XAPP1264 (v1.0) December 03, 2015

Real Time Video Engine 3.1 Implementation in Zynq-7000 All Programmable SoCs

Author: Bob Feng

Summary

This application note leverages the latest Zynq®-7000 All Programmable SoC FPGA architecture to provide a truly scalable video processor reference design to serve multi-stream/multi-pipelined video processing needs. It produces broadcast-quality output and is targeted for applications such as multi-viewer display, video switches, and multichannel video routers, as well as multi-stream up-down converters.

The objective of the Real Time Video Engine (RTVE) reference design is to provide a highly demonstrable, broadcast-quality video processing reference design targeted to a wide range of video applications.

To order a set of reference design files, contact OmniTek or your Xilinx sales representative.

Key Features

Key features of the design include:

- Programmable input and output formats:
  - Support for SD/HD/3G/6G/12G-SDI, Display Port 1.2, and HDMI 1.3 (using external components)
  - Support for a range of Ultra High Definition (UHD)/2160p formats including 2-Sample Interleaved and Square Division UHD formats, in the following operating modes:
    - Single Input: One video path for up to 2160-p60 video on both input and output
    - Dual Inputs: Two video paths for up to 2160-p60 video, without image cropping
    - Quad Inputs: Four video paths for HD input and up to 2160-p60 output
- On-the-fly switchable video sources
- Progressive or interlaced video format
- Fully featured Video Processing:
  - Motion-Adaptive and/or Edge-Adaptive Deinterlacer
  - Polyphase Scaler (resizer) with on-the-fly customizable coefficient table, super-resolution image enhancement (sharpening) and image softening
  - 10-bit 4:4:4 processing engine
  - Contrast enhancement through Colour Correction Matrix

    - Option of multiple overlays
    - Built-in Test Pattern Generator
- Complete Linux 3.10 support with QT framework
    - Complete video I/O device drivers
    - Complete video processing APIs
    - Linux 2D Graphics overlay onto live video
- User Software design examples:
    - QT web client, providing user interface to RTVE3.1 operation
    - QT drawing demonstration
- Built using Xilinx's latest Vivado Development Suite and IPI design interface
- Completely based on Xilinx AXI infrastructure
    - AXI-Lite CPU Control interface
    - AXI-Memory map for external memory access
    - AXI-Streaming for video streaming among video processing blocks

Reference Design

The RTVE 3.1 is built around the OmniTek 3.0 Scalable Video Processor (OSVP) suite and is designed to demonstrate the manipulation of UHD images using this IP.

The OSVP Suite is a highly configurable set of IP blocks and optional features which provides a powerful range of tools for multi-video format conversion and image enhancement for video formats up to 60Hz Ultra HD, with 120Hz Ultra HD output as a further option.

For ease of implementation and to make best use of system resources, the principal IP blocks are packaged as a single OSVP core.

Up to 8 video channels are provided, each of which can be individually configured to offer support for:

- Video input formats up to 4096 x 2160 60Hz
- Video output formats up to 4096 x 2160 120Hz
- Interlaced, progressive or segmented frame (PsF) input; progressive or interlaced output
- YUV in 4:2:0, 4:2:2 or 4:4:4 and RGB Color format
- 8, 10 or 12-bit colour depths
- Up to 8 video processing paths, each individually configurable for video standards and processing actions

Full 12-bit YUV or RGB 4:4:4 processing providing:

- Up/Down/Cross conversion between any supported standards
- Asynchronous input and output timing with frame synchronization
- Frame synchronous transitions when changing frame rate
- Chroma re-sampling
- Full 6-axis YUV/RGB colour correction, brightness/saturation control and hue rotation
- Colour primary mapping
- Motion- and/or Edge-adaptive de-interlacing with best-in-class low-angle handling
- 3:2 and 2:2 film cadence detection and processing, including handling of mixed cadence such as interlaced video over 3:2 film
- Noise reduction
- Crop and resize with Super-Resolution image enhancement
- Alpha blending of multiple video sources

Figure 1 shows the OSVP block diagram

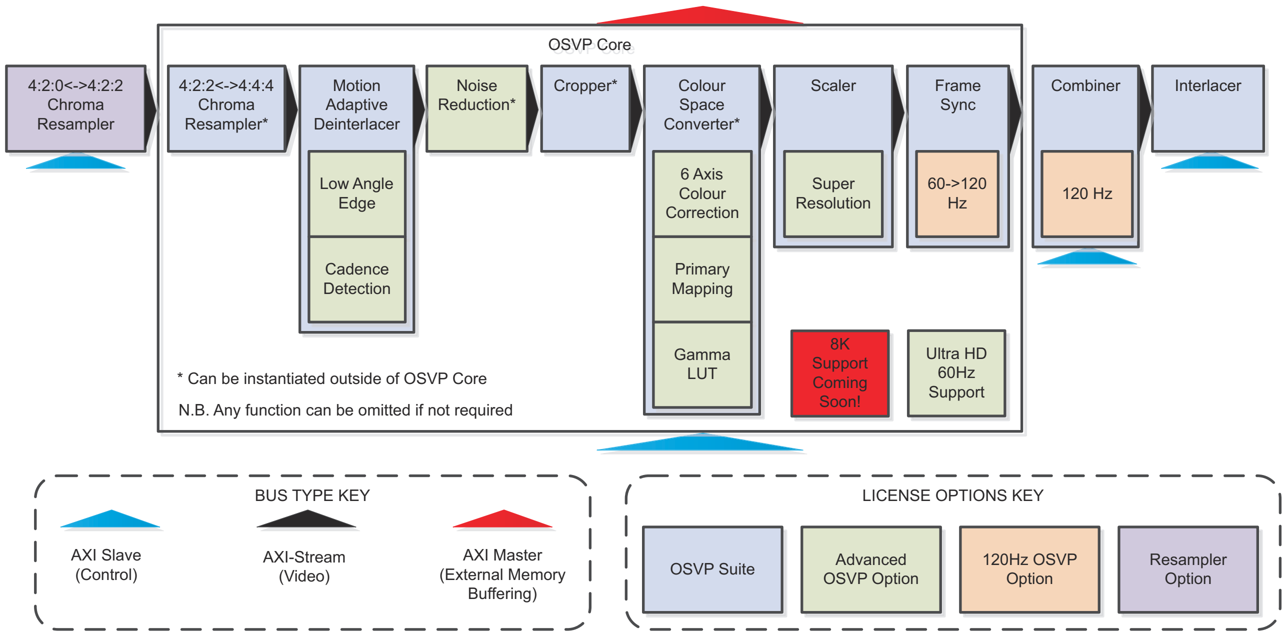


Figure 1: **OSVP Core Block diagram**

More details on the OSVP are available at: http://omnitek.tv/osvp

Hardware

The RTVE 3.1 design is implemented for installation using an OmniTek OZ745 development kit [3], fitted with an OmniTek 12G/DP OZ745 FPGA Mezzanine Card (FMC).

Note: The FMC card is only required when a DisplayPort monitor is used.

The OmniTek 12G/DP expansion card is an FMC high-pin-count (HPC) compatible circuit board which is designed to be fitted on the OmniTek OZ745 assembly. It provides the following video ports:

- 2 x SDI inputs compatible with SD, HD, 3G, 6G and 12Gb/s interface standards
- 2 x SDI outputs compatible with SD, HD, 3G, 6G and 12Gb/s interface standards

Note: There are several versions of the OZ745 and the associated FMC. 6G and 12G are only supported on OZ745 -3 and Rev D FMC (or later). Earlier versions will only support up to 3G SDI. However UHD (4K) is supported on these boards via quad 3G SDI I/O and via the DisplayPort 1.2 output.

Figure 2 shows a block overview of the RTVE 3.1 design.

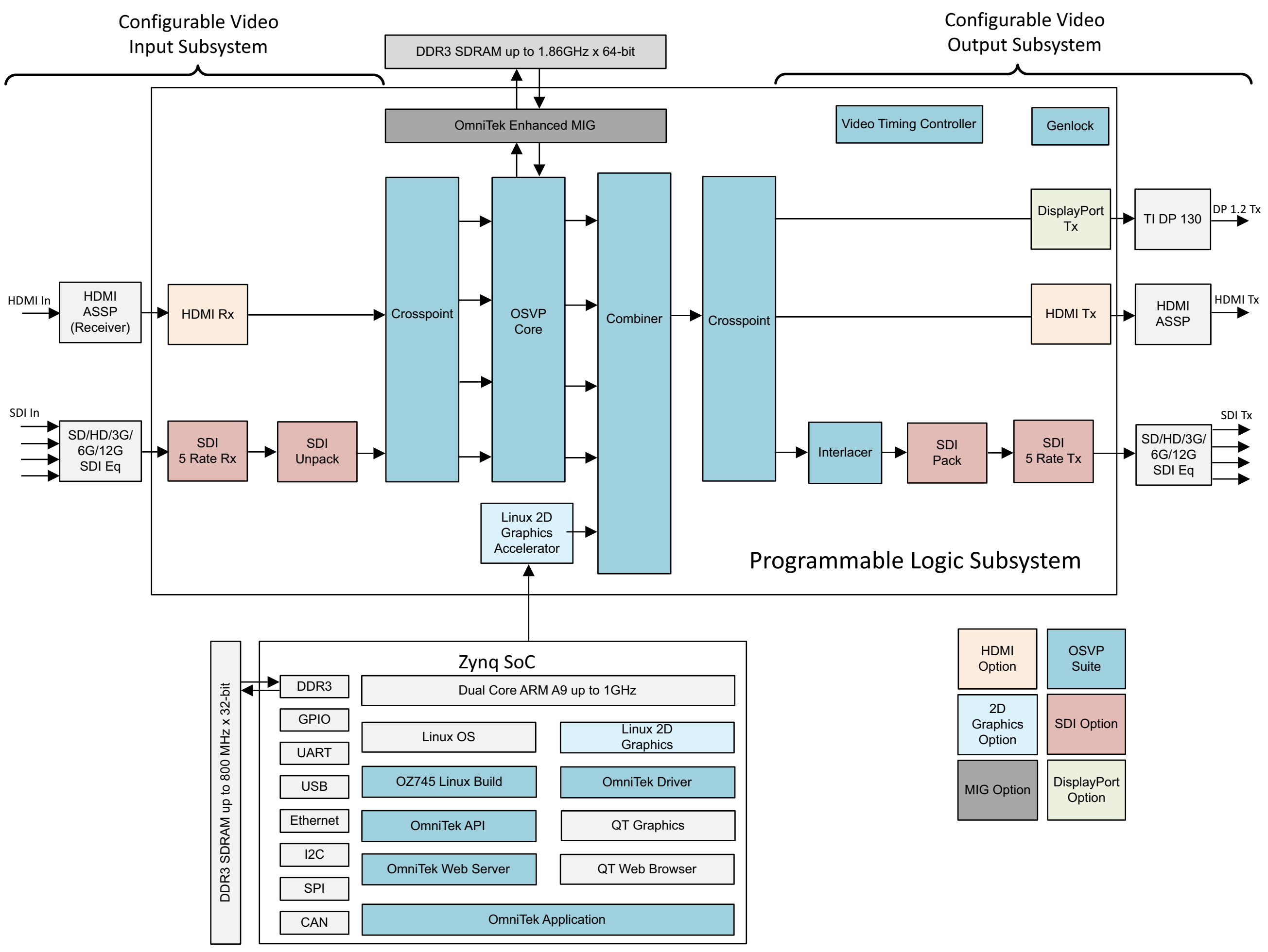


Figure 2: **RTVE 3.1 System Block Diagram**

Omni Offset Capability

Overview

Omni Offset Capability enables easy interaction between the firmware and software domains.

The Omni Offset Capability automatically gathers information from the IPs present in the firmware design and then passes that information to the software, which uses it to configure the firmware IPs according to the target design.

Note: The IPs in the firmware design must be configured for AXI4-Lite based CPU interface to leverage the features of the Omni Offset Capability.

Information gathered is as follows:

- The number of IPs used in the firmware design
- The base address of each IP present in the firmware design
- The allocated memory space for each IP
- The ID Types, ID Tags and ID Associated for each Omni IP

Omni Offset Capability IO Interface

The Omni Offset Capability provides an AXI4-Lite interface (Figure 3), which is used to access and share information between a CPU (for example, Microblaze, ARM) and the Omni Offset Capability.

