SIMD Analog Processor Array Using Time-Encoded Computation

Micah Jeffries
California Polytechnic State University - SLO
micah.j.jeffries@gmail.com

***Abstract*—The proposal of this project is to implement an analog processing element, but to do so using digital circuits and time-encoded computation. Time-encoded computation leverages the strengths of analog circuits, namely low power consumption and high information density while also avoiding its weaknesses such as hardware design difficulty, low precision, limited scalability, and programming difficulty. The datapath components of the processing element include a time domain register file and ALU (arithmetic logic unit). It can also interface with exterior digital circuitry via a data and I/O bus using serial communication. Instructions are received via a separate input signal, and it performs the necessary operations in the time domain which include load, store, add, subtract, etc. The time-based processing elements are implemented in a 16x16 array architecture, operating in the single instruction multiple data (SIMD) mode. The SIMD concurrent processor architecture is ideally suited to implementing low-level image processing algorithms. To evaluate the performance of the processor array, some assembly code is developed to allow the array to perform the Sobel operator on an input image. The processor array is built on the TSMC 180 nm process which demonstrates the functionality and performance benefits of time-encoded computation. Using time-encoded computation, the processor array is able to provide 25.6 GOPS (Giga operations per second) with an average power consumption of 120 mW and an area of 10.5 square millimeters.**

***Index Terms*—Arithmetic logic unit (ALU), energy efficient computing, time-encoded computation, processing element (PE), single instruction multiple data (SIMD)**

I. Introduction

EVER since the 1960's, the modern digital revolution has kept pace with Moore's law which predicts that the areal density of silicon transistors within digital chips will double each year. However, some fundamental physical laws have begun impeding the efforts of scientists and engineers to keep pace with Moore's law. Two of the foremost issues that have presented themselves have been lowering the consumption of power and shrinking transistors down to microscopic levels. Transistors are themselves consuming more energy as process technologies decrease in size as shown in figure 1. In addition, applications such as machine learning and image processing require the use of dedicated digital hardware accelerators, such as the GPU, which requires more energy. The use of these accelerators and application-specific hardware units present many challenges with energy efficiency. As a result, scientists and engineers are on the hunt for alternative solutions that will optimize for energy efficiency and high information density.

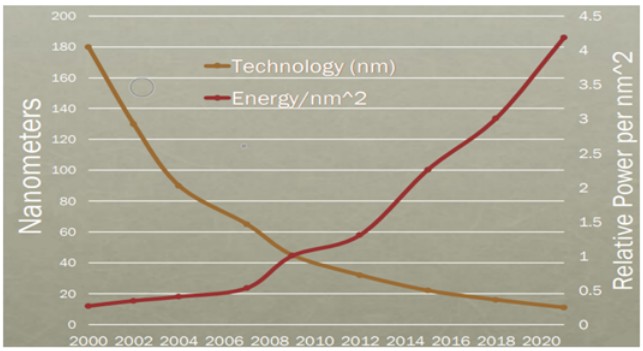

Fig. 1: Power Consumption vs. Process Technology Size [1].

To address the above challenges, this paper presents the design of a time-based processing element which is implemented in an SIMD array architecture to perform low-level image processing applications. Section 2 presents some previous work and other competing processor designs which provide performance benchmarks such as energy consumption and processing speed. Section 3 outlines the overall architecture of the mixed-signal time-domain computing processing element as well as its datapath components such as the register file and arithmetic logic unit (ALU). Section 4 explores the SIMD architecture of the processor array as well as the concept behind the Sobel operator which is used to evaluate the performance of the array. Section 5 contains the experimental results of the processing element and the processor array when compared to its performance benchmarks. Finally, section 6 contains some concluding remarks as well as comments regarding future work to be done on this project.

II. Related Work

One technique that has been researched to improve energy efficiency is analog computing [2]. Analog computing does have a reduced energy consumption when compared to its digital counterpart, as well as the ability to store more information on a single wire with a continuous voltage. There are also some elementary functions such as addition and subtraction that don't require dedicated circuitry to implement these functions. However, there are drawbacks to analog computing which include hardware design difficulty, low precision, limited scalability, and programming difficulty [3]. Analog computing also has a naturally high sensitivity to noise since all the data is represented as continuous voltage as opposed to discrete values.

An alternative to analog computing is time encoding computation. The typical method for translating an analog signal to a digital signal is to sample the analog signal at synchronous intervals and to record the amplitude of the sample as digital information. The traditional sampling representation of an analog signal uses the amplitude domain to represent the signal information. Time encoding machines, on the other hand, encode the signal information in the time domain with asynchronous circuits [4]. In other words, the information of the signal is represented by different time delays as opposed to the traditional method where the information is represented by different amplitudes.

Time encoding has emerged as an alternative to existing technologies for computing with purely a digital or analog implementation. The mixed signal implementation of time encoding combines the advantages of both digital and analog computing. Signal representation in the time domain allows for the information to be processed with digital circuitry which allows for the benefits of scalability and compatibility with conventional digital embedded systems. On the other hand, time encoding also allows for the processing of analog signals within the processor which retains for the benefits of analog computing, namely high information density and energy efficiency [5].

There are many competing digital and analog designs for the SIMD vision chip processor array, many of which would be negatively affected with the successful completion of this project. To ensure that this project can make a profit, it must be on an equal footing of performance with the family of competing vision chip products. To set the benchmark of performance for the analog processor array, this paper identifies different areas of performance from 5 different vision chips. The digital vision chip proposed by [6] is a programmable vision chip for real-time vision applications. The analog vision chip proposed by [7] is a mixed-signal programmable chip for high-speed vision applications. The digital vision chip proposed by [8] is a new vision chip architecture for high-speed target tracking. The digital vision chip proposed by [9] features mixed asynchronous/synchronous processing techniques. The final point of performance comparison comes from the analog vision chip proposed by [10]. The design for this chip features and indeed inspires many of the architectural features for this project which include similar datapath components (registers, ALU) for the PE.

III. Architecture

This paper proposes a fairly simple implementation of mixed-signal time-domain computing processing element. Figure 2 provides an architectural schematic of the processing element which contains a register file and an arithmetic logic unit (ALU). Two registers from the register file, as specified from the two read addresses, can be read during a single instruction and fed into the inputs of the ALU which can perform arithmetic operations such as addition and subtraction. The output of the ALU is fed back into the register file at the register with the specified write address. The input to the register file is also configured to receive an immediate value or a value from a neighbor processing element. The I/O and control signals for each processing element are the same and are generated off chip.

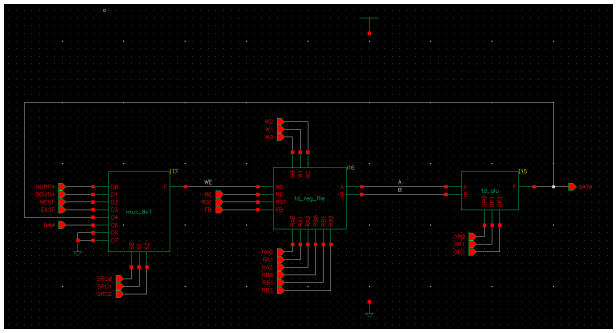

Fig. 2: Schematic of Time-Based Processing Element.

The rest of this section discusses the datapath components of the processing element which include the register file, ALU, and multiplexers. The final subsection explores the physical layout of the processing element built with TSMC 180 nm process technology. It is also worthy of note that many of the elements in this design come previous work prepared by Guzman's thesis [11] who is also a Cal Poly alumni. Credit to his work will be explained through the upcoming sections.

A. Register File

Figure 3 provides the schematic for the register file which contains eight registers and 5 multiplexers. The two multiplexers on the output side select the output registers according to the specified read addresses. The top multiplexer on the left directs the written data to the register according to the specified write address. The read enable signal also requires its own multiplexer since not all eight registers need to be read from for each instruction, just the ones specified from the read addresses. The only control signal which does not require a multiplexer is the reset input since this signal is meant to wipe the value from all the registers simultaneously.

