UG0693

User Guide

Image Edge Detection

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 3.0

In revision 3.0 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 2.0

In revision 2.0 of this document, the steps to simulate the core using testbench were added. For more information, see Testbench (see page 5).

1.3 Revision 1.0

Revision 1.0 was the first publication of this document

2 Introduction

Image Edge Detection is a basic tool, which is used in image processing for feature detection and extraction. The aim is to identify points in a digital image where the brightness of an image changes sharply and find discontinuities. The purpose of edge detection is to significantly reduce the amount of data in an image and preserve the structural properties for further image processing. In a gray level image, the edge is a local feature with in a neighborhood separate regions. The gray level is more or less uniform with in different values on the two sides of the edge. For a noisy image, it is difficult to detect edges as both edge and noise contains high frequency contents, which results in blurred and distorted images.

Image Edge Detection, using Sobel operator is a classical algorithm in the field of image and video processing for the extraction of object edges. Image This works on the premise of computing an estimate of the first derivative of an image to extract edge information. By computing the x and y direction derivatives of a specific pixel against a neighborhood of surrounding pixels, it is possible to extract the boundary between two distinct elements in an image. Due to the computational load of calculating derivatives using the squaring and square root operators, fixed coefficient masks have been adopted as a suitable approximation in computing the derivative at a specific point. In the case of Sobel, the masks used are, as shown in the following figure.

Figure 1 • Sobel Operator Horizontal and Vertical Kernels

Vertical Mask

-1	0	1
-2	0	2
-1	0	1

Horizontal Mask

1	2	1
0	0	0
-1	-2	1

These kernels can be combined together to find the absolute magnitude of the gradient at each point. The gradient magnitude is computed using:

$$G = \sqrt{Gx^2 + Gy^2}$$

Typically an approximate magnitude is computed using:

