ASIC for Real-Time Edge Detection on VGA Video

Jacob Peterson

Daryl Warner

Aaron Pettit

Trent Bennett

University of Utah

Abstract

Presented is an ASIC for real-time edge detection in an image which reveals the edges of objects from a colored video input stream. The input containing the *RGB* pixel values from a digitized VGA-like interface are converted into a grayscaled colorspace which is then convolved with a Sobel filter to compute the gradient between pixels. The ASIC produces a video output stream visualizing the edges of objects in real-time. The design process of the ASIC is explored and simulation results are shown.

Introduction and Motivation

The emerging autonomous vehicle market uses computer vision applications implemented in hardware to increase overall performance in an especially time-sensitive system. One component of vehicle autonomy in the context of computer vision is to use an edge detection algorithm on an image or video feed to identify, for instance, the line markings on a road a car is driving on. This project implements an ASIC for real-time edge detection in an image which reveals the edges of objects from a colored video input stream using the VGA interface.

Related Works

There are several works related to this project, some of which influenced the final ASIC design. Reference [3] demonstrates edge-detection on a video feed using OpenCV (a popular computer vision software library), but with a rather slow algorithm. This project opted for a more efficient method of real-time edge detection that could be implemented in hardware yielding synchronous, real-time performance. Another work related to real-time edge detection ASICs is found in [4] from researchers at the University of New South Wales. They found that “A custom VLSI design for [an edge detection] algorithm using parallel piplined architecture is realisable.” Similar ideas from this research are present in this project.

Design Description

The video input and output streams of this ASIC are designed to comply with the VGA interface industry standard of 640 x 480 @ 60 Hz, but instead of an analog input, a digital equivalent is used [1]. This ASIC assumes an ADC to decode the VGA analog inputs to the digital equivalent and also assumes a DAC to encode the VGA digital outputs to the analog equivalent. The edge detection algorithm convolves a 3x3 matrix consisting of two Sobel filters throughout the input image. One Sobel filter compares the gradient of the rows adjacent to the target pixel and the other compares the columns adjacent. The total magnitude of the two filters is the resulting gradient of the target pixel. If this magnitude is large enough, an edge is detected. Since this requires a single channel colorspace to function properly, the video input stream is converted from the RGB colorspace to a grayscale colorspace. This conversion uses the Luma coefficients to model the human eye's response to red, green, and blue light intensity. The Luma coefficients used in this ASIC are derived from the CCIR 601 luminance conversion: $Y'_{601} = 0.299R' + 0.587G' + 0.114B'$ [2]. The sum-of-divisors method is used for division of each RGB channel since the coefficients are static. The divisors used are powers of two so that right bit-shifting by the $log2$ of each divisor yields real-time division. The quotient of each channel is added together and stored in an intermediary buffer array which the Sobel filter uses to read pixel data from once enough pixels have been stored. This ASIC uses two clock domains with one clock synchronizing the VGA interface and the other synchronizing the core logic. The core logic takes at least 8 clock cycles to filter one input pixel which is determined by the grayscale, Sobel filter, and line buffer state machines.

Block Diagram

Fig. 1. shows the overall block diagram of this design. First, the digitized VGA interface serves as the video source feed which is immediately converted to a grayscale colorspace and stored in a line buffer. This buffer acts as an intermediary between the source feed and the output feed due to the edge detection algorithm requiring valid pixel data directly adjacent to the target output pixel. As a result of this, when the first two rows and the first three columns of the second row within the buffer are filled with valid pixel data, the system then yields the digitized VGA video output feed which is synchronous with the input feed and contains the pixels with the edge detection algorithm applied.

Results

The first working design featured a fully compliant VGA input and output interface with digital RGB channels and a depth of 8-bits per color channel. However, due to $180\ nm$ technology limitations and *TSMC* design constraints, the original design was heavily modified to comply with the foundry specifications and to pass DRC and LVS. Through several design and RTL revisions, simulation iterations, and PnR corrections, the ASIC was successfully implemented using the $180\ nm$ technology library from *TSMC*. Consequently, the internal line buffer uses 3-bits instead of the original 8-bits to greatly reduce its relative core area utilization. A video output bit depth of 4 bits per channel is used instead of 8 bits to reduce the number of I/O ports on the chip. Finally, the standardized VGA pixel clock frequency is halved to improve poor WNS due to large parasitic delay of technology. These modifications still provide a fully functional edge-detection ASIC with a slight compromise in performance and output quality. Throughout the prototyping process, several working simulation outputs and measurements were captured and reported. Fig 2. shows the pre-synthesis simulation output of the given original image, resulting grayscaled image, and final edge-detected image respectively. The original image used here is colorful rocks with high contrast containing several object edges to demonstrate the grayscale conversion retaining image luminance and the edge detection algorithm correctly visualizing many edges. Fig 3. visualizes the noise introduced in the pre-synthesis simulation output when lowering the bit depth of the internal line buffer. Fig 4. 5. and 6. show the pre-synthesis, post-synthesis, and post-PnR simulation outputs respectively using a 4-bit internal line buffer implementation. Fig 7. shows post-PnR simulation using a 3-bit internal line buffer with the top level I/O waveforms visible. As seen from the simulation outputs in Fig 6. and Fig 7, there is increased noise in the output due to the low bit depth used. Fig 8. and Fig 9. shows post-PnR DRC and LVS reports respectively of the final design. The DRC failures are caused by low polysilicon density and connecting the *VSS* to the sealring in our design. Fig 10. 11. and 12. report the power, timing, and area of the final chip respectively. Finally, Fig 13. presents the final die visualization from the CAD software.