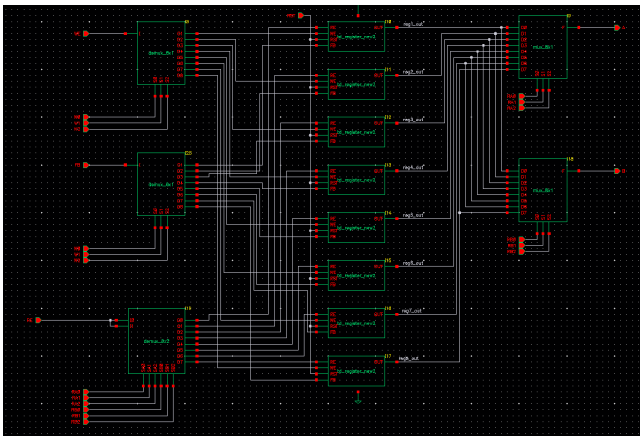

Fig. 3: Schematic of the Register File.

The design for the time domain register circuit is based on the flip-flop described in [12]. It can be described as a ring oscillator attached to a chain of gated delay buffers. Figure 4 shows the schematic and timing diagram for the time-domain flip-flop. The buffers are enabled upon the activation of either the read enable (RE) or write enable (WE) signals. The buffer chain propagates a logical zero when the register is being written to and a logical one when the register is being read from. The reset control signal is included in the third segment to allow for the buffer chain to be cleared while performing a read. Figure 5 shows the schematic for the register originally designed by Guzman [11] which contains 32 gated buffers.

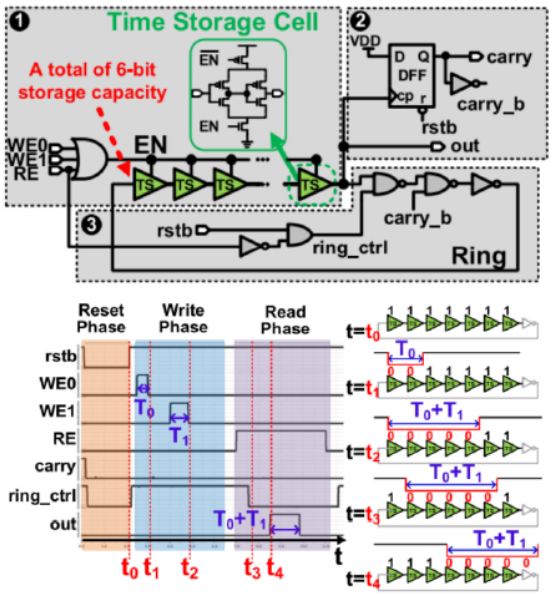

Fig. 4: Schematic and Timing Diagram of Time-Domain Flip-Flop [12].

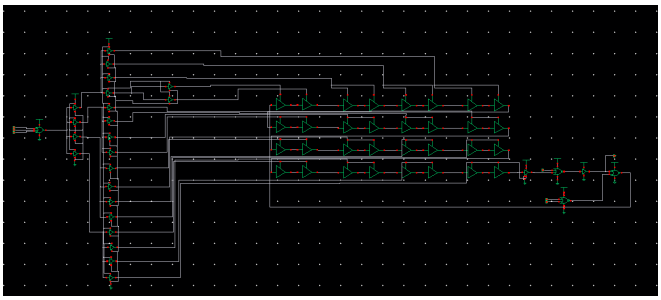

Fig. 5: Schematic of Time-Domain Register.

Some modifications were made to the design to improve the efficiency of the circuit and perform operations within the context of the processing element as a whole. Figure 6 shows two such modifications made by Guzman [11] to the first segment of the circuit which enables the buffer chain. The OR gate includes the reset control signal as an additional input which allows the buffer chain to be cleared without having to activate the read enable signal. Additionally, a buffer tree was inserted to reduce the fan-out of the OR gate since the buffer chain introduces a large capacitive load.

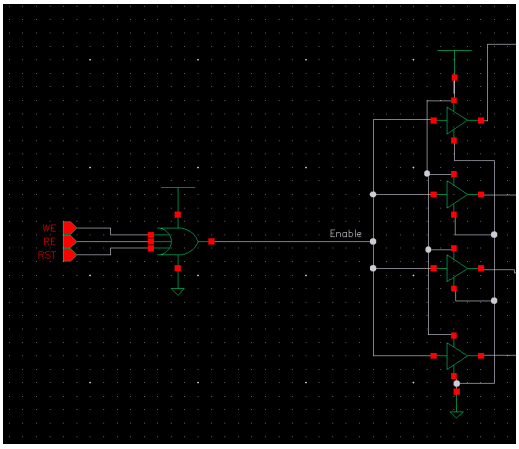

Fig. 6: Buffer Chain Enable Section.

Figure 7 shows the modifications made to the third segment of the circuit. First off, the OR gate at the end was replaced with a NOR gate which allowed the register to read the value from the register while not erasing the value from the buffer chain. In other words, the register can perform multiple non-destructive reads. An additional NOR gate was inserted with the RE and RST signals as inputs. This gate prevents the output signal from being activated when RST is enabled and when RE is disabled. Finally, an additional feedback (FB) control signal was inserted along with an AND gate to turn the destructive read feature on or off. Normally a non-destructive read is desirable but for some arithmetic operations such as subtraction it is useful to erase the value in the buffer chain and replace it with a new value.

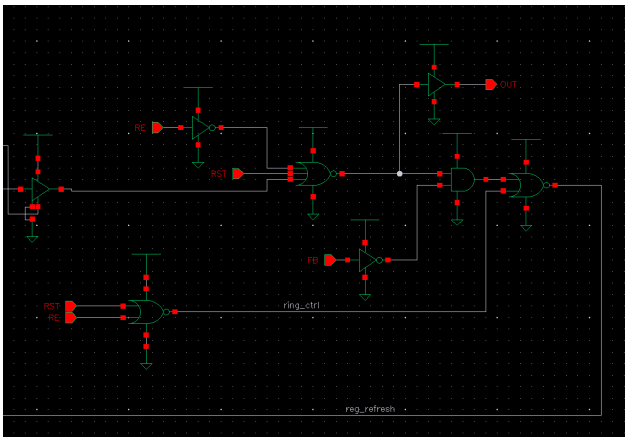

Fig. 7: Ring Control Section.

B. ALU

Figure 8 shows the circuit schematic for the ALU which contains the max, min, compare, and subtract operations. Each operation has its own time-domain circuit and takes

two inputs each with a variable time pulse. The outputs are fed into a multiplexer which selects the output based on the appropriate operation being performed. The selection is controlled by the OP control signal which is a 3-bit address. The multiplexer features two extra inputs for the instance in which the operation requires that the output be simply one of the original inputs. This feature is often used for the addition operation since addition can be carried out in the registers themselves.

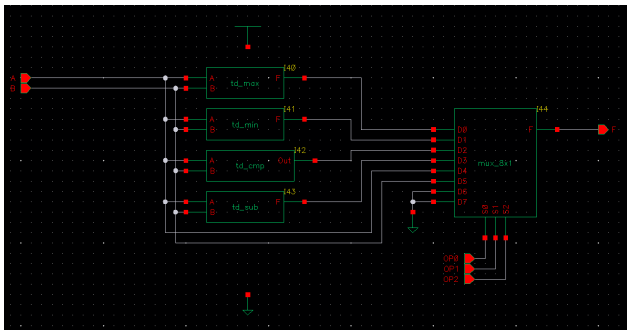

Fig. 8: Schematic of Arithmetic Logic Unit (ALU).

Figure 9 shows the most common ALU operations and their implementations with time-domain circuits. The subtract operation is implemented with a simple XOR gate. The output will only activate when one of the inputs is on while the other is off. The max operation is implemented with an OR gate which means that the output will remain activated for as long as either of the inputs remain high. This effectively extracts the input with the longest time pulse. The min operation is implemented with an AND gate which means that the output will only activate if both inputs are on. This effectively extracts the input with the shortest time pulse. The add operation is carried out in the time domain register because any input to the register will accumulate in its buffer chain. The comparison operation is implemented with feedback and a few logic gates. The output will only activate if the input A has a longer time pulse than input B. The only operation not featured in the ALU is the equal detection circuit.

