Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 5
AIDML 2024

An Optimized 4-bits Absolute Value Detector

Yuyao Tang *

School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, 100044, China

* Corresponding Author Email: 21211190@bjtu.edu.cn

Abstract. The binary absolute value detector is crucial in today's computer domain, especially in computer storage and analysis systems. It ensures the integrity and accuracy of data. Therefore, this paper proposes an optimized design of a 4-bit absolute value detector aimed at finding the circuit with the lowest energy consumption. Firstly, this paper introduces a design different from the traditional absolute value calculator, resulting in a reduced total number of stages in the circuit. Secondly, this paper calculates the delay of the main path can be using the logic effort formula and determine the specific gate sizing value for each logic component. Finally, this paper adjusts sizing and power supply (V_{dd}) ratio to achieve the lowest energy consumption at 1.5 times the minimum delay. The detector exhibits fewer critical path gates, resulting in lower average delay compared to traditional designs and features an even number of gates which means it has lower delay fluctuations across different inputs.

Keywords: Half-adder, Logic effort, Critical path, Energy optimization.

1. Introduction

In computer storage systems, binary absolute value detectors can be used to verify the correctness of stored data. For example, in cyclic redundancy check (CRC) or error detection and correction (EDAC), they can be used to compare the computed checksum with the received checksum to detect errors during data transmission [1]. Binary absolute value detectors play an important role in storage systems, ensuring the integrity and accuracy of data, thereby guaranteeing the reliability and stability of the system. Therefore, this paper proposes an optimized 4-bits absolute value detector for size, delay, and power consumption.

The article is divided into three main sections. The second section introduces basic logic gate circuits and a detector circuit design which optimized differently from traditional methods. The optimization design is further divided into two parts, namely the absolute value calculator and comparator. Based on the binary absolute value algorithm, the adder is optimized, reducing the number of gate circuit units. The third section introduces optimization. Firstly, three circuits with the same stages in the design are compared, and the maximum delay is calculated to identify the critical path. Then, using the logic effort formula to calculate its delay. Finally, this paper performs gate sizing and V_{dd} scaling on the main path. After balancing delay and power consumption, the optimized solution with the lowest energy consumption at 1.5 times the minimum delay is obtained. The fourth section presents the final optimized solution and conclusions, indicating areas where overall design optimization is still possible and prospects for future improvements.

2. Circuit Topology Design

2.1. Theoretical Foundation

The final goal of this project is to devise a 4-bits absolute value detector with minimal energy consumption while ensuring a delay that is 1.5 times its minimum delay. To construct a four-bit absolute value detector, the circuit is split into two sections [2]. The first section's task is to determine the input's absolute value for a 4-bit random variable and the second part is to compare the calculated

value with a threshold value and outputs a sign. This project assumes that when the input value exceeds the predefined threshold, the comparator outputs 1; otherwise, it outputs 0.

Logic circuits refer to circuits designed to perform logical operations. These circuits typically feature multiple input ports and several output ports. When the input signals meet specific logical relationships, the circuit is activated, generating output; otherwise, it remains inactive, producing no output. Therefore, these circuits are also known as logic gate circuits, commonly referred to as gate circuits. The logic circuits to be used in the project are presented in the table 1 below.

Table 1. Logic circuits

Gate Type	Theory	Logic expression
Inverter	To invert the input states from high to low and low to high	$Y = A'$
AND gate	The output will be high only when all inputs are high	$Y = AB$
OR gate	The output will be low only when all inputs are low	$Y = A + B$
NOR gate	Opposite to the OR gate. The output will be high only when all inputs are low	$Y = \overline{A + B}$
XOR gate	When the inputs are the same, the output is low; otherwise, it is high	$Y = A'B + AB'$
XNOR gate	Opposite to the XOR gate. When the inputs are the same, the output is high; otherwise, it is low	$Y = AB + A'B'$

The multiplexer (MUX) is utilized in circuit design. A multiplexer, also known as a data selector or a multiple-way switch, selects one data input from multiple data inputs based on variable control and routes it to the output terminal [3].