AXI4-Lite is a light-weight, single transaction memory mapped interface. It has a small logic foot print and is simple to use from both a design and usage perspective.

The interface consists of five channels:

- Read Address Channel,
- Write Address Channel
- Read Data
- Channel, Write Data Channel
- Write Response Channel

The Omni Offset Capability interface data width is constrained to 32-bits.

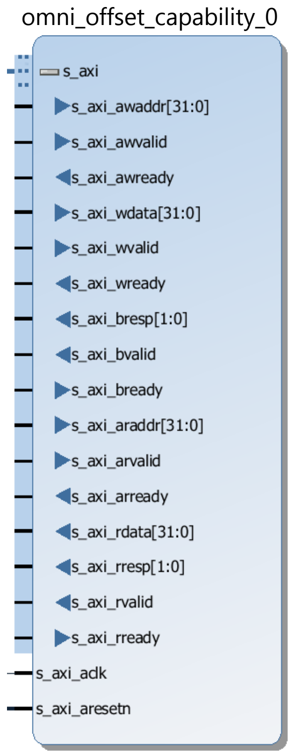
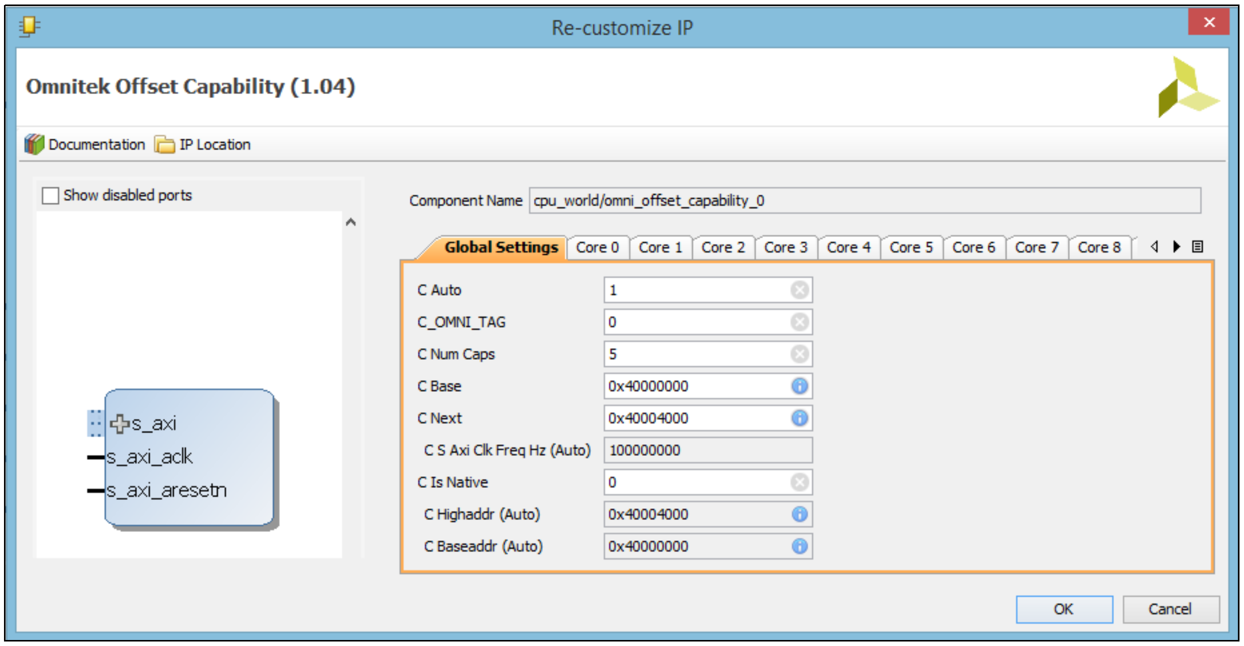


Figure 3: **Omni Offset Capability IO Interface**

GUI

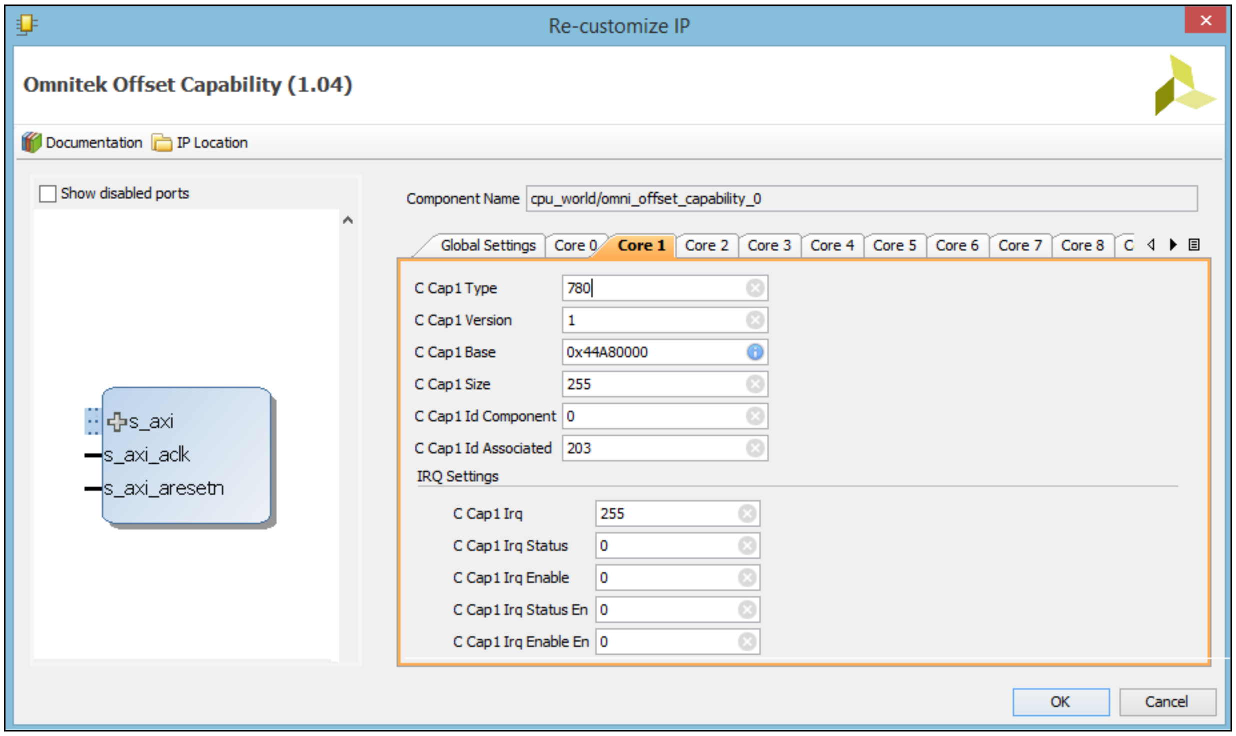
The Omni Offset Capability offers a GUI interface which can be edited during firmware design. GUI design can be split into two categories; Global (Figure 4) and Core (Figure 5).



Component Name cpu_world/omni_offset_capability_0

C Auto	1
C_OMNI_TAG	0
C Num Caps	5
C Base	0x40000000
C Next	0x40004000
C S Axi Clk Freq Hz (Auto)	100000000
C Is Native	0
C Highaddr (Auto)	0x40004000
C Baseaddr (Auto)	0x40000000

Figure 4: **Omni Offset Capability GUI Global Setting Interface**



Component Name cpu_world/omni_offset_capability_0

C Cap1 Type	780
C Cap1 Version	1
C Cap1 Base	0x44A80000
C Cap1 Size	255
C Cap1 Id Component	0
C Cap1 Id Associated	203

IRQ Settings

C Cap1 Irq	255
C Cap1 Irq Status	0
C Cap1 Irq Enable	0
C Cap1 Irq Status En	0
C Cap1 Irq Enable En	0

Figure 5: **Omni Offset Capability GUI Core Setting Interface**

The **Global Settings** tab contains different parameters which allow the configuration of the Omni Offset Capability. This parameters are described in Table 1.

Table 1: **Omni Offset Capability - Global Settings**

Parameter	Supported Values	Description
C Auto	0,1	Configures the Omni Offset Capability to work in either automatic or manual mode. `C Auto = 1` – Automatic mode `C Auto = 0` – Manual mode Automatic mode allows detection of information from all the Omni IPs. Manual mode is used when it is necessary to map third-party IPs.
C_OMNI_TAG	Integer (up to 32 bits)	This is a unique label used by each Omni Offset Capability IP to interact with the rest of the IPs present in the firmware design. This value should be different for each instance of the Omni Offset Capability instantiated in the firmware design.
C Num Caps	Integer (up to 32 bits)	This value corresponds to the total number of IPs associated with the Omni Offset Capability.

Table 1: **Omni Offset Capability - Global Settings**

Parameter	Supported Values	Description
C Base	Hex (up to 32 bits)	If there is only one Omni Offset Capability instantiated, this value corresponds to its base address. If there is more than one Omni Offset Capability, this value correspond to the lowest base address of the Omni Offset Capability instantiated in the design.
C Next	Hex (up to 32 bits)	If there is only one Omni Offset Capability or if this Omni Offset Capability is the last node in the linked list, this value corresponds to zero. Otherwise, this value corresponds to the starting base address of the other Omni Offset Capability instantiated in the design.
C S Axi Clk Freq Hz (Auto)	Integer (up to 32 bits)	Value for the AXI4-Lite clock that is driving the Omni Offset Capability.
C is Native	0,1	This value selects the CPU interface: `C Is Native = 1` - Native CPU Interface `C is Native = 0` - Axi4-Lite Interface
C Highaddr (Auto)	Hex (up to 32 bits)	Shows the high address of the Omni Offset Capability in the firmware design.
C Baseaddr (Auto)	Hex (up to 32 bits)	Shows the base address of the Omni Offset Capability in the firmware design.

The **Core Settings** can be configured in each of the Core tabs. Each Core tab contains different parameters which allow the Omni Offset Capability to be linked with a specific IP Core in the firmware design. These parameters are described in Table 2.

Note: A firmware design can contain more than one Omni Offset Capability. In this scenario, the software expects the Omni Offset Capability to be instantiated as a linked list, (i.e., each Omni Offset Capability should point to the next Omni Offset Capability address in the sequence). In addition, the last node of the linked list should be configured with a `C Next` parameter equal to zero.

Table 2: **Omni Offset Capability - Core Settings**

Parameter	Supported Values	Description
C Cap Type	Integer (up to 32 bits)	This is a unique number which identifies each of the IPs in the firmware design. This value is predetermined by the software and located in the Omni Offset Capability's header file.
C Cap Version	Integer (up to 32 bits)	This value corresponds to the version of the IP associated with the Omni Offset Capability.
C Cap Base	Hex (up to 32 bits)	This is the base address of the IP associated with the Omni Offset Capability.
C Cap Size	Integer (up to 32 bits)	This is the number of bytes that need to be reserved in order to access the memory space of the IP associated with the Omni Offset Capability.
C Cap ID Component	Integer (up to 32 bits)	This value is used to identify different IPs that belong to the same IP category (i.e., they have the same C Cap Type). For example, a firmware design can have multiples video timing generators, all of them having the same C Cap Type parameter, but each of them needs to have a unique C Cap ID Component, so that the software can identify them correctly.
C Cap ID Associated	Integer (up to 32 bits)	This is a unique value that allows the software to group IPs which share a common group in the firmware design, For example, an HDMI video timing generator and the HDMI TX core need to share the same ID for the software to know that they are associated in the same group (e.g., HDMI group) within the firmware design.

Third Party IP

The Omni Offset Capability provides the option to manually add third-party IPs which have an AXI4-Lite interface.

This feature requires modification of the header file which is used by the software to associate the firmware IP to the correct driver. This makes the software aware of its existence in the firmware domain.

Once the third-party IPs are added into the Omni Offset capability's header file, they can be associated with the Omni Offset capability using the manual mode (i.e., C Auto = 0).

For more information, refer to the *OmniTek FPGA Software Interface Framework* document which is delivered with OSVP [Ref 4].

Latency and Genlock

To achieve less than a frame of latency, your video output must be *genlocked* (framelocked) to the video input that you wish to achieve the low latency on.

