UG0649

User Guide

Display Controller

February 2018

Microsemi

Power Matters.™

Contents

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 5.0

In revision 5.0 of this document, the Resource Utilization section and the Resource Utilization Report were updated. For more information, see Resource Utilization (see page 14).

1.2 Revision 4.0

In revision 4.0 of this document, the Testbench section was updated with the steps to simulate the core using testbench. For more information, see Testbench Simulation (see page 7).

1.3 Revision 3.0

The following is a summary of the changes in revision 3.0 of this document.

- Updated the Hardware Implementation section with ddr_rd_video_resolution input signal. For more information, see Hardware Implementation (see page 3).
- Updated the display control resolution to 4096 × 2160. For more information, see Inputs and Outputs (see page 3).
- Added the Testbench Simulation section. For more information, see Testbench Simulation (see page 7).

1.4 Revision 2.0

In revision 2.0 of this document, the Configuration Parameters table was updated with g_DEPTH_OF_VIDEO_PIXEL_FROM_DDR signal. For more information, see Configuration Parameters (see page 6).

1.5 Revision 1.0

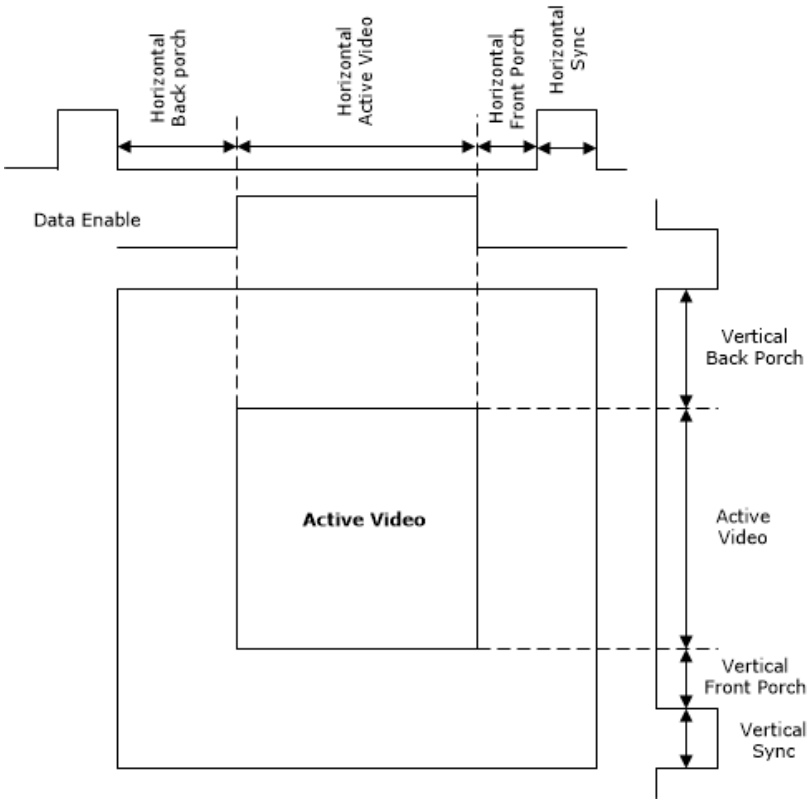
Revision 1.0 is the first publication of this document.

2 Introduction

The display controller generates display synchronization signals based on the display resolution. The synchronization signals are horizontal sync, vertical sync, and data enable. The input video data is also synchronized with these sync signals. The sync signals along with video data can be fed to a DVI, HDMI, or VGA card that interfaces with the display monitor.

The following figure shows the sync signal waveforms.