2.2. Logic Circuit Design and its Selection

To construct this four-bit absolute value comparator, it is imperative to initially compute the binary two's complement. This computation is facilitated by employing a NOT gate to invert the binary digits, whereby a '0' is transformed into a '1', and vice versa. Subsequently, a '1' is added to the least significant bit, A0. The utilization of two half adders in conjunction with a single XOR gate enables the realization of a three-digit carry output. Given that A3 serves as the sign bit, it is feasible to minimize the number of components by excluding A3 from the calculation and instead using it as the control bit for a multiplexer. In instances where A3 is '1', the system should select the complement; conversely, when A3 is '0', indicating a positive value, the original number is to be selected. This delineates the complete procedure for computing the absolute value, with the result being a three-digit output. As for the comparator, it suffices to denote its function by the expression:

$$Y = \boldsymbol{A_2 B_2'} + (\boldsymbol{A_2} \odot \boldsymbol{B_2})\boldsymbol{A_1 B_1'} + (\boldsymbol{A_2} \odot \boldsymbol{B_2})(\boldsymbol{A_1} \odot \boldsymbol{B_1})\boldsymbol{A_0 B_0'} \quad (1)$$

And implement the logic using the appropriate gates. The comparator yields an output of '1' if A is greater than B, and '0' in all other scenarios. The following figure 1 illustrates the process.

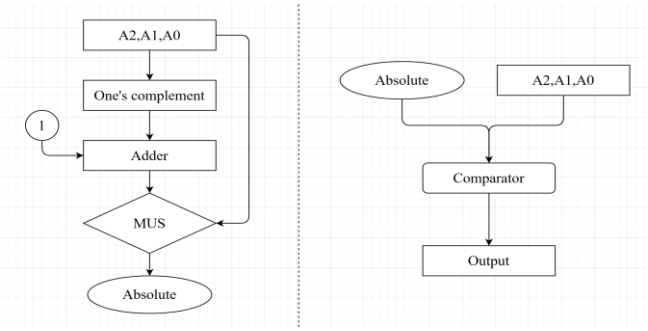

Figure 1. Logic process (Photo/Picture credit: Original)

The absolute value calculator comprises two parts. In two's complement representation, the complement of a positive number is the same as its original representation, while the complement of a negative number is obtained by taking the complement of its corresponding positive number's binary representation and then adding 1 [4]. This ensures that only one set of addition rules is required to handle both positive and negative numbers in a computer. Typically, when the value is positive, no calculation is required; it directly outputs the lower 3-bits value. If the value is negative, to obtain the absolute value of the input, it is necessary to take the complement and add 1. Since only two values (A3A2A1A0 and 1) are added together in the negative case, only a half-adder is needed for the calculation.

In the negative scenario, the less significant bit value (LSB) A0 always adds 1. The equation

$$Sum = A_0' \oplus 1 \tag{2}$$

Indicates that A0 is equal to itself. Therefore, the XOR gate for A0 can be deleted. According to the equation

$$Cout = A_0' \cdot 1 \tag{3}$$

The carry out value is A0′ itself. As a result, the A0′ is linked to the subsequent stage directly and the bottom AND gate is likewise deleted. The formula

$$Cout = A_0' \cdot A_1' \tag{4}$$

It is also used for the A1 carry out calculation. Because an AND gate requires more power and delays more than a single NOR gate due to its two stages in CMOS architecture. Applying De Morgan's rule to convert the AND gate into a two input nor gate, based on the formula $A_0' \cdot A_1' = (A_0 + A_1)'$ [5]. The topology of the half-adder circuit is depicted in the figure 2 below.

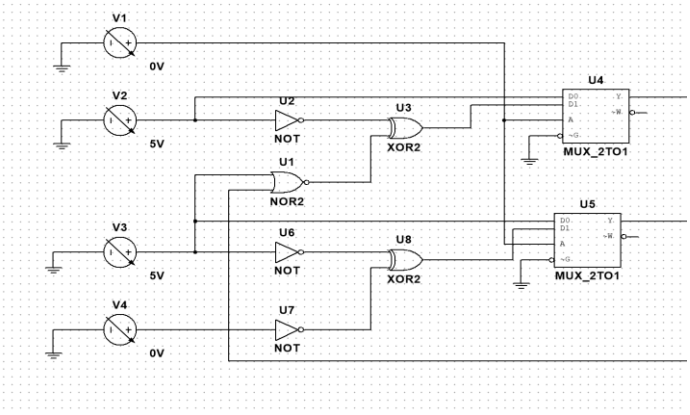

Figure 2. The absolute value calculator circuit (Photo/Picture credit: Original)

Overall, it is clear from comparing our reduced half-adder circuit to the original half-adder design that our reduced version has a critical path delay of just two stages and uses fewer gates than the original half-adder design [6]. The topology of a traditional half-adder is illustrated in the figure 3 below.