Three IP cores are associated with minimizing latency through OSVP:

Deinterlacer

The Deinterlacer core buffers one line of input video before it begins outputting video pixel, except when *Edge* Processing is enabled in which case five lines of video are buffered before it begins outputting video pixels.

Vertical Resizer

The Vertical Resizer core buffers the number of vertical taps per two lines of input video before outputting video pixels. The total number of vertical taps is set in software by writing to a register and is limited by the number of DSPs configured in the OSVP GUI.

The equation used is DSP_TAPS = (total_taps/NUMBER OF DSP)-1. `DSP_TAPS` must be 1 or greater.

Note: Setting a low number of vertical taps will achieve a lower latency but it can also reduce the quality of the resized image. Setting a high number of vertical without sufficient DSPs may result in the resizer being unable to keep up with the required throughput.

The recommended number of taps for the required resizing is calculated by the OSVP software API. That value can then be used to calculate the latency through OSVP for that particular input to output configuration.

Frame Sync

The Frame Sync core buffers the frame of video in memory. When this is configured to be less than one, the latency is set by the read and write latency through the MIG and SDRAM. In most cases the Frame Sync can be bypassed using the OSVP Core's internal crosspoint, in which case the latency through SDRAM is not a factor.

Genlock System Architecture

Genlocking aligns the start of each output video frame to the start of each input video frame (or some configurable offset from the start of each input video frame).

This section describes how to Genlock the video output of OSVP to the timing of a video input.

The output video timing of OSVP is controlled by the OmniTek Video Timebase and the output pixel clock. The OmniTek Video Timebase adds horizontal and vertical blanking to the raw pixel data from OSVP, which, along with the output pixel clock, dictates the frame period of the video output.

Because the output and input video pixel clocks are from different sources, even if the total output frame size, including blanking, is configured to be the same as the input frame size, and the output pixel clock frequency is the same as the input pixel clock frequency, the frame periods of the video input and output will still be different.

This is illustrated in Figure 6.

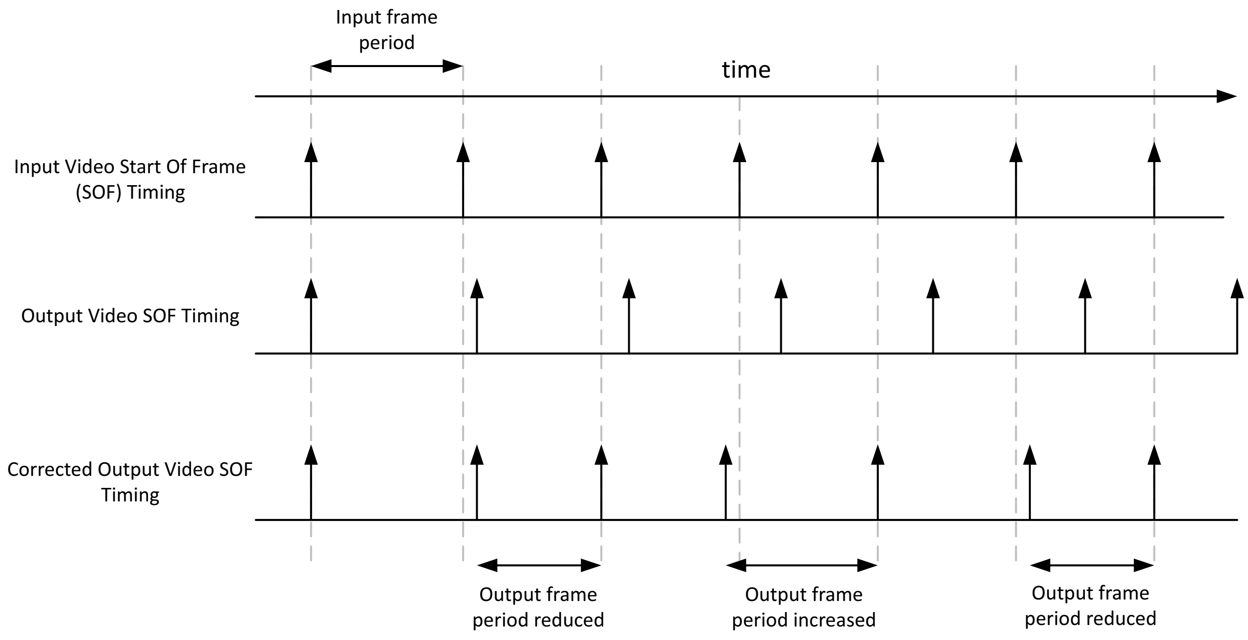

Input frame period
time
Input Video Start Of Frame (SOF) Timing
Output Video SOF Timing
Corrected Output Video SOF Timing
Output frame period reduced
Output frame period increased
Output frame period reduced

Figure 6: **Input and Output Frame Periods**

The difference in frame period caused by clock drift can be corrected in two ways:

1. Adjust the output clock frequency or phase to track the input clock ensuring they don't drift apart. This is the method used by the External Pixel Clock Generation Chip and On FPGA PICXO options shown in Figure 7 and is referred to as *Hard Locking*.

   Refer to *3G/HD/SD Video Clock Generator with Audio Clock* (LMH1983) [Ref 2] for more information on the LMH1983 video clock generator, which is used on OmniTek's OZ745 board.

   Refer to *All Digital VCXO Replacement for Gigabit Transceiver Applications (7 Series/Zynq-7000)* (XAPP589) [Ref 3] for more information on Xilinx's PICXO solution.

2. Frame Synchronization without adjusting the output pixel clock. This is the method used by the Asynchronous Pixel Clock as shown in Figure 7. It can take one of to forms:

   a. Dropping or repeating frames. This ensures that the video output does not drop and works for all input and output combinations but can also introduce unnecessary judder.

   b. Dropping or repeating of blanking pixels. This doesn't introduce judder but only works when the clock drift is minimal. (This method is referred to as *Soft Locking*.)

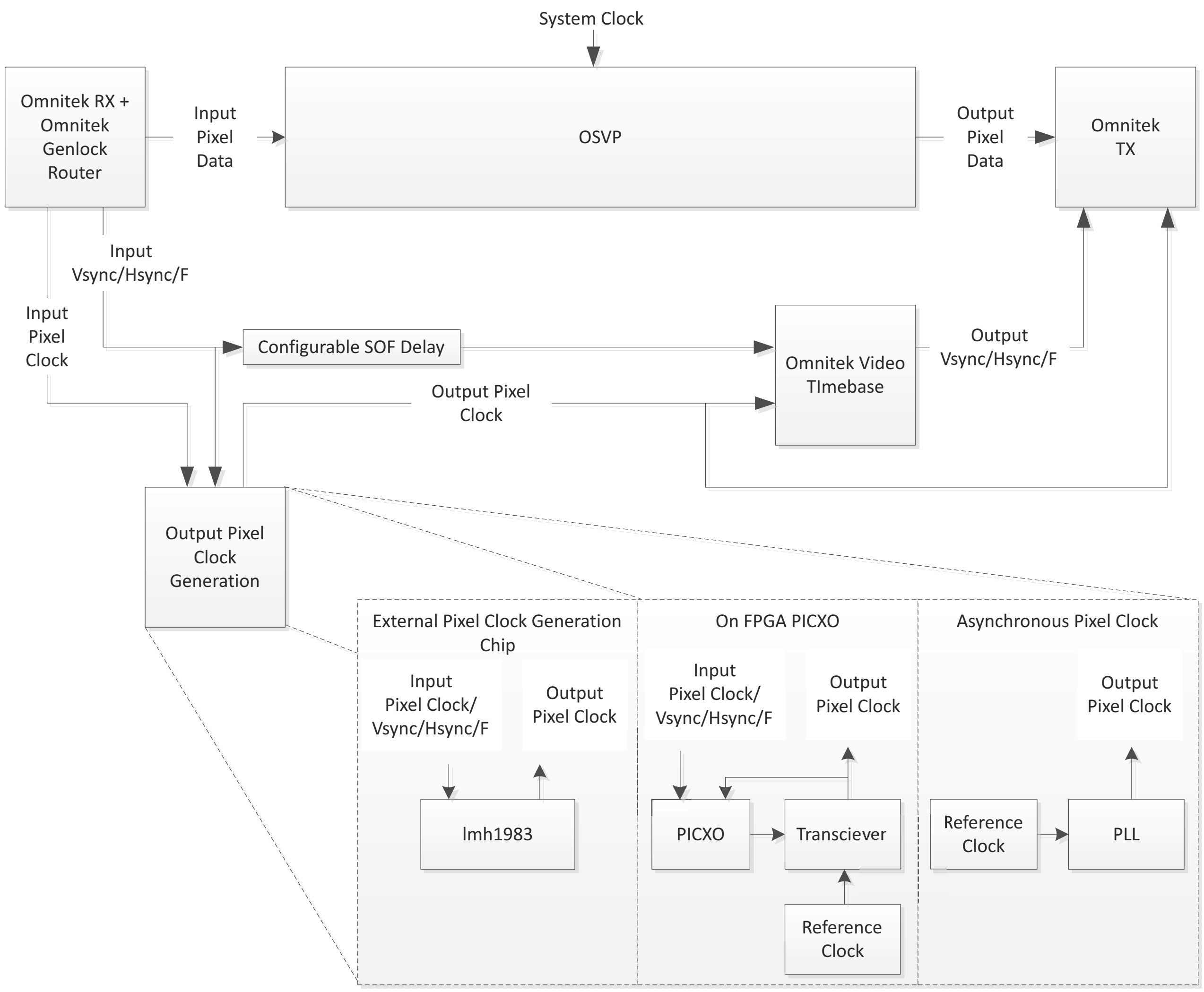


Figure 7: **System Architecture**

IP Configuration

There are two methods of configuring the OmniTek IP:

- **Hard Locking** - Output pixel clock is synchronized with the input pixel clock.
- **Soft Locking** - Output and input clocks are asynchronous and blanking pixels are dropped or repeated to compensate for frame drift.

Both methods require a `genlock_tgl` signal to be connected to the OmniTek Video Timebase which uses this signal to synchronize the output vertical sync, horizontal sync and field signals to.

The `genlock_tgl` signal can be generated by feeding the input video synchronization signals into the OmniTek Genlock Router.

Figure 8 shows the Genlock toggle signal form. The signal toggle at the start of each input frame. The signal is synchronous to the input pixel clock and must be synchronized to the output pixel clock, the clock domain in which the output video synchronization signals are generated by the OmniTek Video Timebase.

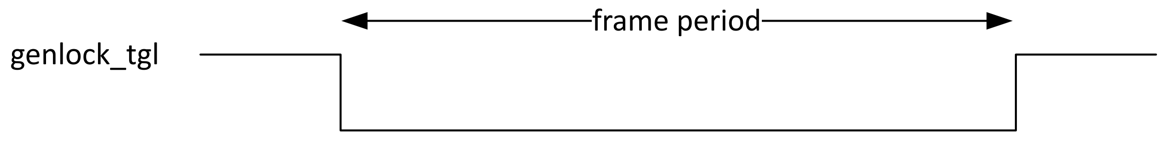


Figure 8: **Genlock Toggle Signal example**

This signal can be delayed to produce output video which is a fixed time offset behind the input video, as shown in Figure 9.

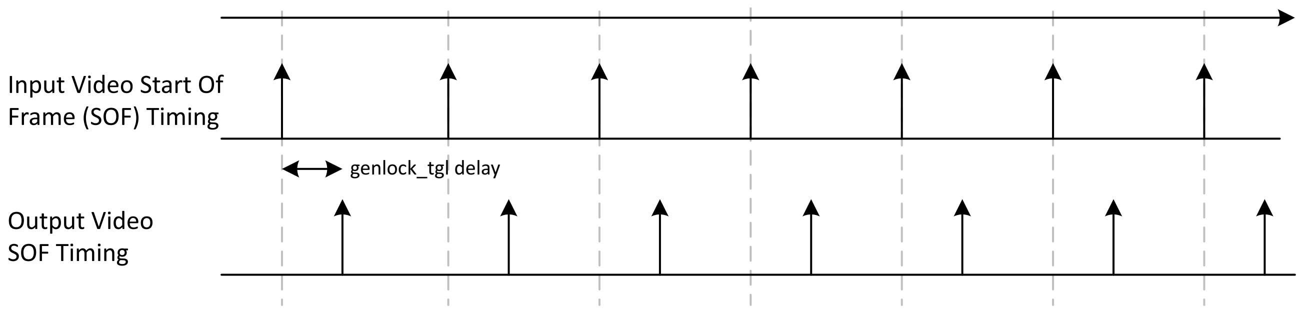


Figure 9: **Genlock_tgl example**

Hard Locking

In *Hard Locking* configuration, the OmniTek Video Timebase accepts a configurable amount of drift in the `genlock_tgl` signal before resynchronizing the output video timing. This is because the output video clock will not track the input video clock exactly and there may be some drift before it is aligned.