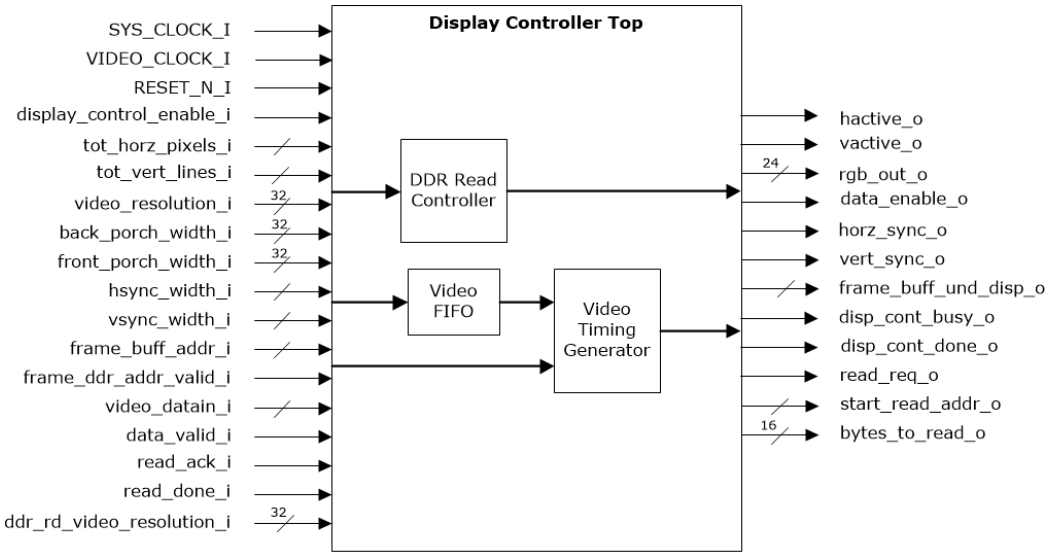
Figure 1 • Sync Signal Waveforms



3 Hardware Implementation

The following figure shows the display controller block diagram.

Figure 2 • Display Controller Block Diagram



3.1 Inputs and Outputs

The following table lists the description of input and output ports.

Table 1 • Inputs and Outputs of Display Controller

Signal Name	Direction	Width	Description
RESET_n_I	Input	-	Active low asynchronous reset signal to design.
SYS_CLOCK_I	Input	-	To synchronize DDR_SDRAM interface input/output signals.
VIDEO_CLOCK_I	Input	-	To synchronize the video data and sync outputs.
display_control_enable_i	Input	-	To start the display.
tot_horz_pixels_i	Input	[(g_INPUT_X_W_RES_WIDTH -1):0]	Total width of one horizontal line.
tot_vert_lines_i	Input	[(g_INPUT_Y_H_RES_WIDTH -1):0]	Total number of vertical lines.
video_resolution_i	Input	[31:0]	Video resolution: [15:0] horizontal [31:16] vertical
back_porch_width_i	Input	[31:0]	Back porch width: [15:0] horizontal [31:16] vertical

Signal Name	Direction	Width	Description
front_porch_width_i	Input	[31:0]	Front porch width: [15:0] horizontal [31:16] Vertical
hsync_width_i	Input	[(g_INPUT_X_W_RES_WIDTH -1):0]	Horizontal SYNC (HSYNC) pulse width.
vsync_width_i	Input	[(g_INPUT_Y_H_RES_WIDTH -1):0]	Vertical SYNC (VSYNC) pulse width.
ddr_rd_video_resolution_i	Input	[31:0]	The video resolution of image to be read from external memory. [15:0] horizontal [31:16] vertical.
frame_buff_addr_i	Input	[(g_DDR_AXI_AWIDTH -1):0]	Base address of display frame buffer.
frame_ddr_addr_valid_i	Input	-	Valid input for base address of display frame buffer.
frame_buff_und_disp_o	Output	[(g_DDR_AXI_AWIDTH -1):0]	Base address of the current display frame buffer.
read_req_o	Output	-	Read request to external memory controller.
read_ack_i	Input	-	Acknowledgment from external memory controller for the read request.
read_done_i	Input	-	Read completion from external memory controller.
start_read_addr_o	Output	[(g_DDR_AXI_AWIDTH - 1):0]	Frame buffer memory address from where read must be started.
bytes_to_read_o	Output	[15:0]	Bytes to be read out from the frame buffer memory.
video_datain_i	Input	[(g_INPUT_VIDEO_DATA_BIT_WIDTH – 1):0]	Video data input from frame buffer memory.
data_valid_i	Input	-	Read data valid from frame buffer memory.
disp_cont_busy_o	Output	-	Display controller busy status signal.
disp_cont_done_o	Output	-	Display controller done status signal.
rgb_out_o	Output	[23:0]	RGB output.
data_enable_o	Output	-	Video data valid output.
horz_sync_o	Output	-	Horizontal sync signal.
vert_sync_o	Output	-	Vertical sync signal.
hactive_o	Output	-	Display horizontal active video.
vactive_o	Output	-	Display vertical active video.
dg_fifo_empty	Output	-	Debug signal indicating that FIFO is empty.