Conclusion

This paper presents an ASIC for real-time edge detection of a VGA video stream input. The motivation of the project stemmed from the emerging autonomous vehicle market in which real-time processing and computer vision play a vital role. Research consisted of finding optimal methods of colorspace conversion and edge-detection using various algorithms

that can be implemented in hardware. The conceptual design is presented along with the step-by-step process of creating a video output stream consisting of an image revealing edges of objects from a video input stream. The design of this ASIC included several iterations of simulation and PnR to acquire a valid tape-out for potential fabrication. Finally, the results of the final design are shown.

References

[1] SECONS Ltd. "VGA Signal Industry Standard Timing," 2008. Online.

[2] "Rec. 601 luma versus Rec. 709 luma coefficients," 2020. Online.

[3] "OpenCV Python Tutorial - Find Lanes for Self-Driving Cars (Computer Vision Basics Tutorial," 2018. Online.

[4] B. Majumdar, N. Sankarayya, and A. Majumdar, "An ASIC design for edge detection in real time," Microprocessing and Microprogramming, vol. 36, no. 2, pp. 55–69, 1993. Online.

[5] Github Open Source repository "EdgeDetectionASIC," 2020 Online.

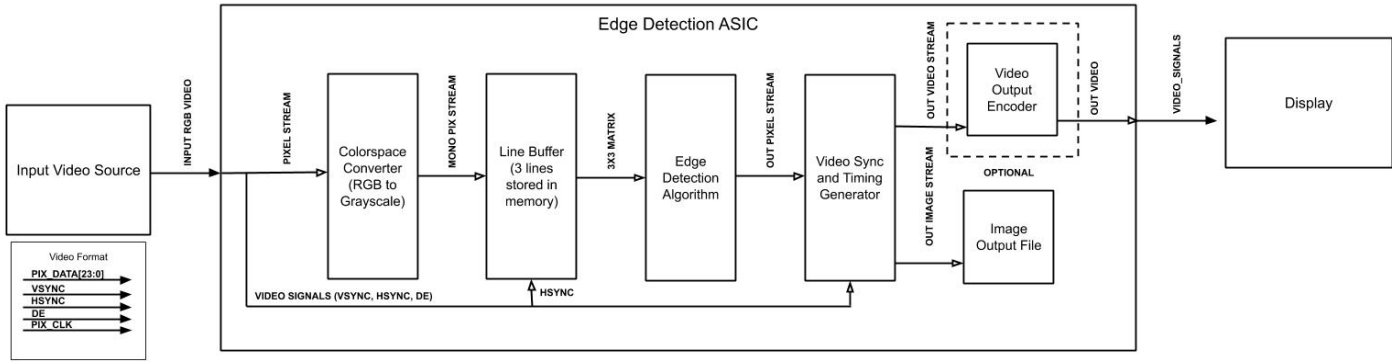


Figure 1: The block diagram the EdgeDetectionASIC.

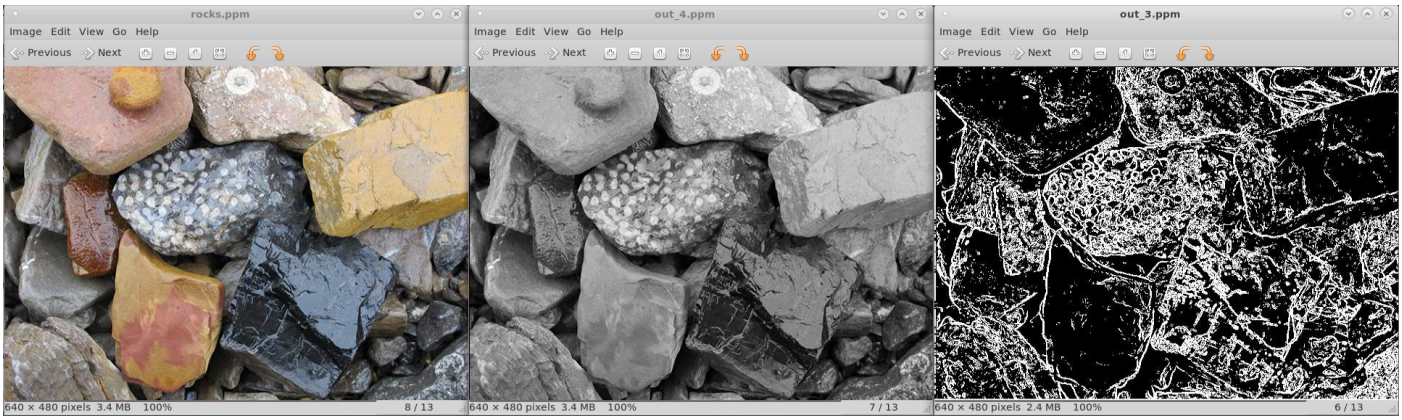
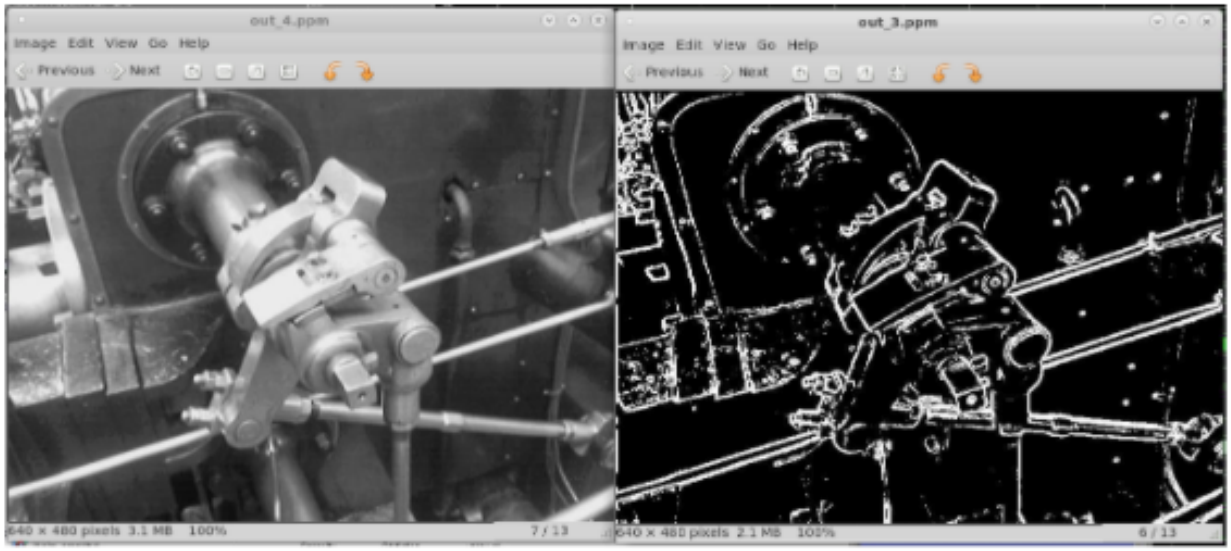