The OmniTek Video Timebase has a window centered around the output video Start Of Frame (SOF).

If the `genlock_tgl` signals changes *inside* this window, the OmniTek Video Timebase does nothing.

If the `genlock_tgl` signal change *outside* of the window, the OmniTek Video Timebase immediately resets the output video timing to align the SOF to the `genlock_tgl` signal. The window can be +/- 7 pixels.

OmniTek Video Timebase registers associated with *Hard Locking* are shown in Table 3 and Table 4.

Table 3: **Genlock Control (Register 10)**

Bit Location	Register Description	Attributes
0	`Genlock Enable` 1 = The Genlock will reset the Timebase counter when a Genlock toggle arrives which is not aligned with the previous Genlock toggle.	WO
1	`Genlock One-Shot` (for debug) 1 = The Genlock will only attempt to re-sync the Timebase counters once. After this you must re-enable the Genlock circuit to force a new re-sync.	WO
2	`Genlock No Window` A lock-out window is used for the input Genlock to capture jitter. During this window, the Genlock controller will not re-sync the Timebase as it assumes it is already locked. This window is between +/-1 to 7 pixels.	WO
31:3	Reserved	RO

Table 4: **Genlock Lockout Window (Register 11)**

Bit Location	Register Description	Attributes
3:0	`Genlock Window` Total window size	WO
31:24	Reserved	RO

Soft Locking

In *Soft Locking* mode, the OmniTek Video Timebase should be configured to have no window (see bit-2 of Genlock Control (Register 10)).

The Timebase then resets the output video timing to align the SOF to the `genlock_tgl` signal on every frame. This, in turn, causes the output video timing to jump from one area of video blanking to another, resulting in a number of video blanking pixels being added or removed from the total frame size, compensating for output pixel clock drift.

The OmniTek Video Timebase must be configured to have one extra line of blanking than is necessary (e.g., a total number of 1125 lines becomes 1126). This allows output video frames to overrun and add extra video blanking pixels if the `genlock_tgl` is late.

OmniTek Video Timebase registers associated with *Soft Locking* are shown in Table 5 and Table 6.

Table 5: **Registers 0 & 1**

Version Added	Offset	Name	Purpose
1.0	0	HTOTAL	Total number of pixels in a line (e.g., 2200).
1.0	1	VTOTAL	Total number of lines in a video frame (e.g., 1125).

Table 6: **Genlock Total Pixels**

Bit Location	Register Description	Attributes
	`Total pixels per frame` Must be programmed with HTOTAL*VTOTAL - 1 Used by the Genlock to determine if the Genlock sync pulses coming in on the `genlock_Tgl` signal are stable.	WO
31:24	Reserved	RO

Software Application

RTVE Application

The RTVE application architecture is shown in Figure 10.

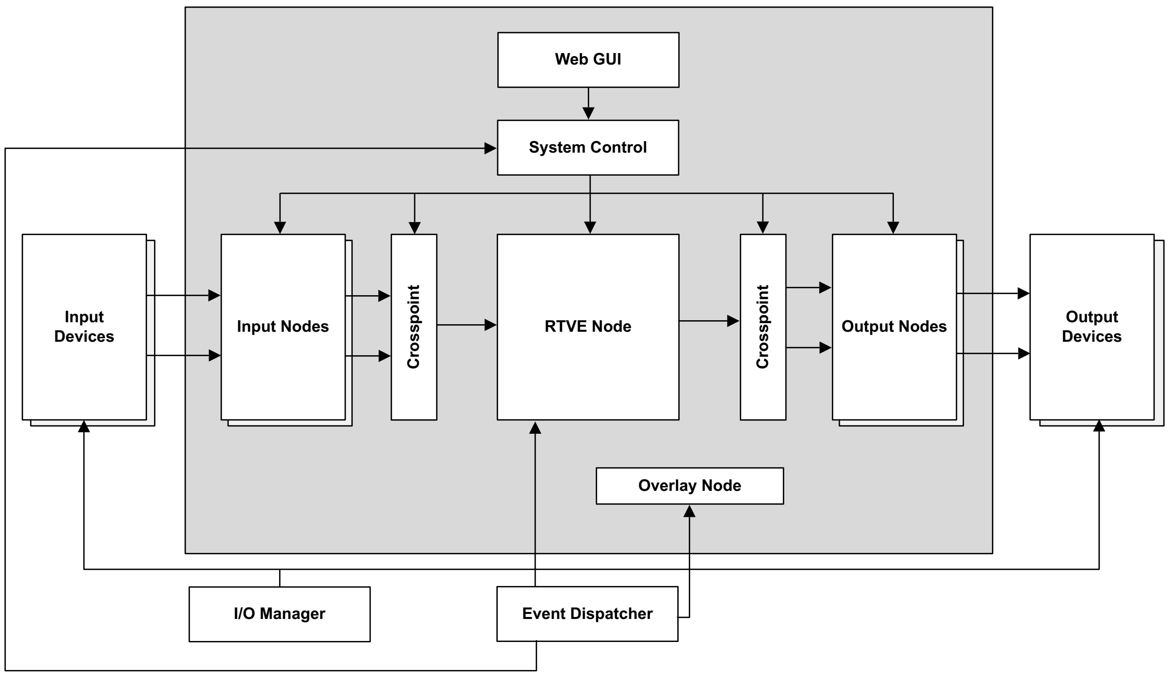


Figure 10: **RTVE 3.1 Application Architecture**

Processor Subsystem Software Design

The architecture of the OmniTek OZ745 Video development board is illustrated in Figure 11.

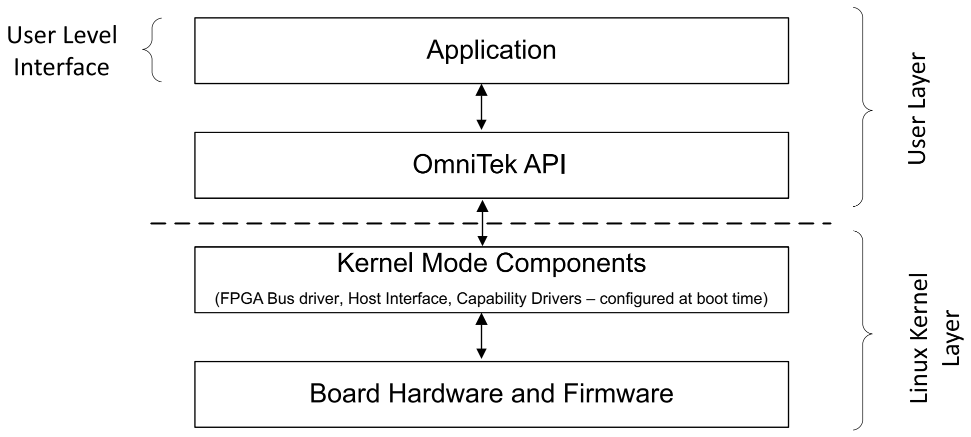

User Level Interface
Application
OmniTek API
User Layer
Kernel Mode Components
(FPGA Bus driver, Host Interface, Capability Drivers – configured at boot time)
Board Hardware and Firmware
Linux Kernel Layer

Figure 11: **RTVE 3.1 Video Architecture**

OmniTek API

The OmniTek API provides a Cross-Platform working environment.

It ensures that the interface presented to the application running on the development board is consistent, regardless of any changes to the facilities offered on the board.

As a result, the same source code can be used for any application running on the board even if the underlying firmware is changed. The code is also usable if the interface to the OZ745 board change to a PCIe bus or the operating system change from Linux to Windows.

This is achieved by establishing a consistent structure and nomenclature for the capabilities offered by the OmniTek IP Blocks implemented in the Zynq PL.

This consistency allows the API at boot time to:

- Discover the OS
- Discover the host interface
- Discover board firmware capabilities
- Create appropriate Kernel mode drivers

This offers two main advantages; the drivers automatically give the application a consistent interface to these components and you do not need to write any kernel mode code.

Note: The module at the top of the RTVE 3.1 design architecture is key to the operation of the OmniTek API. If this block cannot identify the OSVP Block within the firmware on the board, the assumption is made that the OmniTek firmware is not implemented on the board and the set-up process conducted by the OmniTek API is not run.

Video Input

The RTVE 3.1 supports input video streams in any of the formats listed in Table 7.

These formats are all software selectable via the RTVE software application.

In principle, the RTVE 3.1 input supports any combination of SDI, HDMI and Display Port inputs. However, some combinations of video standard cannot be used because they exceed the internal DDR3 bandwidth. For details, see Usage Limitations.

Table 7: **Supported Video Input standards**

Source	Video Standards Supported per Interface Type
SDI	SD PAL 720x576i YUV 4:2:2 50Hz SD NTSC 720x486i YUV 4:2:2 59.94Hz HD 1280x720p YUV 4:2:2 50/59.94/60Hz HD 1920x1080i YUV 4:2:2 50/59.94/60Hz 3G-Level A 1920x1080p YUV 4:2:2 50/59.94/60Hz 3G for 3840 x 2160 via 4 x 3G-Level A 1920x1080p YUV 4:2:2 50/59.94/60Hz: •Square Division also known as *Quad* UHD •2SI Two Sample Interleaved 6G 3840x2160p YUV 4:2:2 25/29.97/30Hz 12G 3840x2160p YUV 4:2:2 50/59.94/60[1]
HDMI	1920x1080p 50/59.94/60Hz 480p 50/60 Hz 576p 50/60 Hz
Test Pattern Generator	Static Images from .BMP or .YUV files, ranging in size from 320x200 up to 3840x2160 with *even* valued ranges for X and Y dimensions. Any image less than 320x200 will be padded with black up to 320x200.

1. When using 12G-SDI Rx on the FMC card, the following considerations need to be taken into account:
   - If 12G-SDI Rx is configured to work in 59.94 mode, the 12G-SDI Tx will only work in 59.94 mode.
   - If 12G-SDI Rx is configured to work in 50 or 60 mode, the 12G-SDI Tx can be configured to work in 50 or 60 mode, including any of the four possible combinations between SDI Rx and SDI Tx (i.e., 50&50, 50&60, 60&50 and 60&60).

Video Output

The RTVE 3.1 generates both an OSVP video stream and a Linux video stream, both of which can be selected for output.

OSVP Output (Primary Output)

The OSVP output is encoded as 10-bit 4:4:4 video, but can be converted to any of video output standards detailed in Table 8. The transformations carried out include down-conversion to 4:2:2 where required to match the selected output video format.

Table 8: **Supported Video Output standards**

Source	Video Standards Supported per Interface Type
SDI Port	SD PAL 720x576i YUV 4:2:2 50Hz SD NTSC 720x486i YUV 4:2:2 59.94Hz HD 1280x720p YUV 4:2:2 50/59.94/60Hz HD 1920x1080i YUV 4:2:2 50/59.94/60Hz 3G-Level A 1920x1080p YUV 4:2:2 50/59.94/60Hz 3G for 3840 x 2160 via 4 x 3G-Level A 1920x1080p YUV 4:2:2 50/59.94/60Hz: •Square Division' also known as 'Quad'4K •2SI, Two Sample Interleaved 6G 3840x2160p YUV 4:2:2 25/29.97/30Hz 12G 3840x2160p YUV 4:2:2 50/59.94/60[1]
HDMI Port	1280x720p YUV 4:2:2 50/59.94/60Hz 1920x1080p YUV 4:2:2 50/59.94/60Hz
DisplayPort 1.2	1280x720p 50/60Hz 1920x1080p 50/60Hz 3840x2160p 50/60Hz

1. When using 12G-SDI Tx on the FMC card, the following considerations need to be taken into account:
   - If 12G-SDI Tx is configured to work in 59.94 mode, the 12G-SDI Rx will only work in 59.94 mode.
   - If 12G-SDI Tx is configured to work in 50 or 60 mode, the 12G-SDI Rx can be configured to work in 50 or 60 mode, including any of the four possible combinations between SDI Rx and SDI Tx (i.e., 50&50, 50&60, 60&50 and 60&60).