Signal Name	Direction	Width	Description
dg_fifo_full	Output	-	Debug signal indicating that FIFO is full.
dg_fifo_hemtpy	Output	-	Debug signal indicating that FIFO is half empty.

The display controller:

- Reads the video data from the specified display frame buffers located in double-data-rate - synchronous dynamic random access memory (DDR-SDRAM) and buffers the data in the internal first-in first-out (FIFO)
- Generates the timing required for the specified display resolution.
  - Synchronizes the input video data according to the video timing
  - Generates the output (the synchronized video data along with the required sync pulses).

The display controller takes the required video resolution parameters from an external controller as inputs. These parameters are the width and height, active video width and height, horizontal and vertical back and front porch, and horizontal and vertical sync pulse widths.

For reading the video data from the frame buffer, the display controller requires the base address of the frame buffer. An external controller must provide the base address on the frame_buff_addr_i signal and validate it with an active high pulse on the frame_ddr_addr_valid_i signal.

The display_control_enable_i signal enables the display controller. Before enabling the display controller, all the video resolution inputs must be provided to the respective inputs. After enabling the signal, the display controller samples the video resolution input values and updates the internal counters terminal values. It has two counters running on the VIDEO_CLOCK_I-horizontal pixel counter and vertical line counter.

The display controller requests for video data from the external memory controller and stores in an internal FIFO. To request data, the display controller provides the starting address on start_read_addr_o output and data bytes to read on bytes_to_read_o output to the external memory controller. It sends a high signal on read_req_o output. It holds the high state on read_req_o till the external memory controller acknowledges it by toggling read_ack_i. The input data video_datain_i signal is stored into the internal FIFO using the input data valid signal data_valid_i. It waits for the completion of the read process by monitoring the read_done_i input from the external memory controller. The video data read interface is synchronized using SYS_CLOCK_I signal.

The internal FIFO acts as an internal buffer and takes care of clock domain crossing between SYS_CLOCK_I and VIDEO_CLOCK_I signals.

When a read process is started on a new frame buffer, display controller fills the internal buffer with multiple horizontal lines of video data, which the g_INITIAL_LINES_TO_BUFFER global configuration parameter determines. Subsequent read requests are generated based on the half-empty status signal from the internal FIFO. The g_SUBSEQUENT_LINES_TO_BUFFER global configuration parameter determines the number of subsequent horizontal lines to read from frame buffer memory.

The display controller has the following status output signals:

- disp_cont_busy_o - Indicates the display process in-progress
- disp_cont_done_o - Indicates the completion of the current display frame buffer

When a new frame buffer is read, its base address is output on the frame_buff_und_disp_o, which can be used by the external controller to know that the particular frame buffer is under display. The display controller supports the maximum resolution of 4096 × 2160. The internal FIFO is configured to a default maximum depth of 4096, which is configured using the global configurable parameter, g_VIDEO_BUFF_AWIDTH.

3.2 Configuration Parameters

The following table lists the description of the generic configuration parameters used in the hardware implementation of display controller, which can vary based on the application requirements.

