Application Note:
Video Switch Example Design
R 0.4

27th Feb, 2008

Authors	VY

APPROVAL

Name	Function	Organisation	Date	Signature

CONTACT INFO

iWave Systems Tech. Pvt. Ltd. 7/B, 29th Main, BTM Layout, 2nd Stage, Bangalore –560 076, India.	**Telephone**	+91-80-2668-3700 +91-80-2678-1643
	e-mail	mktg@iwavesystems.com
	Website	www.iwavesystems.com

DOCUMENT IDENTIFICATION

Project Name	
Document Name	
Document Home	
Revision No	Rev 0.4

DOCUMENT REVISION HISTORY

Revision	Date	Change Description	Author
0.1	5th Feb, 2008	First version	VY
0.2	8th Feb, 2008	Added more IP cores, showed SDRAM as not part of the FPGA, added implementation results for all cores	VY
0.3	26th Feb, 2008	Changed document title, shifted deinterlacer to immediately after video decoder interface in block diagram, added additional descriptive text	VY
0.4	27th Feb, 2008	Removed camera to video decoder path, added block diagrams and description of cores in index	VY

Table of Contents

List of Tables

1 Introduction

The Video Switch Example Design describes a method to apply various processing techniques on a digital video stream in order to meet the resolution, scaling, interlace or color space requirements of a video display. It has the flexibility of interfacing to multiple video input and output devices with the user able to select which input/output device pair to use for streaming.

The design illustrates the usage of the library of video processing blocks that are part of the iWave Systems portfolio of Intellectual Property cores specifically targeted at image/video processing applications in embedded devices. Fully tested as stand-alone designs or as part of successful customer projects, these cores implement commonly used interfaces, are user-programmable and can be easily integrated in your designs. The cores offer the following functions commonly used in current video systems:

- LCD Interfacing
- Camera Interfacing
- Video Decoder Interfacing
- Video Encoder Interfacing
- Chroma Resampling
- Color Space Conversion
- Video Scaling
- Deinterlacing
- SDRAM Controller

The example design employs all cores described above although each one is designed to be able to be used as a stand-alone unit (for more information on the cores, see the section *core descriptions*).

In the example design, video is input from a digital camera or video decoder as selected, and processed by the iWave IP cores inside the FPGA according to the requirements of the output device. The output device can be an LCD or video encoder as selected by the user.

Thus, the following video streaming implementations are possible using the example design: digital camera to LCD, Video Decoder to LCD or Video Decoder to Video Encoder.

2 Functional Description

Figure 1 illustrates the various modules configured in the FPGA.