Differences in output/input frame rates are resolved by either frame repeat or frame drop. For instance, de-interlacing and scaling a 480i 59.94 Hz input to a 720p60 or 1080p60 output will result in frames being repeated. Similarly if the output format is set at 720p50 or 1080p50, frames will be dropped.

Linux Frame Buffer Rendering (Secondary Output)

The Linux Frame Buffer located in the dedicated DDR memory space of the Zynq SoC's processor subsystem (PS) provides a secondary output video stream which gives you an interactive, statically positioned display showing:

- The RTVE 3.1 Application GUI (a Linux Web-Client)
- A Qt Demonstration image

Internally, the Linux Frame Buffer always generates 1920 x 1080 8-bit RGBA. This is not programmable.

If the OSVP is outputting a higher resolution (e.g., 3840 x 2160), the Linux output will be upscaled to 3840x2160 before blending with the OSVP output.

Handling of Quad Link 3G for UHD Video and Latency Consequences

RTVE 3.1 supports 2160p video via *2SI* (two sample interleaved) and *Square Division* modes on both input and output.

Square division mode requires the video to be bounced through SDRAM in order to *de-quad* the video prior to OSVP processing. This process adds a frame of latency.

Similarly, square division output requires an additional SDRAM bounce in order to reorder the video as the quadrants required for square division. This also adds a frame of latency.

The RTVE3.1 design has an input to output latency of approximately three frames to account for the worst case scenario described above. This can be significantly reduced for other design implementations.

Usage Limitations

Due to DDR3 bandwidth limitations as well as other system logistic issues, the RTVE 3.1 does not support certain usage models for every supported video standard.

The main limitations are:

- UHD 60Hz input to UHD 60Hz full screen output is supported on channels 1 and 2, only if channels 3 and 4 are not used.
- RTVE 3.1 can have only one physical output active at any one time. It is not possible to duplicate a video stream across two or more outputs.

System Control

Video processing is driven by the system control block using settings defined via the web GUI and Overlay nodes.

The System Control block and the Overlay Node are shown in Figure 10.

The configuration of the Input and Output Devices is controlled by the separate I/O Manager, which is also responsible for identifying the video standard of the video stream on any input and adjusting the setup of the input devices accordingly.

Input/Output configuration affects how the RTVE and Overlay Nodes are configured. These settings also affect various aspects of the System Control.

The I/O Manager needs to adjust the setup of the Input/Output Devices only when the application is first launched or, when the video standard of an input changes.

The Event Dispatcher is notified of changes and, in turn advises the RTVE Node, the Overlay Node and the System Control that consequent changes may be required to adjust to the changes which have occurred.

SD Card Image

The OZ745 Development Kit includes an SD card which is described in Figure 12.

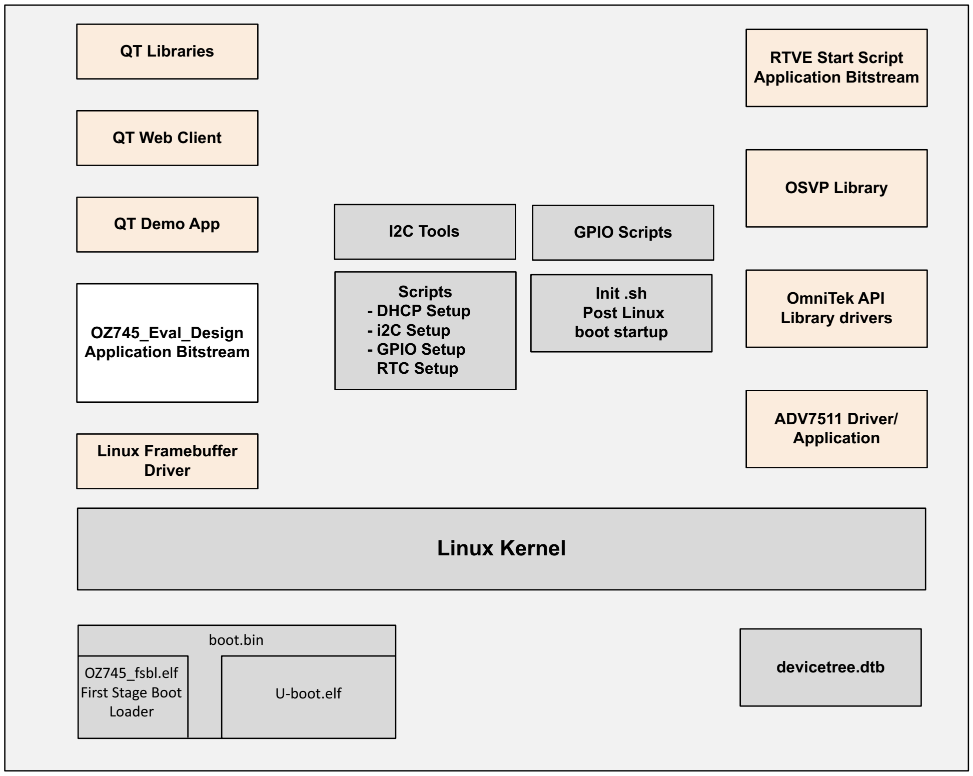

QT Libraries
QT Web Client
QT Demo App
OZ745_Eval_Design Application Bitstream
Linux Framebuffer Driver
I2C Tools
GPIO Scripts
Scripts - DHCP Setup - i2C Setup - GPIO Setup RTC Setup
Init .sh Post Linux boot startup
RTVE Start Script Application Bitstream
OSVP Library
OmniTek API Library drivers
ADV7511 Driver/ Application
Linux Kernel
boot.bin
OZ745_fsbl.elf First Stage Boot Loader
U-boot.elf
devicetree.dtb

Figure 12: **OZ745 SD Card Image**

The items shown in gray are referred to as the *Base* components of the image. These are responsible for booting the hardware and setting up the operating system.

Other items relate to the two reference designs supplied for the OZ745:

- A copy of the RTVE design
- An Evaluation Reference Design which demonstrates, among other things, how to use the QT graphics software which is delivered as part of the OZ745 Board Support Package. For further information about the Evaluation Reference Design, see *OZ745 Evaluation Reference Design (ERD) Guide* (OZ745_ERD_Guide) [Ref 1].

The version of RTVE design delivered on the SD card depends on when the OZ745 kit was purchased. The latest kits will be shipped with RTVE 3.1, but earlier kits may have been shipped with RTVE 3.0 or RTVE 2.1. Updated versions of the SD card image are available from the OmniTek website (registration required).

Depending on the version of the OZ745 you are using, (Evaluation, Standard or Full), various components on the SD image may be supplied as either encrypted source code or full source code. Various build scripts, EDK project and Software project files may also be provided - again depending on the version of the OZ745 you have bought. For more information, see Licensing, page 24.

Tool Flow and Verification

The following checklist indicates the tool flow and verification procedures used for the provided reference design.

Table 9: **Reference Design Checklist**

Parameter	Description
General	
Developer Name	Xilinx
Target Devices	Zynq-7000
Source code provided?	Y (Source level Licensing required)
Source code format (if provided)	VHDL
Design uses code or IP from existing reference design, application note, 3rd party or Vivado software?	OmniTek OSVP suite OmniTek Enhanced MIG OmniTek 12G-SDI Rx/Tx OmniTek HDMI Rx/Tx Xilinx DisplayPort Xilinx GTX 7series Transceiver
Simulation	
Functional simulation performed	Y
Timing simulation performed?	N
Test bench provided for functional and timing simulation?	Y (Source level Licensing required)
Test bench format	VHDL
Simulator software and version	ModelSim
SPICE/IBIS simulations	N
Implementation	
Synthesis software tools/versions used	Vivado synthesis
Implementation software tool(s) and version	Vivado Implementation
Static timing analysis performed?	Y (XDC)
Hardware Verification	
Hardware verified?	Y
Platform used for verification	OmniTek OZ745 development kit [3]

Requirements

Hardware

- One OmniTek OZ745 Zynq SoC Video Development Kit

  Note: If DisplayPort output is required, an OmniTek 12G/DP OZ745 FMC expansion card needs to be fitted to the OZ745 board
- A DisplayPort/HDMI or SDI monitor supporting at least 720p@60 Hz(1280x720p@60) plus video cable.
- A 1080p60 or 1080p50 HDMI or SD/HD/3G SDI Video Source (optional)
- One Ethernet router (providing DHCP server)
- One Ethernet cable
- One SD card loaded with the OZ745 board support package
- A PC with a web browser (Microsoft Internet Explorer, Google Chrome and Mozilla Firefox are supported)
- Linux build machine (CentOS 6.4 64-bit recommended)

Software

- Vivado 2014.3 and later
- Eclipse (https://eclipse.org/cdt/downloads.php)

Reference Design Files

A prebuilt RTVE 3.1 reference design is provided for evaluation purposes. The board support package is located in: `reference_design/oz745_sdcard_rtve_3_1_9_0.zip`.

Unzip the file and copy everything from the `sd_image` directory to an empty, pre-formatted secure digital (SD) card.

Licensing

The RTVE reference design is built based on an OmniTek optimized firmware package which includes an extensive set of VHDL source code for a variety of complex image processing functions, plus a software API and drivers.

You can access a timed out bitstream or a source encrypted RTVE project.

Additionally, if you want to get access to source code, it is possible to purchase licenses.

Four license levels are available as shown in Table 10:

Table 10: **Licensing Levels**

License Level	Description
Evaluation	Compiled software and firmware for running on board.
Extended Evaluation	Encrypted firmware source with timeout. Full source of Reference Design project and software.
Full License	Encrypted firmware source. Full source of Reference Design project and software.
Source License	Unencrypted firmware source. Full source of Reference Design project and software.

Nine Product options are available as shown in Table 11:

Table 11: **Product Options**

Product	Description
OSVP_SUITE	OSVP core offering 1–8 channels of up to 1080p resolution Comprises: • 4:2:2 to 4:4:4 and 4:4:4 to 4:2:2 Chroma resamplers • Colour matrix • Deinterlacer • Crop • Resize and frame synchronization • Also includes the Combiner and Interlacer
OSVP_ADV	Adds Support for: • Up to 4K60 Ultra HD • 6-axis colour correction • Noise reduction • Super-resolution image enhancement • Enhanced cadence detection • Low-angle de-interlacing
OSVP_120	Adds support for output frame rates up to 120Hz
OSVP_4:2:0	Adds 4:2:2 to 4:2:0 and 4:2:0 to 4:2:2 Chroma resamplers
2D_GRAPHICS	Adds an OS frame-buffer and 2D acceleration
SDI	Adds SD up to 12G-SDI I/O connectivity

Table 11: **Product Options**

Product	Description
HDMI2.0	Adds HDMI I/O connectivity
DP1.2	Adds DisplayPort I/O connectivity
OSVP_MIG	Adds MIG enhancement, enabling higher efficiency when using OSVP

Reference Design Steps

Setup

The hardware configuration depends on whether a DisplayPort monitor will be used to show the output from the RTVE. If a DisplayPort monitor is used, the OZ745 needs to be used with an added FMC.

The hardware setup showing all required wire connections for an OZ745 is shown on Figure 13.

The hardware setup showing all required wire connections for an OZ745 with additional FMC is shown on Figure 14.

The HDMI input and output ports are bound together on `J113` where the input connector is on the top and the output connector is on the bottom.

The OZ745 Ethernet port is used to connect to a LAN, which should have a DHCP-capable router so an IP address can be automatically assigned. The Control PC should also be connected to this LAN.

The SD card with the board support package should be plugged into the OZ745 SD card slot.

