

Minilab2 Report

Implementation

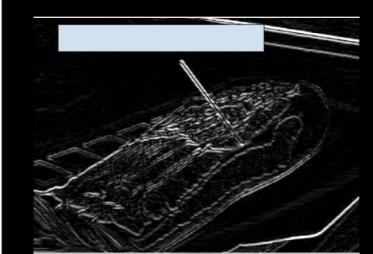
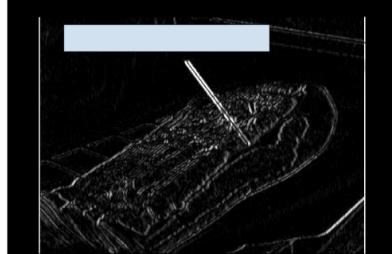
Grayscale: Originally our grayscale took output from the RAW2RGB IP to feed into our module for averaging of the values. The final version takes raw camera data, interpolates it, and then averages for a better grayscale.

Row Buffer: The row buffer buffers a single line of 640 12 bit values implemented as a FIFO with read and write counters.

Edge Detection: We perform a convolution with provided kernels, and the method is determined by a switch. We input the grayscale pixels and buffer them. Once there is enough then convolution can begin. There are 9 registers to store the 3x3 matrix. Then signed arithmetic is executed with the sobel kernels before determining which output to use. Finally they are sent to SDRAM in the demo pipeline.

Testbench

To test our implementation, we wrote a script that converts png images into hex files. We took sample images, converted them to hex, and ran our grayscale and edge detection implementation on them, converted the hex files back to pngs, and looked at the resulting images to determine if it was successful or not. Below is the original image and the resulting edge detected outputs. The png to hex is in the repo but hex to png is not currently.

Original Image	Vertical Detect	Horizontal Detect
		

Quartus Report

The Quartus report can be found as Minilab2.flow.rpt under the Minilab2 folder.

```
+-----+  
; Flow Summary ;  
+-----+  
; Flow Status ; Successful - Tue Feb 10 19:10:43 2026 ;  
; Quartus Prime Version ; 25.1std.0 Build 1129 10/21/2025 SC Lite Edition ;  
; Revision Name ; DE1_SoC_CAMERA ;  
; Top-level Entity Name ; DE1_SoC_CAMERA ;  
; Family ; Cyclone V ;  
; Device ; 5CSEMA5F31C6 ;  
; Timing Models ; Final ;  
; Logic utilization (in ALMs) ; 741 / 32,070 ( 2 % ) ;  
; Total registers ; 1364 ;  
; Total pins ; 226 / 457 ( 49 % ) ;  
; Total virtual pins ; 0 ;  
; Total block memory bits ; 72,144 / 4,065,280 ( 2 % ) ;  
; Total DSP Blocks ; 0 / 87 ( 0 % ) ;  
; Total HSSI RX PCSS ; 0 ;  
; Total HSSI PMA RX Deserializers ; 0 ;  
; Total HSSI TX PCSS ; 0 ;  
; Total HSSI PMA TX Serializers ; 0 ;  
; Total PLLs ; 1 / 6 ( 17 % ) ;  
; Total DLLs ; 0 / 4 ( 0 % ) ;  
+-----+
```

Problems

Our main issue came with integrating our modules with the existing pipeline. We found 8 evenly spaced lines of pixels on the display that were not grayscale and instead some seemingly random color. The color wasn't uniform within or among the lines. Our best guess as to the cause is some mistiming with the SDRAM as without our modules the lines were not present. To solve this we directly inserted our module into the demo files and after doing that the lines disappeared.

Files Made or Modified:

row_buffer.sv
edge_detect.sv
RAW2GRAY.v
DE1_SoC_CAMERA.v

(To solve our vertical line problem we switched to some of the demo files instead of the System Builder generated ones).

Files Copied:

The SDRAM files

camera IP
line_buffer1.v