; set active inputs (design contains TPG as alternate input source); and query the IP core driver for detected streams.

  Source Files: `src/inpss/ -> xinss.c/h`

- Output Subsystem

  This block manages the output domain of the design. It initializes and brings up the backend IP cores (e.g., HDMI TX), including any board peripheral (e.g., DP159, SI5324) required by the output domain. Application-level callback functions required by the included IP core drivers are also implemented in this block.

  This block exposes the system output interface so the user can define the connectivity IP core to be used in the design. Also, the user can define the associated APIs for the system to perform basic operations, such as initialize, start, and stop the output domain; set active outputs (if >1 present in the design); and set system output resolution. This block also includes the blender IP core (video mixer).

  Source Files used: `src/outputs -> xoutss.c/h`, `dp159.c/h`, `si5324.c/h`, `vidpatgen.c/h`, `<>_img.c`

- Phy Subsystem

  FPGA transceivers are tightly coupled with the connectivity MAC layer (e.g., HDMI RX/TX). This block manages the physical layer of the design. It initializes and brings up the transceivers. Application-level callback functions required by the Phy drivers are also implemented in this block.

  Source Files used: `src/phy` -> `physs.c/h`

- Resources

  This block manages the peripheral IP cores included in the design. It initializes and brings up the UART, Interrupt Controller, Test Pattern Generator, input/output muxes (if present), and the system timer. APIs are provided to control and configure the individual components when required.

  Source Files used: `src/res` -> `parser.c/h`, `periph.c/h`, `sleep.c/h`

IP Drivers

The IP core drivers included in the board support package (BSP) for each of the video IP cores, abstract the internal complexity of the core. The drivers provide an out-of-the box solution for the core functionality with an easy-to-use functional API interface.

The drivers for an IP core with an interrupt source provide a mechanism for the user application to be notified of the event. The driver API allows for registering a function to be called when the particular interrupt occurs and before the control is transferred back to the application context.

Note: These callback functions are implemented at the user application level in the design and can serve as a reference for integrating the required IP cores in user-specific designs. For details on the specific IP driver integration requirements and suggestions, see the applicable IP core product guide.

System Control Flow

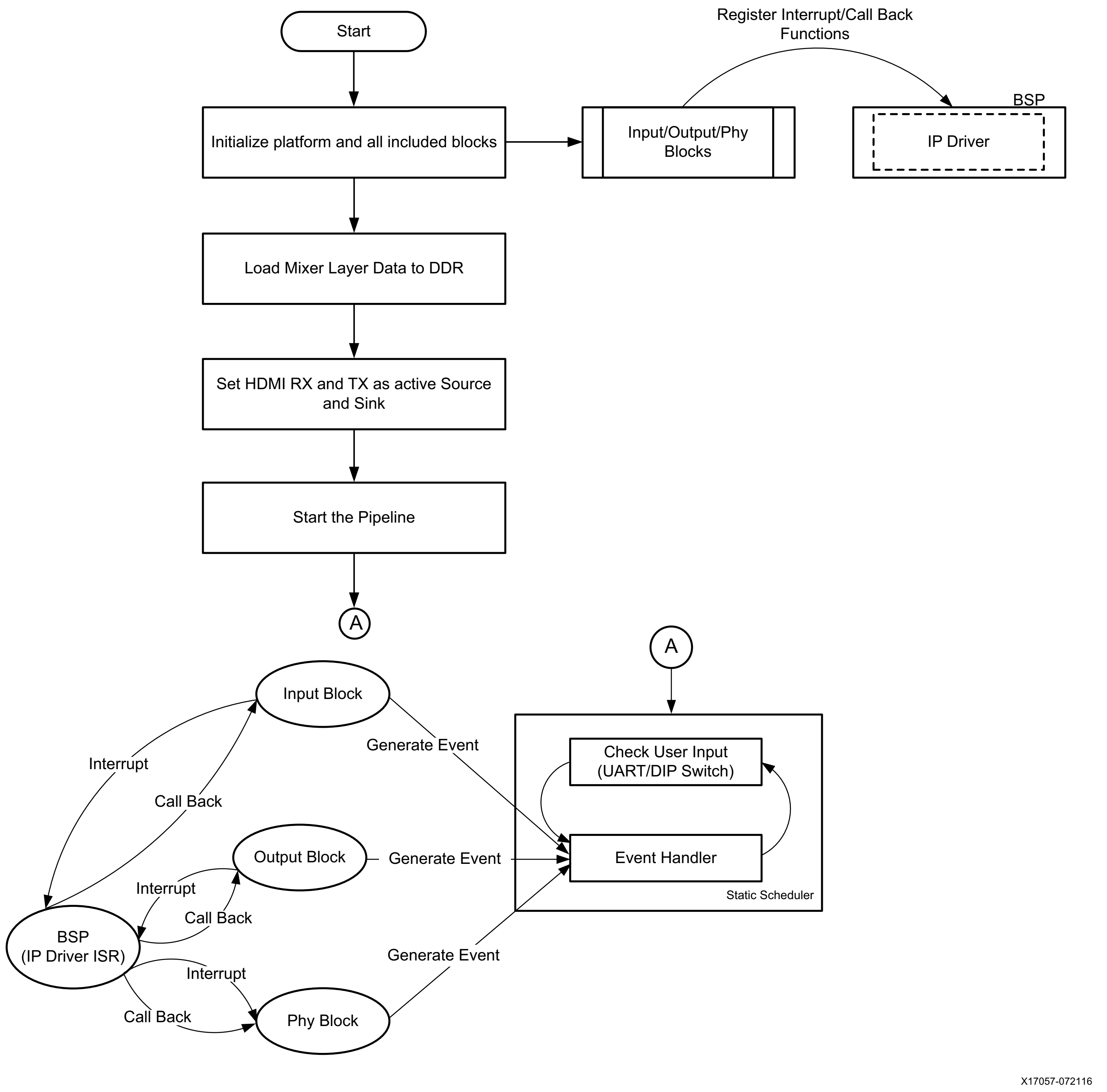


Figure 4: **System Control Flow**

The steps in the system control flow are described in this section.

