Terasic D5M Camera Sensor

ECE492 – Winter 2014 - Group 9

Brent Erickson
Mavis Chan
Sydney Bitner

Introduction

Terasic provides a 5MP camera sensor that is compatible with the DE2 board. There is supplied University IP that helps to interface with and configure the camera. This tutorial will demonstrate the hardware setup required in order to write Camera data to SRAM.

QSys

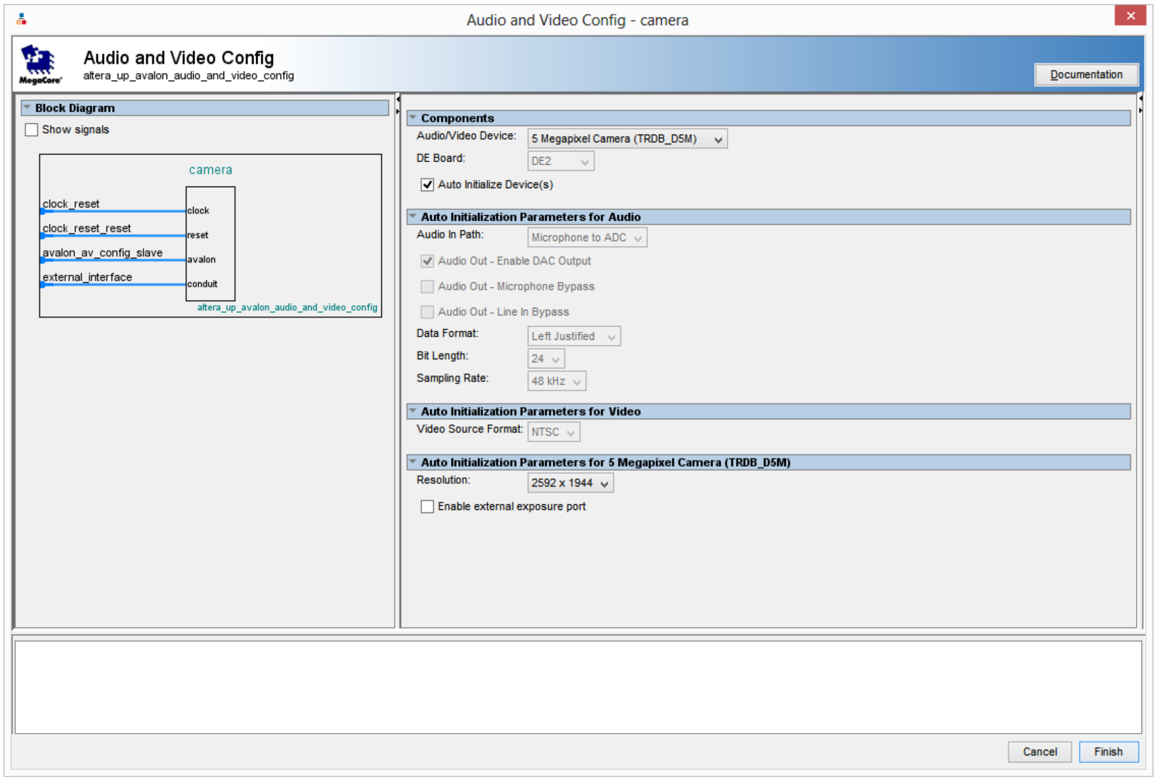
Required Components

- Nios II Processor
- Audo and Video Config
- Video-In Decoder
- Bayer Pattern Resampler
- Clipper
- Scaler
- DMA Controller
- SRAM/SSRAM Controller

Usage

Adding the above components and setting the given configuration values will output a 320x240 8-bit color image.

