



Systematic Analysis and Design of a Four-Bit Absolute Value Comparator

Lanrui Wang¹*

¹School of Electronic Information, Wuhan University, Wuhan, 430072, China

*2023302121172@whu.edu.cn

Abstract. This paper is dedicated to the design of a 4-bit absolute value comparator, aiming to resolve its power consumption issue. Initially, the logic effort method is employed to analyse the circuit structure, precisely identify the critical path, and optimize the sizes of devices on it to achieve the minimum delay state. Subsequently, by adjusting voltage parameters and fine-tuning device sizes, the balance between power consumption and performance is thoroughly explored. Experimental results reveal that when only the voltage is adjusted, with the delay extended to 1.5 times the original, the power consumption can be decreased to 60.84% of the original. Optimizing only the capacitor size can reduce the power consumption to 60.73% of the original under the same delay condition. The combined optimization of voltage and capacitor size is even more effective, reducing the power consumption to 51.42% of the original. This "performance - for - efficiency" strategy is highly applicable in low-power embedded systems. Future research could focus on the synergistic adjustment of other circuit parameters to further optimize the performance - power trade-off.

Keywords: Logic Effort, Circuit Design, Delay Reduction, Size Optimization, Energy Consumption Optimization

1 Introduction

The binary absolute value detector occupies a key position in today's cutting-edge field of neuromorphic computing as well as in the burgeoning field of bioelectronic systems [1]. Its core function shows a striking resemblance to the delicate excitation-inhibition balance of neurons. Delving deeper into the biological nervous system, we find that it encodes information mainly by means of a unique way of threshold comparison of membrane potentials [2]. This is exemplified by the "all-or-nothing" release of action potentials, which are generated instantaneously when the membrane potential reaches a specific threshold, but not when it does not, in order to accurately transmit and process information.

The corresponding absolute value detector, although in a different field, but in the function of the realization of the same wonder. It is through the exquisite digital logic architecture to complete the signal amplitude discrimination work, in the whole bionic

information processing process, this discrimination link is like a cornerstone, for the accurate processing and analysis of subsequent information plays a decisive role.

Based on this, this paper is dedicated to the construction of a 4-bit absolute value comparison circuit to overcome the power consumption problem of the 4-bit absolute value detector, and innovatively proposes a comprehensive and efficient low-power optimization design scheme. At the beginning of the scheme, the professional method of logic effort is cleverly used to deeply analyses the circuit structure and pinpoint the critical path [3]. Once the critical path is identified, the sizes of the various devices on the critical path are finely optimized [4]. Through repeated adjustments and simulation analysis of the device sizes, the optimal size configuration for the minimum delay state was successfully obtained. On the basis of achieving the minimum delay, the balance between power consumption and performance is further explored. By changing the voltage parameters and fine-tuning the device size again, the latency is 1.5 times of the original one. After collecting and analyzing a large amount of experimental data, the optimal parameter combinations that minimize power consumption at this delay state are successfully identified, providing a practical new path for the low-power operation of the 4-bit absolute value detector [5].

2 Design of logic gate circuits

2.1 Definitions

In the complex task of building a 4bit absolute value comparator, the first priority was to clearly define and successfully implement the two core functions.

The first key function is to convert the input complement code to the original code with precision and accuracy. The complement is a widely used form of encoding in digital systems, however, when performing operations such as absolute value comparisons, the original code is often easier to handle and analyses directly. To complete the conversion from complementary code to original code, an in-depth understanding of the encoding rules of complementary code is required. For positive numbers, the complement and the original code in the same form; but for negative numbers, the complement is based on the original code, the sign bit remains unchanged, the rest of the inverse in the lowest bit plus 1 to get. Therefore, in the design of the complementary code conversion circuit, we need to cleverly design the logic to identify the sign bit of the input complementary code, if the sign bit is 0 (indicating a positive number), then directly output the original code; if the sign bit is 1 (indicating a negative number), the circuit needs to carry out the inverse of the operation process and add 1, so as to accurately output the corresponding form of the original code.