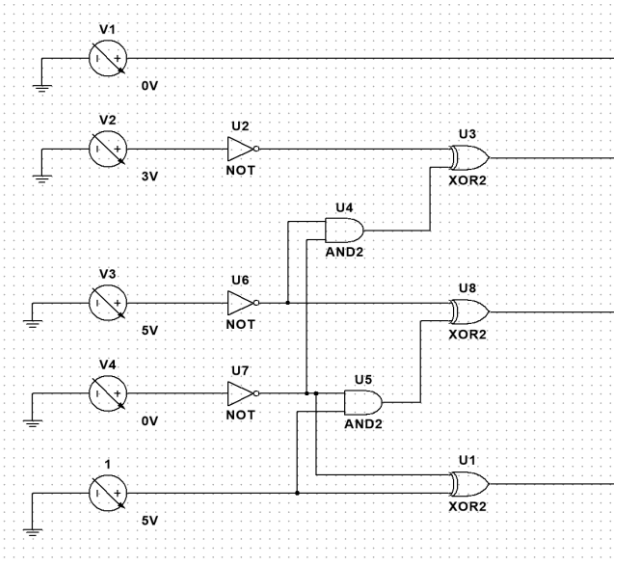

Figure 3. The topology of a traditional half-adder (Photo/Picture credit: Original)

To choose which values to feed into the subsequent comparator step, a multiplexer (MUX) is needed. A2A1A0 is the result of the multiplexer (MUX) when A3 is 0, indicating a positive value; on the other hand, B2B1B0 is the result of the multiplexer (MUX) when A3 is 1, indicating a negative value. Referring to the schematic, this paper incorporates two multiplexer (MUX) circuits for this selection process. Based on the logical expression

$$Y = \boldsymbol{A_2 B_2'} + (\boldsymbol{A_2} \odot \boldsymbol{B_2})\boldsymbol{A_1 B_1'} + (\boldsymbol{A_2} \odot \boldsymbol{B_2})(\boldsymbol{A_1} \odot \boldsymbol{B_1})\boldsymbol{A_0 B_0'} \quad (5)$$

Choose the appropriate gate circuits to build the comparator [7]. The comparator circuit is illustrated in the figure 4 below. The power supply voltage of 5V in the figure represents a logic high level, while 0V represents a logic low level.