Audio and Video Config



Audio and Video Config - camera

Audio and Video Config
altera_up_avalon_audio_and_video_config

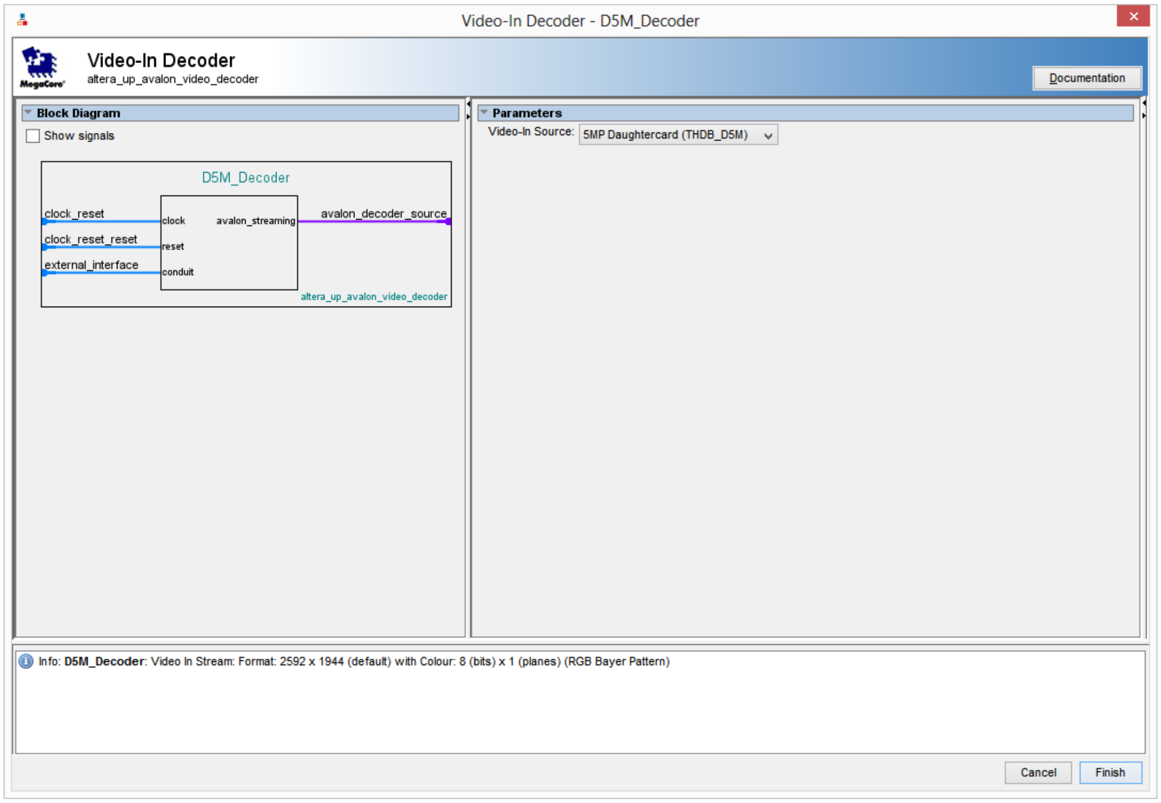
Components
Audio/Video Device: 5 Megapixel Camera (TRDB_D5M)
DE Board: DE2
☑ Auto Initialize Device(s)

Auto Initialization Parameters for Audio
Audio In Path: Microphone to ADC
☑ Audio Out - Enable DAC Output
□ Audio Out - Microphone Bypass
□ Audio Out - Line In Bypass
Data Format: Left Justified
Bit Length: 24
Sampling Rate: 48 kHz

Auto Initialization Parameters for Video
Video Source Format: NTSC

Auto Initialization Parameters for 5 Megapixel Camera (TRDB_D5M)
Resolution: 2592 x 1944
□ Enable external exposure port