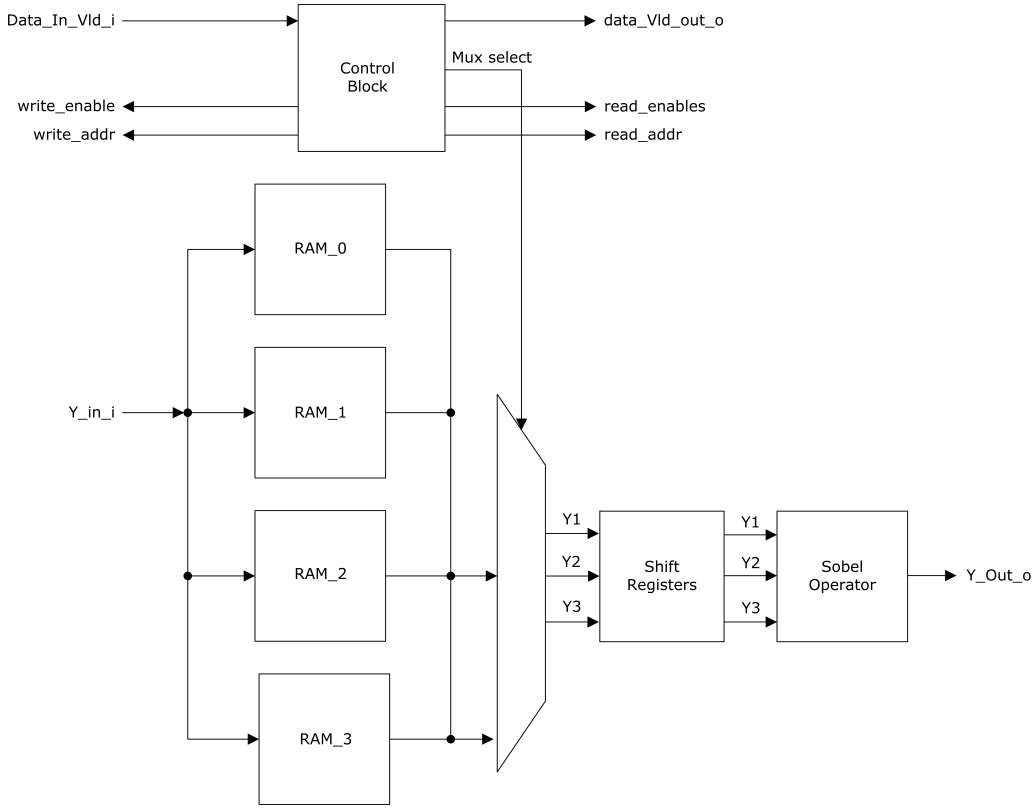
$$|G| = |Gx| + |Gy|$$

This is much faster to compute. The Sobel operator has the advantage of simplicity in calculation.

3 Hardware Implementation

The following figure shows the Image Edge Detection block diagram.

Figure 2 • Image Edge Detection Block Diagram



The incoming data-stream is filled into 4 1-line buffers one by one. Each buffer stores 1 complete horizontal video line. In this design, the Sobel filter is applied on a 3x3 matrix, hence, 3 lines of video are used to form the 3x3 window for Sobel operator.

When the 3rd buffer is filled by 3 pixel values, the read process is initiated.

The Sobel operator is only applied on the Y channel; the Cb and Cr are tied to 0x80 as we find out edges only on a gray input image. For the Y channel, 3 pixels from each of the 3 video lines are read into 3 shift-registers. These 3 shift-registers form the 3x3 2-D array for Edge calculation. The shift registers are applied as input to the Sobel operator. Sobel operator is nothing but convolving the input 3x3 window with horizontal and vertical kernel; summing up both gradients to get the absolute magnitude, which sends out after limiting it to the boundaries. The output of the Sobel operator is updated into the output register. The new column of pixel is shifted into the shift-register with the oldest data being shifted-out.

The 3x3 window moves from left-to-right horizontally and top-to-bottom vertically for a frame.

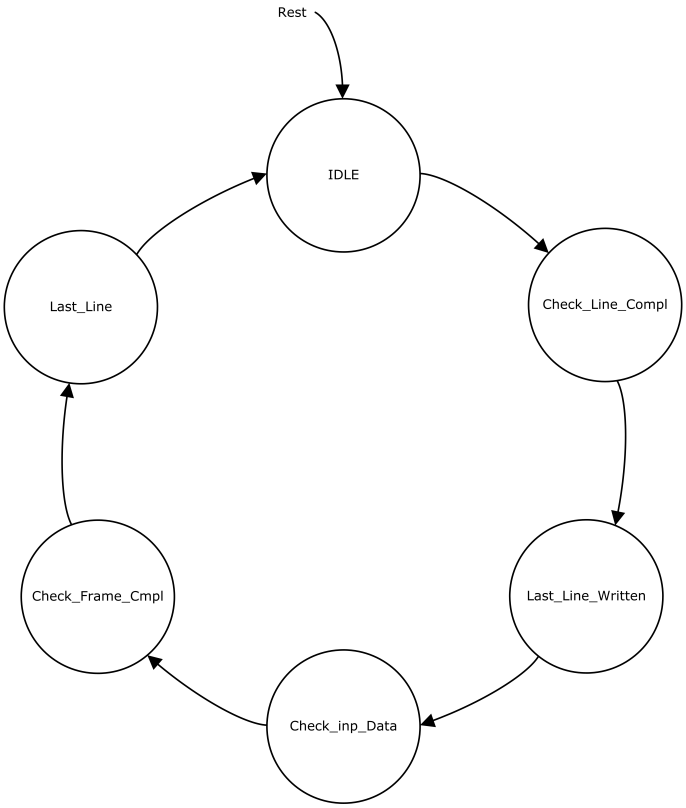
3.1 FSM Implementation

The finite state machine (FSM) of Image Edge Detection has the following states while implementing:

- IDLE STATE: When the module is set to reset, the FSM will be in an idle state and waits for the s_r_read_ready signal to move to the next state.
- CHECK_LINE_COMPL STATE: In this state, the FSM will wait until the input pixel count is equal to the display resolution to move to the next state.
- LAST_LINE_WRITTEN STATE: In this state, the FSM will wait for the last input line and moves to the next state.
- CHECK_INP_DATA: In this state, the FSM will wait for the fourth pixel of the last line written to the line buffer and then moves to the next state.
- CHECK_FRAME_COMPL STATE: In this state, the FSM will check if the output line counter is equal to the vertical resolution width and then moves to the next state.
- LAST_LINE STATE: In this state, the FSM will check if the last output pixel is equal to the last line and then moves to the next state (that is, IDLE).