The second core function performs the task of comparing two 3-bit binary numbers. This function plays a critical discriminatory role in absolute value comparators. When designing the comparator circuit, it is necessary to take into account a variety of situations in which binary numbers are compared. From high to low bit by bit comparison, if the highest bit of the two numbers are different, then the highest bit for the number of 1 is relatively large (here the assumption is that the comparison of unsigned numbers, if the number of signs involved, but also need to be combined with the sign of the bit of the integrated judgement); if the highest bit is the same, then

continue to compare the second highest bit, and so on, until the comparison results. Through a series of well-designed logic gate circuit combinations to achieve this bit-by-bit comparison function, so as to accurately determine the relationship between the size of the two 3-bit binary numbers.

In order to reach these two functions successfully, the design needs to introduce a complement circuit and a comparator circuit. The complement circuit focuses on performing the conversion from the input complement to the original code, and its internal logic is built around the conversion rules of the complement and the original code. The comparator circuit compares the size of two 3-bit binary numbers, and the logic gates are arranged to achieve an efficient and accurate comparison process.

In the operation mechanism of the 4bit absolute value comparator, when the input value corresponding to the original code converted by the complementary code circuit exceeds the pre-set threshold, the comparator outputs 1, which represents that the input value is in a specific high amplitude range; on the contrary, if the input value does not exceed the threshold, the comparator outputs 0, which indicates that the input value is in a relatively low amplitude range. This clear and precise output signal provides a clear and reliable basis for subsequent digital system processing.

2.2 Circuit design

Design of the complementary circuits. In the field of digital circuit design, there are often multiple implementations of circuit construction for specific functions. This time we focus on the design of a circuit that adds one to a three-bit binary number as input, provided that the input is a four-bit binary number $A_3A_2A_1A_0$ and that the sign bit A_3 is not involved in the add-one operation. This design has a wide range of applications in many digital systems such as counters, address generators, etc., so it is important to find an efficient and low-power design solution.

After in-depth research and analysis, we initially identified two feasible design options. The first option is to use an adder to implement the function of adding one to a three-digit binary number. As a basic component used to perform addition operations in digital circuits, the adder has a high degree of versatility and stability. It is capable of performing accurate addition operations on input binary numbers based on the rules of binary addition. For example, when the input is 010, the adder is able to add one to get 011 according to the principle of two into one. However, the internal structure of the adder is relatively complex, usually made up of multiple full adder cascades, which results in the need for a larger number of logic gates to implement. Numerous logic gates not only increase the area of the circuit and increase the manufacturing cost, but also cause a large delay due to the transmission of signals between multiple logic gates, and also consume more power, which is not conducive to the optimization of the overall performance of the circuit.

The second option is to use combinational logic circuits directly to perform the add-one operation. The output of a combinational logic circuit depends only on the current input and is characterized by a relatively simple structure and flexible design. By cleverly designing the way the logic gates are combined, specific logic functions can be achieved. In this circuit design for adding one to a three-bit binary number, we can construct a specialized combinational logic circuit based on the logic rules for adding

one to a binary number. For example, for the lowest bit, if it is a 0, it becomes a 1 after adding one, and if it is a 1, it creates a rounding. For the middle and highest bits, their values depend not only on the input of this bit, but are also affected by the lower rounding bit.

In order to determine which solution is superior, a thorough and detailed comparison of the two solutions was conducted. The key metrics of the comparison focus on both the number of gates used and the circuit performance. After detailed theoretical calculations and simulation experiments, we found that the direct use of combinational logic circuits has significant advantages. In terms of the number of gates used, combinational logic circuits avoid the use of a large number of cascaded logic gates as in the case of adders by means of well-designed logic combinations, thus significantly reducing the number of gates used. In terms of circuit performance, fewer gate circuits mean shorter signal transmission paths, which can effectively reduce the delay time of signals and enable the circuit to respond to input signals more quickly. At the same time, the reduction in the number of gates results in a corresponding reduction in power consumption, which is critical for digital systems that need to run for long periods of time and have stringent power consumption requirements.