Video-In Decoder



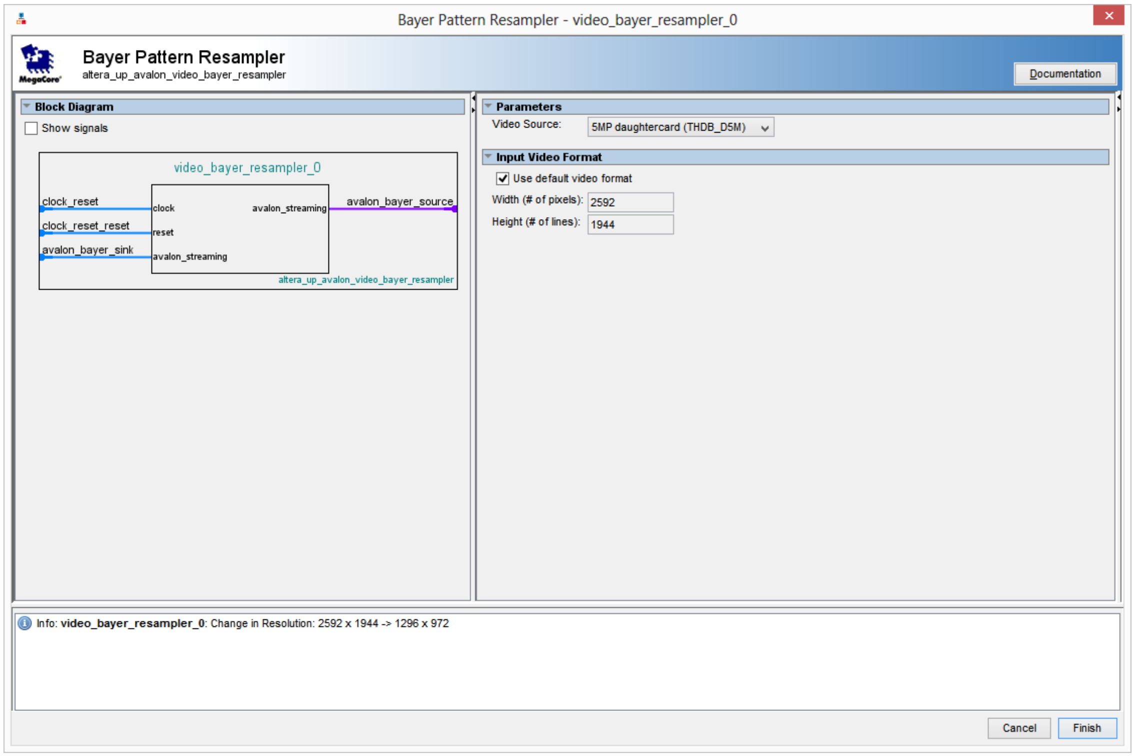
Video-In Decoder - D5M_Decoder

Video-In Decoder
altera_up_avalon_video_decoder

Video-In Source: 5MP Daughtercard (THDB_D5M)

Info: **D5M_Decoder**: Video In Stream: Format: 2592 x 1944 (default) with Colour: 8 (bits) x 1 (planes) (RGB Bayer Pattern)

Bayer Pattern Resampler



Bayer Pattern Resampler - video_bayer_resampler_0

Bayer Pattern Resampler
altera_up_avalon_video_bayer_resampler

Video Source: 5MP daughtercard (THDB_D5M)