1. The main program is started with these initialization tasks:
   - Define objects for each IP block driver (`input/output/phy/peripherals`).
   - Initialize platform and enable data and instruction caches.
   - Bind the objects to the system `input/output/phy` interfaces.
   - Initialize all the blocks including board peripherals. During this step each block registers required or optional callback functions (e.g., HDMI RX/TX cable connect/disconnect, frequency change, etc.)
2. The video mixer layer data is preloaded into external memory during boot up from the embedded image data in source files, which is then sourced by the mixer IP every frame.
3. The processing pipeline is started with the default input and output resolution set to 1080p at 60 Hz RGB.
4. At power on, if no input is detected at the HDMI RX, the output video is muted to maintain stable output on the screen. The default is a solid blue frame.
5. The static scheduler (background task) continuously monitors the system for user input (via serial terminal) and reacts to external interrupts to configure the video pipe accordingly.
   - When an interrupt occurs, control transfers to the source IP driver interrupt status register (ISR). Before the control returns to the user application, the registered callback is executed. The application logs the change by posting an event.
   - The event handler processes the events in the preset priority order.
   - The serial terminal parser decodes user commands and initiates the requested action.

Features

The reference design demonstrates these features of the included video IP cores:

- Auto detect input resolution change and configure the video pipe.
- Auto detect "No Input" condition and mute output video.
- Supports crop, zoom, and PIP functionality.
- Read three video layers from memory and compose an output frame with four windows blended with master stream layer (HDMI RX stream).
- Support for many input and output resolutions up to 4K2K @ 60 Hz.
- Video processing functions, such as deinterlacing, color space and format conversions, scaling, letterboxing, etc.
- Includes alternate input source (TPG) to bring up the design even when no input source is available.

When the system is first powered on, the default behavior is to have the HDMI RX stream, if present, scaled to a 1080p output resolution. All mixer layers are disabled. To achieve the window output as shown in Figure 5, configure the system in PIP mode with mixer layers enabled:

Note: At power on, the default PIP window is set to a 400 x 400 resolution at 0,0. For demonstration purposes, this needs to be set to {0,0,width/2,height/2}, where width/height refers to the output resolution.

1. On the UART, press **s** to invoke the setup menu.
2. Select option 4 to set PIP window and set these parameters:

   StartX: 0
   StartY: 0
   Width: 960
   Height: 540

3. Select option **2** to enable PIP mode. The HDMI RX core input is positioned in the PIP window in the first quadrant with a black background.
4. Select **m** to invoke the mixer menu and select option **1** to enable the desired layers.
5. Enable layers 1, 2, 3, and 8. The output should be similar to what is shown in Figure 5.

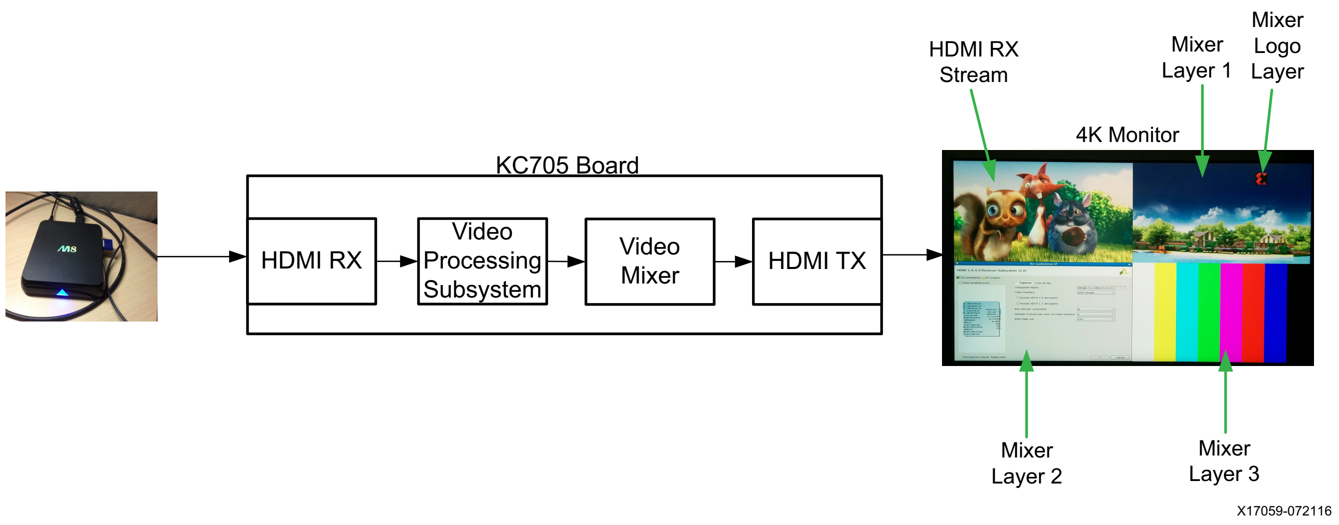


Figure 5: **Demonstration**

Tool Flow and Verification

The checklist in Table 3 indicates the tool flow and verification procedures used for the provided reference design.

Table 3: **Reference Design Checklist**

Parameter	Description
General	
Developer name	Bob Slous and Rohit Consul
Target devices	Kintex-7 FPGAs
Source code provided	Yes
Source code format	Tcl scripts and C code
Design uses code and IP from existing Xilinx application note and reference designs or third party	No
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/versions used	Vivado Design Suite 2016.2
Implementation software tools/versions used	Vivado Design Suite 2016.2
Static timing analysis performed(1)	Yes
Hardware Verification	
Hardware verified	Yes
Hardware platform used for verification	KC705 evaluation board

Notes:
1. See Known Issues.

Requirements

Hardware

The hardware requirements for this reference system are:

- Xilinx Kintex-7 FPGA KC705 Evaluation Kit.
- One inrevium HDMI 2.0 daughter card (TB-FMCH-HDMI4K).
- Two HDMI cables.
- HDMI 2.0 video source (e.g., DVD Player).
- HDMI 2.0 sink (e.g., ultra HD TV set).
- JTAG USB Platform cable or USB cable Type-A to micro-B.
- USB cable with Type A to mini B.

Software

The software requirements for this reference design are:

- Vivado Design Suite 2016.2.
- Software Development Kit (SDK) 2016.2.
- Software terminals (for example, Tera Term, HyperTerminal, or PuTTY).

External Memory

Three components in the reference design require external memory. The requirements for each are described in this section.

- The Video Processing Subsystem IP needs to store frames into DDR to perform frame rate conversions by the frame drop/repeat method, and also requires field buffers for the video deinterlacer. The buffer requirements are:
    - Five UHD (4096x2160) buffers at max 48 bit/pixel = 265 MB
    - Three 1080i field buffers at max 48 bit/pixel = 18 MB

  Total memory required for Video Processing Subsystem buffer storage = 283 MB.