Based on the results of these comparisons, we finally decided to design it in a way that directly uses combinational logic circuits to list truth tables and write expressions. The core of this design approach is to first list the truth table based on the logic rule of adding three binary digits to one. A truth table is a comprehensive table that shows all possible combinations of inputs and their corresponding outputs. For a three-digit binary number $A_2A_1A_0$, there are $2^3 = 8$ different combinations of inputs. By carefully analyzing the output of each input combination after the add-one operation, we can accurately record them in the truth table.

Based on the truth table, we can derive the logical relationship between the output variables and the input variables, and then write the logic expression. Without introducing the A_3 chip select signal, we have obtained a partial logical expression. For Y_0 , the logical expression is $Y_0 = A_0$. This is because in the operation of adding one to a binary number, the result of adding one to the lowest bit, A_0 , is either unchanged (when $A_0 = 0$) or becomes 0 (which results in a rounding off when $A_0 = 1$), so that Y_0 is directly equal to A_0 . For Y_1 , the logical expression is $Y_1 = A_0 \oplus A_1$. The rule for the operation of heteroskedasticity is that the output of the two inputs is 1 when they are different and 0 when they are the same. In the process of adding one to a binary number, when $A_0 = 1$, there will be a rounding, and this rounding will affect the value of A_1 , which will be reversed after the addition operation, and the different-or operation can accurately describe the effect of this rounding on the output of A_1 . For the logical expression of Y_2 (the original content is not given here in full), it will be determined according to the logical relationship between A_0 , A_1 and A_2 , as well as the rules of the plus one operation of the rounding rule, which usually involves a combination of basic logical operations such as and, or, and not.

After obtaining the complete logic expressions, we can draw the circuit diagram based on these expressions. To further optimize the performance of the circuit, we have used a transmission gate instead of the conventional data selector in the design of the circuit diagram. A data selector is a commonly used digital circuit component to select an output from multiple input signals. However, the internal structure of a data selector is relatively complex, containing multiple logic gates and control signals, which results

in a large signal transmission delay and also consumes more power. The transmission gate is a simple switching element based on the MOS tube principle, which is able to conduct or cut off according to the state of the control signal, thus enabling data selection and transmission [6]. When the control signal is high, the gate conducts, allowing the signal to pass; when the control signal is low, the gate cuts off, preventing the signal from passing. By using transmission gates, we have successfully simplified the circuit structure, reduced the delay time of signal transmission, and also reduced the energy consumption of the circuit.

Specifically, when A3 is 1, we use the control signal to turn on Y3Y1 in the figure, and then the circuit outputs the negative original code. Here, the negative original code refers to a form of representation of negative numbers under specific coding rules. When A3 is 0, we change the control signal to energize Y4Y2 in the diagram, which also outputs a negative source code. By introducing a transmission gate and combining it with the A3 chip select signal, we have achieved a flexible data selection function. This function enables the circuit to select the appropriate output path according to different situations, thus improving the practicality and reliability of the circuit. In practice, this flexible data selection function allows the circuit to switch between different operating modes to meet diverse needs. Complementary circuit design is shown in Figure 1.

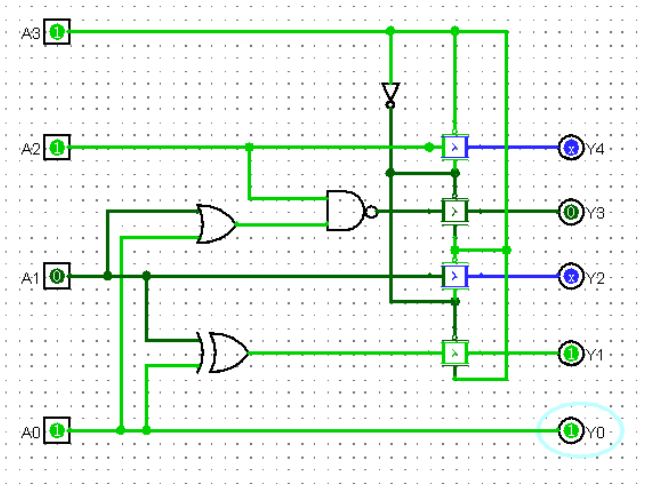


Fig. 1. Complementary circuit design (Photo/Picture credit: Original)

Comparator circuit design. A common way of designing comparators is by using logic expressions [7].

$$Y = A_2B_2' + (A_2 \odot B_2)A_1B_1' + (A_2 \odot B_2)(A_1 \odot B_1)A_0B_0' \quad (1)$$

However, constructing the circuit using the same-or gate requires more MOS tubes than the same-or gate, and the circuit has more delay and power consumption, so use your same-or gate to construct the circuit, using Moore's law to make changes to the

logic expression, here replacing $(A_2 \odot B_2)$ as a whole with $\overline{A_2B_2}$. It is possible to build the circuit without using the same-or gate.