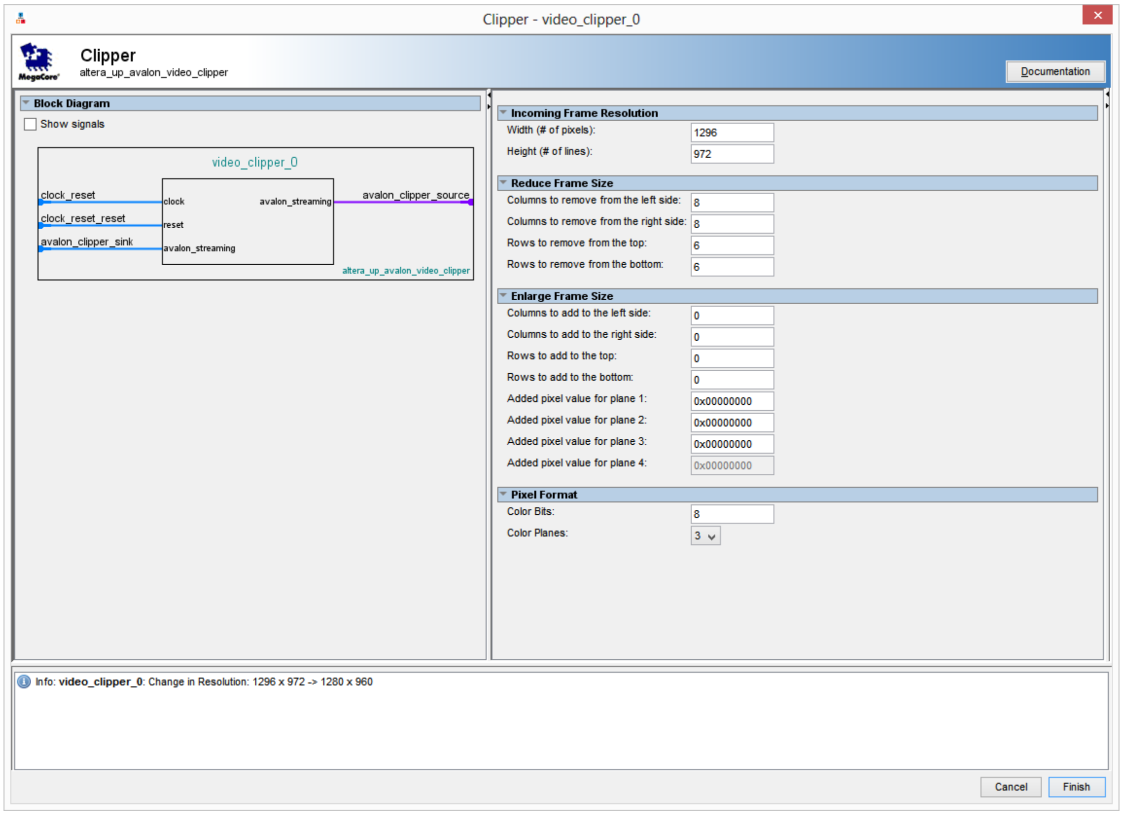
☑ Use default video format

Width (# of pixels): 2592

Height (# of lines): 1944

Info: **video_bayer_resampler_0**: Change in Resolution: 2592 x 1944 -> 1296 x 972