- The Video Mixer IP core in the design is preconfigured with three memory layers, a logo layer, and the main streaming layer. Typically, there are multiple sources (e.g., GPU, second input source, such as DP/SDI, etc.) that generate data for the mixer memory layer. However, the reference design does not have this source, so the mixer layer image data is included as a C source file and is compiled into the application binary. The details of each layer data are:
    - Layer 1 is a 1600x542 resolution image. Every frame rendered on the screen is a consecutive subset (960x540) of the image, offset from the previous location in the image. On screen this appears as an image that is panned from left to right.

      Memory required = 1600*542*32 bits = 3.5 MB
    - Layer 2 is a graphics image of 960x540 resolution.

      Memory required = 960x540*32 bits = 2 MB
    - Layer 3 is a color bar test pattern, generated on-the-fly at power on by the included test pattern generator module, configured for 960x540 resolution.

      Memory required = 960x540*32 bits = 2 MB
    - Logo data is loaded into the block RAM allocated inside the IP core. No external memory is needed.

  The total memory required for the Video Mixer IP core layer data storage = 7.5 MB.

Note: The static images resolution, embedded in the application software, is optimized for output resolution ≥ 1080p. If output resolution < 1080p is selected, the Video Mixer IP core image will not work.

- The application software binary is approximately 9 MB (800 KB of text and 7.5 MB of video mixer image data) and runs out of DDR. At power on, the application software loads the image data into the defined mixer layer buffer addresses in DDR.

The reference design memory storage requirement is ~300 MB.

Reference Design Files

Figure 6 shows the reference design directory structure.

IMPORTANT: *The reference design should be unzipped close to root.*

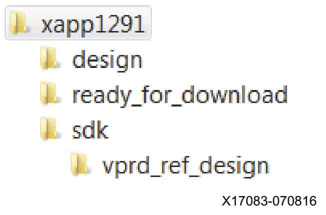
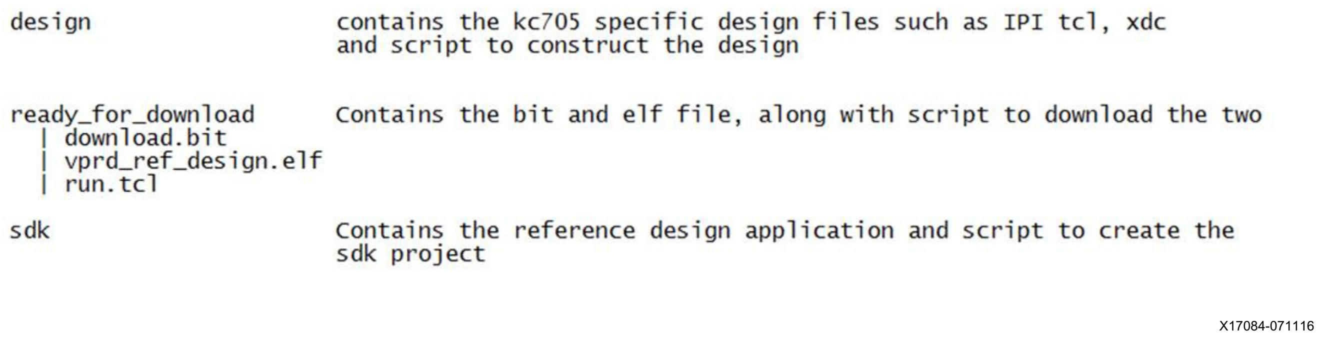


Figure 6: **Reference Design Directory Structure**

Figure 7 shows the file structure of the reference design.



```
design                   contains the kc705 specific design files such as IPI tcl, xdc
                         and script to construct the design

ready_for_download       Contains the bit and elf file, along with script to download the two
   | download.bit
   | vprd_ref_design.elf
   | run.tcl

sdk                      Contains the reference design application and script to create the
                         sdk project
```

X17084-071116

Figure 7: **Reference Design Files**

Licensing

The HDMI RX and TX subsystems, Video Processing subsystem, Video Mixer, and Video Test Pattern Generator are licensed cores. Ensure that the licenses for these cores are installed for successful design compilation.

Reference Design Steps

This section describes the Video Processing Subsystem IP core reference design setup, execution, and results.

Board Setup

The reference design is targeted for the Kintex-7 FPGA KC705 development board. The design supports the inrevium TB-FMCH-HDMI4K FMC card. The board setup is shown in Figure 8.