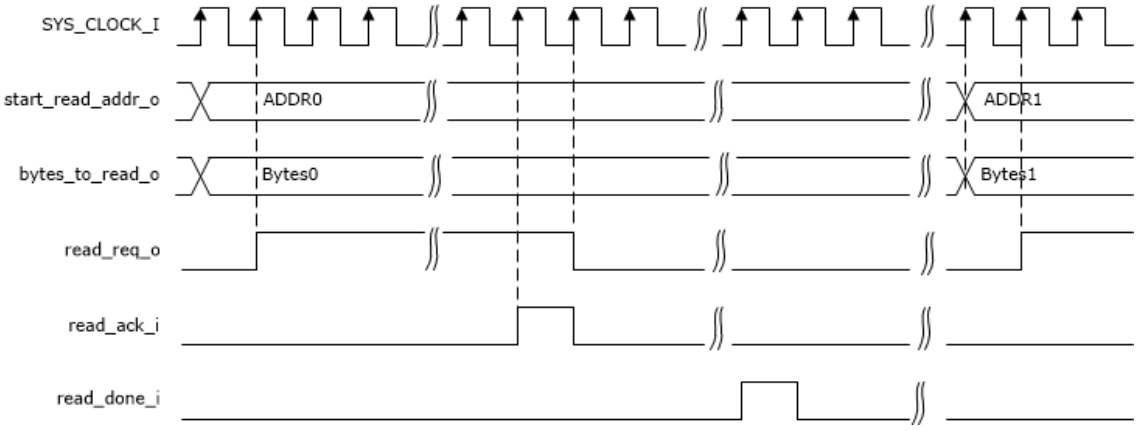
Table 2 • Configuration Parameters

Name	Description
g_DDR_AXI_AWIDTH	DDR-SDRAM AXI address bus width.
g_INPUT_X_W_RES_WIDTH	Bit width of total width of one output frame.
g_INPUT_Y_H_RES_WIDTH	Bit width of total number of lines in output frame.
g_VIDEO_FIFO_AWIDTH	Address bus width for the internal FIFO, which stores the video data.
g_INPUT_VIDEO_DATA_BIT_WIDTH	Input video data bit width.
g_DEPTH_OF_VIDEO_PIXEL_FROM_DDR	Number of bytes used to represent each pixel.
g_HORZ_SYNC_PULSE_POLARITY	Polarity of the HSYNC.
g_VERT_SYNC_PULSE_POLARITY	Polarity of the VSYNC.
g_INITIAL_LINES_TO_BUFFER	Number of display horizontal lines that are buffered for the first read from frame buffer memory.
g_SUBSEQUENT_LINES_TO_BUFFER	Number of display horizontal lines that are buffered in subsequent reads from frame buffer memory.
g_HALF_EMPTY_THRESHOLD	Half empty threshold of the internal video FIFO. This threshold determines the DDR read transactions.

3.3 Timing Diagrams

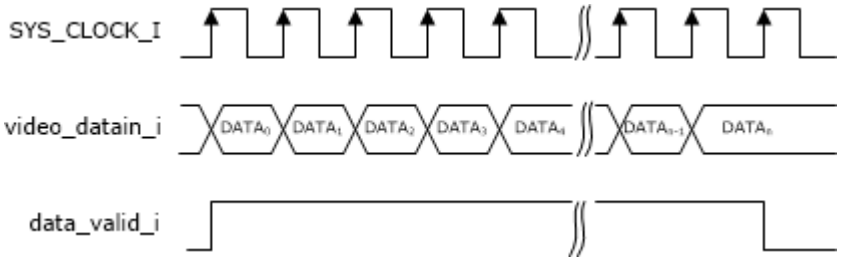
The following figure shows the read request, starting memory address, bytes to read sent from display controller, read acknowledgment, and read done from external memory controller in the timing diagram.

Figure 3 • Timing Diagram



The following figure shows the timing diagram for video data and data valid input signals received from the memory controller.

Figure 4 • Timing Diagram For Video Data Input



3.4 Testbench Simulation

A testbench is provided to check the functionality of the display controller. The following table lists the parameters that can be configured.