Clipper



Incoming Frame Resolution

Width (# of pixels): 1296

Height (# of lines): 972

Reduce Frame Size

Columns to remove from the left side: 8

Columns to remove from the right side: 8

Rows to remove from the top: 6

Rows to remove from the bottom: 6

Enlarge Frame Size

Columns to add to the left side: 0

Columns to add to the right side: 0

Rows to add to the top: 0

Rows to add to the bottom: 0

Added pixel value for plane 1: 0x00000000

Added pixel value for plane 2: 0x00000000

Added pixel value for plane 3: 0x00000000

Added pixel value for plane 4: 0x00000000

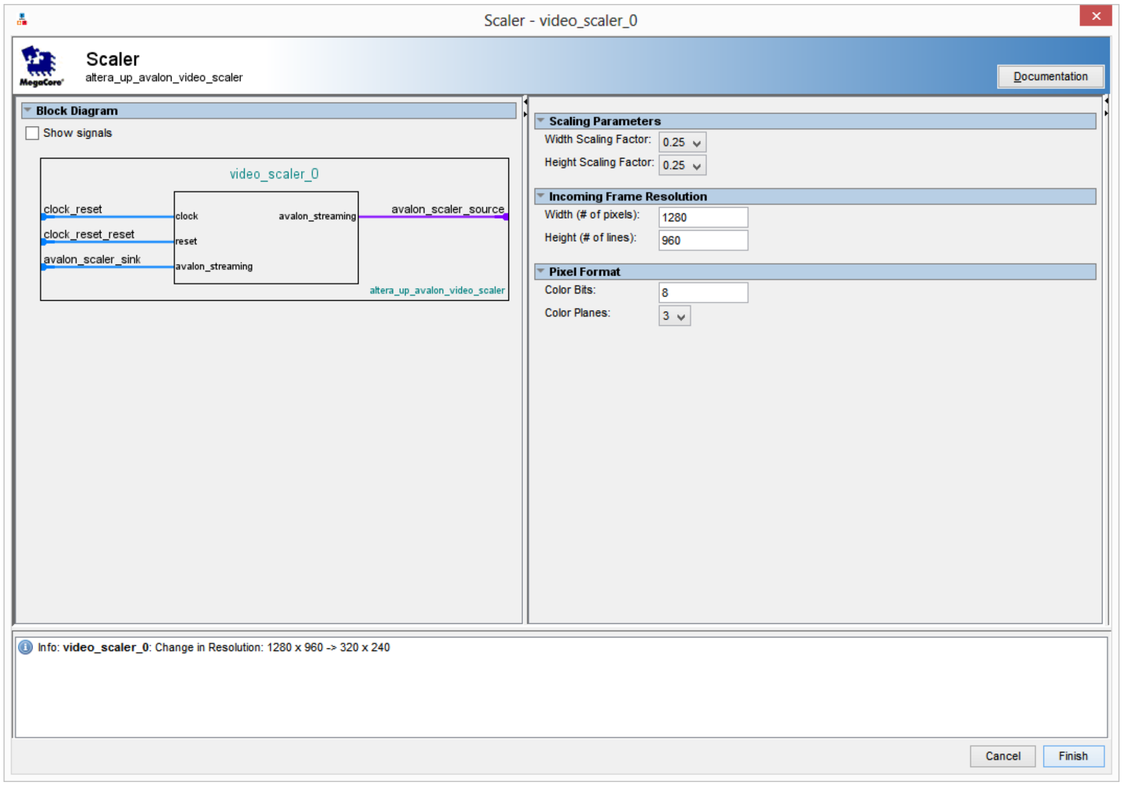
Pixel Format

Color Bits: 8

Color Planes: 3

Info: **video_clipper_0**: Change in Resolution: 1296 x 972 -> 1280 x 960

Scaler



Scaling Parameters

Width Scaling Factor: 0.25

Height Scaling Factor: 0.25

Incoming Frame Resolution

Width (# of pixels): 1280

Height (# of lines): 960

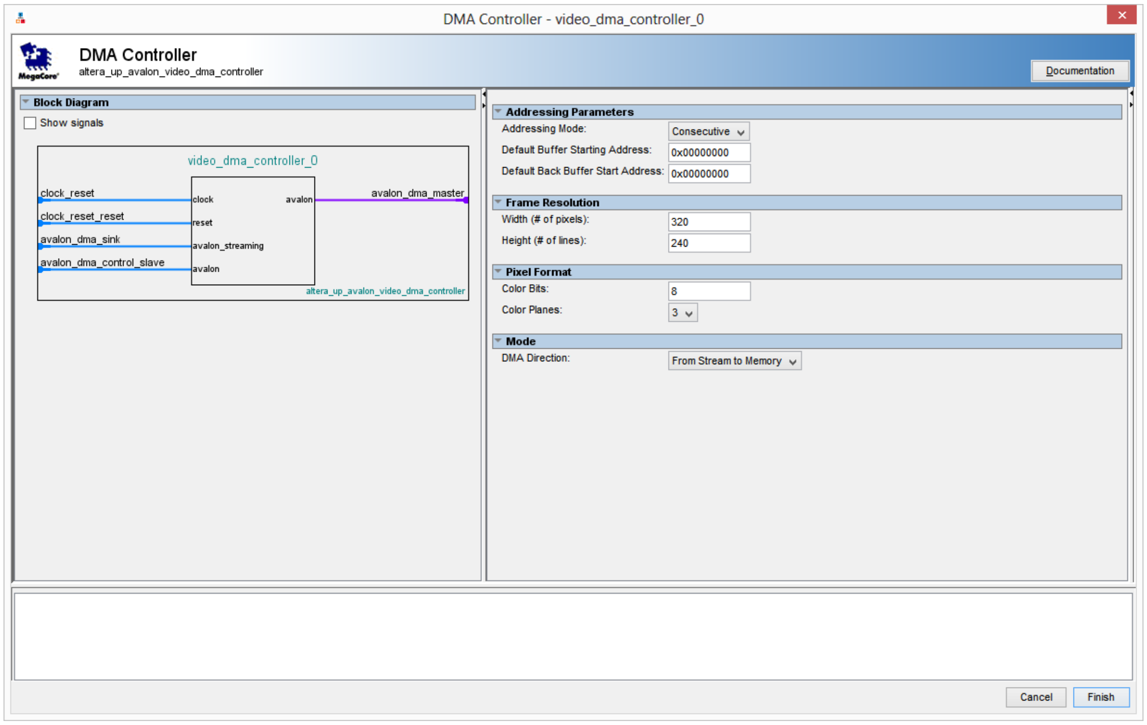
Pixel Format

Color Bits: 8

Color Planes: 3

Info: **video_scaler_0**: Change in Resolution: 1280 x 960 -> 320 x 240

DMA Controller



Addressing Parameters

Addressing Mode: Consecutive

Default Buffer Starting Address: 0x00000000

Default Back Buffer Start Address: 0x00000000

Frame Resolution

Width (# of pixels): 320

Height (# of lines): 240

Pixel Format

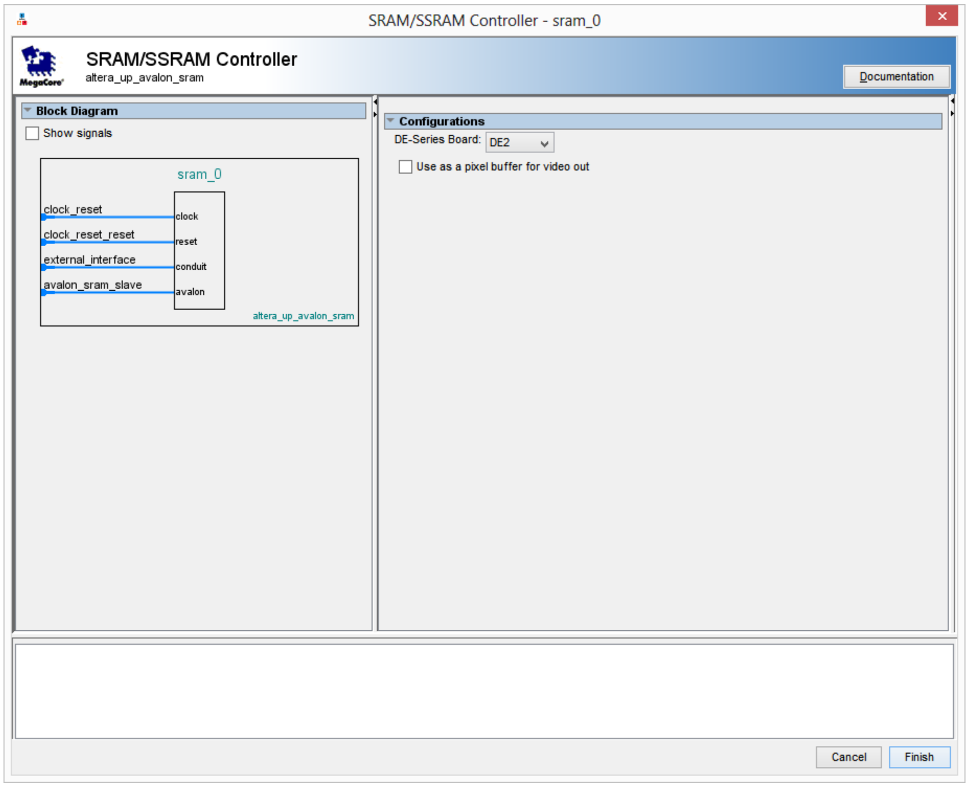
Color Bits: 8

Color Planes: 3

Mode

DMA Direction: From Stream to Memory

SRAM/SSRAM Controller

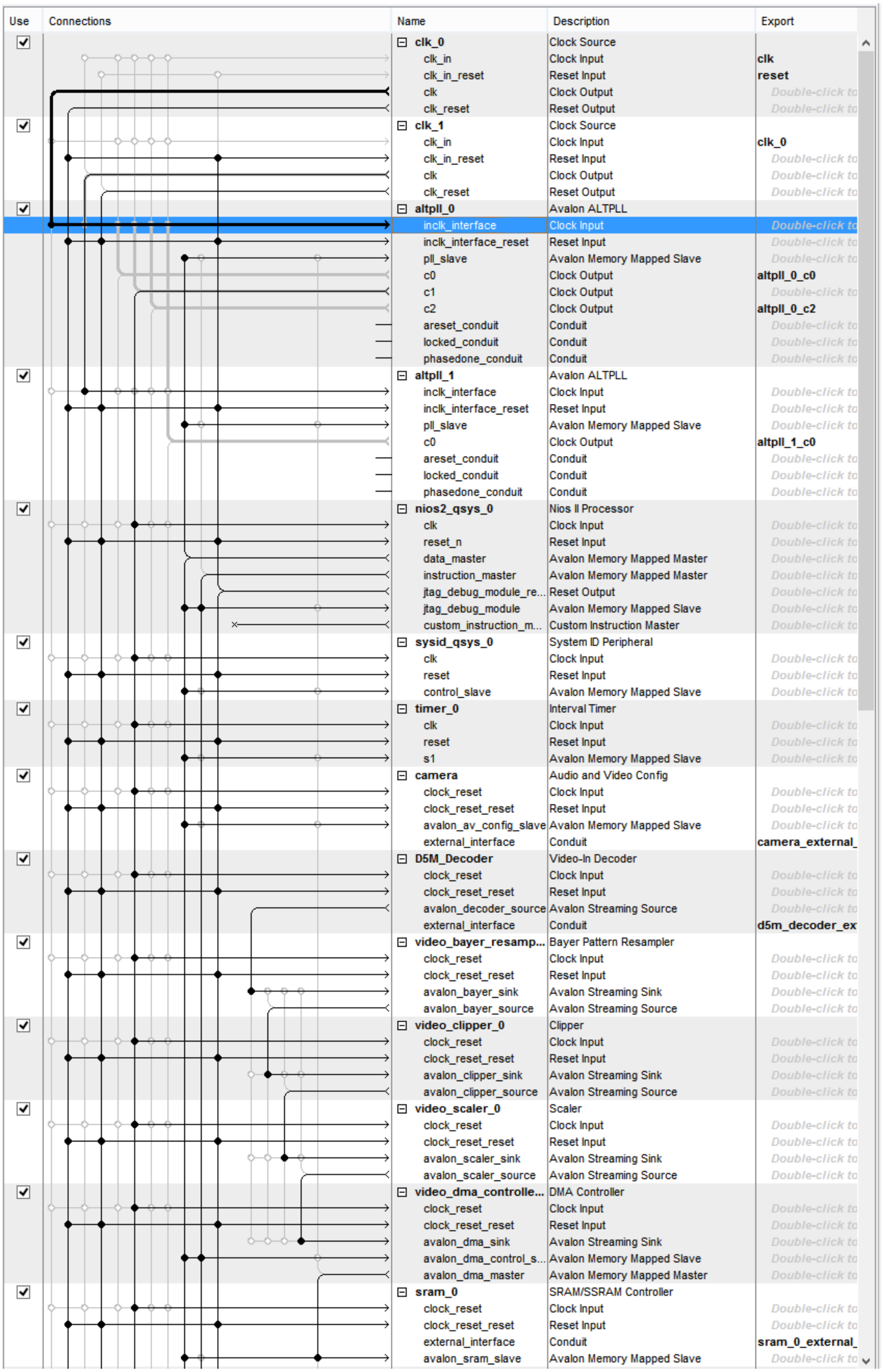


Configurations

DE-Series Board: DE2

□ Use as a pixel buffer for video out

Full Qsys Machine with interconnects



Use	Connections	Name	Description	Export
☑		⊟ **clk_0**	Clock Source	
		clk_in	Clock Input	**clk**