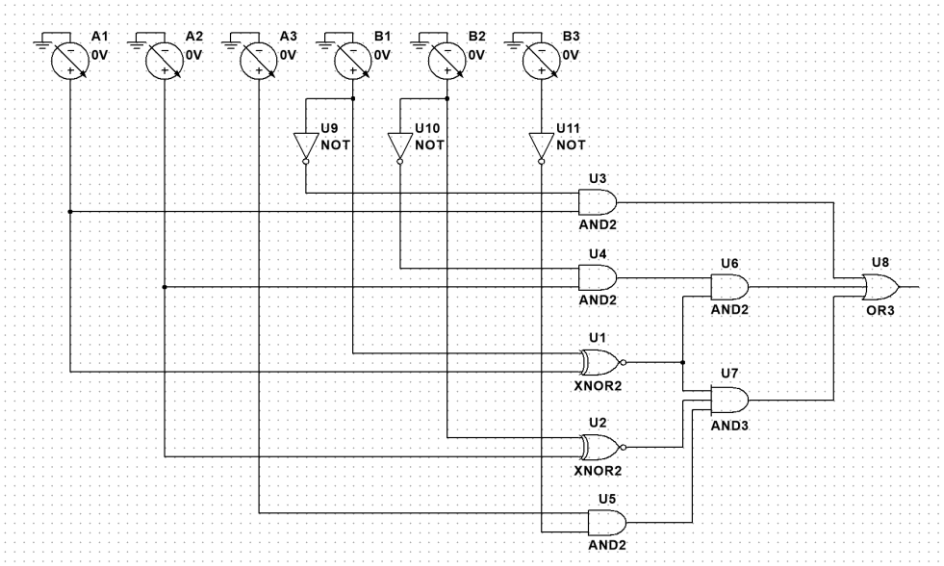

Figure 4. The comparator circuit (Photo/Picture credit: Original)

To obtain the final 4-bit absolute value detector, connect all the circuit topologies mentioned above. The whole schematic for the circuit, as seen in the figure 5 below.

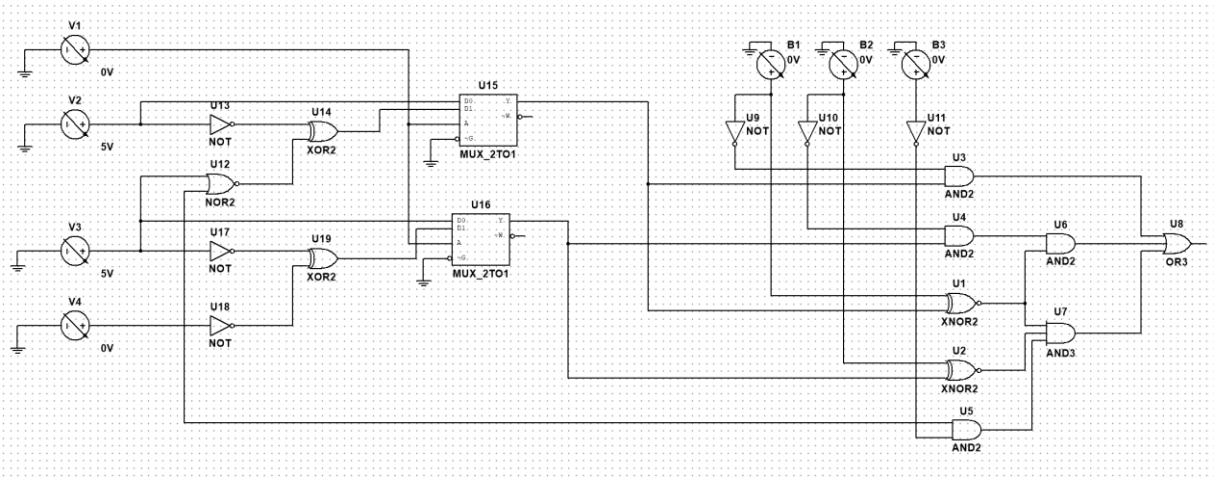

Figure 5. The whole schematic for the circuit (Photo/Picture credit: Original)

3. Circuit Optimization

3.1. Theoretical Foundation (Critical path analysis)

Based on the circuit topology designed above, there are three potential critical paths with the same stages (all with 6 stages). The three potential critical paths as shown in Figure 6 below.