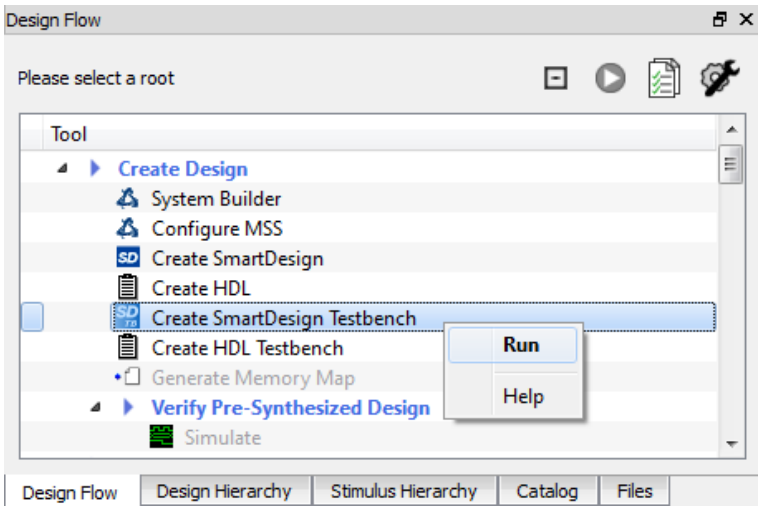
Table 3 • Testbench Configuration Parameters

Name	Description
SYS_CLOCK_PERIOD	Clock period for input.
g_DISPLAY_RESOLUTION	Width of the image.
g_VERT_DISPLAY_RESOLUTION	Height of the image.
IMAGE_FILE_NAME	Input file name.

The following steps describe how to simulate the core using the testbench.

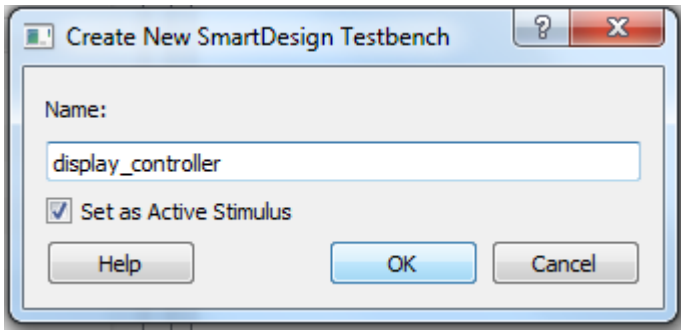
1. In the **Libero Design Flow** window, expand **Create Design**, double-click **Create SmartDesign Testbench** or right-click **Create SmartDesign Testbench** and click **Run** as shown in the following illustration.

Figure 5 • Create SmartDesign Testbench



2. Enter a name for the SmartDesign testbench in the **Create New SmartDesign Testbench** dialog box and click **OK**.

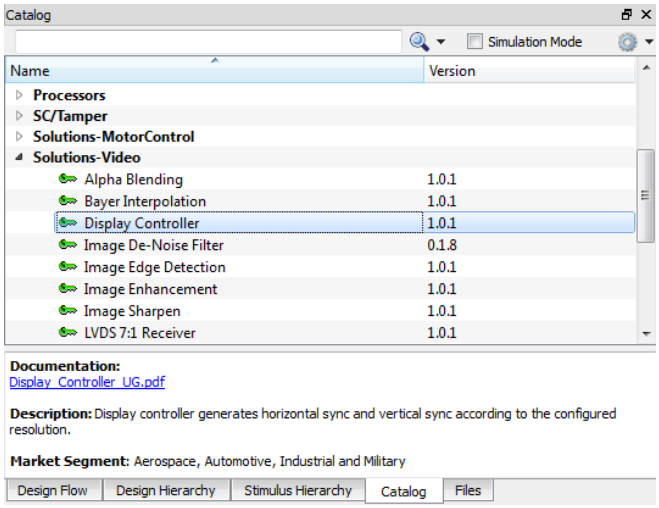
Figure 6 • Naming SmartDesign Testbench



A SmartDesign testbench is created, and a canvas appears to the right of the **Design Flow** pane.