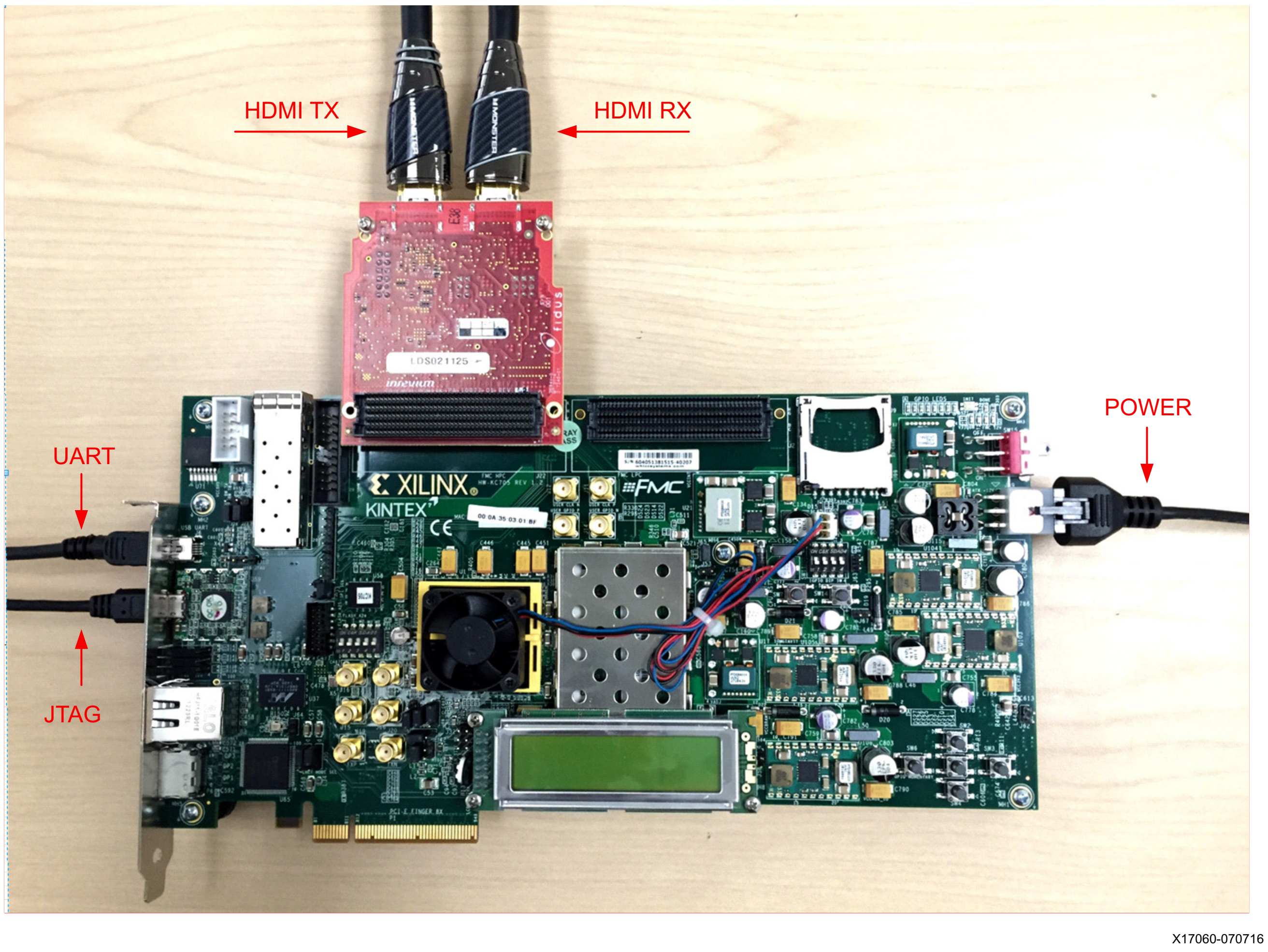


Figure 8: **KC705 Development Board Setup**

Hardware Setup

1. Connect a USB cable from the host PC to the USB JTAG port. Ensure the appropriate device drivers are installed.
2. Connect a second USB cable from the host PC to the USB UART port. Ensure that USB UART drivers described in Hardware have been installed.
3. Connect the TB-FMCH-HDMI4K board to the HPC FMC connector of the KC705 board.
4. Connect the HDMI TX port to an HDMI 1.4/2.0 sink or monitor (e.g., an ultra HD TV set).
5. Connect the HDMI RX port to an HDMI 1.4/2.0 source (e.g., a DVD Player).
6. Connect the KC705 board to power supply slot J49.
7. Switch on the KC705 board.
8. Start a terminal program (e.g., Hyper Terminal) on the host PC with these settings for the Standard COM port:
   - Baud Rate: 115200
   - Data Bits: 8
   - Parity: None
   - Stop Bits: 1
   - Flow Control: None

Running the Reference Design

There are two ways to work with the reference design:

- Run the reference design using the precompiled binary files in the `ready_for_download` directory.
- Re-create the reference design (both hardware and software) from scratch using script files.

This section describes the steps for executing the system using the files in the `ready_for_download` directory.

1. Launch the XSCT shell:

   Start > All Programs > Xilinx Design Tools > SDK 2016.2 > Xilinx Software Command Line Tool

2. Change to the bitstream directory:

   ```
   xsct% cd {<path-to-ready_for_download-directory}
   ```

3. Switch to the xmd shell:

   ```
   xsct% xmd
   ```

4. Download the bitstream and `.elf` files to the FPGA:

   ```
   xmdt% source run.tcl
   ```

5. Exit the xmd shell:

```
xmd% exit
```

IMPORTANT: *The software application starts immediately after the completion of the FPGA configuration. Messages displayed on the serial terminal show the progress of the application powering up.*

Creating the Reference Design Using Script Files

For users who prefer to create the design from scratch, TCL scripts (compatible with the Vivado 2016.2 tools) are provided in the design folder. The `setup.tcl` file generates an example design for the KC705 board with the FMC HDMI 2.0 card.

Building the Hardware Design

To generate the bit file in the Vivado Design Suite 2016.2:

1. Open the Vivado Design Suite.
2. At the Tcl console, change to the workspace directory:

```
> cd <unzip_dir>\xapp1291
```

3. Run the Tcl script to create the block design:

```
> source ./design/setup.tcl
```

4. Generate the bitstream by clicking **Generate Bitstream** from the **Flow** menu.

IMPORTANT: *The bitstream generation can take about an hour to complete on a typical Windows-based computer.*

5. Export the hardware to SDK by selecting **File > Export > Export Hardware**. Select the option to include the bitstream. This creates a `vprd.sdk` directory under the Vivado project directory and exports the Xilinx platform project file into the directory.

Known Issues

- The design fails to meet timing during compilation and is a known issue with the Video Mixer core. The Video Mixer core is in early access. This issue will be addressed in the Vivado Design Suite 2016.3 release. A workaround is to disable the logo layer in the Video Mixer to remove timing violations.
- The Vivado Design Suite on computers running Windows is sensitive to project path length. To avoid any path length issues, unzip the design to c:\.

Creating SDK Project

The SDK folder has a script (`vprd.tcl`) that automates the process of creating an SDK project for the reference design and generating the downloadable `.bit` and `.elf` files.

To run the provided Tcl script:

1. Copy the exported hardware design file (`vprd_wrapper.hdf`) to the sdk folder.
2. Launch the Xilinx Software Command Line Terminal (XSCT) shell:

   Start > All Programs > Xilinx Design Tools > SDK 2016.2 > Xilinx Software Command Line Tool

3. Change to the sdk directory:

   ```
   xsct% cd {<path-to-sdk-directory>}
   ```

4. Source the tcl file

   ```
   xsct%>source vrpd.tcl
   ```

5. Execute the script:

   ```
   xsct%>vprd vprd_wrapper.hdf
   ```

The Tcl script performs these tasks:

- Creates workspace.
- Creates hardware project.
- Creates BSP.
- Creates application project.
- Builds BSP and application project.

After the process is complete, the required files are available here:

- Bit file is available in the `vrpd.sdk/vprd_kc705_hw_platform_0` folder.
- Elf file is available in the `vprd.sdk/vprd_ref_design/{Debug/Release}` folder.