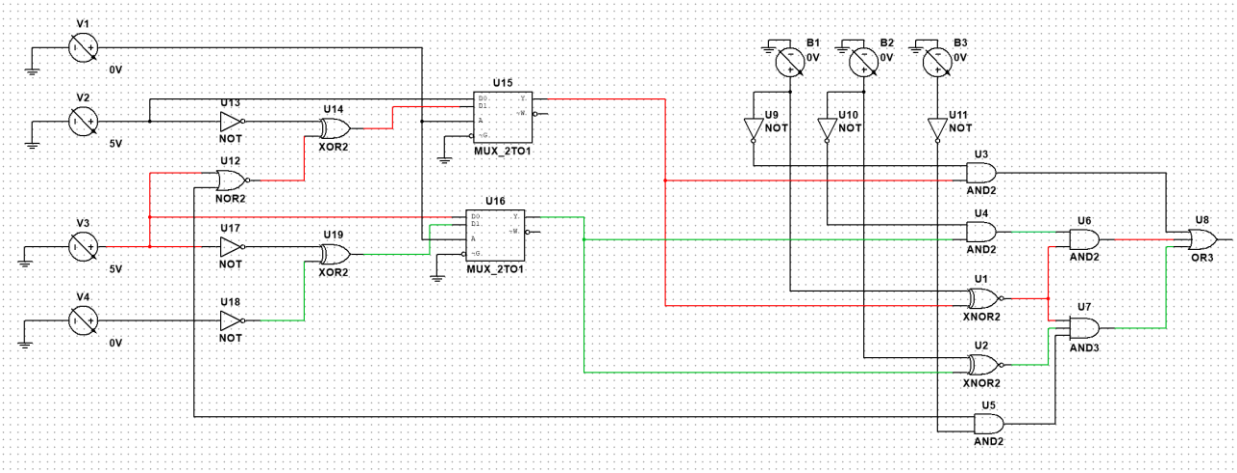

Figure 6. Three potential critical paths (Photo/Picture credit: Original)

Assuming that the inputs to our adder are driven by a unit sized buffer (chain of two-unit sized inverters) and after the output bit there is a 32 times unit-sized inverter capacitance load. The unit-sized inverter is W_p equal to 650nm, W_n equal to 430nm and L_p equal to L_n equal to 100nm. The g and parasitic delay of each gate are shown in the table 2 and table 3 below.

Table 2. g for Different number of inputs

Gate Type	g for Different number of inputs					
	1	2	3	4	5	n
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n+2)/3
NOR		5/3	7/3	9/3	11/3	(2n+1)/3
Multiplexer		2	2	2	2	2
XOR, XNOR		4	12	32		

Table 3. Parasitic delay

Gate Type	Parasitic delay
Inverter	P_{inv}
n-input NAND	nP_{inv}
n-input NOR	nP_{inv}
n-way Multiplexer	$2nP_{inv}$
2-input XOR,XNOR	$2^{n-1}nP_{inv}$

In the calculation process, this paper treats AND and OR gates as NAND and NOR gates with an additional inverter. By using the parameters of each gate circuit displayed in the table 2 and table 3, the corresponding logic effort and parasitic effort can be obtained [8].

To find the critical path with the maximum delay, the time delay needs to be calculated for each of the three possible paths and then compare them. Here, the circuit design of Critical Path 1 is shown in figure 7.

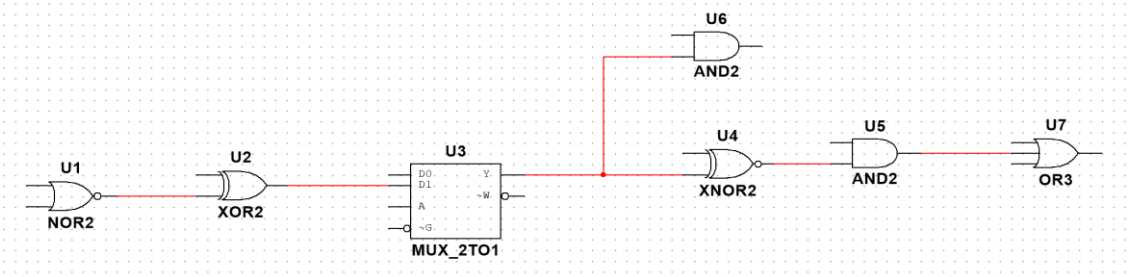

Figure 7. Critical path 1 (Photo/Picture credit: Original)

Through the provided computation process, the time delay of 49.128 seconds for Critical Path 1 can be obtained.

$$G = \sum g = \frac{5}{3} * 4 * 2 * 4 * \frac{4}{3} * 1 * \frac{7}{3} * 1 = 4480/27 \approx 165.926 \tag{6}$$

$$H = {}^{C_L}/_{C_{in}} = 32 \tag{7}$$

$$B = \sum b = 1 * 1 * 2 * 1 * 1 = 2 \tag{8}$$

$$F = G \cdot H \cdot B = 10619.259 \tag{9}$$

$$f^* = \sqrt[N]{F} = \sqrt[6]{G \cdot H \cdot B} = 4.688 \tag{10}$$

$$D = N \cdot f^* + \sum p = 49.128 \tag{11}$$

Here, the circuit design of Critical Path 2 is presented in figure 8.