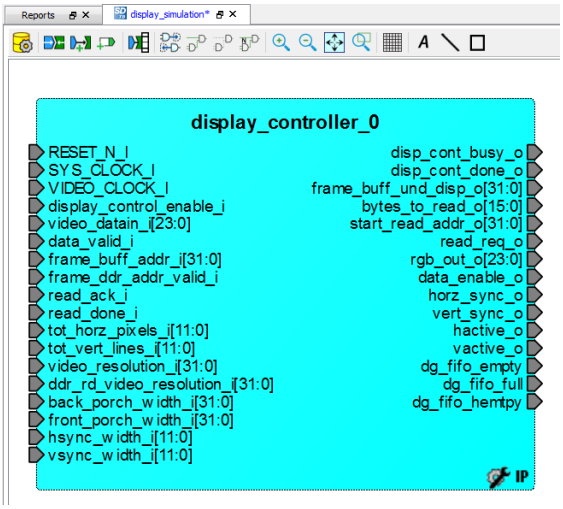
3. In the **Libero SoCCatalog** (**View** > **Windows** > **Catalog**), expand **Solutions-Video** and drag-and- drop the **Display Controller** core onto the **SmartDesign testbench** canvas as shown in the following figure.

Figure 7 • Display Controller in Libero SoC Catalog



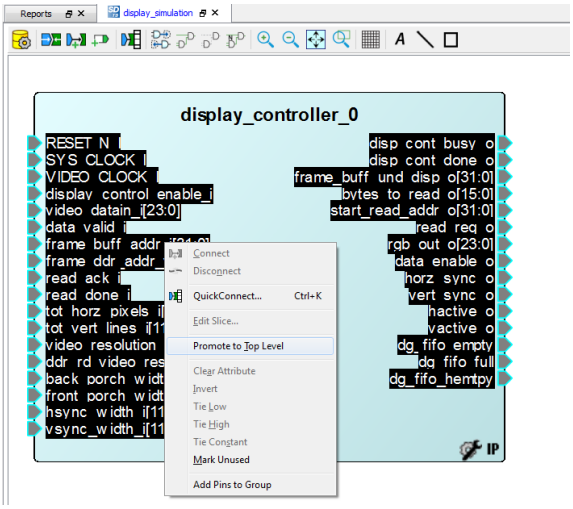
The core appears on the canvas as shown in the following illustration.

Figure 8 • Display Controller Core on SmartDesign Testbench Canvas



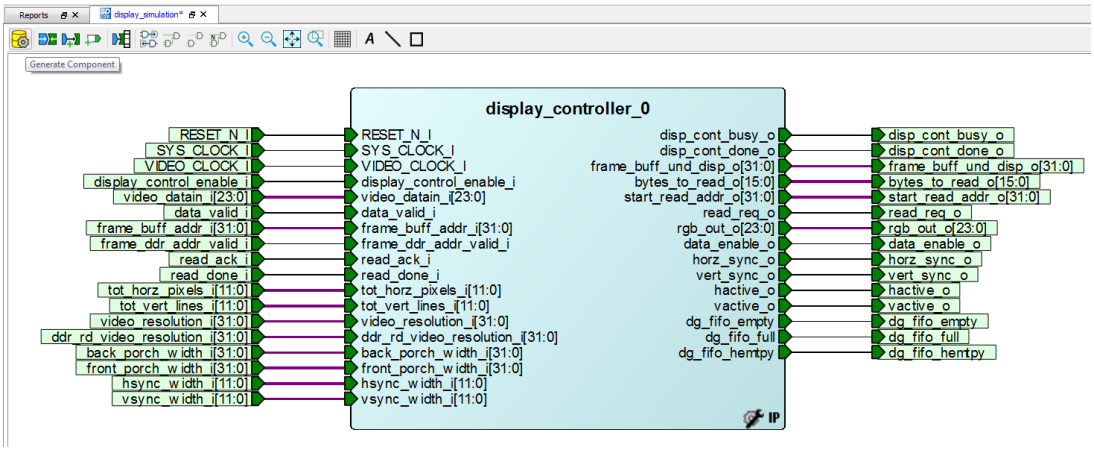
4. Select all the ports of the core, right-click, and select **Promote to Top Level** as shown in the following illustration.