The following figure shows the Image Edge Detection FSM implementation.

Figure 3 • Image Edge Detection FSM



3.2 Inputs and Outputs

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

Table 1 • Input and Output Ports of Image Edge Detection

Signal Name	Direction	Width	Description
RESET_n_I	Input		Active low asynchronous reset signal to design
SYS_CLK_I	Input		System clock

Signal Name	Direction	Width	Description
Y_in_i	Input	[(g_DATAWIDTH) - 1 : 0]	Luminance input channel
Data_In_Vld_i	Input		This signal is set when input data is valid
Y_Out_o	Output	[(g_DATAWIDTH) - 1 : 0]	This signal will give convolved output channel
data_Vld_out_o	Output		All output channels data valid output

3.3 Configuration Parameters

The following table lists the description of the configuration parameters used in the hardware implementation of Image Edge Detection. They are the generic parameters and can vary based on the application requirements.

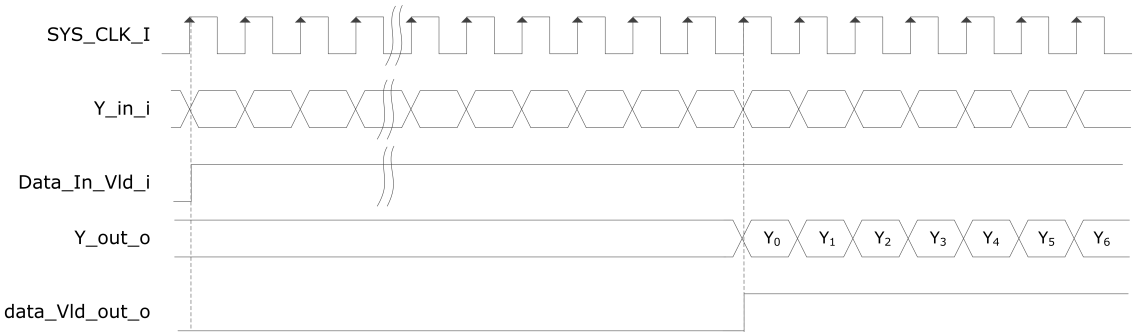
Table 2 • Configuration Parameters

Name	Description
g_DATAWIDTH	Defines Y data bit width
g_X_RES_WIDTH	Defines resolution bit width
g_DISPLAY_RESOLUTION	Defines the horizontal display resolution
g_VERT_DISPLAY_RESOLUTION	Defines the Vertical resolution data width

3.4 Timing Diagrams

The following figure shows the timing diagram of Image Edge Detection.

Figure 4 • Timing Diagram



3.5 Testbench

A testbench is provided to check the functionality of Image Edge Detection core. The following table lists the parameters that can be configured according to application.