Write the logic expression and draw and circuit diagram based on the truth table: realise that the output is 1 when A2A1A0 is greater than B2B1B0 and 0 for the rest of the cases.

Logical expressions:

$$Y = \overline{\overline{(A_2 B_2)} [(A_2 B_2) A_1 \overline{B_1}]} \overline{\overline{(A_2 B_2) (A_1 B_1) A_0 B_0}} \tag{2}$$

The final comparator circuit diagram is obtained as follows in Figure 2.

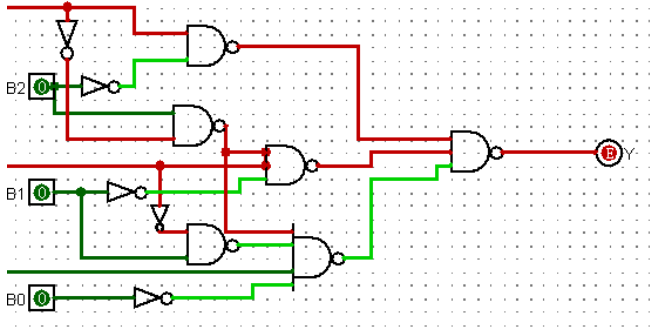


Fig. 2. Comparator circuit design (Photo/Picture credit: Original)

2.3 Circuit diagram as a whole

Overall circuit diagram of the absolute comparator is shown in Figure 3.

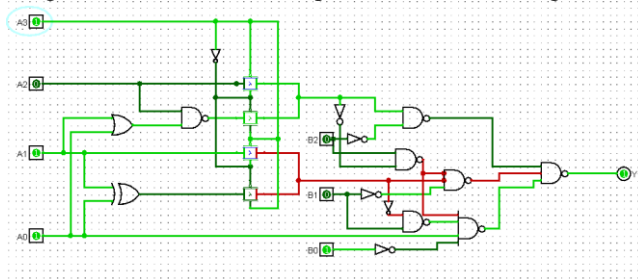


Fig. 3. Overall circuit diagram of the absolute comparator (Photo/Picture credit: Original)

3 Critical path selection

3.1 Comparison of critical paths

Based on the most logic gates passed through, two paths were first selected and then their delays were calculated for each of the two paths and the one with greater delay was defined as the critical path [8].

3.2 Calculation of path delay

In order to find the critical path with the maximum delay, it is necessary to calculate the time delays of two possible paths and then compare them. Here, the circuit design of Critical Path 1 is shown in Figure 4. Determine the optimal capacitor sizes with logical effort [9].

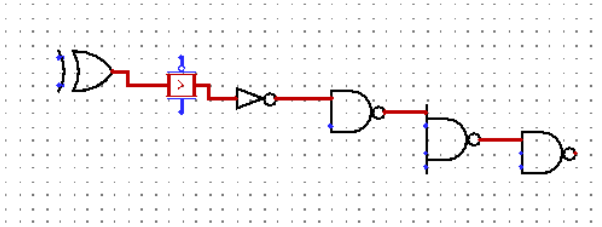


Fig. 4. The design of Critical Path 1 (Photo/Picture credit: Original)

$$G = \sum g = 4 * 1 * 1 * \frac{4}{3} * 2 * \frac{5}{3} = \frac{80}{9} = 8.89 \tag{3}$$

$$H = \frac{C_L}{C_{in}} = \frac{32}{4} = 8 \tag{4}$$

$$B = \sum b = \frac{3+1}{1} = 4 \tag{5}$$

$$F = G \cdot H \cdot B = 284.48 \tag{6}$$

$$f^* = \sqrt[N]{F} = \sqrt[6]{G \cdot H \cdot B} = 2.565 \tag{7}$$

$$D = N \cdot f^* + \sum p = 6 * 2.565 + \sum p = 30.387 \tag{8}$$

And the circuit design of Critical Path 2 is shown in Figure 5.