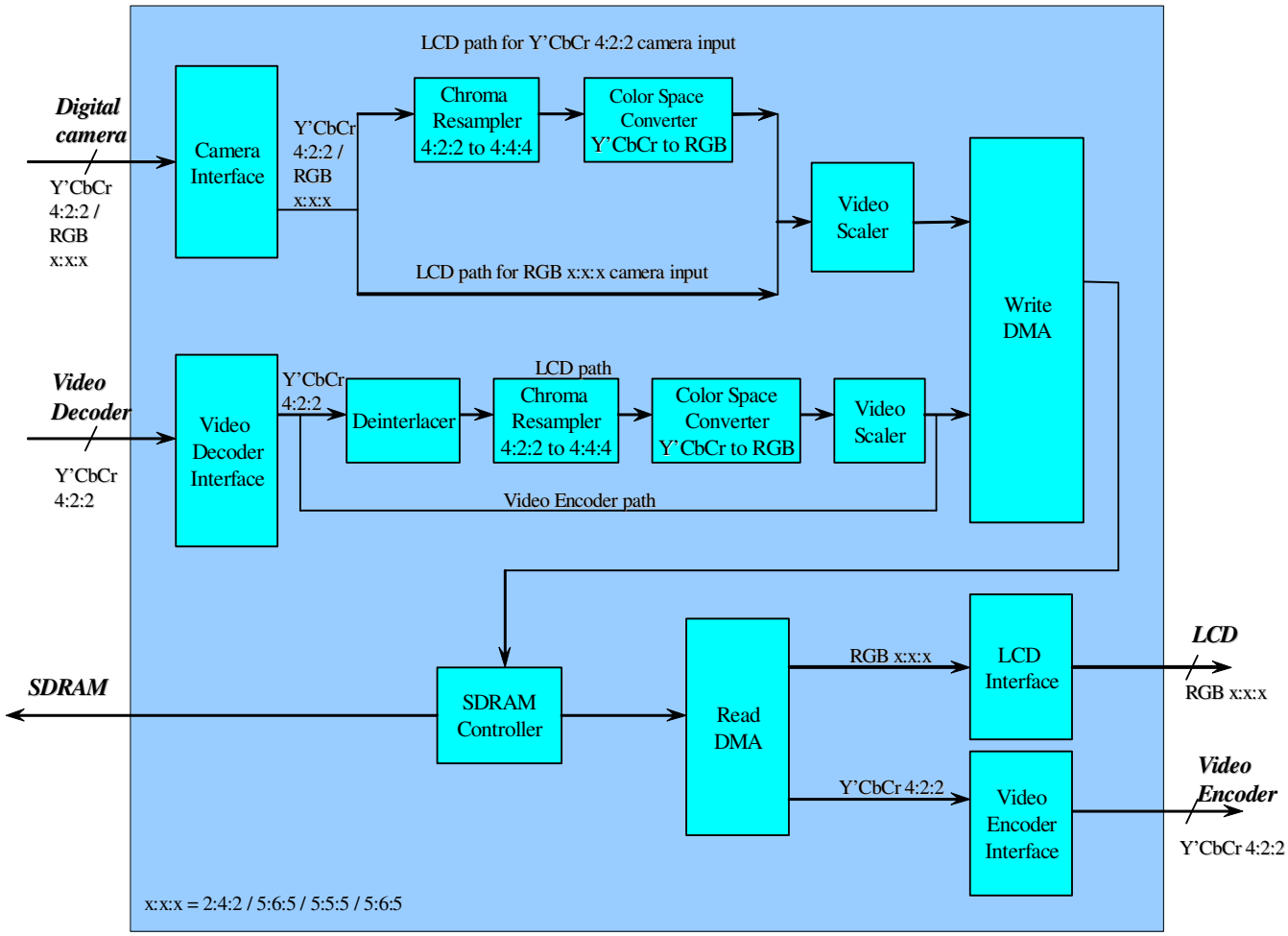

Figure 1: Block diagram of switch for video streaming

All modules shown (except write/read DMA functions) are part of the iWave Video IP library. Video data with associated synchronization signals is input into the design through either of the digital camera or video decoder interfaces and after processing, is output to the LCD or video decoder interfaces.

When streaming to the LCD, if the input video from the camera/decoder is not progressively scanned, a deinterlacer performs the conversion from interlaced to progressive. Since the LCD also expects RGB formatted pixels, a color space converter converts them from Y´CbCr to RGB format.

Functions performed by various modules in the design are described below.

Camera Interface: This block accepts synchronized Y´CbCr 4:2:2 / RGB x:x:x input from a digital camera (or image sensor).

Video Decoder Interface: This block accepts synchronized Y´CbCr 4:2:2 NTSC/PAL input (as per recommendation 601/656) from a video decoder like ADV 7180.

Chroma resampler: This block upsamples or downsamples the chrominance components of images e.g. from Y´CbCr 4:2:2 to 4:4:4 (upsampling) or Y´CbCr 4:4:4 to 4:2:2 (downsampling).

Color Space Converter: This block converts Y´CbCr to RGB colour space or RGB to Y´CbCr colour space as required. There can be two situations when it can be used.

- When the video transmission is from digital camera to the LCD, and the digital camera outputs Y´CbCr 4:2:2 data.
- When video transmission is from the video decoder to LCD

Video Scaler: In case resolutions of the input and output device are different, this block up/down scales the input resolution to meet the output resolution.

Deinterlacer: This block separates the intermeshed odd and even fields of the NTSC/PAL frames output by the video decoder, into a combined, sequential de-interlaced frame, in order to display on LCD.

SDRAM Controller: Processed pixels are be buffered in memory before transmitting to the output device.

Write/Read DMA: Using DMA enables fast data transfers from peripheral to peripheral.

LCD Interface: This block outputs synchronized RGB x:x:x video data to an LCD.

Video Encoder Interface: This block outputs synchronized Y´CbCr 4:2:2 video (as per recommendation 601/656) to a video encoder like ADV 7390.

3 I/O formats

In order to drive the peripherals supported by the design, standard Y´CbCr 4:2:2 and RGB video in different color depths are supported. A description of the requirements of each interface is given below:

- Camera interface: Y´Cbcr 4:2:2 or RGB x:x:x data input/output where x:x:x might be 2:4:2, 5:6:5, 5:5:5 or 4:4:4
- Video decoder interface: Y´CbCr 4:2:2 input/output as per recommendation 601/656.
- LCD interface: RGB x:x:x input/output where x:x:x might be 4:4:4, 5:5:5, 5:6:5, 6:6:6 or 8:8:8
- Video encoder interface: Y´CbCr 4:2:2 input/output as per recommendation 601/656.

4 Core descriptions

- **LCD Interface**

This module takes unformatted (without synchronization information) 24-bit RGB video data as input and formats them into frames comprising of separate 8-bit R, G and B outputs along with horizontal and vertical synchronization signals, which can be sent to an LCD controller/panel for display. The RGB formats supported are RGB 4:4:4, RGB 5:5:5, RGB 5:6:5, RGB 6:6:6 and RGB 8:8:8. The display size and pixel clock frequency are programmable. Polarities of data enable, frame clock and line clock can also be programmed. The interface supports progressive type of displays.