Perform these steps to run the software application from within the SDK:

IMPORTANT: *Make sure the hardware is powered on and a Digilent Cable or an USB platform cable is connected to the host PC. Also, ensure that a USB cable is connected to the UART port of the KC705 board.*

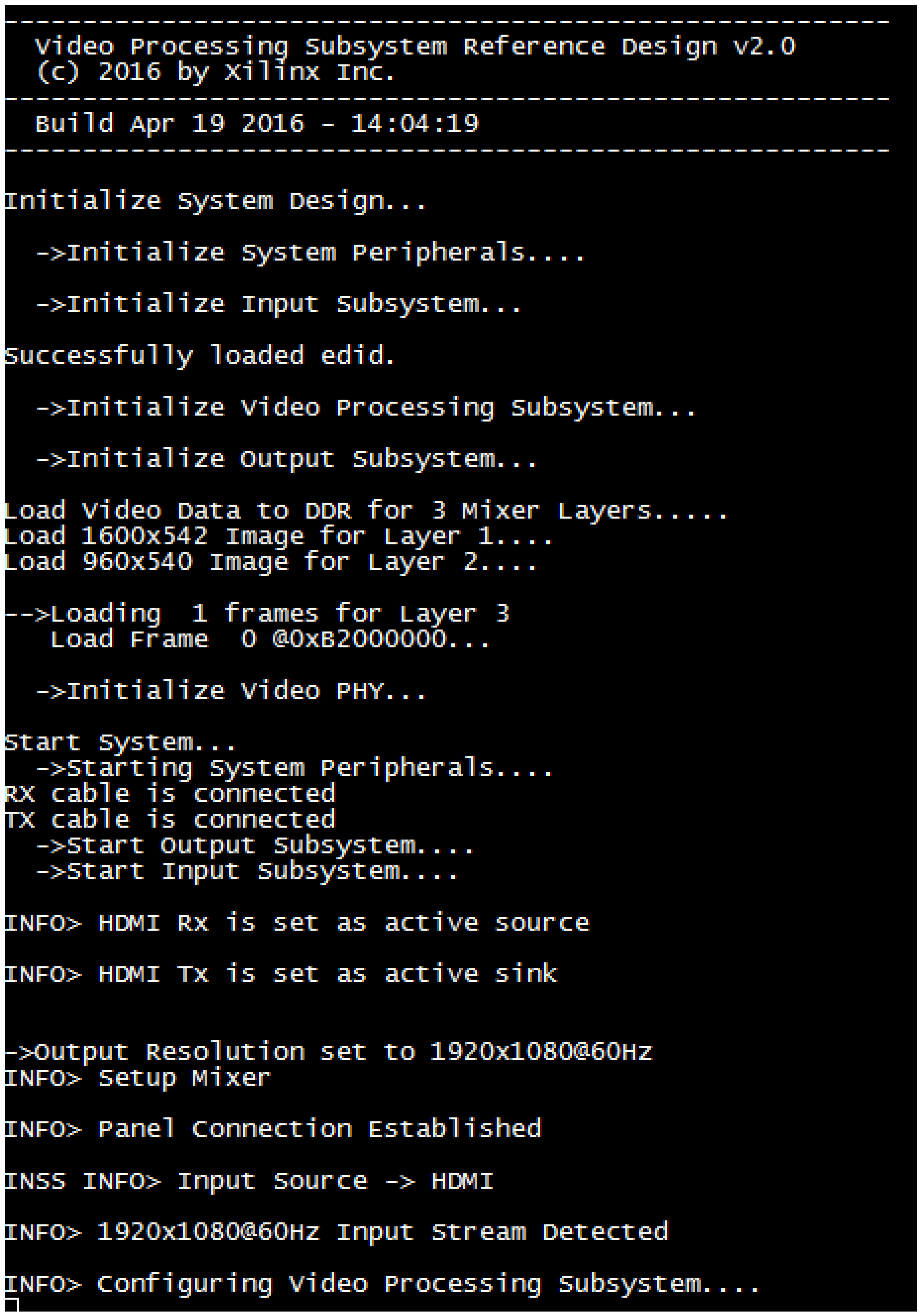
1. Launch SDK.
2. Set workspace to the `vprd.sdk` folder in the prompted window. The SDK project opens automatically (if a welcome page appears, close it.)
3. Download the bitstream into the FPGA by selecting **Xilinx Tools > Program FPGA**. The program FPGA dialog box opens.
4. Ensure that the bitstream field shows the bitstream file generated by the Tcl script, and then click **Program**.

   Note: The DONE LED on the board turns green if the programming is successful.
5. A terminal program (HyperTerminal or PuTTY) is needed for UART communication. Open the program, choose the appropriate port, set baud rate to 115 kb/s, and establish the serial port connection.
6. Select and right-click the application `vprd_ref_design` in the Project_Explorer panel.
7. Select **Run As > Launch on Hardware (System Debugger)**.
8. Select **Binaries and Qualifier** in the window and click **OK**.

The design execution progress is shown in the terminal program.

Results

Figure 9 shows the reference design powering up.



```
----------------------------------------------------------
  Video Processing Subsystem Reference Design v2.0
  (c) 2016 by Xilinx Inc.
----------------------------------------------------------
  Build Apr 19 2016 - 14:04:19
----------------------------------------------------------

Initialize System Design...

  ->Initialize System Peripherals....

  ->Initialize Input Subsystem...

Successfully loaded edid.

  ->Initialize Video Processing Subsystem...

  ->Initialize Output Subsystem...

Load Video Data to DDR for 3 Mixer Layers.....
Load 1600x542 Image for Layer 1....
Load 960x540 Image for Layer 2....

-->Loading  1 frames for Layer 3
   Load Frame  0 @0xB2000000...

  ->Initialize Video PHY...

Start System...
  ->Starting System Peripherals....
RX cable is connected
TX cable is connected
  ->Start Output Subsystem....
  ->Start Input Subsystem....

INFO> HDMI Rx is set as active source

INFO> HDMI Tx is set as active sink


->Output Resolution set to 1920x1080@60Hz
INFO> Setup Mixer

INFO> Panel Connection Established

INSS INFO> Input Source -> HDMI

INFO> 1920x1080@60Hz Input Stream Detected

INFO> Configuring Video Processing Subsystem....
```

X17051-051816

Figure 9: **Reference Design Power Up**

User Interface

The video processing reference design (VPRD) provides two interfaces for interacting with the design:

- Serial terminal.
- Pushbuttons.