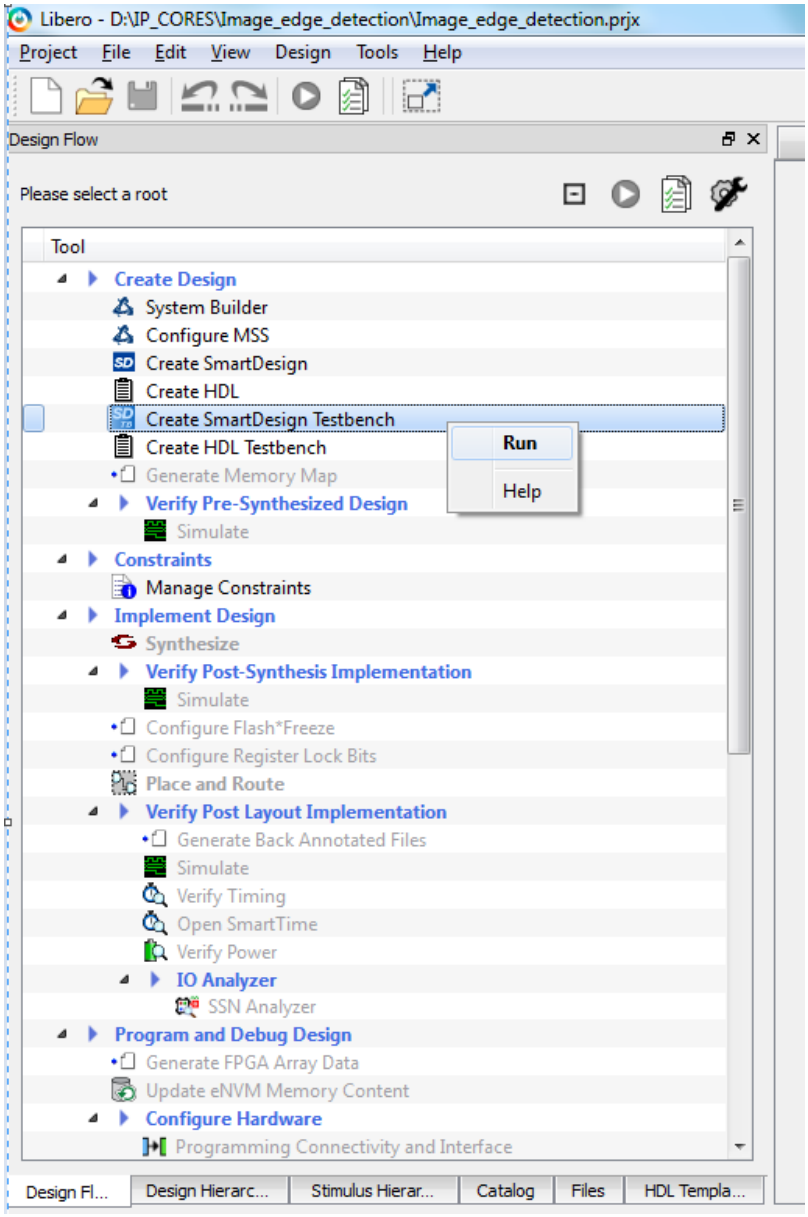
Table 3 • Configuration Parameters

Name	Description
CLKPERIOD	Clock Period
HEIGHT	Height of the image
WIDTH	Width of the image
WAIT	Amount of delay after each line of input image
DATA_BIT_WIDTH	Bit width of data (RGB)
IMAGE_FILE_NAME	Input image name

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

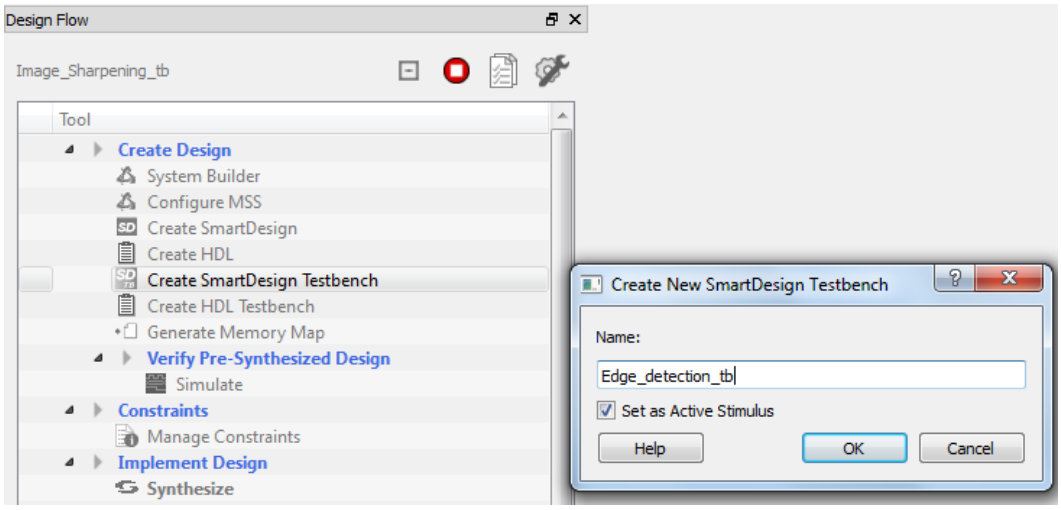
1. In the **Design Flow** window, expand **Create Design**. Right-click **Create SmartDesign** testbench and click **Run**, as shown in the following figure.