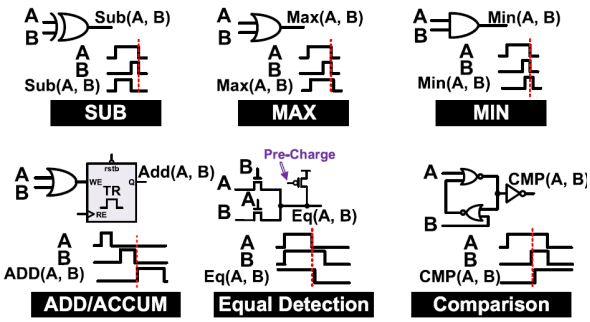

Fig. 9: Arithmetic Operations of the ALU and their time-domain implementation [12].

C. Multiplexers

This datapath component of the processor was designed with purely digital technology as there is yet no known multiplexer with a time-domain implementation. The multiplexer and demultiplexer are both key building blocks for the processing element, register file, and ALU as demonstrated in the previous sections. So a good design is critical for accurate operation and compatibility with time-domain signals.

There are some standard components in Cadence Virtuoso for multiplexers but this paper instead proposes a custom design. The main reason for this comes from the inherent noise variability in analog-based time-domain signals where the timing of the input and output signals are critical for computational accuracy. Thus, it is favorable in these conditions to insert some buffers which enhances the sharpness of the signal. Figure 10 shows the custom design for the four by one multiplexer and Figure 11 shows the custom design for the eight by one multiplexer. The logical design for each unit is equivalent for the standard implementation, however there are some inserted buffers for the input and output signals which is necessary since these signals are in the time-domain. The control signals are digital signals and thus do not require buffers to enhance their sharpness.

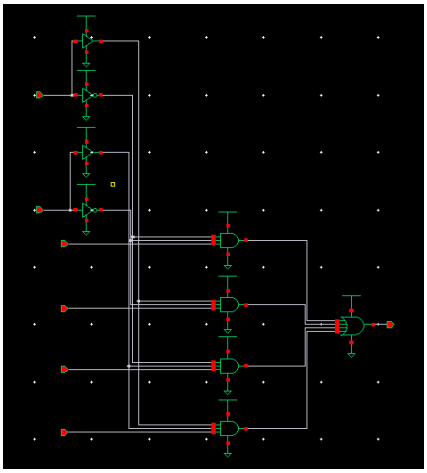

Fig. 10: Schematic of Four by One Multiplexer.

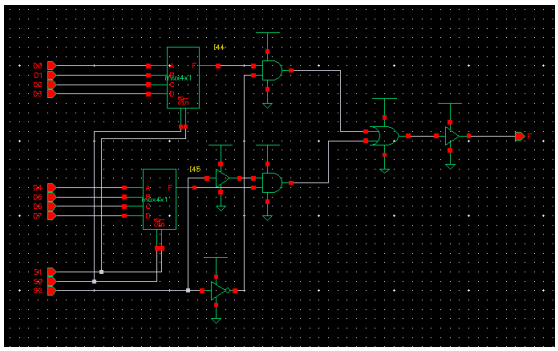

Fig. 11: Schematic of Eight by One Multiplexer.

This paper also proposes a couple custom designs for the demultiplexer which is required by the register file to route the appropriate control signals to its eight registers. Figure 12 shows the design for the eight by one demultiplexer which features buffers at the output to sharpen the time-domain signal and is otherwise the standard logical design. Figure 13 shows the custom design for another similar demultiplexer except that it features two inputs. This demultiplexer is used exclusively to route the RE signal to all the registers in the register file. Since two registers can be read from at any one time, the RE signal is fed into both of the inputs and is routed to two different registers depending on the register address inputs which serve as the control signals for the demultiplexer. This custom design also accounts for the case in which both register addresses are equivalent by inserting an OR gate at each output.

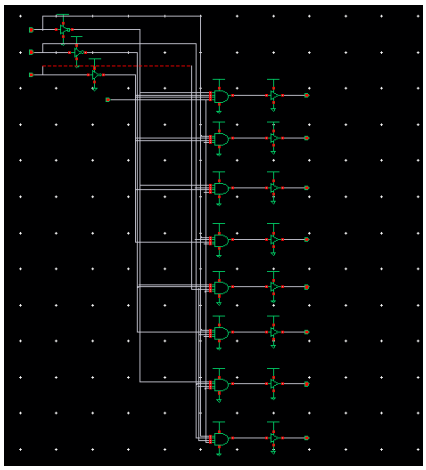

Fig. 12: Schematic of Eight by One Demultiplexer.

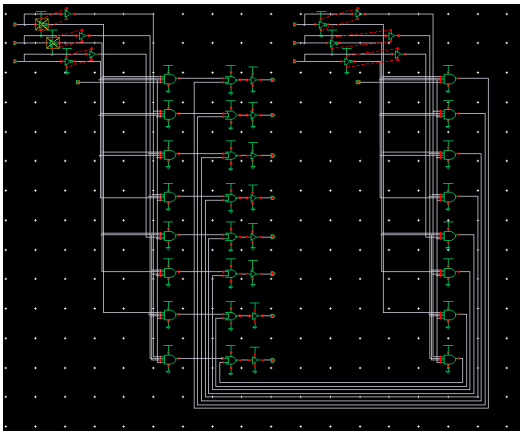

Fig. 13: Schematic of Eight by Two Demultiplexer.

D. Physical Layout

The architecture described thus far has all been designed in Cadence Virtuoso with TSMC 180 nm process technology. All the physical layouts for the standard digital logic gates are defined in Cadence and can be used to develop the physical layout of the processing element and its datapath components. There are programs in Cadence which allow for the automatic placement and routing for each component, yet these predefined algorithms would often not optimize for minimal area. In these cases, the placement for each component was done manually and then the automatic routing tool would be used.

Figure 14 shows the physical layout for the time-domain register which takes up an area of 2975 um^2. The bulk of the area comes from the gated buffer chain which consists of 32 gated buffers which can store a time-domain pulse up to 2 ns in duration. To decrease the size of the register would necessarily imply removing some buffers from the chain which would subsequently shorten the maximum pulse duration that the chain can store.

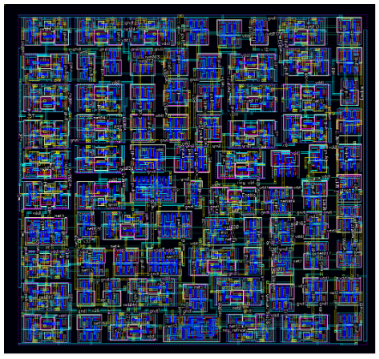

Fig. 14: Physical Layout of Time-Domain Register.

Figures 15 and 16 show the physical layout for the register file and the ALU respectively. Given that these are the two datapath components of the processing element, the register file takes up 84.1% of the total area while the ALU takes up only 3.6% of the total area. The rest of the area comes from the multiplexer which selects the input for the register file. The register file takes up an area of 28230 um^2 while the ALU takes up an area of only 1187 um^2. The amount of logic gates and physical area for implementing arithmetic operations in a computer processor is significantly better for time-domain circuitry than the traditional digital implementation. However, the design of the register demands that most of the physical space for the processor be allocated for registers.

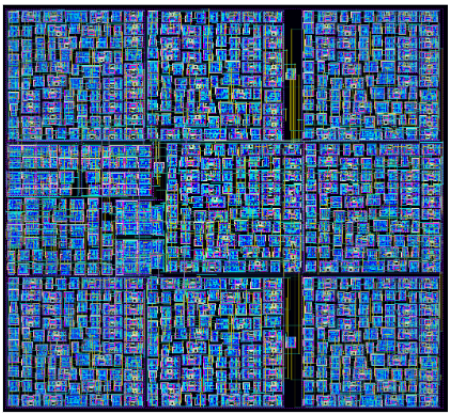

Fig. 15: Physical Layout of Register File.