Figure 2: Pre-synthesis simulation output of original image, grayscaled image, and edge-detected image respectively.

Pixel depth of 8:



Pixel depth of 4:

Pixel depth of 3:

Figure 3: Pre-synthesis simulation output using grayscaled color bit depth of 8, 4, and 3 respectively.

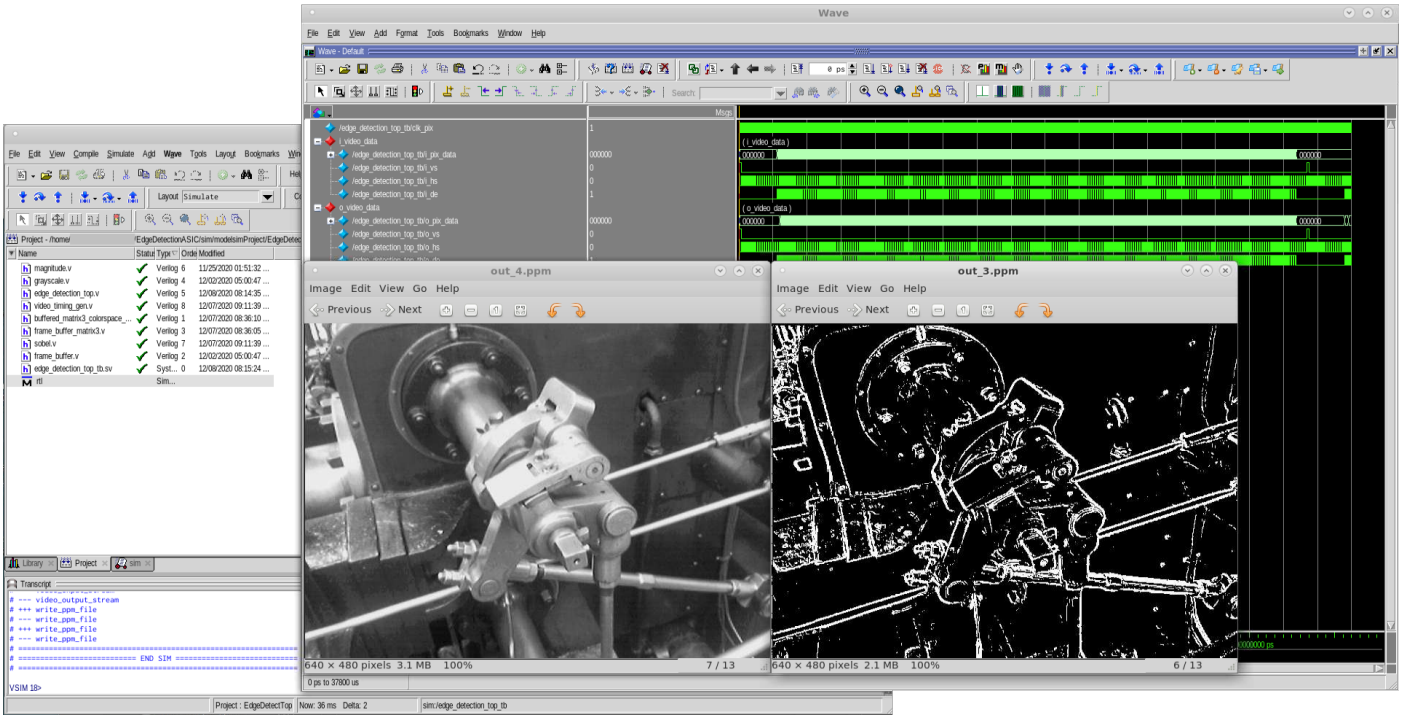


Figure 4: Pre-synthesis simulation output.