Figure 5 • Create SmartDesign Testbench



2. Enter a name for the SmartDesign testbench, and click **OK**.

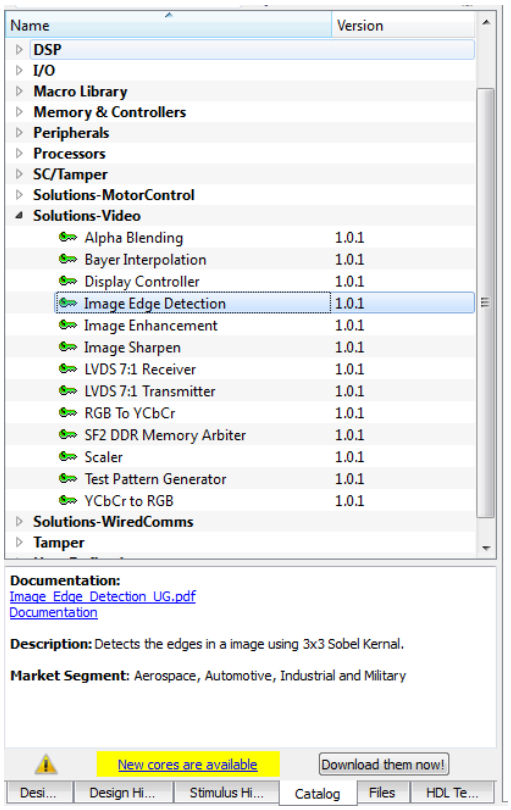
Figure 6 • Create New SmartDesign Testbench



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

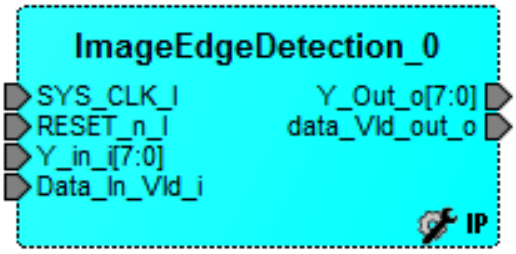
3. In the **Catalog** window, expand **Solutions-Video**, and drag the **Image Edge Detection** core onto the SmartDesign testbench canvas.

Figure 7 • Image Edge Detection



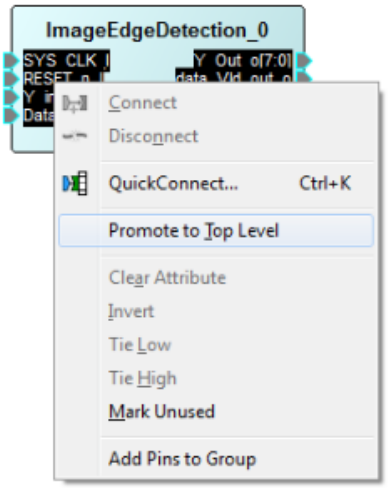
The core appears on the canvas, as shown in the following figure.