A block diagram of the LCD interface is shown below:

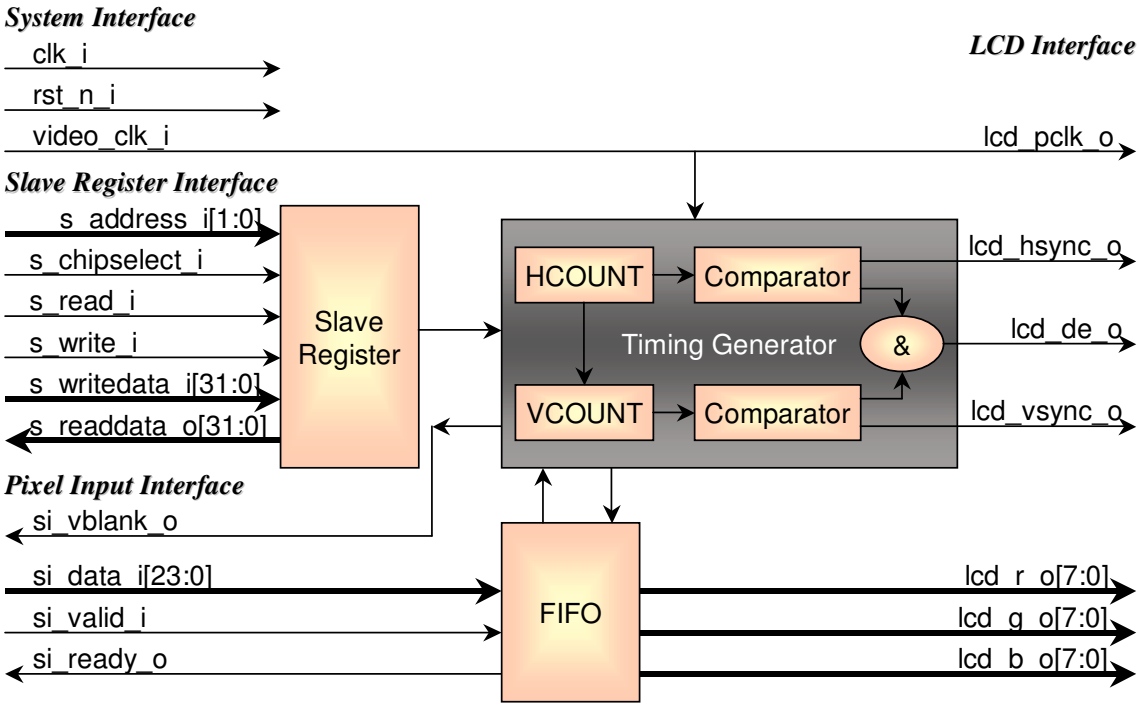

Figure 2:LCD Interface block diagram

Here, the slave register block contains programmable control registers, which are configured prior to start of operation through the Slave Register interface. The FIFO stores the incoming pixels from the Pixel Input interface when the valid signal is high. The Timing Generator controls read access to the FIFO using values programmed in the control registers of the slave register block and also provides pixel clock, data enable and synchronization signals to the LCD.

- **Camera Interface**

Pre-processed frames from a camera/image sensor comprising of 10-bit Y′CbCr 4:2:2 or RGB video data along with vertical and horizontal synchronization signals, can be connected as input to this module. The interface decodes these incoming frames and outputs 16-bit video data only in a continuous stream (without synchronization/blanking information) in the same format for further processing by a downstream device. Specific RGB formats supported are RGB 2:4:2, RGB 5:6:5, RGB 5:5:5 and RGB 4:4:4.The vertical and horizontal resolutions as well as the frame and line clock polarities can be programmed by the user.

A block diagram of the Camera interface is shown below:

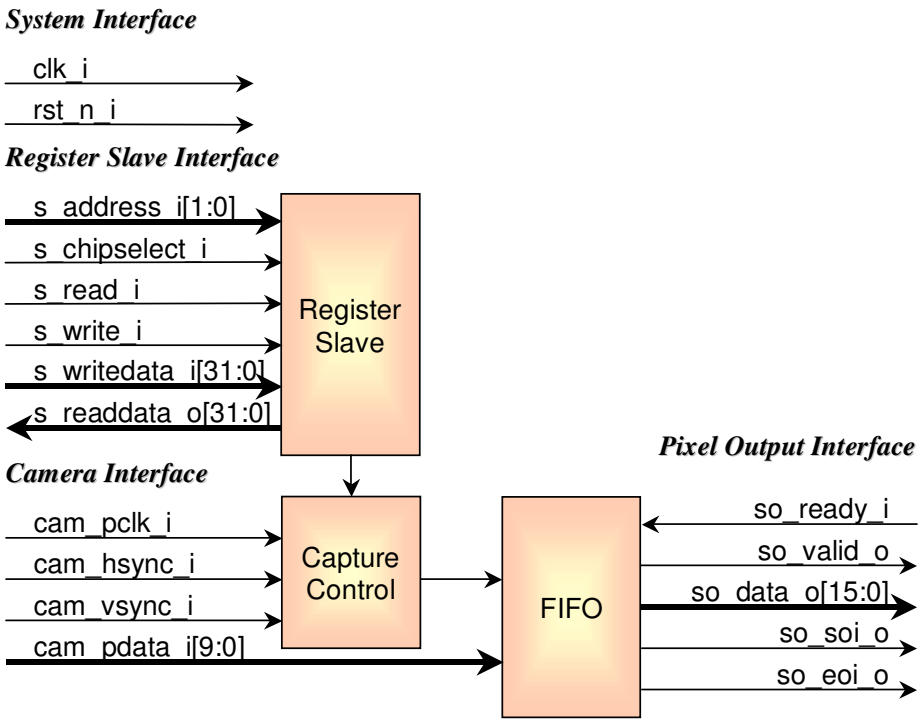

Figure 3:Camera Interface block diagram

Here, the slave register block contains programmable control registers, which are configured prior to start of operation through the Register Slave interface. The FIFO stores the incoming pixels from the Camera interface, which are read out to the Pixel Output interface. The Capture Control block decodes the incoming sync signals and generates FIFO write access when the data bus has valid pixel data. Pixels are read from the FIFO when the ready signal is high at the Pixel Output interface. The valid signal at this interface indicates valid data on the output data bus. Soi and eoi lines indicate that the first and last data of a picture frame are present on the data bus.