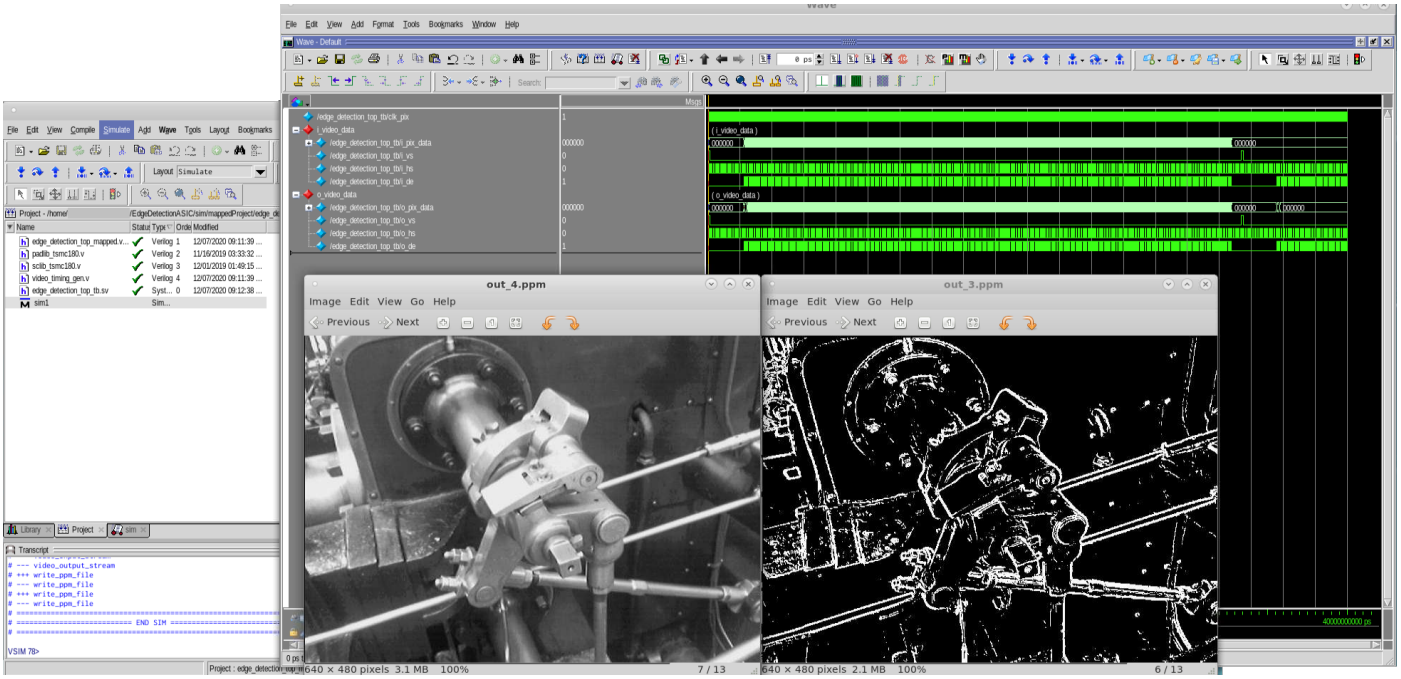


Figure 5: Post-synthesis simulation output.