		clk_in_reset	Reset Input	**reset**
		clk	Clock Output	*Double-click to*
		clk_reset	Reset Output	*Double-click to*
☑		⊟ **clk_1**	Clock Source	
		clk_in	Clock Input	**clk_0**
		clk_in_reset	Reset Input	*Double-click to*
		clk	Clock Output	*Double-click to*
		clk_reset	Reset Output	*Double-click to*
☑		⊟ **altpll_0**	Avalon ALTPLL	
		inclk_interface	Clock Input	*Double-click to*
		inclk_interface_reset	Reset Input	*Double-click to*
		pll_slave	Avalon Memory Mapped Slave	*Double-click to*
		c0	Clock Output	**altpll_0_c0**
		c1	Clock Output	*Double-click to*
		c2	Clock Output	**altpll_0_c2**
		areset_conduit	Conduit	*Double-click to*
		locked_conduit	Conduit	*Double-click to*
		phasedone_conduit	Conduit	*Double-click to*
☑		⊟ **altpll_1**	Avalon ALTPLL	
		inclk_interface	Clock Input	*Double-click to*
		inclk_interface_reset	Reset Input	*Double-click to*
		pll_slave	Avalon Memory Mapped Slave	*Double-click to*
		c0	Clock Output	**altpll_1_c0**
		areset_conduit	Conduit	*Double-click to*
		locked_conduit	Conduit	*Double-click to*
		phasedone_conduit	Conduit	*Double-click to*
☑		⊟ **nios2_qsys_0**	Nios II Processor	
		clk	Clock Input	*Double-click to*
		reset_n	Reset Input	*Double-click to*
		data_master	Avalon Memory Mapped Master	*Double-click to*
		instruction_master	Avalon Memory Mapped Master	*Double-click to*
		jtag_debug_module_re...	Reset Output	*Double-click to*
		jtag_debug_module	Avalon Memory Mapped Slave	*Double-click to*
		custom_instruction_m...	Custom Instruction Master	*Double-click to*
☑		⊟ **sysid_qsys_0**	System ID Peripheral	
		clk	Clock Input	*Double-click to*
		reset	Reset Input	*Double-click to*
		control_slave	Avalon Memory Mapped Slave	*Double-click to*