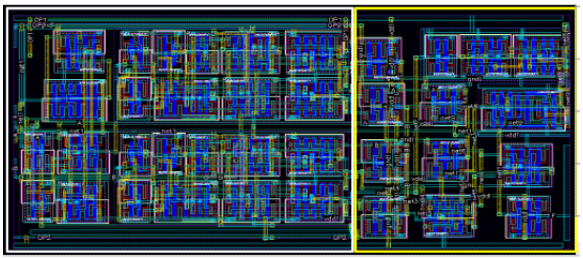

Fig. 16: Physical Layout of ALU.

Figure 17 shows the complete layout for the processing element which takes up an area of 33539 um^2 or 0.034 mm^2. Using this area, the area for the chip can be predicted by scaling up this area by the amount of processing elements that will be included in the chip. The SCAMP work was able to produce a similar chip with 16,384 (or an array of 128 by 128) parallel processing elements and a chip area of 54 mm^2 [10]. If this processing element were to be duplicated with 16,384 elements in a chip, the area of the chip would be around 550 mm^2 about 10x the size of the SCAMP chip. Unfortunately, this means that the current design of the processing element does not compete well with other works of literature yet, at least in terms of area.

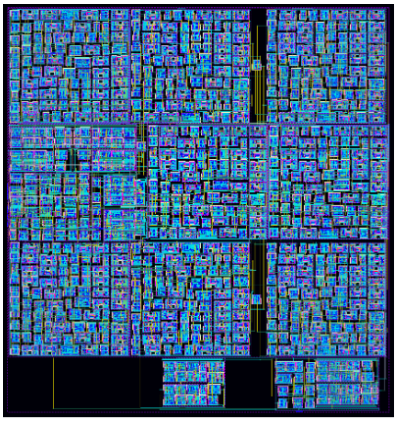

Fig. 17: Physical Layout of Processing Element.

IV. Methodology

A time-domain processor has many promising attributes when compared to the conventional digital implementation. The most promising attributes of this technology are higher information density and lower power consumption. To test the power consumption of the processor proposed in this paper, the processor will be programmed to run a Sobel filter to perform edge detection on a variety of images.

A. Sobel Operator

The Sobel operator is used in image processing and computer vision applications, particularly within edge detection algorithms, where it creates an image emphasizing its edges. It works by applying two convolution filters to each pixel in the image. The two 3x3 kernels which are convolved with the original image calculate approximations of the horizontal and vertical derivatives of the center pixel. If A is defined as the source image, then the computations for the horizontal and vertical derivative approximations are as follows [13]:

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Fig. 18: Horizontal and Vertical Derivative Approximations for Sobel Filter Computation [13].

The two outputs are referred to as the horizontal and vertical gradients respectively. What is normally done next is to take the squared sum of these gradients to get the gradient magnitude. Finally, the gradient magnitude for each pixel is compared to some predefined threshold which determines if the magnitude is strong enough to warrant that particular pixel as an edge pixel. Figure 19 shows a picture of a steam engine while figure 20 shows the Sobel operator applied to that same image. The stronger the brightness of a particular pixel, the larger the calculated gradient magnitude.

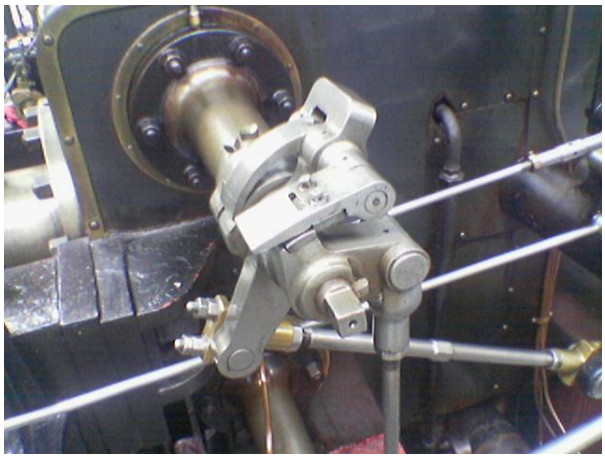

Fig. 19: A Color Picture of a Steam Engine [13].

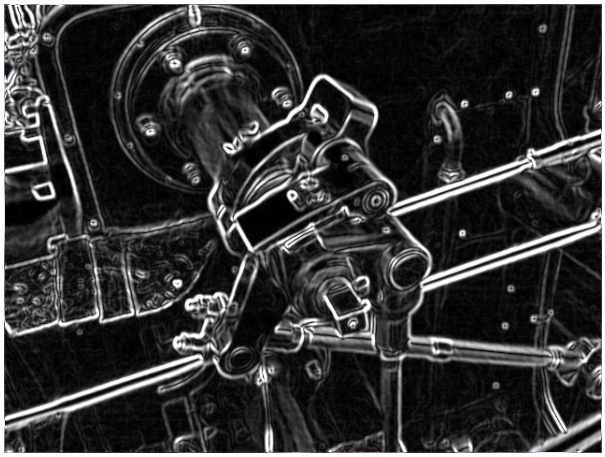

Fig. 20: The Sobel Operator Applied to that Image [13].

B. Processor Array

To implement the Sobel operator, 256 processing elements are interconnected in a parallel architecture operating in SIMD mode. Each of these processing elements has its own dedicated pixel and performs the gradient operation for that pixel by extracting the pixel values from its eight nearest neighbors. In figure 21, there are 16 black boxes, each of which represents one processing element. The control signals are routed to each element and each element is connected to its closest neighbor elements. The schematic from figure 21 is encapsulated in a black box and inserted 16 times into another schematic shown in figure 22. The result is 256 processing elements all connected in parallel and which can perform the Sobel operator on a 16x16 pixel image. The same control signals are routed to these 16 black boxes which allow the processor array to operate in SIMD mode. The same instruction is sent to all 256 processing elements and is performed simultaneously.

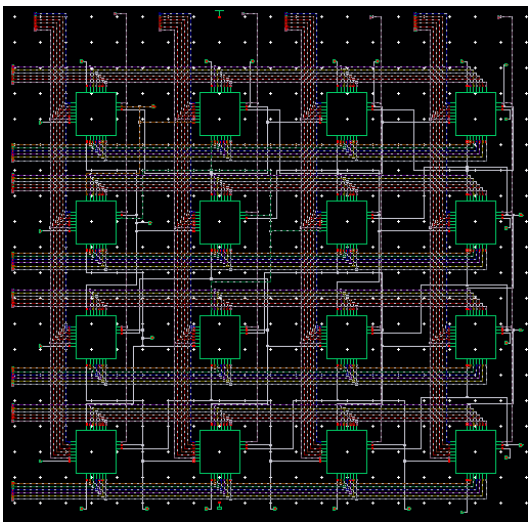

Fig. 21: SIMD Processor Array of 16 Processing Elements.

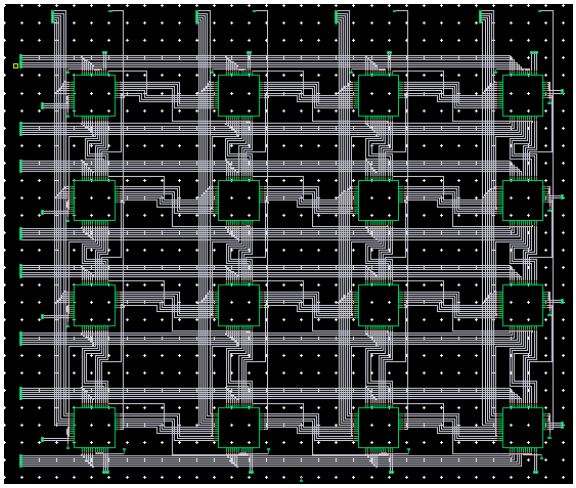

Fig. 22: SIMD Processor Array of 256 Processing Elements With 16 Elements in Each Black Box.

C. Assembly Code

Four assembly code instructions were developed specifically to accomplish the computations for the Sobel operator: mv, ld, add, and sub. The mv instruction is used specifically to transfer register values between neighbor processors, not to move register values internally inside one processing element. The ld instruction enables the programmer to load an immediate value directly into a specified register. The add and sub instructions are used to carry out the computation of the horizontal and vertical gradients as shown in figure 9. The assembly code for the Sobel operator along with a description of each register is shown in Appendix D.