Figure 8 • Image Edge Detection Core on SmartDesign Testbench Canvas



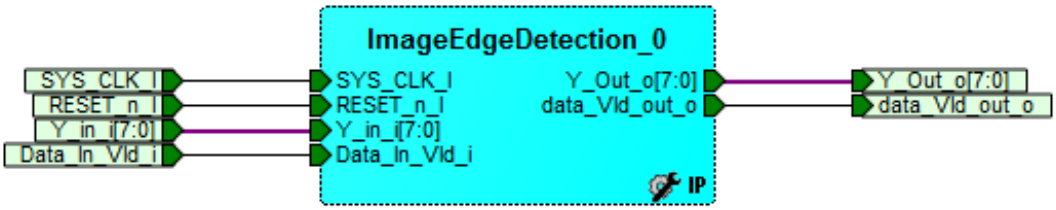
4. Select all the ports of the core, right-click, and then click **Promote to Top Level**, as shown in the following figure.

Figure 9 • Promote to Top Level Option



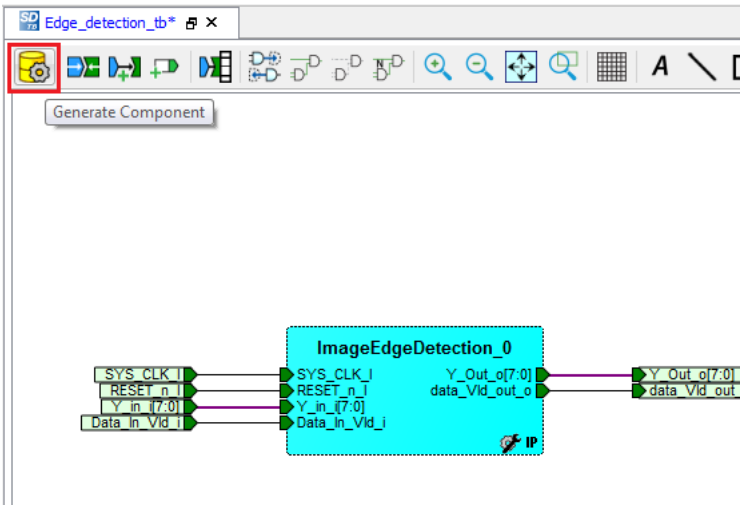
The ports are promoted to the top-level, as shown in the following figure.

Figure 10 • Image Edge Detection Ports Promoted to Top Level



5. Click the Generate Component highlighted in the following figure.

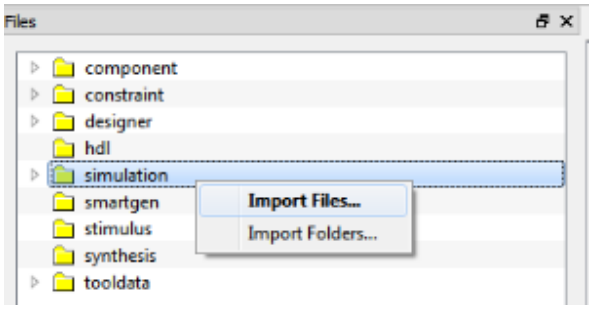
Figure 11 • Generate Component Icon



A sample image is available in the Stimulus hierarchy.

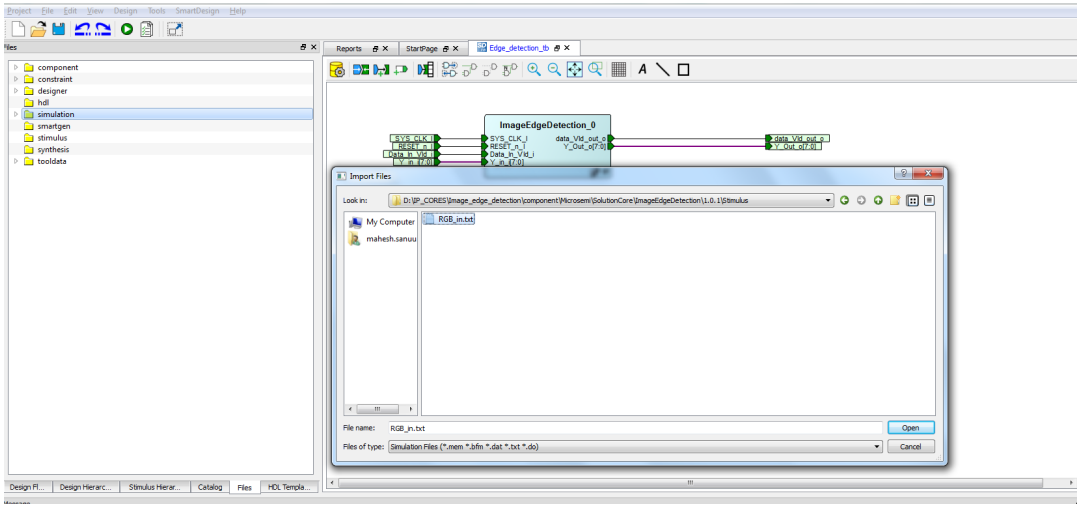
6. In the **Files** window, right-click the **simulation**, and click **Import files...**, as shown in the following figure.