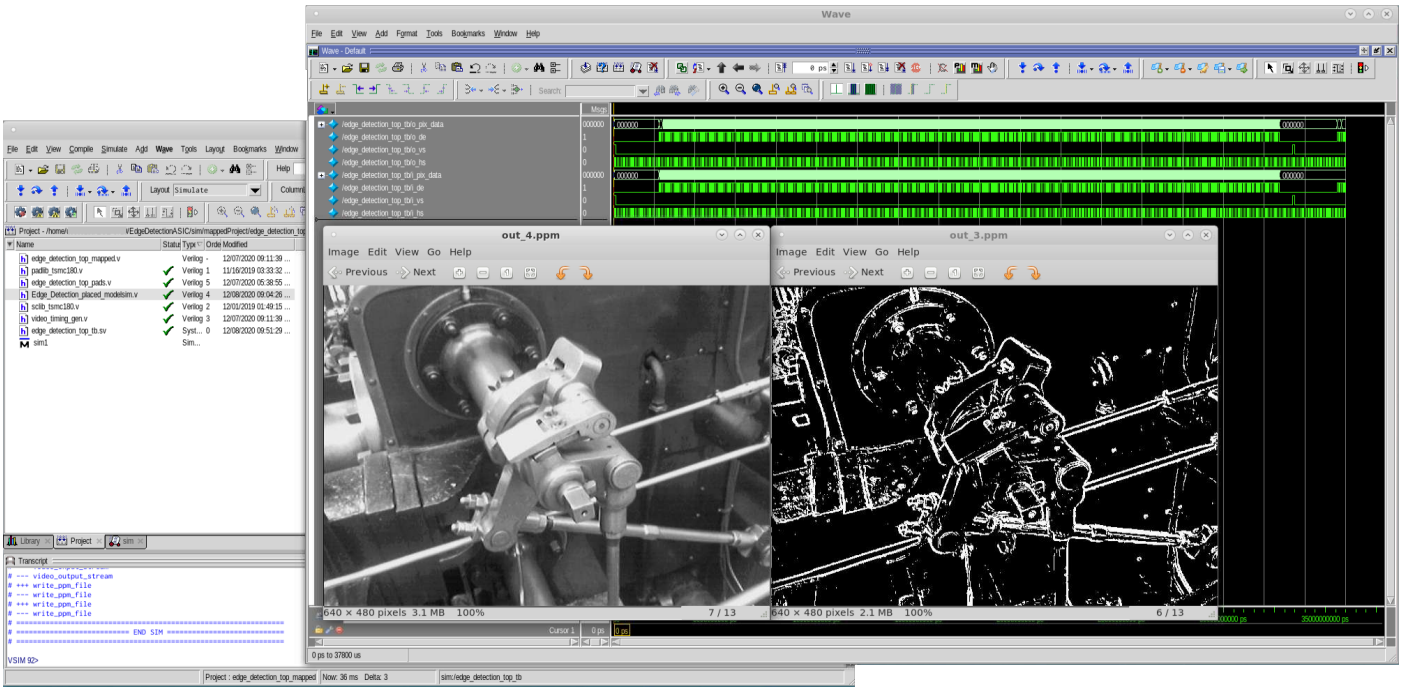


Figure 6: Post-PnR simulation output of with 4-bit internal line buffer.

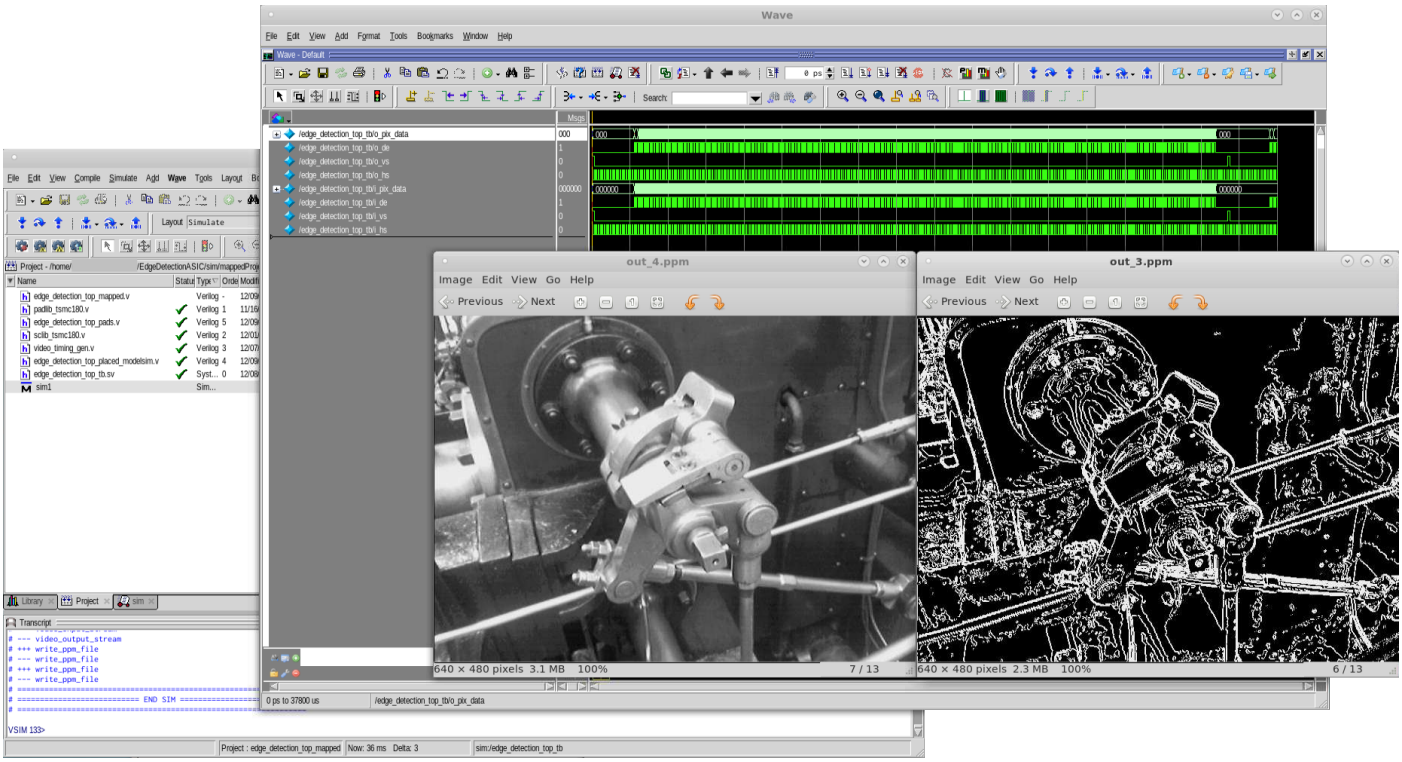


Figure 7: Post-PnR simulation output of with 3-bit internal line buffer.