The Sobel operator requires the use of all eight registers in the register file. The first four registers are used to store variables such as the pixel value or the gradient value. The last four registers are used to store neighbor pixel values and transfer them back and forth between neighbor processors. This is important because the gradient computation for each pixel depends on its eight nearest neighbors. To perform the horizontal and vertical gradient calculations, the center processor has to extract the pixel values from each of its eight neighbor processors and store them in its own register file. Since the processing elements are arranged in a parallel architecture, the mv instruction allows all processing elements to transfer their pixel value either to their north, south, west, or east neighbor simultaneously.

V. Experimental Results

The section will analyze the simulation performances of the analog processor array and compare these metrics against that of other digital and analog vision chip designs. The ostensible advantages of time-encoded computation include lower power consumption and higher information density. But first, some analysis is conducted on the datapath components of the processing element which include all the registers and arithmetic operations.

A. Arithmetic Operations

The arithmetic operations described in section 3B have fairly simple time-domain implementations involving one or more digital logic gates. It is important to know the power and timing metrics for each circuit because each will have an impact on the power consumption and processing speed of the processing element as a whole. Figures 23-25 show the simulation for the subtraction, maximum, and minimum time-domain circuits while table I notes the performance metrics for each operation. The static power consumption for each circuit is on the order of nW and is inconsequential as compared to the dynamic power consumption which is on the order of mW.

The timing measurements for these circuits are far more critical than the power measurements. The propagation delays set the practical limit for how fast the processing element can be clocked. Together with the propagation delay from the various multiplexers and registers in the processing element, this adds significant delay for each instruction that has to be processed. Also consequential are the output rise and fall times

TABLE I: Arithmetic Operation Power, Energy, and Timing Measurements [11]

Operation	Dynamic Power (mW)	Static Power (nW)	Propagation Delay (ps)	Output Rise Time (ps)	Output Fall Time (ps)
Subtraction	0.812	3.15	89.67	32.55	43.72
Maximum	0.591	139	83.87	33.18	33.63
Minimum	1.15	85	64.85	28.97	15.89

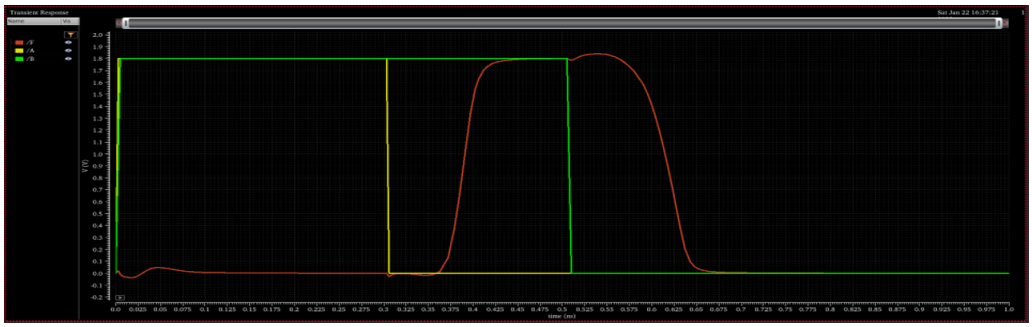

Fig. 23: Simulation of Subtraction Circuit.

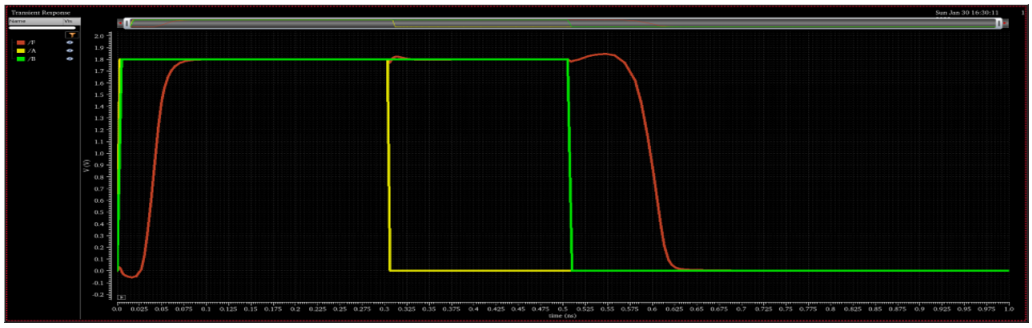

Fig. 24: Simulation of Maximum Circuit.

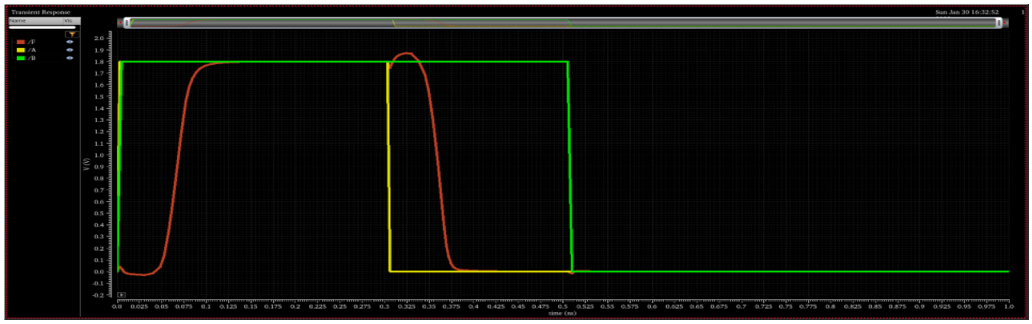

Fig. 25: Simulation of Minimum Circuit.

for each operation which all lie in the range of 15-45 ps. The maximum pulse duration for any time-domain input signal was determined previously to be 2 ns. Together with the rise and fall times for each arithmetic operation, this implies that an error is introduced whenever a signal is processed with any of these operations.

This also implies that the shorter the input signal, the greater the percentage error is going to be. This sets a practical limit on how short the minimum pulse duration can be. For example, if the minimum pulse duration was defined to be 100 ps, this would imply an introduced error on the order of 30% each time the signal was processed which is obviously very detrimental to the overall accuracy of the system.

B. Register

Figure 26 shows an example simulation of the time-domain register where the WE signal is pulsed with two separate pulses each 0.5 ns in duration and the result, when RE is pulsed, is a 1 ns signal at the output. At least that would be the ideal outcome, however, upon immediate inspection of the waveforms, one notices the discrepancy at the output. Just as with the arithmetic operations, the capacitance from

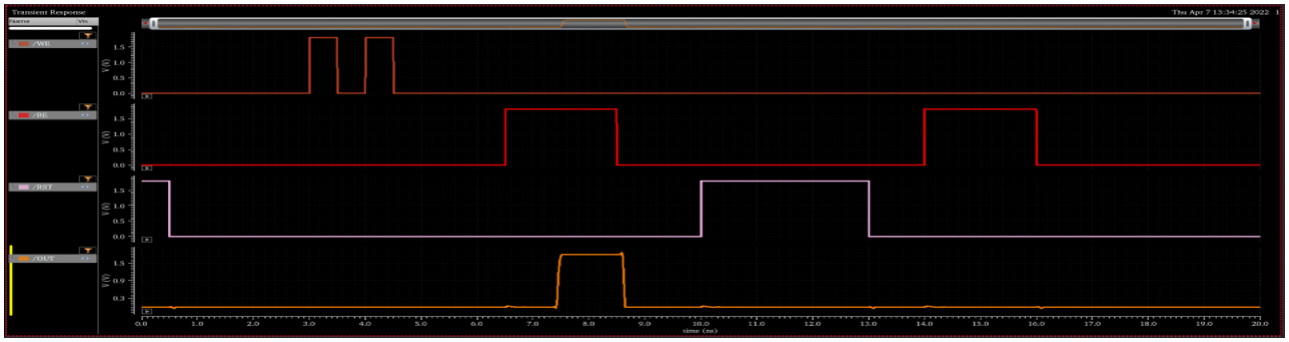

Fig. 26: Simulation of Time-Domain Register.

the internal nodes of the register slow down the rise and fall times of the data which negatively impacts its accuracy.