☑		⊟ **timer_0**	Interval Timer	
		clk	Clock Input	*Double-click to*
		reset	Reset Input	*Double-click to*
		s1	Avalon Memory Mapped Slave	*Double-click to*
☑		⊟ **camera**	Audio and Video Config	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_av_config_slave	Avalon Memory Mapped Slave	*Double-click to*
		external_interface	Conduit	**camera_external_**
☑		⊟ **D5M_Decoder**	Video-In Decoder	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_decoder_source	Avalon Streaming Source	*Double-click to*
		external_interface	Conduit	**d5m_decoder_ex**
☑		⊟ **video_bayer_resamp...**	Bayer Pattern Resampler	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_bayer_sink	Avalon Streaming Sink	*Double-click to*
		avalon_bayer_source	Avalon Streaming Source	*Double-click to*
☑		⊟ **video_clipper_0**	Clipper	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_clipper_sink	Avalon Streaming Sink	*Double-click to*
		avalon_clipper_source	Avalon Streaming Source	*Double-click to*
☑		⊟ **video_scaler_0**	Scaler	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_scaler_sink	Avalon Streaming Sink	*Double-click to*
		avalon_scaler_source	Avalon Streaming Source	*Double-click to*
☑		⊟ **video_dma_controlle...**	DMA Controller	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		avalon_dma_sink	Avalon Streaming Sink	*Double-click to*
		avalon_dma_control_s...	Avalon Memory Mapped Slave	*Double-click to*
		avalon_dma_master	Avalon Memory Mapped Master	*Double-click to*
☑		⊟ **sram_0**	SRAM/SSRAM Controller	
		clock_reset	Clock Input	*Double-click to*
		clock_reset_reset	Reset Input	*Double-click to*
		external_interface	Conduit	**sram_0_external_**
		avalon_sram_slave	Avalon Memory Mapped Slave	*Double-click to*