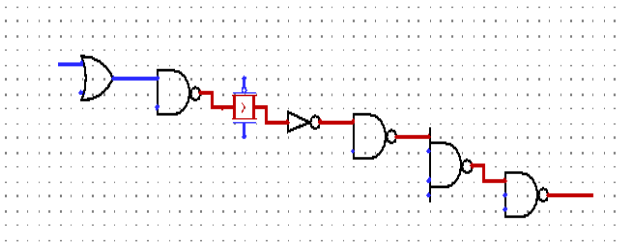


Fig. 5. The design of Critical Path 2 (Photo/Picture credit: Original)

$$G = \sum g = \frac{8}{3} * \frac{4}{3} * 1 * 1 * \frac{4}{3} * 2 * \frac{5}{3} = 15.802 \tag{9}$$

$$H = \frac{C_L}{C_{in}} = \frac{32}{3} = 10.667 \tag{10}$$

$$B = \sum b_{path} = \frac{4+3}{4} = 1.75 \quad (11)$$

$$F = G \cdot H \cdot B = 294.980 \quad (12)$$

$$f^* = \sqrt[7]{F} = \sqrt[7]{G \cdot H \cdot B} = 2.25 \quad (13)$$

$$D = N \cdot f^* + \sum p = 7 * 2.25 + \sum p = 31.75 \quad (14)$$

3.3 Optimization of critical path dimensions

By means of the formula, Capacitor sizes at various stages are shown in Table 1.

Table 1. Capacitor sizes at various stages

Stage	C_{in}
1	2.667
2	1.46
3	2.466
4	5.549
5	12.486
6	21.070
7	23.704
C_{load}	32

4 Optimization of energy consumption

4.1 Calculate the energy consumption of the critical path

According to formula $E = P * C * VCC^2$ [10]. Firstly, the probability of each node on the critical path converting from 0 to 1 is calculated and the probabilities are shown in table 2 below.

Table 2. The probability of each node transitioning from 0 to 1

Stage	P (0)	P (1)	P (0→1)
1	1/4	3/4	3/16
2	7/8	1/8	7/64
3	11/16	5/16	0.215
4	5/16	11/16	0.215
5	55/256	201/256	0.169
6	0.9165	0.035	0.034
7	0.54	0.46	0.248

Bringing in the energy consumption formula at $Vcc = 1V$ calculates the energy consumption to be 17.402.

4.2 Optimization of energy consumption

Voltage change only. According to the formula

$$\frac{V}{(V-2)^2} \bigg/ \frac{1}{(1-0.2)^2} = 1.5V, V = 0.78 \tag{15}$$

$$E' = 17.402 * (0.78)^2 = 60.84\% \tag{16}$$

Calculated critical path energy consumption is 10.587 reduced to 61% of original

Changing capacitor size only. Based on the formula

$$t' = \sum g_i h_i + \sum p_i = 47.625 \tag{17}$$

$$\sum g_i h_i = \frac{8}{3}h_1 + \frac{4}{3}h_2 + h_3 + h_4 + \frac{4}{3}h_5 + 2h_6 + \frac{5}{3}h_7 = 31.625 \tag{18}$$

The energy consumption of the critical path is calculated to be E' = 10.569. The capacitor sizes on the critical path are as follows in Table 3.

Table 3. Capacitor sizes at various stages

Stage	1	2	3	4	5	6	7	C_{load}
h	0.49 8	0.75 2	1	1.3 3	1.5	11.29 9	1.416	32
C_{in}	2.67	1.33	1	1	1.3 3	2	22.59 8	32

Simultaneous change of voltage and size. Based on the formula

$$\frac{V}{(V-0.2)^2} = 1.5625m = \frac{1.5}{m} * 31.75 - 16 \tag{19}$$

Obtain $m = 1.302$ and $V = 0.844$ when energy consumption is minimized. The dimensions of the capacitors are obtained as follows in Table 4.