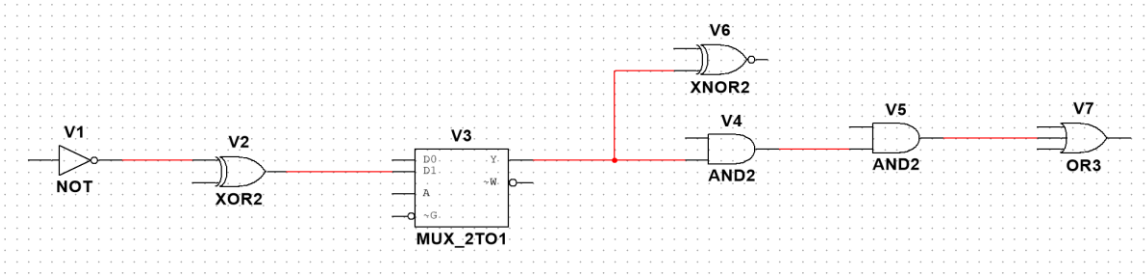

Figure 8. Critical path 2 (Photo/Picture credit: Original)

Through the provided computation process, the time delay of 40.51 seconds for Critical Path 2 can be obtained.

$$G = \sum g = 33.185 \tag{12}$$

$$H = {C_L}/{C_{in}} = 32 \tag{13}$$

$$B = \sum b = 1 * 1 * 2 * 1 * 1 = 2 \tag{14}$$

$$F = G \cdot H \cdot B = 2123.852 \tag{15}$$

$$f^* = \sqrt[N]{F} = \sqrt[6]{G \cdot H \cdot B} = 3.585 \tag{16}$$

$$\mathrm{D} = \mathrm{N} \cdot \mathrm{f}^* + \sum \mathrm{p} = 40.51 \tag{17}$$

Here is the circuit design of Critical Path 3 is presented in figure 9.

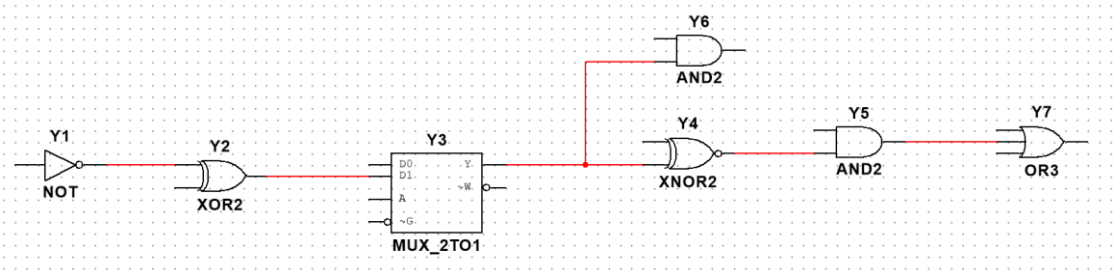

Figure 9. Critical path 3 (Photo/Picture credit: Original)

Through the provided computation process, the time delay of 40.51 seconds for Critical Path 3 can be obtained.

$$G = \sum g = 99.556 \tag{18}$$

$$H = {C_L}/{C_{in}} = 32 \tag{19}$$

$$B = \sum b = 1 * 1 * 2 * 1 * 1 = 2 \tag{20}$$

$$F = G \cdot H \cdot B = 6371.556 \tag{21}$$

$$f^* = \sqrt[N]{F} = \sqrt[6]{G \cdot H \cdot B} = 4.306 \tag{22}$$

$$D = N \cdot f^* + \sum p = 45.836 \tag{23}$$

Based on the calculation results above, the delay of Path 1 is the maximum. Therefore, Path 1 is determined as the critical path of the circuit.

3.2. Sizing optimization and V_{dd} scaling

Based on the sizing formula

$$C_{in} = g \cdot {C_{out}}/{f^*} \tag{24}$$

And the assumption that the C_L is 32 times unit sized inverters. It can calculate backwards for the sizes of each stage and get the sizes of critical path components which is shown in table 4.