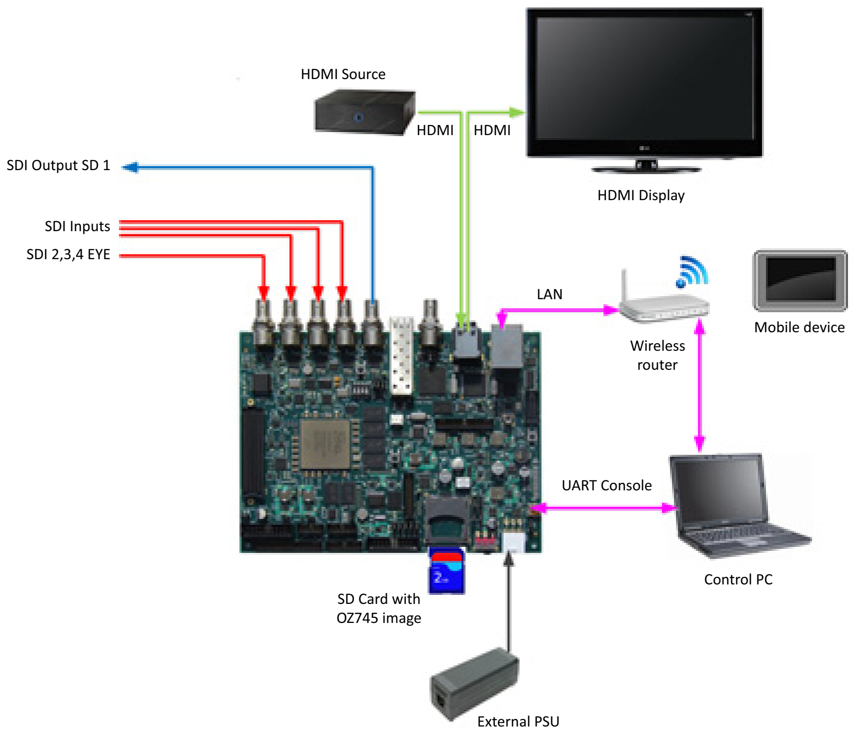


Figure 13: **OZ745 board**

Note: In the configuration shown in Figure 13, inputs can either be used individually or as a Quad link.

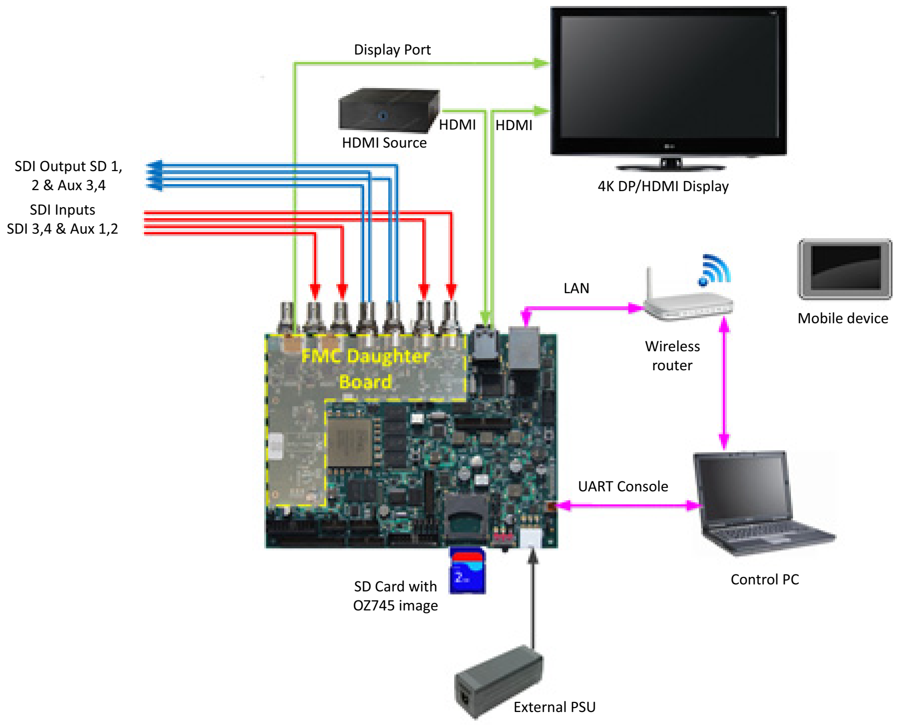


Figure 14: **OZ745 board with FMC**

Note: In the configuration shown in Figure 14, both Inputs and Outputs can either be used individually or as a Quad Link.

Running the Reference Design

Booting the Reference Design (HA Insert)

1. Add the OZ745 board to the existing LAN using the Ethernet port.
2. Connect the monitor to the appropriate output port.
3. Plug the HDMI input to the HDMI Input port.
4. Plug the SD card into the OZ745 SD card slot then
5. Power up the OZ745 board and allow to boot.
6. Call up the User Interface (UI):
   a. Open web browser on the Control PC
   b. Type the URL `http://oz745-000000000000` (replace 000000000000 in the URL with the MAC address of your board). The User Interface (UI) displays as shown in Figure 15.

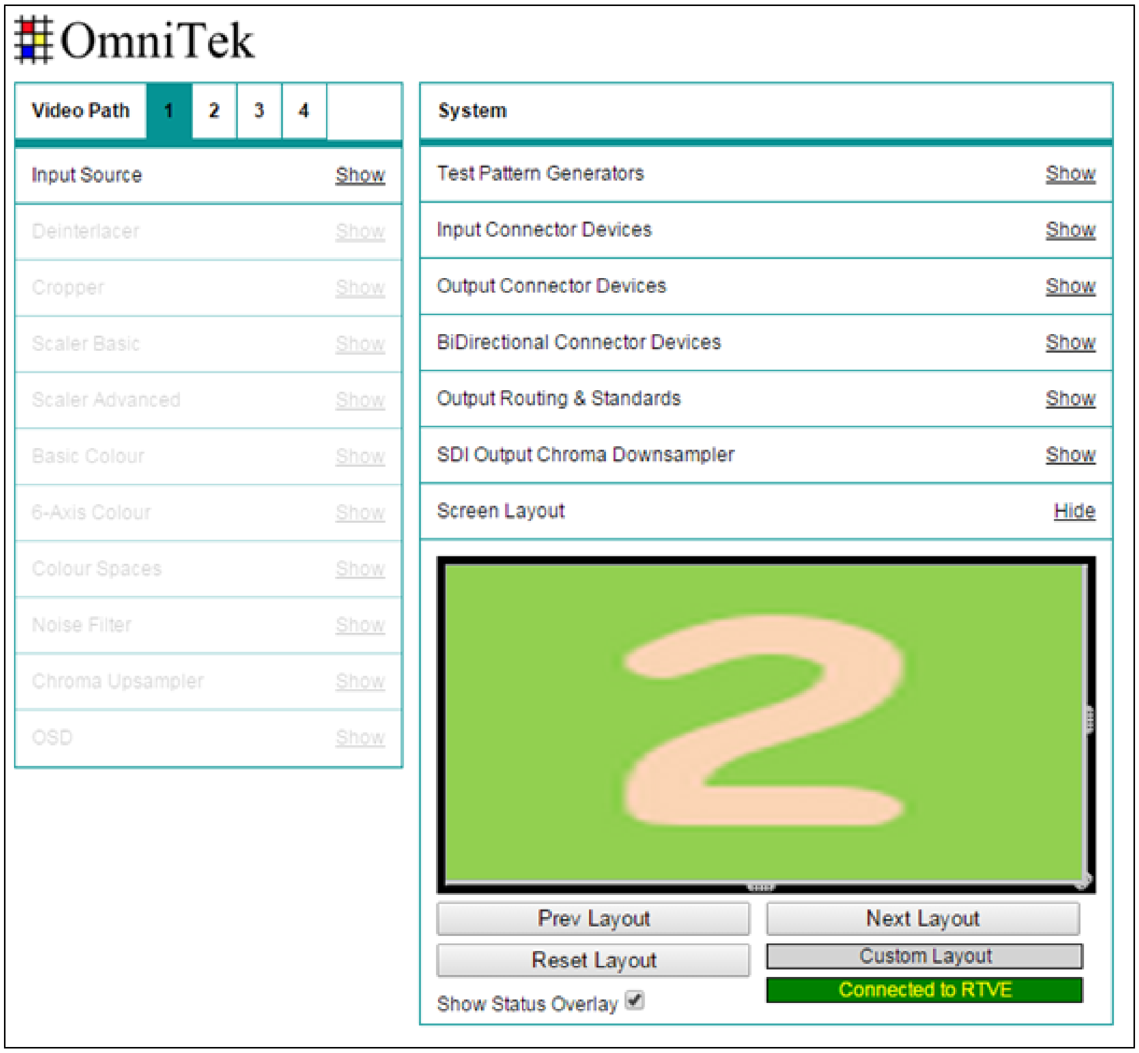
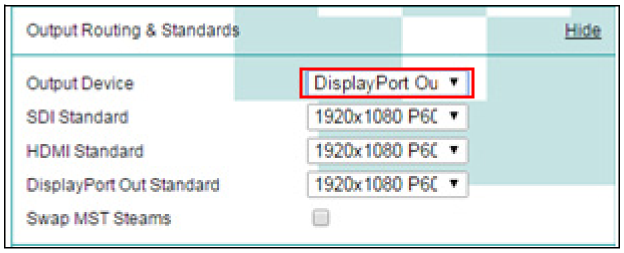


Figure 15: **RTVE User Interface**

As shown in Figure 15, the UI is split into two main sections; the left-hand display panel is used to configure video paths, the right-hand panel is used to configure overall system settings.

7. Click **Show** in the **Output Routing and Standards** section on the right panel. The **Output Routing & Standards options** are displayed as shown in Figure 16. Use the pull-down menu to select the appropriate Output Device (For example, if you are using a DisplayPort monitor, set the Output Device to Display Port Output).

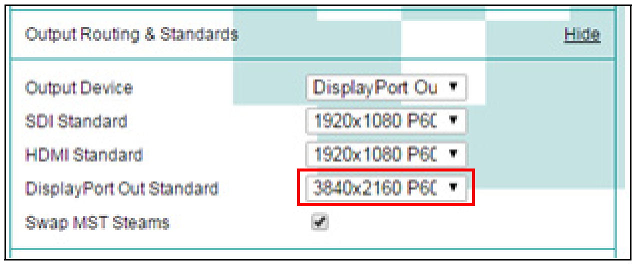

Output Routing & Standards Hide

Output Device	DisplayPort Ou
SDI Standard	1920x1080 P6(
HDMI Standard	1920x1080 P6(
DisplayPort Out Standard	1920x1080 P6(
Swap MST Steams	☐

Figure 16: **Output Routing & Standards**

System Status information should now be displayed on your monitor.

8. Set the appropriate SDI/DisplayPort Out Standard to 3840x2160 P60 as shown in Figure 17.



Output Device: DisplayPort Ou
SDI Standard: 1920x1080 P6C
HDMI Standard: 1920x1080 P6C
DisplayPort Out Standard: 3840x2160 P6C
Swap MST Steams: ✓

Figure 17: **Output Routing & Standards**

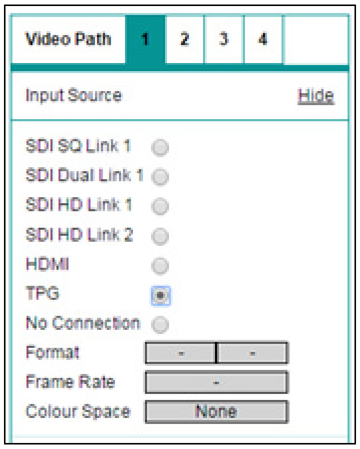
Note: The monitor should now lock to the OZ745. If the message **No Input Detected** is displayed, select DisplayPort Out Standard 3840x2140 P50.

9. If Status information is displaying on the right-hand display panel when using a DisplayPort monitor, select the **Swap MST Streams** check box.

Displaying Images

After booting the reference design, follow these steps to display images:

1. Click **Video Path 1** on the left-hand display panel.
2. Click **Show** beside Input Source. A list of possible sources displays.
3. Select the **TPG** option for this demonstration as shown in Figure 18. This selects the built in Test Pattern Generator.



Input Source: TPG
Format: - -
Frame Rate: -
Colour Space: None

Figure 18: **Input Source Selection**

4. Click **Show** on the right-hand panel beside Test Pattern Generators. A **Filename** drop-down menu displays a list of available TPGs.
5. Select Filename `TSA_3840_2160xxx` from the drop-down menu. The selected pattern is loaded and displays on the monitor.
6. Select **Video Path 2** on the left-hand panel.
7. Select **HDMI** as the input source.