Figure 9 • Promote to Top Level



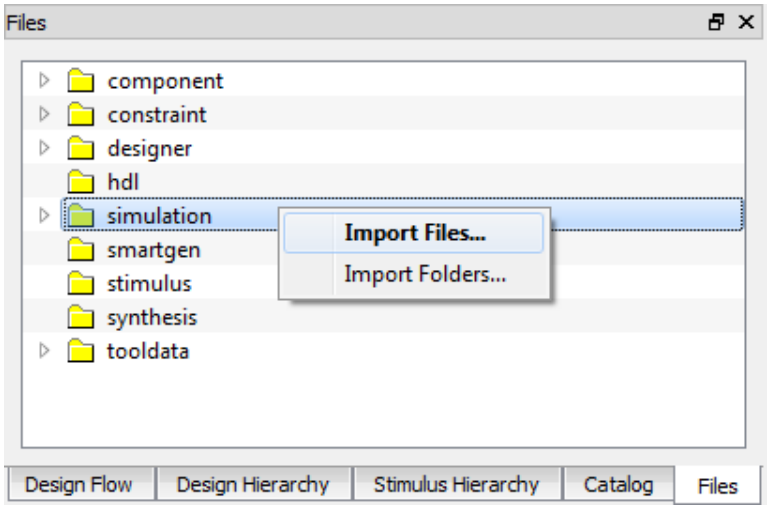
5. Click **Generate Component** from the SmartDesign toolbar as shown in the following illustration.

Figure 10 • Generate Component



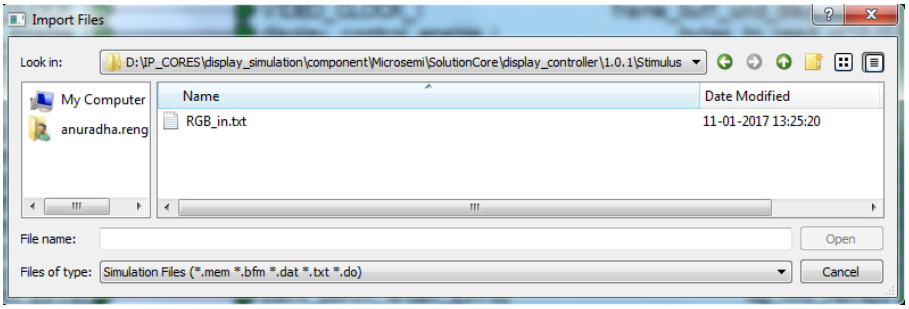
6. In the **Libero SoC Files** window, from **Files** tab, right-click **simulation** and click **Import Files...** as shown in the following illustration.

Figure 11 • Import Files



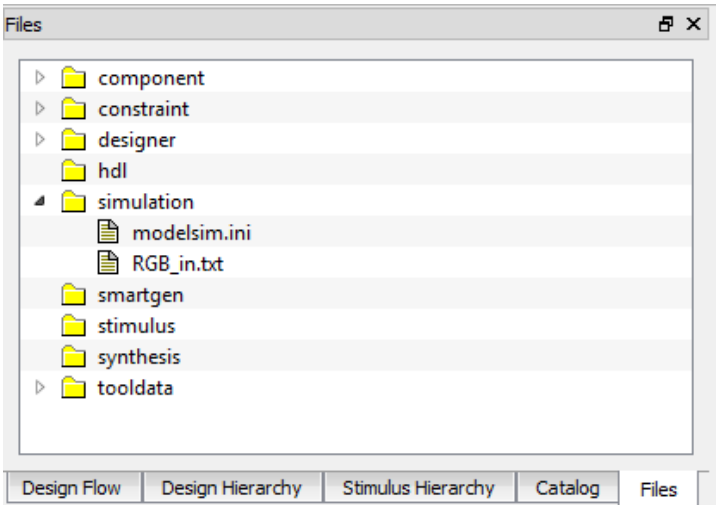
7. Do one of the following:
   - To import the sample testbench input image, browse to sample testbench input file to the stimulus directory, and click **Open** as shown in the following figure. The sample RGB_in.txt file is provided with the testbench at `..\Project_name\component\Microsemi\SolutionCore\display_controller\0.0\Stimulus`
   - To import a different image, browse to the folder containing the image file, and click **Open**.