Table 4. The sizes of critical path components

Stages	C_{in}
1	0.500194
2	1.406946
3	1.648941
4	3.865117
5	4.529918
6	15.92719
C_{load}	32

As for the sizing approach for non-critical paths, as the non-critical paths naturally have a lower delay than critical paths, the delay redundancy can be utilized to optimize the energy performance [9]. Thus, for non-critical paths, the sizes are decreased as close as 1 after gate sizing by global deflation, within the limit of maximum delay that is not longer than the optimized critical path delay. After the gate sizing, the total C_{path} is 59.878.

Utilizing formula

$$E = C_{path} \cdot V_{dd}^2 \tag{25}$$

the energy after sizing can be calculated as 59.878.

This paper has chosen to prioritize reducing total energy consumption over minimizing delay, with the current maximum delay being 1.5 times the minimum delay. Assuming the delay is proportional to the V_{dd} as shown in formula, the maximum supply voltage is 1V and $V_T = 0.2$V.

$$D = {k \cdot V_{dd}}/{(V_{dd} - V_T)^2} \tag{26}$$

Then plug in the value and then get the parameter k.

$$D_{min} = {k \cdot 1}/{(1 - 0.2)^2} \tag{27}$$

After getting the parameter k, the minimum voltage value can be obtained.

$$1.5D_{min} = {k \cdot V_{dd,min}}/{(V_{dd,min} - V_T)^2} \tag{28}$$

The minimum voltage is 0.775V. Utilizing formula

$$E = C_{path} \cdot V_{dd}^2 \tag{29}$$

The energy can be calculated after V_{dd} scaling as 35.964.

After gate sizing and V_{dd} scaling, this paper finds that in delay optimization, adjusting the size can reduce delay, but lower delay implies a greater energy consumption. In energy optimization, providing a lower supply voltage means reduced energy consumption. Adjusting V_{dd} helps relax

energy level, then get a relative optimal for both delay and energy. In that case, sizing is to find the minimum delay and scaling V_{dd} is to balance the performance of both delay and energy in practice [10].

In the end, some extra delay is ignored in exchange for a significant energy savings. Our approach to optimize minimum energy at 1.5Delay for the circuit is by finding an optimal combination of gate sizing and scaling V_{dd}. This paper comes to an approximate solution by stepping up D from unit (minimum) delay increments of 10% while stepping down the sizing component from 50%. But it is necessary to ensure that the delay remains within 1.5D. It is found that the V_{dd} of 0.808V at 1.5D with sizing contribution of 0.1D and supply voltage scaling contribution of 0.4D. This leads to a reduced dynamic energy to 35.272, which is a 41.1% reduction. The result is shown in table 5.

Table 5. The optimization results

V_{dd}%	0%	10%	20%	30%	40%	50%
V_{dd}	1V	0.939V	0.888V	0.845V	0.808V	0.775V
Sizing%	50%	40%	30%	20%	10%	0%
Energy	44.939	41.963	39.622	37.773	35.272	35.964

Energy calculations are conducted for various combinations of sizing and V_{dd} ratios, and then focused on detailing three of the most common approaches: utilizing V_{dd} only, sizing only, and combining sizing with V_{dd}. Basically, if the voltage is the minimum value which is 0.775V and keep the sizing as the original minimum delay value. The energy will decrease by 39.9% while the critical path delay will rise by 50%, from 49.128 to 73.692. Additionally, there will be a 50% increase in delay but a 24.9% energy decrease if the voltage is left at its original level and just the size optimization is done. However, if both optimizations are done, the supply voltage will be 0.808V and the lowest energy reduction of 41.1% will occur. All the detail value are shown in the figure 10 below.