- **Video Decoder Interface**

This module accepts 10-bit Standard Definition TV frames (digitized PAL/NTSC) in ITU-R BT.656 / BT.601 formats from a video decoder like ADV7180 and outputs 10-bit video data only in a continuous stream, stripped of blanking and synchronization information. The user can program the frame and line clock polarities.

A block diagram of the Video Decoder interface is shown below:

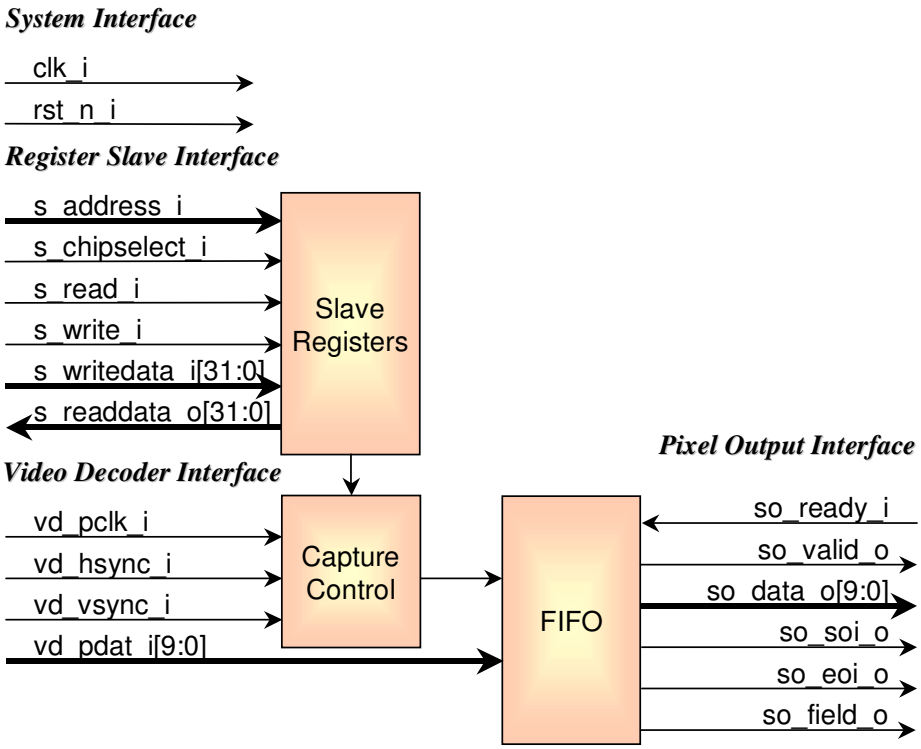

Figure 4:Video Decoder interface block diagram

Here, the slave register block contains programmable control registers, which are configured prior to start of operation through the Register Slave interface. The FIFO stores the incoming pixels from the Video Decoder interface, which are read out to the Pixel Output interface. The Capture Control block decodes the incoming sync signals and generates FIFO write access when the data bus has valid pixel data. Pixels are read from the FIFO when the ready signal is high at the Pixel Output interface. The valid signal at this interface indicates valid data on the output data bus. Soi and eoi lines indicate that the first and last data of a picture frame are present on the data bus. The field signal indicates whether the pixels on the data bus belong to the first or second field of the interlaced frame.

- **Video Encoder Interface**

This module takes as input 10-bit unformatted Y′CbCr 4:2:2 NTSC/PAL video and formats them into frames comprising of 10-bit Y′CbCr 4:2:2 video output along with horizontal and vertical synchronization signals in ITU-R BT.601 / BT.656 format. The output of the interface can be connected to a video encoder like ADV 7390. The user can program the frame and line clock polarities. Interlace scanned signals are supported by the interface.

A block diagram of the Video Encoder interface is shown below:

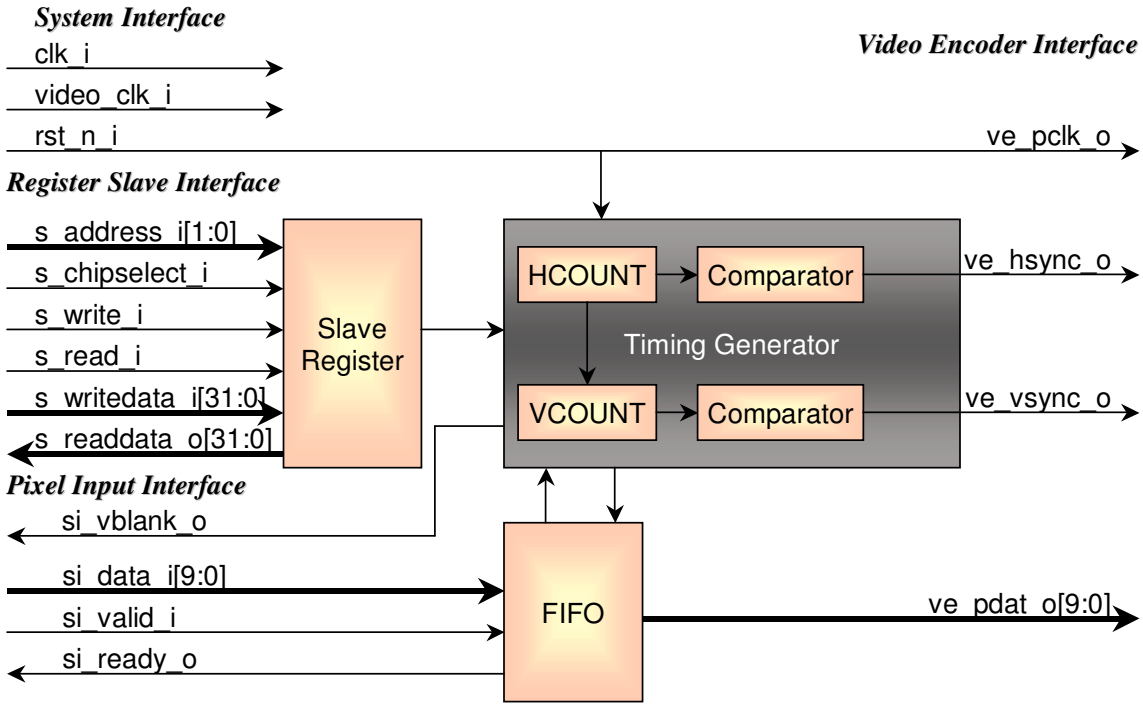

Figure 5:Video Encoder interface block diagram

Here, the slave register block contains programmable control registers, which are configured prior to start of operation through the Register Slave interface. The FIFO stores the incoming pixels from the Pixel Input interface when the valid signal is high. The Timing Generator controls read access to the FIFO using values programmed in the control registers of the slave register block and also provides pixel clock, data enable and synchronization signals to the Video Encoder.

- **Chroma Resampler**

This block upsamples or downsamples the chrominance components of images e.g. from Y´CbCr 4:2:2 to 4:4:4 (up sampling) or Y´CbCr 4:4:4 to 4:2:2 (down sampling).

A block diagram of the Chroma Resampler is shown below:

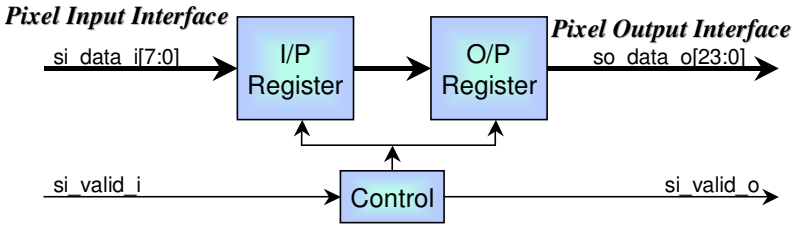

Figure 6: Chroma Resampler block diagram

- **Color Space Converter**

RGB to Y´CbCr or Y´CbCr to RGB color space conversion can be performed with this module. A block diagram of the Color Space Converter is shown below:

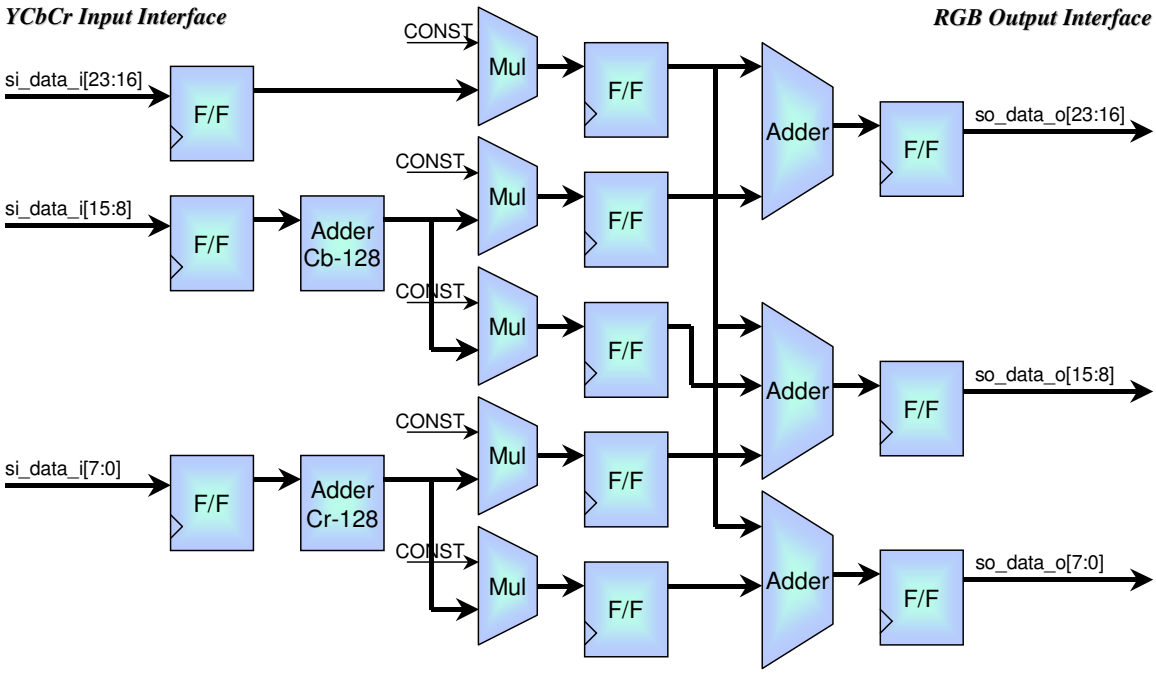

Figure 7:Color Space Converter block diagram

- **Video Scaler**

This module can upscale the input resolution by any value from 1 to 4 or downscale by any value from 1 to 16. Upscaling is accomplished by pixel duplication while downscaling is accomplished by pixel dropping. Upto a maximum of 4096*4096 input resolution is supported.