Figure 12 • Input Image File Selection



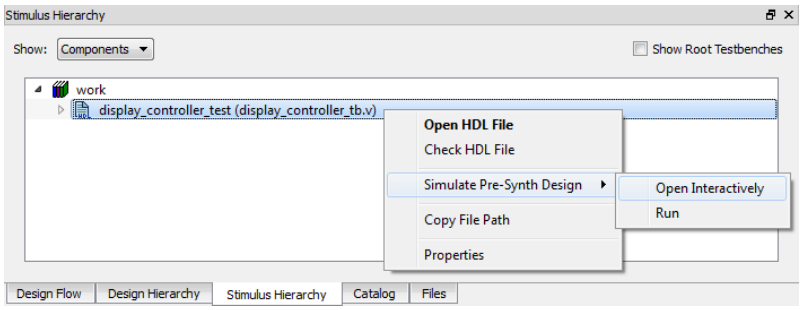
The input image file appears in the simulation directory as shown in the following figure.

Figure 13 • Input Image File in Simulation Directory



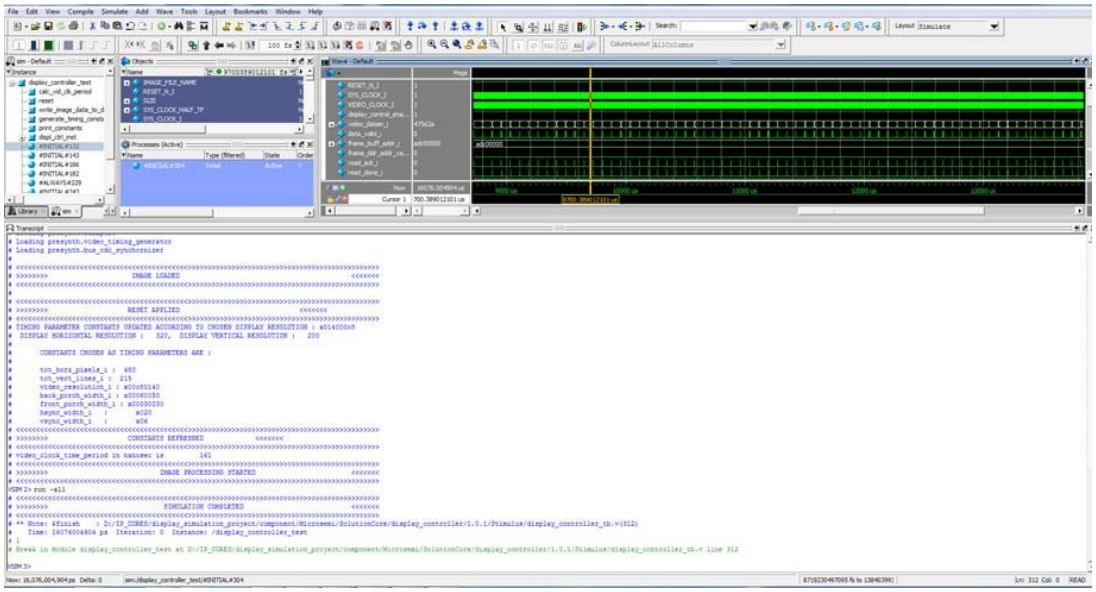
8. On **Stimulus Hierarchy** tab, right-click **display_controller_test** (display_controller_tb.v) testbench file and click **Open Interactively** from **Simulate Pre-Synth Design**. It simulates the core for one frame.

Figure 14 • Simulating Testbench



The ModelSim tool appears with the test bench file loaded on to it as shown in the following figure.

Figure 15 • ModelSim Tool with Display Controller Testbench File



If the simulation is interrupted because of the runtime limit in the DO file, use the `run -all` command to complete the simulation. After the simulation is completed, the test bench output image file appears in the simulation folder (**View** > **Files** > **simulation**).
For more information about updating the testbench parameters, see Testbench Configuration Parameters (see page 7).

3.5 Resource Utilization

The display controller is implemented in the SmartFusion®2 system-on-chip (SoC) FPGA (M2S150T-1FC1152 package) and PolarFire FPGA (MPF300TS_ES - 1FCG1152E Package). The following table lists the resources utilized by the FPGA.

Table 4 • Resource Utilization Report

Resource	Usage
DFFs	913
4-Input LUTs	1442
MACC	0
RAM1Kx18	12, supports up to horizontal resolution of 1920
RAM64x18	0

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

50200649