Top Level

PIN Mapping Changes

Note: We assume that the camera is connected to GPIO_1
Remap the following pins currently assigned to GPIO_1 to CCD_DATA. This is necessary to access the block of data pins required for decoding. The pins are "backwards" in relation to traditional pin mapping and are not a contiguous set of pins. Re-mapping allows us to access them as if they were.

CCD_DATA[11]	Location	PIN_K26
CCD_DATA[10]	Location	PIN_M23
CCD_DATA[9]	Location	PIN_M19
CCD_DATA[8]	Location	PIN_M20
CCD_DATA[7]	Location	PIN_N20
CCD_DATA[6]	Location	PIN_M21
CCD_DATA[5]	Location	PIN_M24
CCD_DATA[4]	Location	PIN_M25
CCD_DATA[3]	Location	PIN_N24
CCD_DATA[2]	Location	PIN_P24
CCD_DATA[1]	Location	PIN_R25
CCD_DATA[0]	Location	PIN_R24

VHDL Changes

Entity

The newly added PINs need to be added to the toplevel entity:

```
entity BioLock is
    port (
              -- Reset and Clock
              KEY           : in std_logic_vector (0 downto 0);
              CLOCK_50      : in std_logic;
              CLOCK_27 : in std_logic;



              -- Off Chip
              GPIO_1        : inout std_logic_vector(35 downto 0);
              CCD_DATA              : in std_logic_vector(11 downto 0);
…
…
```

Nios Component

The camera components from the qsys machine also need to be added to the nios_system component:

```
--Camera
  camera_external_interface_SDAT   : inout std_logic   := 'X';       -- SDAT
  camera_external_interface_SCLK   : out   std_logic;    -- SCLK
  d5m_decoder_external_interface_PIXEL_CLK     : in    std_logic   := 'X';       -- PIXEL_CLK
  d5m_decoder_external_interface_LINE_VALID    : in    std_logic   := 'X';       -- LINE_VALID
  d5m_decoder_external_interface_FRAME_VALID   : in    std_logic   := 'X';       -- FRAME_VALID
  d5m_decoder_external_interface_PIXEL_DATA    : in    std_logic_vector(11 downto 0) := (others => 'X');
-- PIXEL_DATA
```

Port Map

The following need to be added to the process for the camera reset and clock pins

```
            GPIO_1(17) <= '1'; --reset
            GPIO_1(16) <= CLOCK_50;
```

The following need to be added to the port map for the nios system:

```
  camera_external_interface_SDAT   => GPIO_1(23),   --    camera_external_interface.SDAT
  camera_external_interface_SCLK   => GPIO_1(24),   --    .SCLK
  d5m_decoder_external_interface_PIXEL_CLK     => GPIO_1(0),    --    .PIXEL_CLK
  d5m_decoder_external_interface_LINE_VALID    => GPIO_1(21),   --    .LINE_VALID
  d5m_decoder_external_interface_FRAME_VALID   => GPIO_1(22),   --    .FRAME_VALID
  d5m_decoder_external_interface_PIXEL_DATA    => CCD_DATA,
```

Other Info

At this point, data is being written directly into SRAM. There are 320x240x4 bytes used and can be read directly from SRAM by the nios processor. Each set of 4 bytes corresponds to a single pixel (R, G, B, NULL). This format correlates to the BMP format.

Readings

The following readings proved useful when researching how to get the D5M module working:

- http://www.terasic.com.tw/attachment/archive/282/TRDB_D5M_UserGuide.pdf
- http://www.alteraforum.com/forum/showthread.php?t=29018
- http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=39&No=281&PartNo=2
- https://www.youtube.com/watch?v=dPq7ROJiM90 (for proof that it actually worked for someone else)