Serial Terminal (UART)

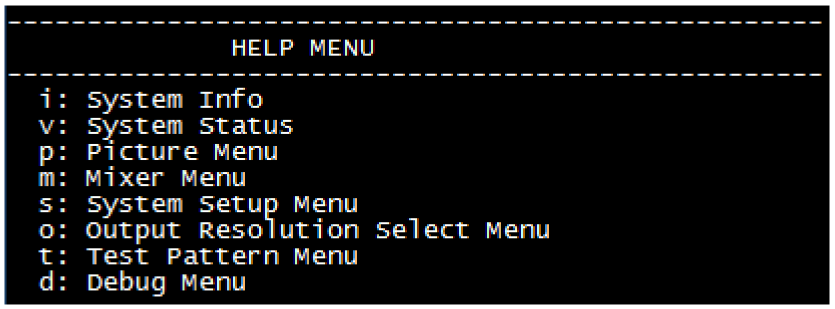
A serial terminal interface is provided for interacting with the design. Connect the host computer to the USB UART, open the terminal, and set the terminal properties to 115200 baud rate, 8 data bits, no parity, and 1 stop bit.

After the application has been downloaded to the FPGA, the serial terminal prints a welcome message and shows the progress.

UART Menus

The reference design functionality can be accessed through the serial terminal. There are different menus available for each feature. The 'h' key displays the help menu, which lists all of the feature menus available in the reference design.

Note: The menu selection keys are not case sensitive.



```
----------------------------------------------------
              HELP MENU
----------------------------------------------------
  i: System Info
  v: System Status
  p: Picture Menu
  m: Mixer Menu
  s: System Setup Menu
  o: Output Resolution Select Menu
  t: Test Pattern Menu
  d: Debug Menu
```

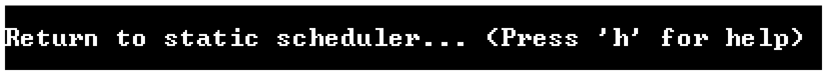
X17073-051816

Figure 10: **Help Menu**

Key Entry Mechanism

The reference design includes a serial terminal parser that provides these usage characteristics:

1. A key entry is not committed until the Return key is pressed.
2. When in the UART menu, the static scheduler is disabled because the software is waiting for user input. The system will not react to external events.
3. The user must exit the menu to enable the static scheduler (confirmation is provided when the system enables the static scheduler and resumes the background monitoring task).



```
Return to static scheduler... (Press 'h' for help)
```

Figure 11: **Return to Static Scheduler**

System Info (i)

The System Info command provides information on the IP blocks in the reference design, categorized by the associated subsystem.

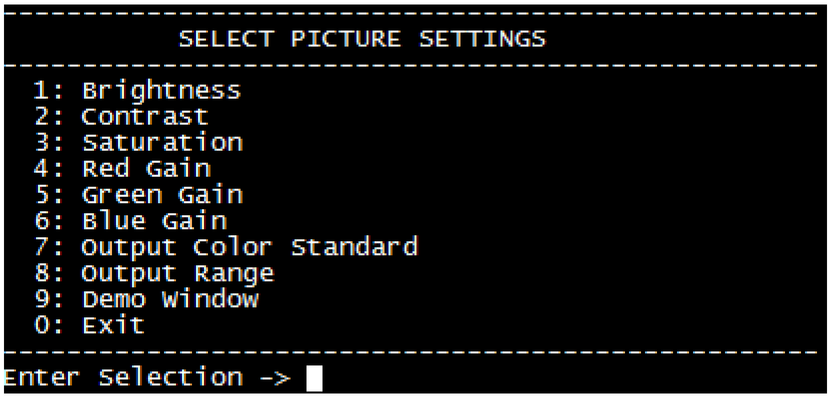
System Status (v)

The System Status command provides status information about the different blocks in the reference design:

- System status overview.
- Detected input timing (HDMI RX).
- Generated output timing (HDMI TX).
- Phy status.
- Link status.

Picture Menu (p)

The Picture Menu command provides options that impact the picture settings. Unless modified, the picture settings impact the full image on the screen. There is also a demonstration mode provided that allows a window to be defined within the output resolution where these changes will have an impact. Pixels outside the demonstration window remain as is.



```
------------------------------------------------------
              SELECT PICTURE SETTINGS
------------------------------------------------------
  1: Brightness
  2: Contrast
  3: Saturation
  4: Red Gain
  5: Green Gain
  6: Blue Gain
  7: Output Color Standard
  8: Output Range
  9: Demo Window
  0: Exit
------------------------------------------------------
Enter Selection ->
```

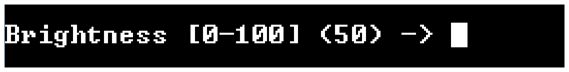
X17075-051816

Figure 12: **Picture Menu**

This menu provides these selection characteristics:

<Feature>   [Range]   (Current set value) --> <New Value>

For example:

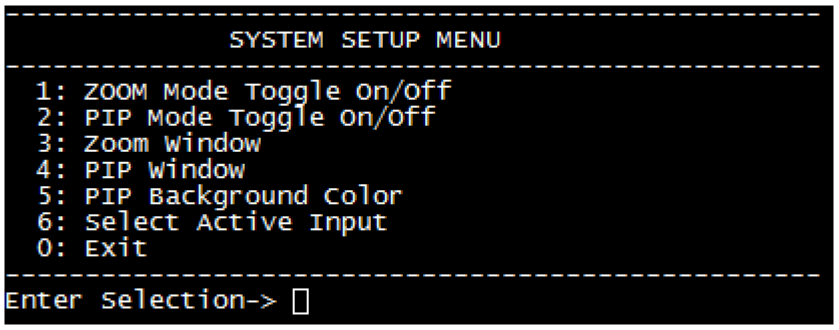


```
Brightness [0-100] (50) ->
```

X17076-061416

System Setup Menu (s)

The System Setup Menu command provides access to the zoom and PIP features, and allows the window to be set for each feature independently. The default settings are listed in Table 4.



```
--------------------------------------------------------------
                  SYSTEM SETUP MENU
--------------------------------------------------------------
  1: ZOOM Mode Toggle On/Off
  2: PIP Mode Toggle On/Off
  3: Zoom Window
  4: PIP Window
  5: PIP Background Color
  6: Select Active Input
  0: Exit
--------------------------------------------------------------
Enter Selection->
```