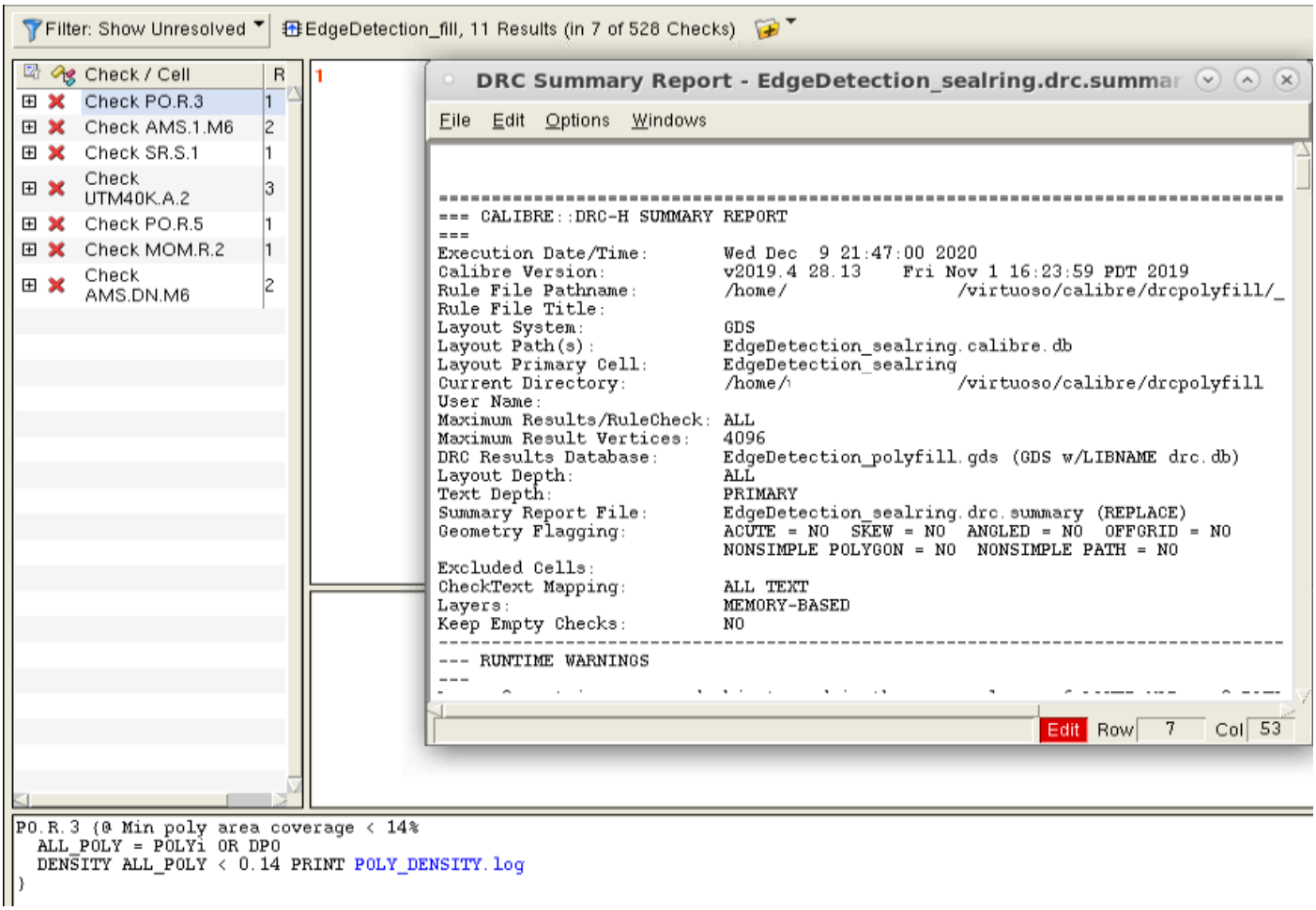


Figure 8: Screenshot of post-PnR DRC validation.