Layouts

At the bottom of the right-hand display panel there are a set of Screen Layout options (see Figure 19). These allow you to select various layouts to compare schematics with what is shown on your monitor. The area highlighted in red provides a representation of the layout currently selected.

At this point in the demonstration, the screen should display a single video stream showing a large number **1** as shown in Figure 19. The **1** indicates that the current display is a full screen image of the input from **Video Path 1**.

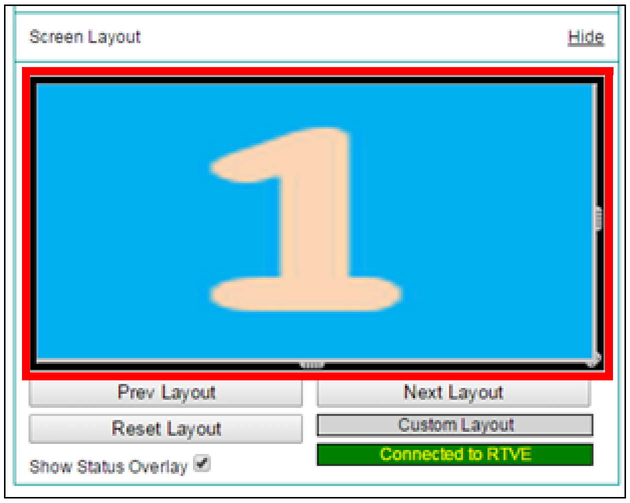


Figure 19: **Screen Layout Options**

1. Use the **Prev Layout** and **Next Layout** buttons to call up other provided layouts and compare the schematics shown with what is shown on your monitor.

   Note: Areas displaying numbers 2, 3 and 4 show the HDMI image. This is because when no input has been associated with a video path that is specified in a layout, RTVE 3.1 automatically substitutes video from another video path.

2. Use the **Show Status Overlay** check box to toggle show/hide Status Information. (All layouts show status information overlaid by default.)

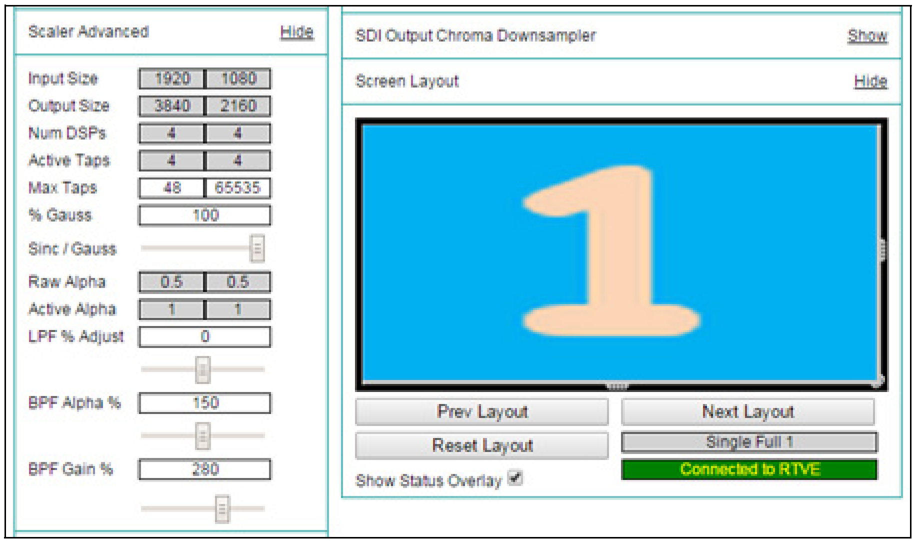
Scaling

Different layouts use the Test Pattern Image and the HDMI image at various sizes. You can view scaling details for a Video Path by:

1. Selecting the Video Path.
2. Click **Show** in the **Advanced Scaler** section in the left-hand panel of the display.

Note: As different layouts are selected, you will notice the **Output Size** changes.

See Figure 20 for reference.


Scaler Advanced Hide

Input Size 1920 1080
Output Size 3840 2160
Num DSPs 4 4
Active Taps 4 4
Max Taps 48 65535
% Gauss 100
Sinc / Gauss
Raw Alpha 0.5 0.5
Active Alpha 1 1
LPF % Adjust 0
BPF Alpha % 150
BPF Gain % 280

SDI Output Chroma Downsampler Show

Screen Layout Hide

Figure 20: **Scaler Advanced**

Scaling Algorithms

Different algorithms are used to resize images. Some algorithms give smoother images while some apply *Super-resolution* techniques to produce sharper images.

You can observe the effects different algorithms have on images by:

1. Select the screen layout **Single Fill 1**.
2. Use the Test Pattern Generator to select **TP_Vision_Sharpen_Image**. This is a 1980x1080 image.
3. Select **Video Path 1** in the left-hand panel.
4. Click **Show** in the **Scaler Basic** section.
5. Select different options from the drop-down menu of **Image Presets** (see Figure 21).

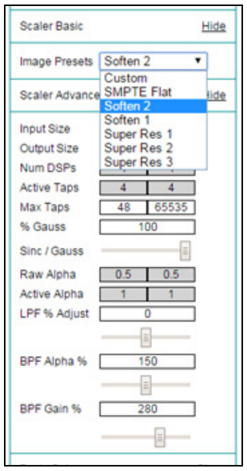

Scaler Basic Hide

Image Presets Soften 2

Scaler Advanced Hide

Input Size
Output Size
Num DSPs
Active Taps 4 4
Max Taps 48 65535
% Gauss 100
Sinc / Gauss
Raw Alpha 0.5 0.5
Active Alpha 1 1
LPF % Adjust 0
BPF Alpha % 150
BPF Gain % 280

Figure 21: **Image Presets**

The **SMPTE Flat** option selects the mathematically-correct interpolation algorithm. Other options select algorithms that produce varying degrees of softening/sharpening.

These selection correspond to settings within the **Scaler Advanced** section of the display.

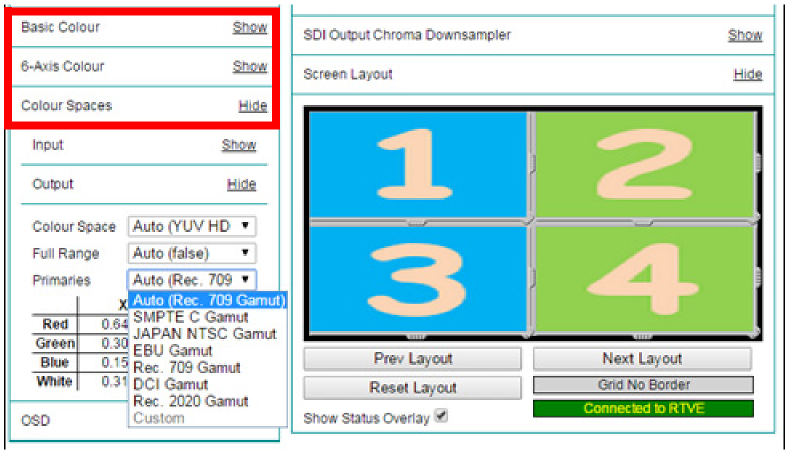
Selecting an option in the **Scaler Basic** section sets these parameters appropriately for the selected interpolation algorithm. Any subsequent changes you make within the **Scaler Advanced** section will modify the details of the algorithm applied.

Note: The **Custom** option is not available in release 3.1

Colour Manipulation

1. Select the Screen Layout **Grid No Border**.
2. Configure Video Path 2 to take input from **TPG**. The top two quadrants of the display should show identical images.

   Colour manipulation is performed in the left-hand panel in three different sections as shown in Figure 22.



Colour Space: Auto (YUV HD
Full Range: Auto (false)
Primaries: Auto (Rec. 709

Figure 22: **Colour Manipulation**

3. **Basic Colour**.
   a. Select **Show** in the **Basic Colour** section in the left-hand panel. Change settings for **Brightness**, **Saturation** and **Hue** and compare the top right-hand image on the monitor (the video path you are changing) against the top left-hand image (which does not change).
4. **6-Axis Colour**.
   a. Select **Show** in the **6-Axis Colour** section in the left-hand panel. Change settings for **Lift** and **Gain** for the different colour components.

      Note: Acceptable **Lift** values are between 0-100%, acceptable **Gain** values are between 0-200%.
5. **Colour Spaces**.
   a. Select **Show** in the **Colour Spaces** section in the left-hand panel. This shows colour space data for the output path of Video Path **2**.

Building the Reference Design

This section describes the steps required to build the firmware and software of the RTVE 3.1 reference design. To order the full set of reference design files, contact OmniTek or your Xilinx sales representatives.

See the Requirements section for a complete list of software and hardware requirements for this reference design.

Build Steps

1. Set the `VIVADO_HOME` environment variable to your Vivado 2014.4 bin directory (for example `C:\Xilinx\Vivado\2014.4\bin`).
2. Unzip the `reference_design/RTVE_3_1_OZ745_4X.zip` file.
3. Change to the `release/hw/rtve_3_1_oz745_4x` directory.
4. Run the `create_project.bat` (Windows) or `create_project.csh` (Linux) this creates the `rtve_3_1_oz745_4x.xpr` Vivado project.
5. Open `rtve_3_1_oz745_4x.xpr` using Vivado.
6. If you have purchased a licensed version of OSVP follow step 7, otherwise skip to step 12.
7. Click **Tools**->**Project Settings**.
8. Select the **IP** tab and click **Add Repository**
9. Browse to your licensed OSVP bundle and select the `release\hw\ip_catalog` directory.

   Note: You will receive a number of critical warnings stating the IP in the local ip_catalog will take precedence.
10. Click the move up arrow to force the licensed OSVP bundle to take precedence.
11. Click **OK**.
12. Check the **Critical Warning Messages** to ensure the licensed OSVP bundle takes precedence.
13. To review the design, click **IP Integrator**->**Open Block Design** in the **Flow Navigator**.

    The following blocks should be present:

    a. `video_in`: contains the IP for all the video inputs.

    b. `video_core`: contains OSVP, Combiner and other video processing IP.

    c. `cpu_word`: contains the CPU and supporting peripherals.

    d. `video_out`: contains the IP for all the video outputs.

    e. `gtx_core`: contains the transceivers used by `video_in` and `video_out` for video I/O.

    A block diagram showing each of the IP cores in the RTVE 3.1 design is shown in Figure 23.

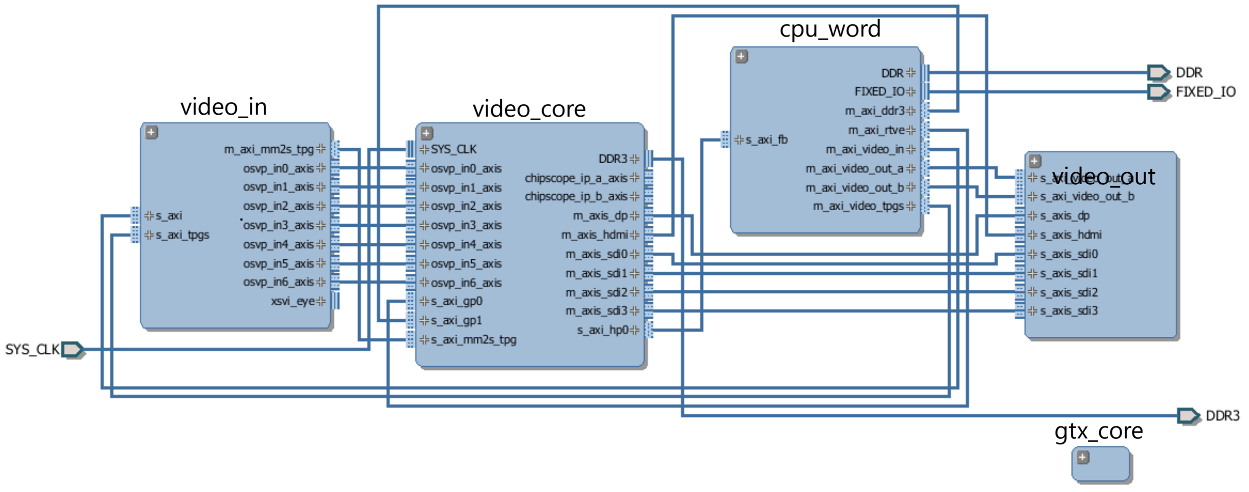


Figure 23: **RTVE 3.1 Block Diagram**

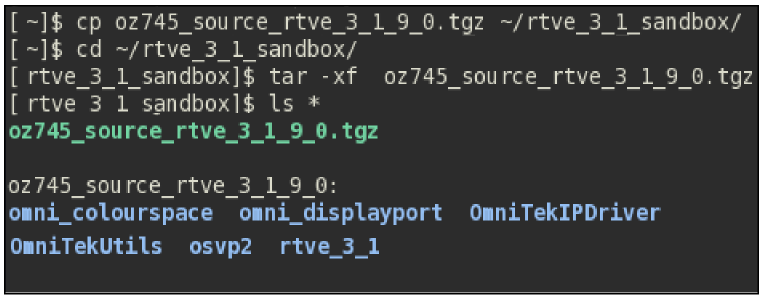
14. Click **Program and Debug**->**Generate Bitstream**. This builds the `rtve_3_1_oz745_4x.runs\impl_1\rtve_3_1_oz745_4x_top_wrapper.bit` file which has to be renamed `rtve_3_1\bit\rtve.bit` and copied to the SD card.