Figure 12 • Import Files Option



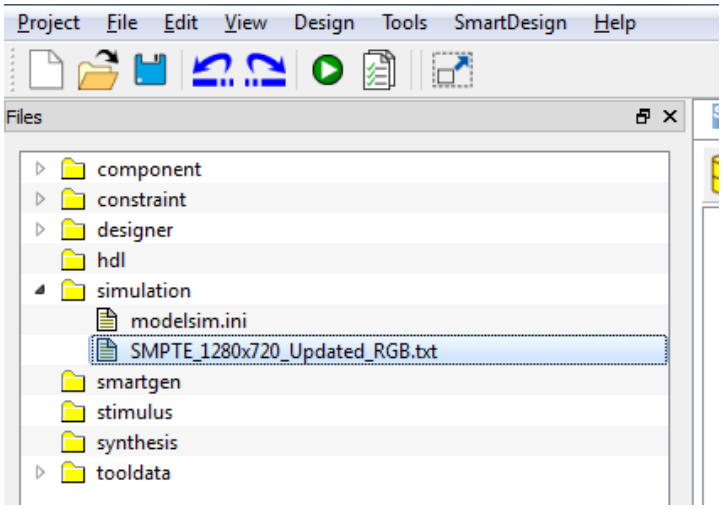
7. Do one of the following:
   • To import a different image, browse to the folder containing the image file, and click **Open**.
   • To import the sample testbench input image, browse to sample testbench input file to the stimulus directory, and click **Open**, as shown in the following figure.

Figure 13 • Input Image File Selection



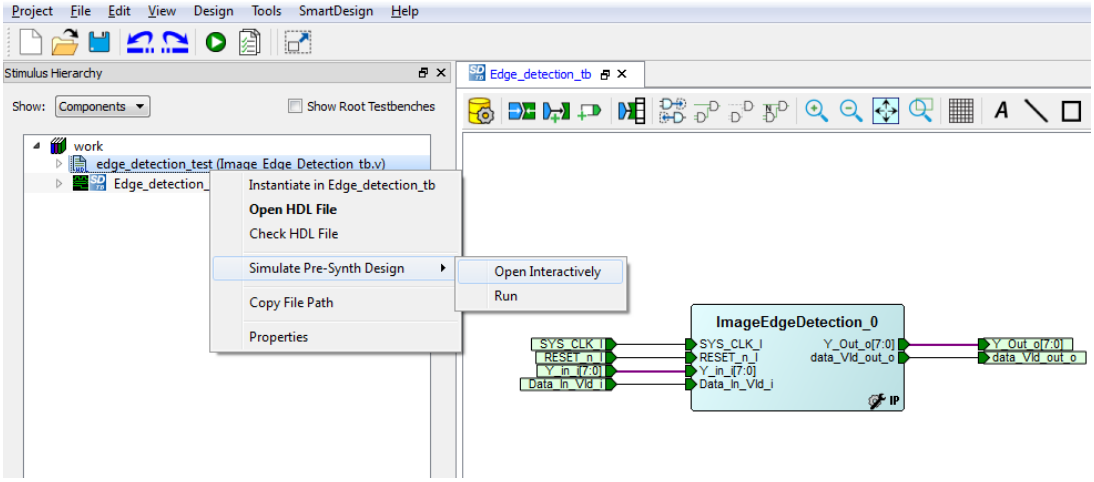
8. Import this file into the Simulation folder in the Files folder.