Table 4. Capacitor sizes at various stages

Stage	1	2	3	4	5	6	7	C_{load}
h	0.49 8	0.75 2	1	1.3 3	1.79 5	2.88 7	4.64 2	32
C_{in}	2.67	1.33	1	1	1.33	2	6.89	32

The energy consumption of the critical path is calculated to be E' = 8.753.

5 Conclusion

In circuit design, there exists a notable trade - off between performance, exemplified by delay, and energy consumption. Through optimization experiments on absolute value comparators, it has been proven that appropriately sacrificing circuit performance can lead to a substantial reduction in energy consumption.

When only the voltage is adjusted, stretching the delay to 1.5 times the original and decreasing the operating voltage can cut the energy consumption to 60.84% of the initial value. This is due to the squared relationship between voltage and energy consumption, indicating that even with the increased delay, there is still a significant improvement in energy efficiency.

Merely optimizing the capacitor size can reduce energy consumption to 60.73% of the original under the same delay condition. This is achieved by methods like reducing switching activity or load capacitance, demonstrating that capacitor optimization effectively lessens dynamic power consumption related to capacitor charging and discharging.

The combined optimization of voltage and capacitor sizing is even more efficient, further reducing energy consumption to 51.42% of the original. This showcases the potential of co - optimization, maximizing energy efficiency gains while still meeting the delay requirements.

In general, circuit power consumption can be remarkably decreased through voltage scaling, capacitance optimization, or their combination, even with a slight compromise in performance such as increased latency. This validates the "performance - for - efficiency" strategy, which is particularly useful in latency - insensitive yet efficiency - oriented application scenarios like low - power embedded systems. Future research can explore the synergistic adjustment of additional circuit parameters, such as threshold voltage and architecture - level optimization, to expand the Pareto - optimal performance - energy boundary.

References

1. Iranmanesh, S., Raikos, G., Jiang, Z., et al.: CMOS Implementation of a Low Power Absolute Value Comparator Circuit. Proc. 14th IEEE Int. New Circuits and Systems Conf. (NEWCAS), 2016
2. Appali, R., Petersen, S., van Rienen, U.: A Comparison of Hodgkin-Huxley and Soliton Neural Theories, *Adv. Radio Sci.*, 8(115), 75–79(2010)
3. Saravanakumar, S., Srinivasan, K., Rukkumani, V., et al.: Delay Minimization and Evaluation in Logic Paths of RC Interconnects by Unified Logical Effort. Proc. Int. Conf. Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), 2021
4. Karandikar, S.K., Sapatnekar, S.S.: Technology Mapping for Solving the Load Distribution Problem Using Logical Effort, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 27(1), 45–58(2008)
5. Piguet, C.: Logic Design for Low-Power CMOS Circuits. Proc. IEEE TENCON Region 10 Int. Conf. Microelectronics and VLSI, Asia-Pacific Microelectronics 2000, 1995
6. Hao, S.Y., Kim, K.H., Kim, Y.H.: Critical Path Analysis Considering the Signal Transition Time. Proc. 6th Int. Conf. VLSI and CAD (ICVC), 37–40(1999)

7. Hassanpourghadi, M., Zamani, M., Sharifkhani, M.: A Low-Power Low-Offset Dynamic Comparator for Analog to Digital Converters, *Microelectron. J.*, 45(2), 256–262(2014)
8. Yi, M., et al.: A Method for Understanding and Designing Transmission Gate Logic and Static CMOS Logic Circuits, *Electron. World*, 12, 2–6(2013)
9. Singh, K., Jain, A., Mittal, A., et al.: Optimum Transistor Sizing of CMOS Logic Circuits Using Logical Effort Theory and Evolutionary Algorithms, *Integration*, 60, 25–38(2018)
10. Z.S. Zumsteg, C. Kemere, S. ODriscoll, G. Santhanam, R.E. Ahmed, K.V. Shenoy, T.H. Meng, "Power feasibility of implantable digital spike sorting circuits for neural prosthetic systems", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13(3), 272-279, 2005.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