15. Power cycle the OZ745 board to load the new bitstream.

Building the Software (HA Insert)

See the Requirements section for a complete list of software and hardware requirements for this reference design.

1. Create a Sandbox; from a terminal type:
   a. **mkdir -p ~/rtve_3_1_sandbox**
   b. **cp reference_design/oz745_source_rtve_3_1_9_0.tgz ~/rtve_3_1_sandbox/**
   c. **cd ~/rtve_3_1_sandbox/**
   d. **tar -xf oz745_source_rtve_3_1_9_0.tgz**



```
[~]$ cp oz745_source_rtve_3_1_9_0.tgz ~/rtve_3_1_sandbox/
[~]$ cd ~/rtve_3_1_sandbox/
[rtve_3_1_sandbox]$ tar -xf  oz745_source_rtve_3_1_9_0.tgz
[rtve_3_1_sandbox]$ ls *
oz745_source_rtve_3_1_9_0.tgz

oz745_source_rtve_3_1_9_0:
omni_colourspace  omni_displayport  OmniTekIPDriver
OmniTekUtils  osvp2  rtve_3_1
```

Figure 24: **Create Sandbox**

2. Create an Eclipse workspace.

   Note: These instructions assume that the Eclipse binary is in your path and the Xilinx Toolchain is installed in the recommended location. In this example, a 64-bit machine is used with just the SDK installed. For other configurations, you may need to adjust the paths in the following steps to reflect your installation.

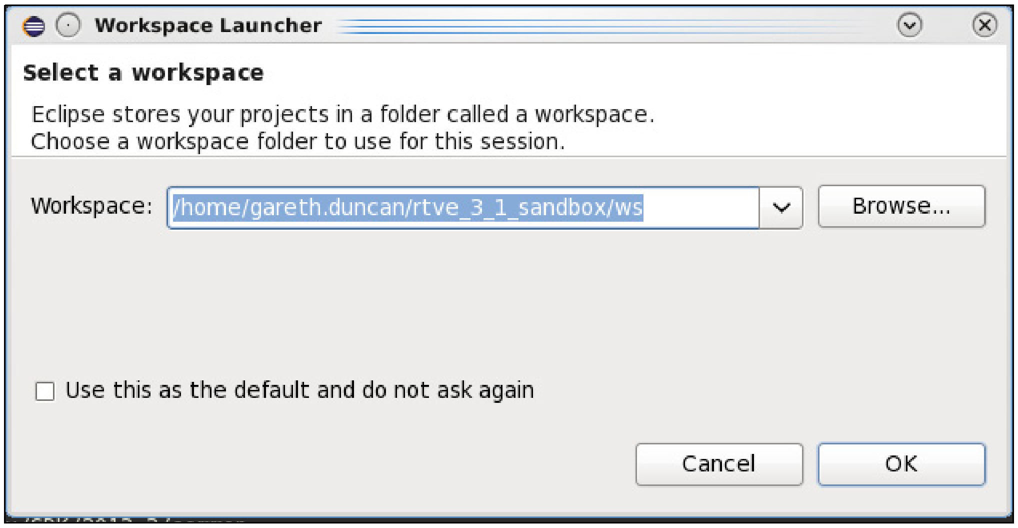
   From a terminal type:

   a. source /opt/Xilinx/SDK/2013.3/settings64.sh && unset LD_LIBRARY_PATH

   b. eclipse

   Eclipse launches.

   c. In the Workspace field type **/home/<user name>/ rtve_3_1_sandbox/ws** as shown in Figure 25.



Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

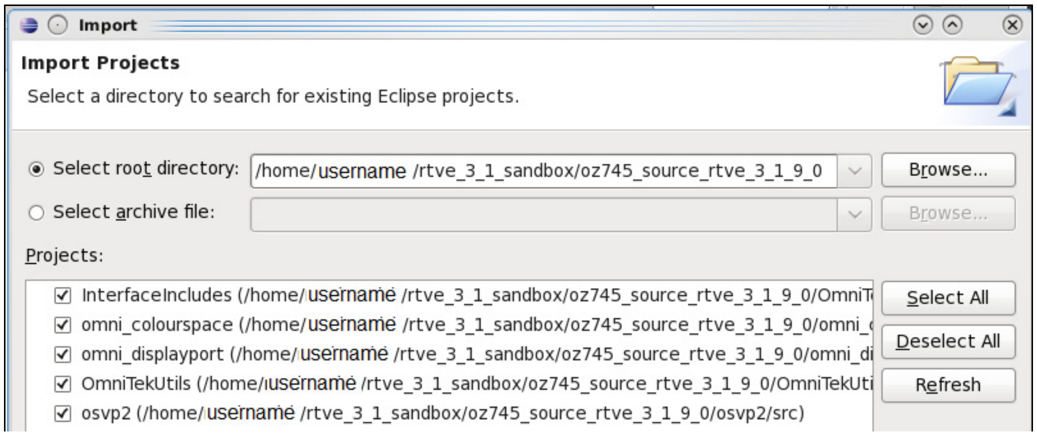
Workspace: /home/gareth.duncan/rtve_3_1_sandbox/ws

☐ Use this as the default and do not ask again

Figure 25: **Eclipse Workspace launcher**

   d. Select **Project** -> **Build Automatically** to disable the Eclipse project auto building.

   e. Select **File** -> **Import** -> **General** -> **Existing Projects** into Workspace (see Figure 26). This imports the sandbox into the workspace.



Select a directory to search for existing Eclipse projects.

● Select root directory: /home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0
○ Select archive file:

Projects:

☑ InterfaceIncludes (/home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/OmniT
☑ omni_colourspace (/home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/omni_
☑ omni_displayport (/home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/omni_di
☑ OmniTekUtils (/home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/OmniTekUti
☑ osvp2 (/home/username /rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/osvp2/src)

Figure 26: **Import Existing Projects**

f. Browse to the extracted source location and select **Finish**.

3. Build the Workspace:

   a. Verify the active build configuration of each project is set to `arm_linux_debug`. This is the default.

      i. Right click on a project and select **Build Configurations** -> **Set Active** -> `arm_linux_debug`.

   b. Build each of the projects in the following order:

      i. omni colourspace

      ii. omni displayport

      iii. omniTekUtils

      iv. osvp2

      v. RTVE 3.1

4. Run the OZ745 board:

   a. Copy the new RTVE binary to the SD card. The RTVE binary is located at: `~/rtve_3_1_sandbox/oz745_source_rtve_3_1_9_0/rtve_3_1/src/arm_linux_debug/RTVE`

      and should be copied to:

      `/mnt/rtve_3_1/RTVE`.

   b. Power cycle the OZ745 board. The program will automatically run.

      Note: To disable automatically launching the RTVE program, remove the `autorun.sh` file from the SD card.

   c. To run the RTVE program multiple times without a reboot, the `rtve_3_1/scripts/rtve.sh` file on the SD card needs to be modified. To do this, comment out the line which launches the binary in the start routine. You can then manually run the RTVE:

      i. Via SSH or serial console, navigate to `/mnt/rtve_3_1`:

      ii. Type **scripts/rtve.sh start.** This programs the FPGA, loads the drivers, and sets up links for the shared libraries.

      iii. Type **./RTVE**. This runs the application. Press **Ctrl+d** at any time to perform a graceful shutdown or **Ctrl+c** to kill the application.

5. Eclipse Remote Debugging:

   a. Determine the IP address or hostname of the OZ745 board (type hostname on the console).

   b. Load the firmware and drivers but do not run the RTVE application (see step 4.C).

   c. Start Eclipse.

   d. Right click on the RTVE 3.1 project.

      i. Select **Debug As** ->**Debug Configurations**

      ii. Double click on **C/C++ Remote Application**

      iii. Click **Search Project...**, select the RTVE binary and click **OK**.

      iv. Add a new connection.

         1. Click **New**.

         2. Select **SSH Only**.

         3. Set the **Hostname** field to be either the IP address or the host name obtained earlier

         4. Click **Finish**.

      v. Set the **Remote Absolute File Path for C/C++ Application** as `/mnt/rtve_3_1/RTVE`

      vi. Set **Command to execute before application** to `cd /mnt/rtve_3_1 && LD_LIBRARY_PATH=/mnt/rtve_3_1/lib`

      vii. Switch to the **Debugger** tab.

      viii. Change **GDP Debugger** to `arm-xilinx-linux-gnueabi-gdb`.

      ix. It is recommended to create (and set) a `GDB command file`:

         1. Create a .gdbinit with the following line: **set sysroot /opt/Xilinx/SDK/2013.3/gnu/arm/lin/arm-xilinx-linux-gnueabi/libc**

      x. Switch to the **Shared Libraries** tab.

      xi. Add `<extracted source location>/OmniTekIPDriver/lib` and `<extracted source location>/rtve_3_1/src/common/FreeImage` as paths to search

      xii. The **Debug** button is now usable. You will prompted for a username and password. These are both **root**.

Debugging

Table 12 provides a list of the most common problems when trying to bring-up the RTVE3.1 on the OZ745 development board.

Table 12: **Common bring-up problems**

Problem	Solution
The RTVE 3.1 is running but the monitor shows a dark-green screen.	Regardless of having an input video source connected to any of the input ports of the OZ745, by default the RTVE 3.1 provides an overlay image when is running. Therefore, make sure that the correct video output (i.e, DisplayPort, HDMI or SDI) is selected on the GUI. This be found in the **Output Routing & Standards** section of the GUI.
An SDI input source is driving the SDI 1 input port, but the RTVE3.1 does not detect it.	By default, the RTVE3.1 configures the SDI 1 and SDI 2 as output ports. Therefore, it is necessary to change their configuration through the GUI. This can be done in the **Bidirectional Connection** Devices section of the GUI.
A 12G-SDI video stream is connected to the Eye input of the OZ745, but the channel # 3 on the RTVE does not detect the input video stream.	In the RTVE3.1, only Channel 1 and Channel 2 are capable of processing 12G/6G/3G/HD-SD video streams, whereas Channel 3 and Channel 4 are constrained to handle HD and SD video streams.
The video on the output screen starts flickering when a 12G-SDI video stream is driving Channel 1 in RTVE and Channel 2, 3 and 4 are driven by HD signals.	Due to DDR3 bandwidth limitations, no more than two RTVE channels can be active at the same time when a 12G video stream is driving one of the RTVE channels.

References

1. *OZ745 Evaluation Reference Design (ERD) Guide (OZ745_ERD_Guide)*
2. *3G/HD/SD Video Clock Generator with Audio Clock (LMH1983)*
3. *All Digital VCXO Replacement for Gigabit Transceiver Applications (7 Series/Zynq-7000)* XAPP589. http://www.xilinx.com/support/documentation/application_notes/xapp589-VCXO.pdf
4. *OmniTek FPGA Software Interface Framework.* http://omnitek.tv/guides

Revision History

The following table shows the revision history for this document.

Date	Version	Changes
12/03/2015	1.0	Initial release.

Please Read: Important Legal Notices