Figure 14 • Input Image File in Simulation Directory



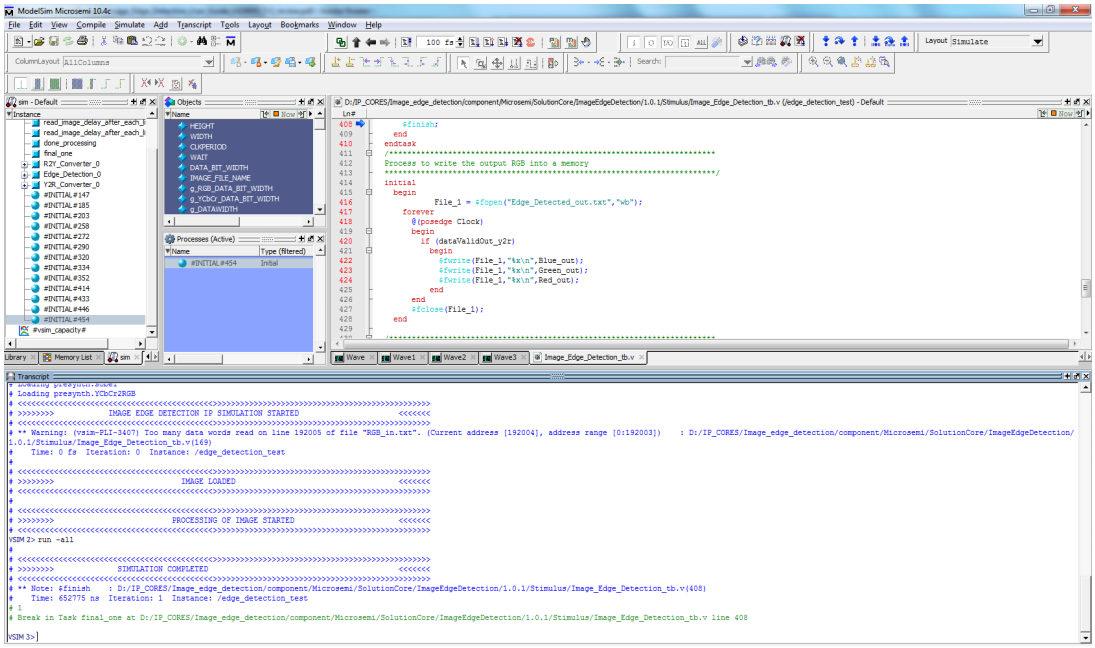
9. In the Stimulus Hierarchy, expand **Work**, and right-click the **Image_edge_detection_test** file (`Image_Edge_Detection_tb.v`). Select **Simulate Pre-Synth Design**, then click **Open Interactively**.

Figure 15 • Open Interactively Option



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

Figure 16 • ModelSim Tool with Image Edge Detection Testbench File



If the simulation is interrupted because of the runtime limit in the DO file, use the `run -all` command in the transcript window to complete the simulation.

After the simulation is completed, the test bench output image file appears in the simulation folder.

3.6 Simulation Results

This section shows an image before and after being processed using the Image Edge Detection.

The following figure shows the input image.

Figure 17 • Input Image

The following figure shows the output image.

Figure 18 • Output Image

3.7 Resource Utilization

The image edge detection block is implemented on an M2S150T SmartFusion®2 System-on-Chip (SoC) FPGA in the FC1152 package) and PolarFire FPGA (MPF300TS_ES - 1FCG1152E package).

Table 4 • Resource Utilization Report

Resource	Utilization
DFFs	703
4-Input LUTs	814
MACC	0
RAM1Kx18	5
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

50200693