Figure 27 displays the percent error of the output signal given an input signal of duration ranging from 0-2 ns. The output rise and fall times introduce significant error with any input signal less than 0.5 ns in duration. This severely limits the range of input signals that can be used with reasonable accuracy. Only when the input signal is 1 ns or longer in duration does the percent error become negligible and tolerable. Depending on the thresholds for accuracy, this can limit the range of input signals from 1-2 ns. Extending the range beyond that would necessarily imply a reduced accuracy performance.

Figure 27 also shows an additional issue with successive reads. A new feature of the register design proposed in this paper is non-destructive reads as previously explained in section 3A. Though the value will not be erased when a read is performed, there is a small issue with the pulses being truncated by a certain amount every time a read is performed on the register. As shown in the graph, the issue is most prevalent with input signals ranging from 1-1.5 ns. Though the error might be comparatively small, it still can not be ignored.

The graph in figure 28 also quantifies the error performance of the register. It is displaying the pulse difference between the input and output signals for both the current register design as discussed in section 3A and the previous design as proposed by Guzman's thesis in [11]. The graphs follow the same trend as shown in the graph in figure 27, namely large errors for short duration inputs and small errors for longer inputs. The modifications proposed in this paper helped to practically reduce the error by a factor of two for inputs with a duration longer than 0.5 ns.

C. Processor Array

As discussed in section 4, the processor array of 256 processing elements operating in SIMD mode are programmed with the assembly code in Appendix D to run the Sobel operator on an input image. Each processing element computes the gradient magnitude for its own dedicated pixel of the 16x16 pixel image. The accuracy of the overall processing element still requires some work to be done to achieve adequate image processing results so there are yet no available images which show the results of the Sobel operator. Not to mention that the 16x16 pixel image is very small and is not practicable to display the accuracy results of the processor array. Nevertheless, the assembly code can still be ran on the array all the while recording the performance metrics.

As far as the speed of the processor is concerned, it is running on a 200 MHz clock signal which is fed into the RE signal of each processing element. Given that the maximum pulse duration for the register is 2 ns and other factors such as propagation delay for the arithmetic circuits, this clock speed was the fastest that could reasonably be run to ensure proper operation.

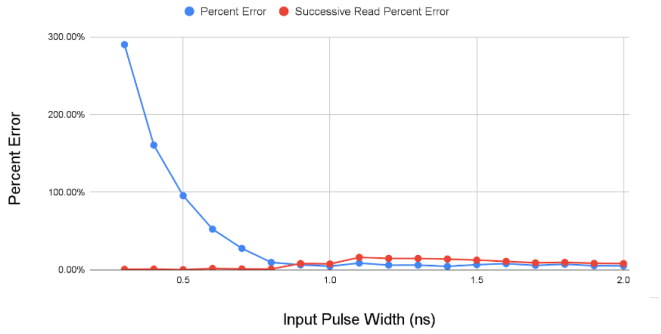

Fig. 27: Time-Domain Register Percent Error and Successive Read Error.

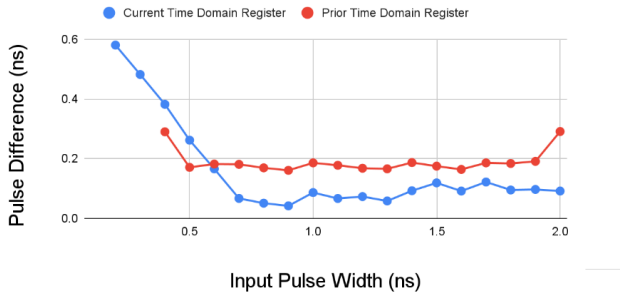

Fig. 28: Time-Domain Register Error Difference with Current Design and Previous Design by [11].

TABLE II: Comparison of Digital and Analog Vision Chips

Reference	**Time-Domain**	**SRVC [6]**	**ACE16k [7]**	**SPE [8]**	**ASPA [9]**	**SCAMP [10]**
Processing Type	analog	digital	analog	digital	digital	analog
Array Size	16x16	16x16	128x128	64x64	128x128	128x128
Technology, um	0.18	0.18	0.35	0.35	0.35	0.35
Performance, GOPS	25.6	0.213	330	6.4	157	20
Chip Size, mm^2	10.5	2.25	145.2	29.2	213.5	54
Power Per Chip (mW)	120	8.72	2900	N/A	5400	240
P_A, MOPS/mm^2	2438	94	3800	343	820.8	512
P_E, GOPS/W	213	24.4	180	N/A	29	85.3

The ostensible advantages of time-encoded computation include higher information density and lower power consumption. With this in mind, table II records the speed performance in GOPS, the chip size in mm^2, the power consumption of the chip, the performance per area, and the performance per watt. These performance metrics are then compared against similar digital and analog vision chips from previous work done by researchers in the field as described in section 2. The array size and process technology for each vision chip are also recorded in the table for the reader to adequately asses the power and performance figures.

As far as chip size is concerned, the best metric to compare area performance comes from the performance per area metric. For every mm^2 of chip area, the array proposed in this paper can perform at 2438 MOPS which is very competitive with its analog and digital chip counterparts. The only competitor with a better area performance comes from the analog vision chip proposed by [7] with a performance per area figure of 3800 MOPS/mm^2.

The power and speed performance of the time-domain array also have promising figures when compared to the other vision chips. The best metric to compare power and speed performance comes from the performance per watt metric. For every watt of input power to the processor array, it achieves a speed performance of 213 GOPS, the highest performance among its competing vision chips. This far outperforms the speed performance for digital chips, but it also exceeds its analog competitors. Its closest competitor comes from the analog vision chip proposed by [7] with a performance per watt figure of 180 GOPS/W.

VI. Conclusion and Future Work

This paper proposed a design for a processing element based on time-encoded computation and implemented it in an analog processor array of 16x16 processing elements. Experimental results and measurements showed that computationally accuracy is still one major area of concern for time-encoded computation. The capacitance of the element's internal nodes causes excessive output rise and fall times which introduce significant error when computing with signals with a maximum duration of only 2 ns. The error are especially troublesome for inputs pulses of short duration.

However, experimental results have also verified the improved area and speed performance of the processor array. The array proposed in this paper was able to achieve a higher performance per watt figure than its analog and digital counterparts, thus lending to the claim that time-encoded computation lowers power consumption. It also has a fairly competitive performance per area figure with it being exceeded by only one other analog chip, thus lending to the claim that time-encoded computation has a higher information density.

This work has proven that time-encoded computation holds some promising rewards for future applications and it definitely requires further critiquing to ensure that it can compete on a level playing field with its digital and analog competitors. The work for this project is still in its verification stage and can use some work from future students to address the issue of computational accuracy. There is also potential to extend the arithmetic capabilities of the processing element to include multiplication and division operations implemented with time-domain circuitry. Some work has already been done by Guzman [11] to design a multiplication unit which can eventually be integrated into the processing element's ALU. The last step would be to eventually fabricate and test the chip with external circuitry inputting the instructions to run the Sobel operator on an input image.

References

[1] J. Hennessy. The End of Moore's Law Faster General Purpose Computing, a New Golden Age. In DARPA ERI Summit 2018, 2018.

[2] Y. Tsividis, "Not your Father's analog computer," in IEEE Spectrum, vol. 55, no. 2, pp. 38-43, February 2018, doi: 10.1109/MSPEC.2018.8278135.

[3] Y. Huang et al. Evaluation of an analog accelerator for linear algebra. In 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), pp. 570-582, 2016.

[4] A. A. Lazar and E. A. Pnevmatikakis, "A video Time Encoding Machine," 2008 15th IEEE International Conference on Image Processing, 2008, pp. 717-720, doi: 10.1109/ICIP.2008.4711855.