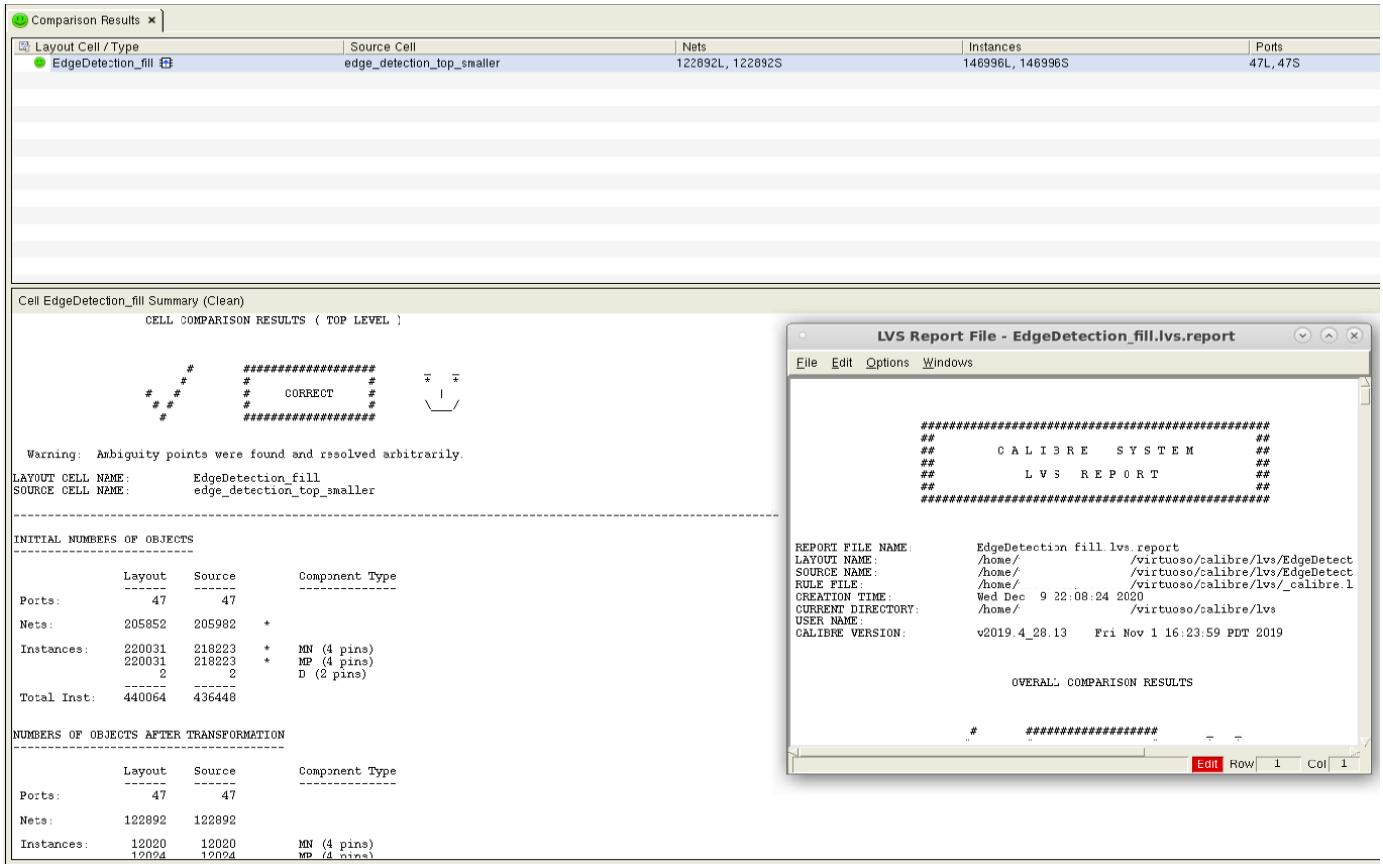


Figure 9: Screenshot of post-PnR LVS validation.

```
Global Operating Voltage = 1.8
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =   6.0400 mW   (86%)
  Net Switching Power  =   1.0054 mW   (14%)
                         ---------
Total Dynamic Power    =   7.0454 mW  (100%)

Cell Leakage Power     =   1.2963 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000              0.0000           0.0000  (   0.00%)
memory             0.0000           0.0000              0.0000           0.0000  (   0.00%)
black_box          0.0000           0.0000              0.0000           0.0000  (   0.00%)
clock_network  1.6008e-03       1.4529e-03              0.3405       3.0541e-03  (   0.04%)
register           5.7329       3.0648e-03             416.1527           5.7363  (  81.41%)
sequential         0.0000           0.0000              0.0000           0.0000  (   0.00%)
combinational      0.3055           1.0009             879.8292           1.3072  (  18.55%)
--------------------------------------------------------------------------------------------------
Total              6.0399 mW        1.0055 mW     1.2963e+03 nW         7.0465 mW
1
```

Figure 10: Post-PnR Power Report.

```
Endpoint:   dut/iBMCC/iGrayscaledFrameBufferMatrix3/q_o_pixel_matrix_reg[45]/D
(^) checked with  leading edge of 'I_CORE_CLK'
Beginpoint: dut/iBMCC/q_frame_buffer_write_enable_reg/Q
(v) triggered by  leading edge of 'I_CORE_CLK'
Path Groups: {I_CORE_CLK}
Analysis View: wc
Other End Arrival Time          0.012
- Setup                         0.278
+ Phase Shift                  10.000
= Required Time                 9.734
- Arrival Time                 74.705
= Slack Time                  -64.971
     Clock Rise Edge                   0.000
     + Clock Network Latency (Prop)   -0.084
     = Beginpoint Arrival Time        -0.084
```

Figure 11: Post-PnR Timing Report.