X17077-051816

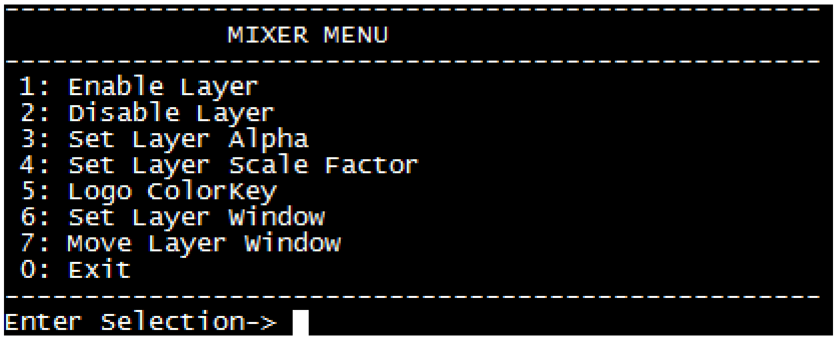
Figure 13: **System Setup Menu**

Table 4: **System Setup Menu Default Settings**

Feature	Start Coordinate	Window Size
Zoom	0,0	400x400
PIP	0,0	400x400

Video Mixer Menu (m)

The Mixer IP in the reference design is preconfigured with three memory layers, a logo layer, and the main streaming layer. Each memory layer has the alpha and scale feature enabled. This menu allows the user to experiment with these features.



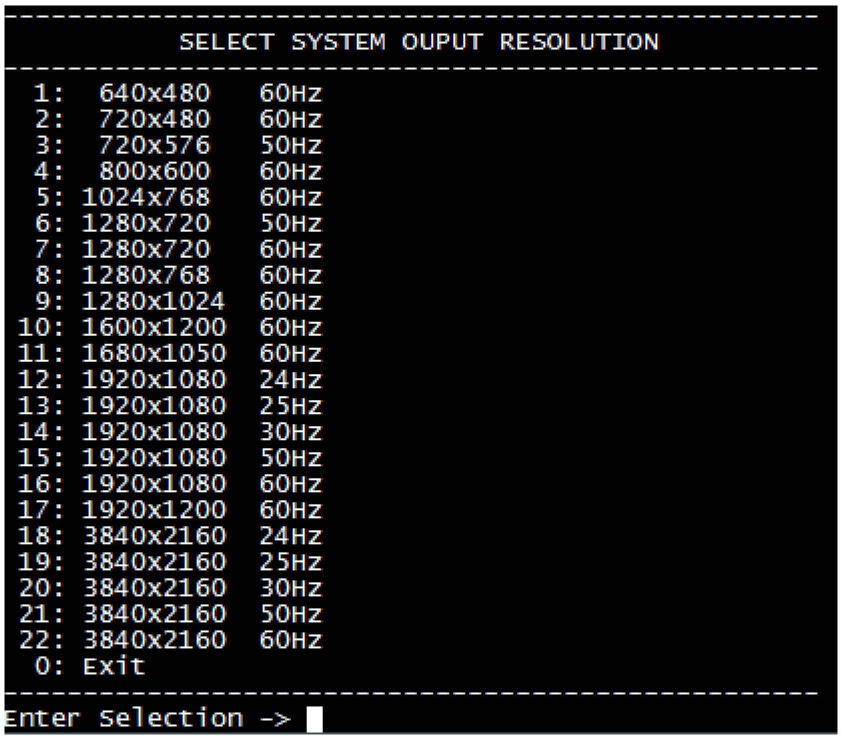
```
--------------------------------------------------------------
                  MIXER MENU
--------------------------------------------------------------
 1: Enable Layer
 2: Disable Layer
 3: Set Layer Alpha
 4: Set Layer Scale Factor
 5: Logo ColorKey
 6: Set Layer Window
 7: Move Layer Window
 0: Exit
--------------------------------------------------------------
Enter Selection->
```

X17078-051816

Figure 14: **Mixer Menu**

Output Resolution (o)

The output resolution menu provides the list of output resolutions supported by the reference design.



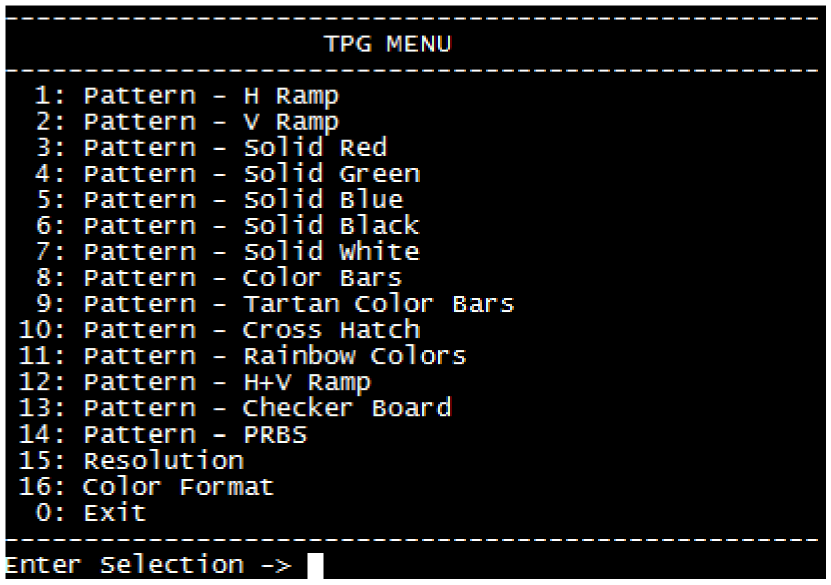
```
-------------------------------------------------------
           SELECT SYSTEM OUPUT RESOLUTION
-------------------------------------------------------
  1:   640x480    60Hz
  2:   720x480    60Hz
  3:   720x576    50Hz
  4:   800x600    60Hz
  5: 1024x768     60Hz
  6: 1280x720     50Hz
  7: 1280x720     60Hz
  8: 1280x768     60Hz
  9: 1280x1024    60Hz
 10: 1600x1200    60Hz
 11: 1680x1050    60Hz
 12: 1920x1080    24Hz
 13: 1920x1080    25Hz
 14: 1920x1080    30Hz
 15: 1920x1080    50Hz
 16: 1920x1080    60Hz
 17: 1920x1200    60Hz
 18: 3840x2160    24Hz
 19: 3840x2160    25Hz
 20: 3840x2160    30Hz
 21: 3840x2160    50Hz
 22: 3840x2160    60Hz
  0: Exit
-------------------------------------------------------
Enter Selection ->
```

X17082-051816

Figure 15: **Output Resolution**

Test Pattern Menu (t)

The TPG is used in the reference design as a secondary input stream source. The user can switch between the TPG and the HDMI RX (default) via the option in the system setup menu. Different aspects of the TPG stream can be configured from this menu.