[5] Z. Chen and J. Gu, "A Time-Domain Computing Accelerated Image Recognition Processor With Efficient Time Encoding and Non-Linear Logic Operation," in IEEE Journal of Solid-State Circuits, vol. 54, no. 11, pp. 3226-3237, Nov. 2019, doi: 10.1109/JSSC.2018.2883394.

[6] W. Miao, Q. Lin, W. Zhang and N. Wu, "A Programmable SIMD Vision Chip for Real-Time Vision Applications," in IEEE Journal of Solid-State Circuits, vol. 43, no. 6, pp. 1470-1479, June 2008, doi: 10.1109/JSSC.2008.923621.

[7] G. L. Cembrano, A. Rodriguez-Vazquez, R. C. Galan, F. Jimenez-Garrido, S. Espejo and R. Dominguez-Castro, "A 1000 FPS at 128/spl times/128 vision processor with 8-bit digitized I/O," in IEEE Journal of Solid-State Circuits, vol. 39, no. 7, pp. 1044-1055, July 2004, doi: 10.1109/JSSC.2004.829931.

[8] T. Komuro, I. Ishii, M. Ishikawa and A. Yoshida, "A digital vision chip specialized for high-speed target tracking," in IEEE Transactions on Electron Devices, vol. 50, no. 1, pp. 191-199, Jan. 2003, doi: 10.1109/TED.2002.807255.

[9] A. Lopich and P. Dudek, "A SIMD Cellular Processor Array Vision Chip With Asynchronous Processing Capabilities," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 58, no. 10, pp. 2420-2431, Oct. 2011, doi: 10.1109/TCSI.2011.2131370.
[10] Dudek, Piotr. (2011). SCAMP-3: A vision chip with SIMD current-mode analogue processor array. Focal-Plane Sensor-Processor Chips. 10.1007/978-1-4419-6475-52.
[11] E. Guzman, "Energy Efficient Computing using Scalable General Purpose Analog Processors", Cal Poly Digital Commons, San Luis Obispo, CA, rep., 2021.
[12] Z. Chen and J. Gu, "High-Throughput Dynamic Time Warping Accelerator for Time-Series Classification With Pipelined Mixed-Signal Time-Domain Computing," in IEEE Journal of Solid-State Circuits, vol. 56, no. 2, pp. 624-635, Feb. 2021, doi: 10.1109/JSSC.2020.3021066.
[13] "Sobel operator," Wikipedia, 11-May-2022. [Online]. Available: https://en.wikipedia.org/wiki/Sobeloperator. [Accessed: 01-Jun-2022].
[14] "Advanced circuits," PCB Standard amp; Custom Manufacturing — Advanced Circuits. [Online]. Available: https://www.4pcb.com [Accessed: 11-Nov-2021].
[15] "PCB manufacturing, Quick Turn PCB manufacturers," Bay Area Circuits, 30-Oct-2021. [Online]. Available: https://bayareacircuits.com [Accessed: 11-Nov-2021].
[16] "The economics of Asics: At what point does a custom SOC Become Viable?," ElectronicDesign, 15-Jul-2019. [Online]. Available: https://www.electronicdesign.com/technologies/embedded-revolution/article/21808278/ensilica-the-economics-of-asics-at-what-point-does-a-custom-soc-become-viable. [Accessed: 01-Dec-2021].
[17] "An ultimate guide to the PCB manufacturing process," mcl, 21-Oct-2021. [Online]. Available: https://www.mclpcb.com/blog/pcb-manufacturing-process/. [Accessed: 11-Nov-2021].
[18] "Printed Circuit Board Recycling Methods," Oct-2012. [Online]. Available: https://www.epa.gov/sites/default/files/2014-05/documents/handout-10-circuitboards.pdf. [Accessed: 14-Nov-2021]
[19] "IEEE Code of Ethics." IEEE, www.ieee.org/about/corporate/governance/p7-8.html.
[20] What are the human health effects of PCBS? [Online]. Available: https://www.clearwater.org/news/pcbhealth.html. [Accessed: 11-Nov-2021].

APPENDIX

Appendix A - Analysis of Senior Project Design

A. Summary of Functional Requirements

This project implements a hardware accelerator for digital circuitry. The hardware accelerator uses digital circuits and performs computation in the time domain. The datapath components of the processor include a time domain register file and ALU (arithmetic logic unit). The processor interfaces with exterior digital circuitry via a data and I/O bus using serial communication. It receives instructions via a separate input signal and performs the necessary operations in the time domain. This processor also has the potential to be implemented in an array architecture. It is equipped with input and output signals that allow it to communicate with nearby neighbor processors.

B. Primary Constraints

Though time domain computation has its benefits, some arithmetic operations such as multiplication are particularly vulnerable to electrical noise and thermal variation. This adds significant restraints to the type of environment that the processor can operate in. Another significant hurdle comes with the parallel processing capabilities when the processor is implemented in an array architecture. Managing communication between neighbor processors and ensuring synchronization of computation led to some difficult problems.

C. Economic

This processor contributes to human capital because it requires technical expertise to keep the technology up to date with other competing products. This technology requires engineers who specialize in hardware design, architecture design, hardware verification, software development, etc. Jobs will be created for people who can provide this expertise in VLSI system design. The processor requires the financial capital of the Cal Poly EE department to invest in the development of this project. We will also require financial capital to market this technology to the target consumer base. The manufacturing process for this technology requires elements of earth's natural resources which include silicon, fiberglass, copper, and other metal elements. Heat and power dissipation eventually wear down the silicon transistors which puts an expiration date for this technology at around 5-10 years.

The cost of developing a custom ASIC is very expensive even for a relatively large process technology of 0.18 um. The one-time cost of the production mask is approximately 100,000 dollars. However, according to competing vendors such as 4pcb [14], Bay Area Circuits [15], and PCBWay [16], the PCB for this project can be cheaply manufactured for 100 dollars per board. Similar competing hardware accelerators sell on the market for approximately 1000 dollars per unit which is more than enough of a profit margin for manufacturing expenses. This high profit margin also justifies the development expenditure for this technology which required approximately 560 hours of labor which equates to a labor cost of 20,000 dollars.

D. If manufactured on a commercial basis

With a purchase price of 1000 dollars and a manufacturing cost of 100 dollars, the total revenue per unit sold comes out to approximately 900 dollars. Labor costs, as calculated in the Gannt charts, comes to around 20,000 dollars per year for the development of the product. Taking into account the cost for the production mask, the total cost to get this product out on the market requires 120,000 dollars. To break even, approximately 130 units of the product would have to be sold every year. Clearly, for the business to be profitable and to provide substantial returns for its shareholders, it must sell much more than 130 units.

The good news is that 130 units is a very conservative estimate for the number of sales. The market for hardware accelerators has great potential and demand as our computers take on more computationally heavy applications such as computer vision and artificial intelligence. There are also very few competing products who produce hardware accelerators with analog technology. The low power dissipation and high information density benefits of this technology make this product very competitive in this market of hardware accelerators.

E. Environmental

The main elements in the PCB include the substrate, copper layer, and solder mask [17]. The substrate is usually made of fiberglass which provides the skeletal structure for the

PCB as well as all the silicon transistors designed as part of the hardware processor. Silicon is a rapidly renewable resource because it is available in high quantities in the Earth's sand. The copper layer and solder mask comprise various metal elements and are not a renewable resource. This necessitates that the hardware accelerator, and likewise with other competing products, minimize the length and density of traces on each PCB.

As for the PCB itself, the recycling process is extremely complex and is a consistent subject of study for scientists and engineers. In order to recover the materials for reuse as mentioned above, the PCB has to undergo a complex subtraction method in which it is carefully deconstructed into its individual materials [18]. This subtraction process generates its own type of waste, some of which is hazardous waste.

F. Manufacturability

Due to the high concentration of silicon transistors and electrical traces in this design, there may arise manufacturing errors on the part of the vendor. This may result in a nonfunctioning unit due to some open or short trace which electrically damages the product. Depending on the number of faulty traces and the nature of these faulty traces, the entire functionality of the product could be compromised, or some feature of the product may not work as intended. If there does arise this type of error, it must be on the vendor since all ASIC designs are required to pass a standard of testing which checks for these open or short traces before the design is sent to the vendor.