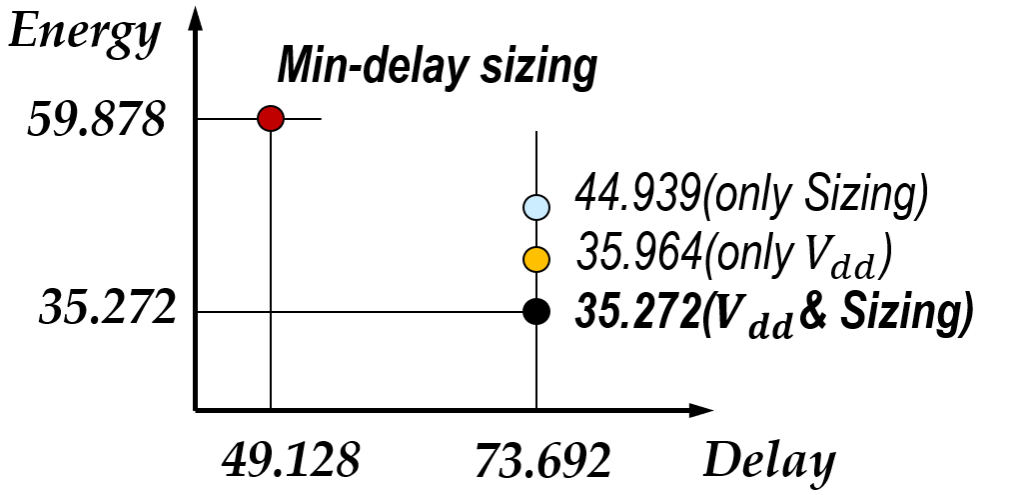

Figure 10. The detail values (Photo/Picture credit: Original)

4. Conclusion

In the second section, the combination of the designed absolute value calculator and comparator resulted in an absolute value detector. The optimized half adder was utilized in the design of the absolute value calculator. In the third section, the delay of the critical path is calculated using the logic effort formula. Then, by applying gate sizing and V_{dd} scaling optimization at different ratios, the final optimized solution is obtained. It is found that the V_{dd} of 0.808V at 1.5D with sizing contribution of 0.1D and supply voltage scaling contribution of 0.4D. This leads to a reduced dynamic energy to 35.272, which is a 41.1% reduction.

In summary, this detector exhibits fewer critical path gates, resulting in lower average delay compared to traditional designs. Additionally, it features an even number of gates which means it has lower delay fluctuations across different inputs. However, there are areas in the design that can be improved. One option is to explore a better topology to address overflow cases, such as when dealing with the inputs is -7 (1000). Another improvement could involve refining the optimization tuning by calculating for additional settings of V_{dd} and 1.x Delay, aiming to achieve better performance in terms of both delay and energy consumption.

References

[1] Sridevi N, Jamal K, Mannem K. Implementation of Cyclic Redundancy Check in Data Recovery. 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2021: 17-24.

[2] Iranmanesh S, Raikos G, Jiang Z, et al. CMOS implementation of a low power absolute value comparator circuit. 2016 14th IEEE International New Circuits and Systems Conference (NEWCAS). IEEE, 2016: 1-4.

[3] Prakash J, Kiran B R, Sathish P. The Efficient Implementation to Optimize Power and Delay Using Data Selector. ICTACT Journal on Microelectronics, 2021: 1159-1165.

[4] Zwillinger D. CRC standard mathematical tables and formulas. Chapman and hall/CRC, 2018.

[5] Lin C C, Lin C S, Tsai Y H, et al. don't care computation and De Morgan transformation for threshold logic network optimization. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 41(5): 1412-1422.

[6] Wang J, Sun J, Sun Q. Single-PPLN-based simultaneous half-adder, half-subtracter, and OR logic gate: proposal and simulation. Optics express, 2007, 15(4): 1690-1699.

[7] Hassanpourghadi M, Zamani M, Sharifkhani M. A low-power low-offset dynamic comparator for analog to digital converters. Microelectronics Journal, 2014, 45(2): 256-262.

[8] Anand S, Ghosh P K, Kumar M, et al. Logical Effort to study and Compare the Performance of VLSI Adders. UACEE International Journal of Advances in Electronics Engineering, 2011, 1: 23-27.

[9] Singh K, Jain A, Mittal A, et al. Optimum transistor sizing of CMOS logic circuits using logical effort theory and evolutionary algorithms. Integration, 2018, 60: 25-38.

[10] Li G, Yan J, Chen L, et al. Energy consumption optimization with a delay threshold in cloud-fog cooperation computing. IEEE access, 2019, 7: 159688-159697.