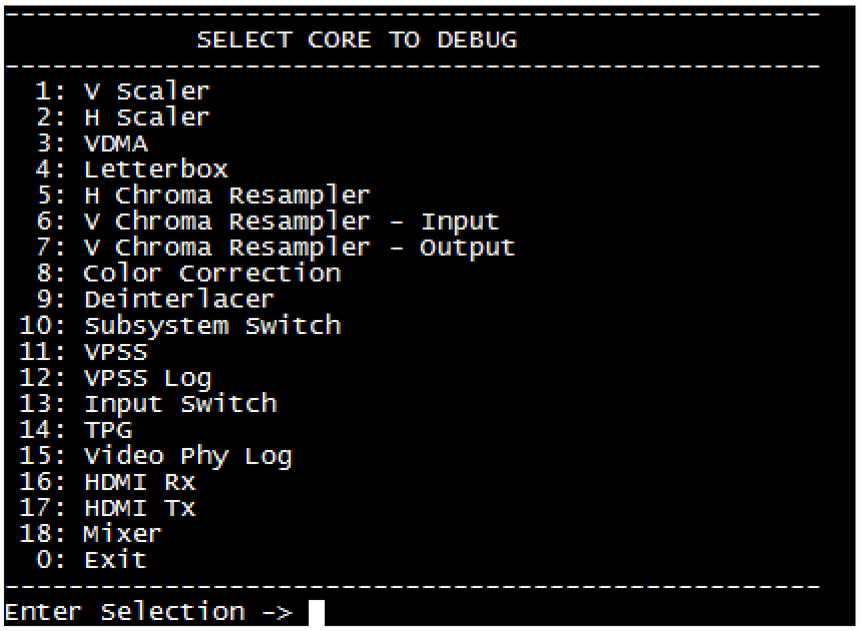
```
--------------------------------------------------
                    TPG MENU
--------------------------------------------------
  1: Pattern - H Ramp
  2: Pattern - V Ramp
  3: Pattern - Solid Red
  4: Pattern - Solid Green
  5: Pattern - Solid Blue
  6: Pattern - Solid Black
  7: Pattern - Solid White
  8: Pattern - Color Bars
  9: Pattern - Tartan Color Bars
 10: Pattern - Cross Hatch
 11: Pattern - Rainbow Colors
 12: Pattern - H+V Ramp
 13: Pattern - Checker Board
 14: Pattern - PRBS
 15: Resolution
 16: Color Format
  0: Exit
--------------------------------------------------
Enter Selection ->
```

X17080-051816

Figure 16: **Test Pattern Menu**

Debug Menu (d)

The debug menu provides a software debug interface for obtaining state and register information for a core.



```
---------------------------------------------------
             SELECT CORE TO DEBUG
---------------------------------------------------
  1: V Scaler
  2: H Scaler
  3: VDMA
  4: Letterbox
  5: H Chroma Resampler
  6: V Chroma Resampler - Input
  7: V Chroma Resampler - Output
  8: Color Correction
  9: Deinterlacer
 10: Subsystem Switch
 11: VPSS
 12: VPSS Log
 13: Input Switch
 14: TPG
 15: Video Phy Log
 16: HDMI Rx
 17: HDMI Tx
 18: Mixer
  0: Exit
---------------------------------------------------
Enter Selection ->
```

X17081-051816

Figure 17: **Debug Menu**

Pushbuttons

Pushbuttons on the KC705 board are used to control zoom and PIP window movement on the screen.

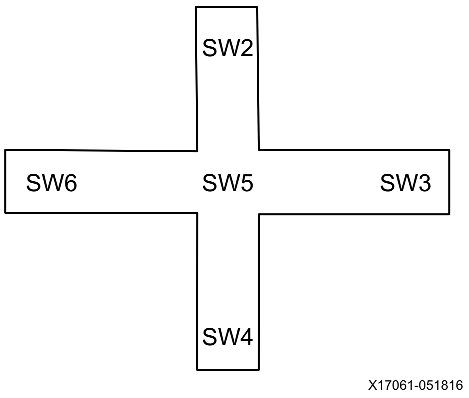


Figure 18: **Pushbuttons**

- SW2: move window up.
- SW4: move window down.
- SW6: move window left.
- SW3: move window right.
- SW5: feature (zoom/PIP) off, if on, else don't care.

Debugging

This section includes tips for resolving typical user issues.

- LED0

  HDMI TX subsystem lock. If this LED is off, it suggests that there is a mismatch in video timing information between the HDMI TX stream and what the HDMI_TX_SS is expecting. Confirm that the HDMI TX stream timing is correct, or disconnect and then reconnect the TX cable to reinitialize the HDMI TX.

- There are multiple debugging taps within the pipeline that can be queried to determine system or IP status. These debug taps are shown by the application software via a serial terminal. See Debug Menu (d) for more information.

References

1. Xilinx Kintex-7 FPGA KC705 Evaluation Kit
2. Video Processing Subsystem
3. Inrevium TB-FMCH-HDMI4K
4. *Video Processing Subsystem v2.0 Product Guide* (PG231)
5. *Video Mixer v1.0 LogiCORE IP Product Guide* (PG243)
6. *HDMI 1.4/2.0 Transmitter Subsystem v1.0 Product Guide* (PG235)
7. *HDMI 1.4/2.0 Receiver Subsystem v1.0 LogiCORE IP Product Guide* (PG236)
8. *Video PHY Controller v2.0 Product Guide* (PG230)
9. *AXI Reference Guide* (UG761)
10. *Video Test Pattern Generator LogiCORE IP Product Guide* (PG103)
11. *KC705 Board User Guide* (UG810)
12. *HDMI 2.0 Implementation on Kintex-7 FPGA GTX Transceivers Application Note* (XAPP1287)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/22/2016	1.0.1	Typographical update.
07/20/2016	1.0	Initial Xilinx release.

Please Read: Important Legal Notices