G. Sustainability

The only materials that are not renewable in this product are the metal elements that comprise the copper layer and solder mask portions of the PCB. The good news is that practically all metals are recyclable including copper, tin, and lead which are primarily used in the manufacturing of a PCB. The functionality of these metals may deteriorate over time, but they are not limited to a one-time use. Subsequent designs and upgrades to this product would be under the same constraint as in the original design: minimize trace length and density so as to minimize the use of these nonrenewable metals.

The market demand for this product ensures the economic sustainability for this type of hardware accelerator. As long as computers continue to take on more computationally heavy applications, the demand will be sustained. This product also ensures social sustainability by providing a cheap and affordable energy efficient processor for all consumers.

H. Ethical

As further explained in section 9 on health and safety, there are studies being conducted that suggest PCBs as containing harmful chemicals that may cause cancer to the user. However, the studies performed thus far are inconclusive. This presents a moral dilemma for the developers of the product. According to the IEEE code of ethics [19], members of IEEE make a commitment to "disclose promptly factors that might endanger the public or the environment". Does this mean that the developers of the product are under obligation to disclose the findings of this study, even if they are inconclusive?

This moral dilemma can be analyzed from a utilitarian perspective as well which posits that the correct decision is the one that maximizes utility (or happiness) for the maximum amount of people. There are two factors in this dilemma that influence utility, namely health and economy. If the results of the study were not disclosed to the public, that decision would make the company the most money which benefits the companies' employees and its shareholders. However, if the results of the study were disclosed, that would maximize the utility of the general public by prioritizing the health of its users.

I. Health and Safety

The safety of the user is paramount in the design of this product. The design of the product undergoes rigorous analysis to ensure there are no electrical discontinuities that may cause damage to the product or harm to the user. The PCB must pass the standard as specified by the IPC 9252 specification to test for continuity and isolation before the plans are sent off to the manufacturer.

However, "The International Agency for Research on Cancer and the Environmental Protection Agency classify PCBs as a probable human carcinogen" [20]. The term "probable human carcinogen" is used here because PCBs contain chemicals that are known to cause cancer in animals and there is evidence to suggest that it also causes cancer in humans, although it is not conclusive. The studies performed thus far are inconclusive due to small sample sizes and high exposure of the participants to other chemicals.

J. Social and Political

The direct stakeholders include the investor in this project which is the Cal Poly EE department and companies who produce competing designs for general purpose processors. The Cal Poly EE department has a stake in the successful completion of this project as that would produce revenue for them. Other competing companies would potentially lose a share in the marketplace with the successful introduction of this product.

The general public would be considered an indirect stakeholder in this project. The hazardous waste that is generated from the deconstruction of a PCB could negatively impact the environment. If the impact is serious enough, legislation could be introduced to curb the amount of acceptable hazardous waste that is produced in this process. There are also studies being conducted that suggest PCBs as containing harmful chemicals that may cause cancer to the user. Depending on the severity of this issue, the social and political implications could be catastrophic. If serious health effects were eventually observed as a consequence of PCBs, serious legislation would have to be introduced which would radically change the market for PCBs.

K. Development

During the development of this project, I learned about the entire lifecycle that goes into the development of a VLSI. The first stage is dedicated to researching the market for the various capabilities and features that should be present in the product. Next, a general architecture is drawn up which breaks down the function of the unit into various subfunctions. Then a team of engineers, generally broken up into the design team and verification team, go back and forth to ensure the functionality of each subsystem and the functionality when integrated as a whole. The design team has to ensure the design of the VLSI adheres to the original architectural plans while the verification team is responsible for verifying the VLSIs functionality. Then, the design plans are synthesized into logic gates that implement the high-level C and Verilog code. Then the synthesized logic is converted to the appropriate files to be sent off to the vendor for manufacturing. Once the product has returned, a validation team performs various tests with the actual hardware to ensure its functionality and that it meets the benchmarks of performance as drawn up by the research team so that it can compete in the marketplace.

Appendix B - Gannt Charts

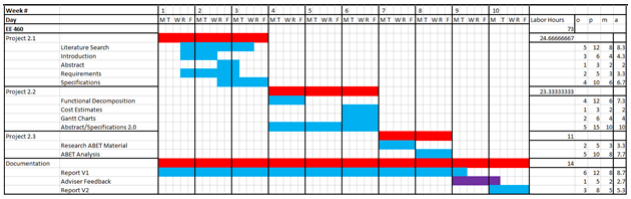

Fig. 29: EE 460 Gannt Chart.

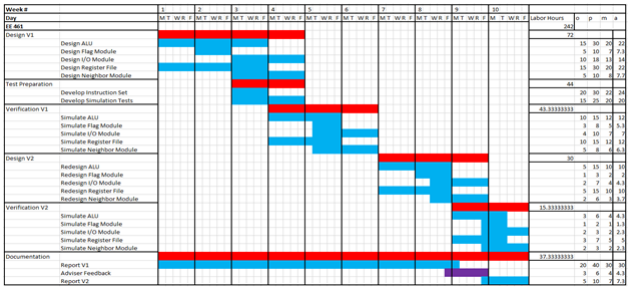

Fig. 30: EE 461 Gannt Chart.

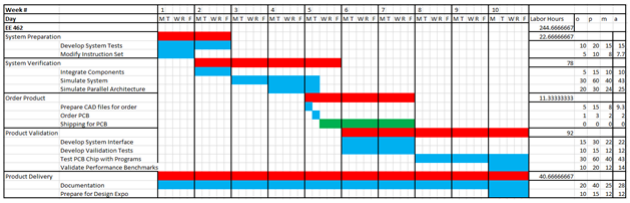

Fig. 31: EE 462 Gannt Chart.

Appendix C - Sobel Operator Assembly Code

```
//-------------------------------------------------------------
// Sobel Filter program
// x0 = Original pixel
// x1 = Filtered pixel
// x2 = Horizontal Gradient
// x3 = Vertical Gradient
// x4 = top neighbor
// x5 = bottom neighbor
// x6 = left neighbor
// x7 = right neighbor
//-------------------------------------------------------------
mv x5, x0 // load neighbor's top pixel
mv x6, x5 // load neighbor's right pixel
add x2, x2, x6 // add value to horizontal gradient
mv x4, x0 // load neighbor's bottom pixel
mv x6, x4 // load neighbor's right pixel
sub x2, x2, x6 // subtract value from horizontal gradient
mv x5, x0 // load neighbor's top pixel
add x2, x2, x5 // add value to horizontal gradient
add x2, x2, x5 // add value to horizontal gradient
mv x4, x0 // load neighbor's bottom pixel
sub x2, x2, x4 // subtract value from horizontal gradient
sub x2, x2, x4 // subtract value from horizontal gradient
mv x5, x0 // load neighbor's top pixel
mv x7, x5 // load neighbor's left pixel
add x2, x2, x7 // add value to horizontal gradient
mv x4, x0 // load neighbor's bottom pixel
mv x7, x4 // load neighbor's left pixel
sub x2, x2, x7 // subtract value from horizontal gradient
mv x4, x0 // load neighbor's bottom pixel
mv x7, x4 // load neighbor's left pixel
add x3, x3, x7 // add value to vertical gradient
mv x4, x0 // load neighbor's bottom pixel
mv x6, x4 // load neighbor's right pixel
sub x3, x3, x6 // subtract value from vertical gradient
mv x7, x0 // load neighbor's left pixel
add x3, x3, x7 // add value to vertical gradient
add x3, x3, x7 // add value to vertical gradient
mv x6, x0 // load neighbor's right pixel
sub x3, x3, x6 // subtract value from vertical gradient
sub x3, x3, x6 // subtract value from vertical gradient
mv x5, x0 // load neighbor's top pixel
mv x7, x5 // load neighbor's left pixel
add x3, x3, x7 // add value to vertical gradient
mv x5, x0 // load neighbor's top pixel
mv x6, x5 // load neighbor's right pixel
sub x3, x3, x6 // subtract value from vertical gradient
add x1, x2, x3 // Sum together horizontal/vertical gradients
```