Depth	Name	#Inst	Area (um^2)
0	edge_detection_top_pads	83660	2379415.5328
1	dut	83609	1175450.5728
2	dut/iSobel	1107	16558.3936
2	dut/iBMCC	82096	1151088.2432
2	dut/iVidGen	345	7029.0304
3	dut/iSobel/add_0_root_add_0_root_add_94_2	50	731.0016
3	dut/iSobel/sub_66_S2	63	902.2272
3	dut/iVidGen/add_129	27	443.4304
3	dut/iSobel/add_1_root_add_0_root_add_89_2	50	731.0016
3	dut/iSobel/add_0_root_add_0_root_add_99_2	50	731.0016
3	dut/iSobel/add_1_root_add_0_root_add_94_2	50	731.0016
3	dut/iSobel/sub_67	63	902.2272
3	dut/iSobel/add_0_root_add_0_root_add_104_2	50	731.0016
3	dut/iSobel/sub_64_S2	63	902.2272
3	dut/iBMCC/sub_221	27	333.6704
3	dut/iSobel/add_1_root_add_0_root_add_99_2	50	731.0016
3	dut/iSobel/sub_65	63	902.2272
3	dut/iVidGen/add_133	27	443.4304
3	dut/iSobel/add_0_root_add_0_root_add_89_2	50	731.0016
3	dut/iBMCC/iGrayscale	217	3547.4432
3	dut/iSobel/add_69	42	564.1664
3	dut/iSobel/add_1_root_add_0_root_add_104_2	50	731.0016
3	dut/iBMCC/iGrayscaledFrameBufferMatrix3	81404	1138283.6416
4	dut/iBMCC/iGrayscale/add_3_root_add_1_root_add_76_9	42	625.632
4	dut/iBMCC/iGrayscale/add_0_root_add_1_root_add_76_9	43	649.7792
4	dut/iBMCC/iGrayscale/add_8_root_add_76_9	27	412.6976
4	dut/iBMCC/iGrayscaledFrameBufferMatrix3/sub_73	26	316.1088
4	dut/iBMCC/iGrayscale/add_1_root_add_1_root_add_76_9	28	417.088

Figure 12: Post-PnR Area Report.

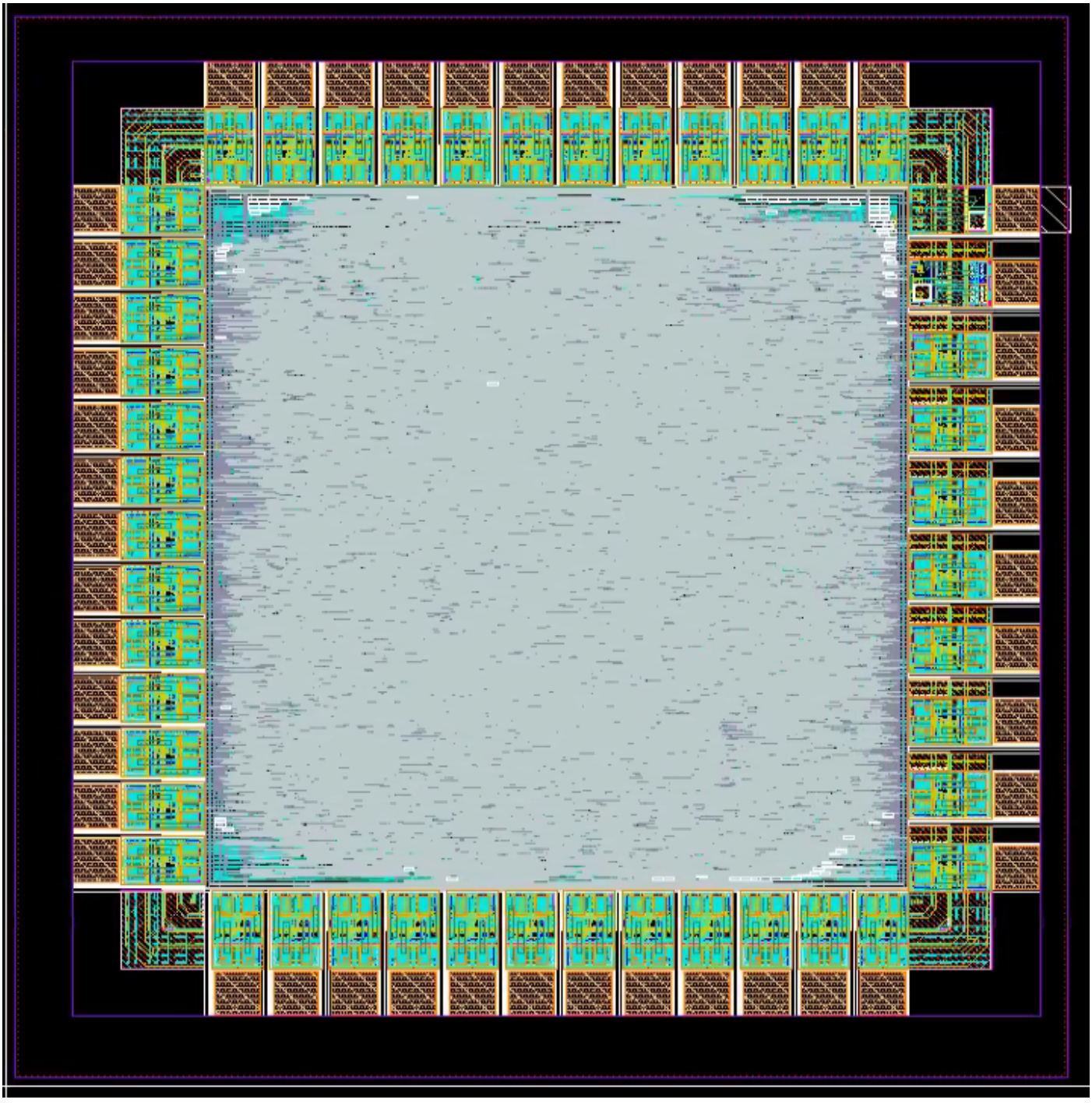


Figure 13: The final die visualization from the CAD software.