A block diagram of the Video Scaler is shown below:

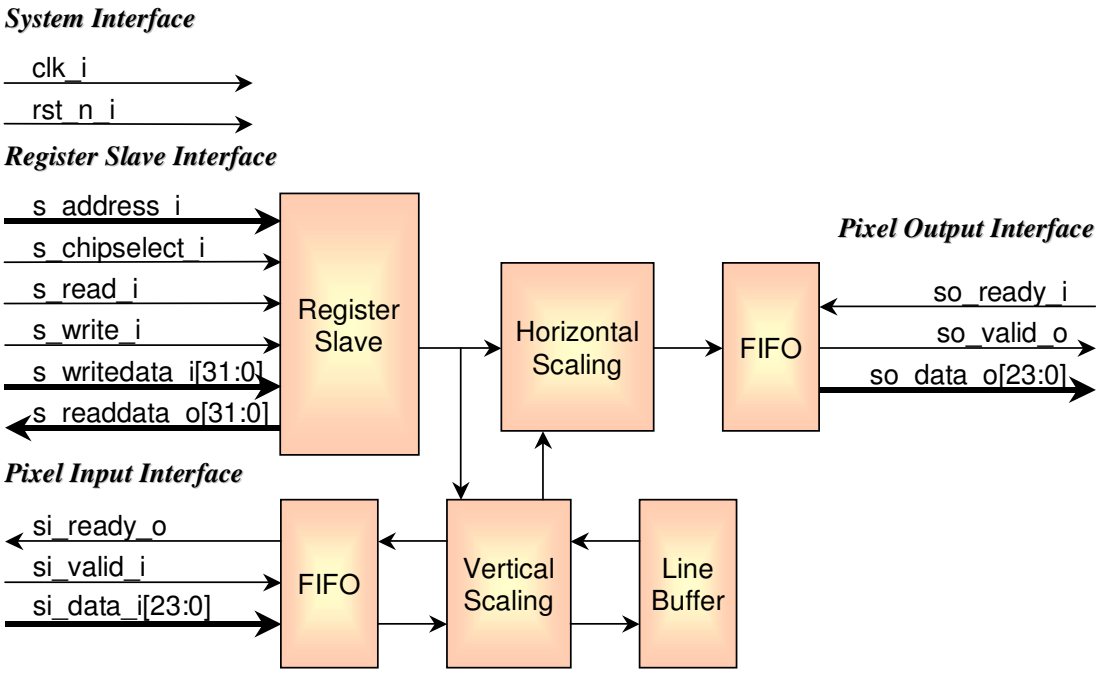

Figure 8:Video Scaler block diagram

Here, the register slave block contains the control registers for the scale values in the horizontal and vertical direction. Two synchronous FIFOs are used to store the input and scaled output video. The vertical scaling block scales the incoming image in vertical direction and stores in the DPRAM line buffers. The vertically scaled image is then passed on to the horizontal scaling block for scaling in the horizontal direction and stored in the output FIFO.

- **DeInterlacer**

Deinterlacing converts input interlaced (NTSC / PAL) video into a non-interlaced, sequential form suitable for modern display technologies such as LCDs.

A block diagram of the DeInterlacer is shown below:

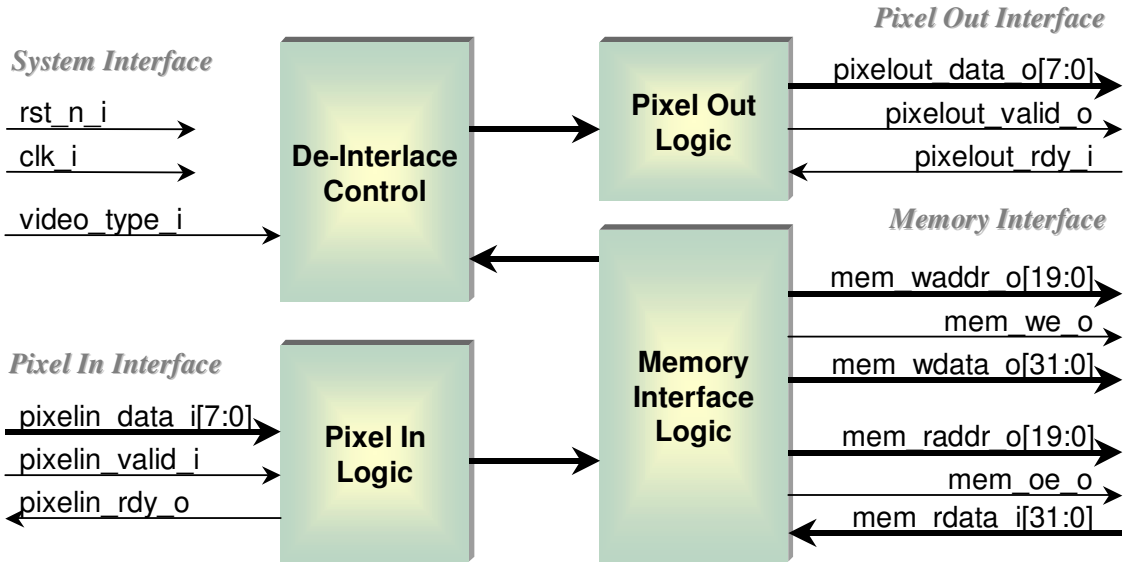

Figure 9: Video DeInterlacer block diagram

Here, interlaced frame comprising of 8-bit pixels are input through the Pixel In interface to the Pixel In logic when the ready signal is high. The Pixel In Logic converts it to 32-bit and forwards to the Memory Interface Logic for writing to memory. The Memory Interface Logic takes care of write address generation and read and write control signal generation. The De-Interlace Control block then generates read address and performs deinterlacing by the weave method. The de-interlaced video is passed onto the Pixel Out Logic and presented on the Pixel Out interface along with the valid signal.

- **SDRAM Controller**

This module is an interface to a Single Data Rate SDRAM and features configurable timing parameters (CAS latency, tRP, tRCD and tREFC), mode register configuration and 64Mbit or 128Mbit SDRAM size configuration. 2 or 4 banks per device are supported which can be configured. The controller can perform Sequential Read/Write burst operations with Burst Length of 1, 2, 4, 8 or Full Page. Commands supported for SDRAM Operation are Normal Operation (Read/Write), Refresh, Precharge and Load Mode Register. An Auto Refresh with a programmable Refresh Counter is available. A simple DMA interface is featured for easy integration in any application.

A block diagram of the SDRAM Controller is shown below:

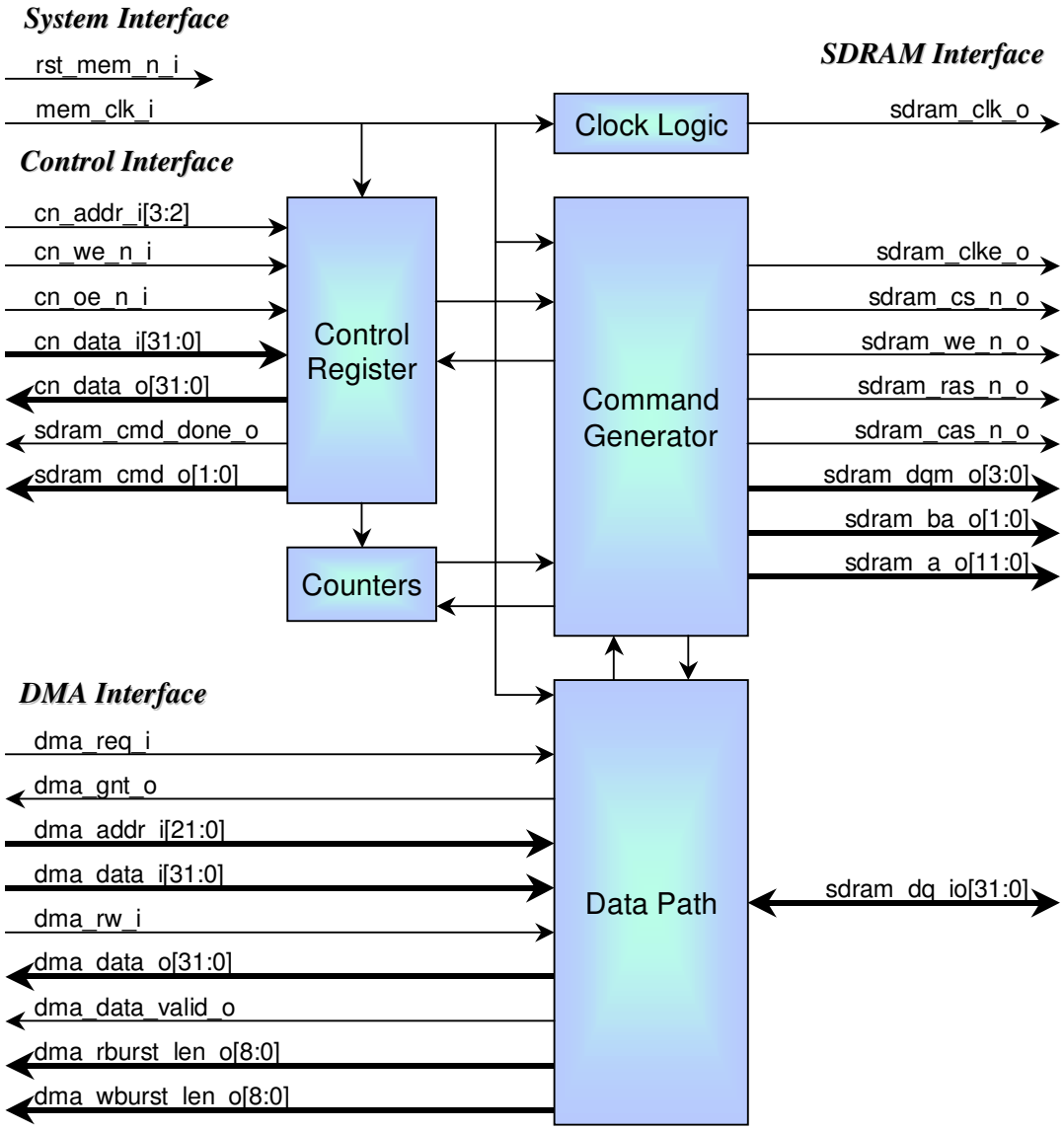

Figure 10:SDRAM Controller block diagram

Here, the Control Register block implements control registers for correct operation. The counters block implements different counters like Burst Counter, CAS Latency Counter, Precharge Counter, RCD Counter and Refresh Counter. On getting the grant, the DMA interface sends data enter to the Data Path block, which passes it on to the SDRAM with control inputs from the Command Generator block. The Command Generator block also sends commands to the SDRAM for load mode register, read, write, precharge and refresh operations using the values configured in the Counters block.

5 Implementation results

5.1 LCD Interface

Table 1: LCD Interface Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	445	1	0	127	0	78
Virtex4	XC4VLX40-10FF668	453	1	0	127	0	101

Table 2: LCD Interface Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	1928	2	127	0	34
IGLOO	AGL600FBGA256	1916	2	127	0	17

Table 3: LCD Interface Resource Utilization Summary - Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
Stratix II	EP2S15F484 C3	645	6144	0	127	0	136
Cyclone II	EP2C5F256 C6	715	6144	0	127	0	103

5.2 Camera Interface

Table 4: Camera Interface Resource Utilization Summary – Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	363	1	0	104	0	100
Virtex4	XC4VLX40-10FF668	358	1	0	104	0	133

Table 5: Camera Interface Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	1261	1	104	0	63

IGLOO	AGL600FBGA256	1243	1	104	0	32

Table 6: Camera Interface Resource Utilization Summary – Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
StratixII	EP2S15F484 C3	556	4608	0	104	0	200
CycloneII	EP2C5F256 C6	605	4608	0	104	0	137

5.3 Video Decoder Interface

Table 7: Video Decoder Interface Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	282	1	0	96	0	124
Virtex4	XC4VLX40-10FF668	274	1	0	96	0	135

Table 8: Video Decoder Interface Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	850	1	96	0	68
IGLOO	AGL600FBGA256	832	1	96	0	35

Table 9: Video Decoder Interface Resource Utilization Summary – Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
StratixII	EP2S15F484 C3	443	3328	0	98	0	271
CycloneII	EP2C5F256 C6	458	3328	0	98	0	197

5.4 Video Encoder Interface

Table 10: Video Encoder Interface Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	610	1	0	96	0	69

Virtex4	XC4VLX40-10FF668	608	1	0	96	0	100

Table 11: Video Encoder Interface Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	2449	1	96	0	38
IGLOO	AGL600FBGA256	2494	1	96	0	21

Table 12: Video Encoder Interface Resource Utilization Summary - Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
Stratix II	EP2S15F484 C3	819	2560	0	98	0	152
Cyclone II	EP2C5F256 C6	895	2560	0	98	0	102

5.5 Chroma Resampler

Table 13: Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	11	0	0	36	0	221
Virtex4	XC4VLX40-10FF668	15	0	0	36	0	382

Table 14: Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	60	0	36	0	250
IGLOO	AGL600FBGA256	60	0	36	0	143

Table 15: Resource Utilization Summary – Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
Stratix2	EP2S15F484 C3	55	0	0	36	0	500
Cyclone2	EP2C5T144 C6	55	0	0	36	0	420

5.6 Color Space Converter

Table 16: Color Space Converter Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	57	0	5	52	0	124
Virtex4	XC4VLX40-10FF668	57	0	5	52	0	204

Table 17: Color Space Converter Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P250PQFP208	872	0	52	0	77
IGLOO	AGL600FBGA256	930	0	52	0	37

Table 18: Color Space Converter Resource Utilization Summary - Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
Stratix2	EP2S15F484C3	238	0	0	52	0	310
Cyclone2	EP2C5T144C6	138	0	4	52	0	189

5.7 Video Scaler

Table 19: Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-5FG320	356	18	0	122	0	87
Virtex4	XC4VLX40-10FF668	356	18	0	122	0	101

Table 20: Resource Utilization Summary - Altera

Family	Device	Logic Elements	Memory bits	DSP/ MULT	IO	PLL/ DLL	Fmax (MHz)
Stratix 2	EP2S15F484C3	589	294912	0	122	0	149
Cyclone 2	EP2C35F484C6	674	294912	0	122	0	95

5.8 DeInterlacer

Table 21: Resource Utilization Summary - Xilinx

Family	Device	Slices	BRAM	DSP/ MULT	IO	DCM/ DLL	Fmax (MHz)
Spartan3E	XC3S500E-4FG320	158	0	0	129	0	100
Virtex4	XC4VLX15-10ff676	160	0	0	129	0	150

Table 22: Resource Utilization Summary - Actel

Family	Device	Cores	RAM	IO	PLL	Fmax (MHz)
ProASIC3	A3P600-1 FBGA484	699	0	0	0	80
Igloo	AGL600V5STD FBGA484	766	0	0	0	45