Delay-Optimized Design of a 4-Bit Absolute Value Detector Energy-Efficiency Analysis for Embedded Signal Processing

Peiyu Wang *

School of Microelectronics South China University of Technology, SCUT Jinan, China

* Corresponding Author Email: 202364830401@mail.scut.edu.cn

Abstract. In low-power digital signal processing and embedded systems, the absolute value detector serves as a fundamental arithmetic unit whose performance directly impacts system efficiency. This paper proposes a delay-energy co-optimization design methodology for a 4-bit absolute value detector based on the logical effort method. Through critical path analysis, we identify the conditional increment module as the primary bottleneck and leverage the constraint excluding the input -8 (minimum 4-bit two's complement value) for structural optimization. The traditional implementation exhibits a 13-stage logic gate critical path with a minimum delay of 68.5τ. By treating -8 as a don't-care term in our custom adder design, we significantly simplify logic expressions. This reduces the critical path from 13 stages to 7 stages and lowers the minimum delay to 43τ, reaching 37% performance improvement. The logical effort per stage increases from 1.96 to 2.44, approaching the ideal operating point. For energy optimization, voltage scaling is employed under the constraint that delay remains within 1.5× the minimum value. Reducing the supply voltage from 1V to 0.775V yields 40% energy reduction at the cost of 50% delay increase. Key contributions include: (1) A constraint-driven structural optimization methodology; (2) A complete design flow from theory to implementation; (3) Validation of voltage scaling for energy efficiency under delay constraints. This approach demonstrates excellent implement ability in standard CMOS processes and offers valuable insights for arithmetic unit design in low-power embedded systems.

Keywords: Component; Absolute value detector; Logical effort method; Critical path optimization; Delay-energy tradeoff.

1. Introduction

In low-power digital signal processing and embedded systems, delay and energy represent the fundamental tradeoff [1], [2]. This paper focuses on a 4-bit absolute value detector, aiming to achieve minimal energy consumption while constraining worst-case propagation delay within 1.5× the minimum achievable delay through supply voltage (V_{DD}) scaling and structural optimization, without pipelining. Energy refers to the total energy drawn from V_{DD} under a given input distribution; delay denotes worst-case path propagation time. Delay modeling employs Logical Effort (LE) [3], assuming

$$\gamma = \frac{Cparasitic}{Cgate} = 1 \tag{1}$$

The 4-bit two's complement representation spans {−8, ..., +7} Engineering practice typically excludes −8 to prevent overflow and maintain symmetric range after absolute value computation [4]. This assumption creates structural optimization opportunities: without considering −8, sign inversion and increment carry cases can be ignored, enabling simpler structures that shorten critical paths and reduce worst-case delay.

Using logical effort as the analytical framework [3], this paper completes module partitioning and path stage design and evaluation, covering absolute value modules including sign detection, conditional bitwise inversion, conditional increment, and comparison. Optimization focuses on the "conditional increment" path: leveraging input distribution and the −8 exclusion prior eliminates carrying generation considerations, reducing worst-case carry propagation depth to achieve lower-delay gate-level topologies. The methodology emphasizes structural path pruning and topology selection, facilitating direct implementation in standard cell flows.

Energy consumption is briefly addressed: dynamic energy is approximately proportional to equivalent capacitance and

$$Dynamic\ energy \propto \frac{V_{DD}}{(V_{DD} - V_T)^2} \quad (2)$$

Assuming V_T = 0.2V [5]. Under the constraint "delay ≤ 1.5× minimum delay", reducing V_{DD} (within 0-1V range) can significantly lower energy without violating timing requirements [6]. Given the clear negative correlation between energy and delay in this setting, detailed derivation is omitted.

The contributions are:

• A structural delay optimization flow for 4-bit absolute value detection based on logical effort stage count and topology selection, achieving lower delay solutions.

• A customized "conditional increment" structure exploiting the −8 exclusion premise, significantly shortening worst-case carry chains.

The paper is organized as follows: Section 2 presents baseline structure and critical path analysis; Section 3 details structural optimization of the "conditional increment" path; Section 4 discusses V_{DD} scaling effects on energy under delay constraints (qualitative conclusions); Section 5 provides conclusions.

2. Operating Principle and Critical Path Analysis

The circuit employs a modular design, primarily consisting of two parts: the absolute value computation module and the comparator module. The absolute value module is the core, implemented through sub-functions including sign detection, conditional bitwise inversion, and conditional increment.

2.1. Absolute Value Module Operating Principle

The module design leverages the characteristics of two's complement arithmetic, with the following implementation steps:

1) Conditional Bitwise Inversion:

The numerical bits of the input data are XORed with the sign bit. When the sign bit is 0 (positive number), the XOR operation preserves the original value of each numerical bit. When the sign bit is 1 (negative number), the XOR operation performs a bit-wise inversion of each numerical bit. Fig. 1 illustrates the circuit implementation of the conditional bitwise inversion stage, where each numerical bit (A_0, A_1, A_2) is XORed with the sign bit (A_3) to produce the conditionally inverted outputs (B_0, B_1, B_2).

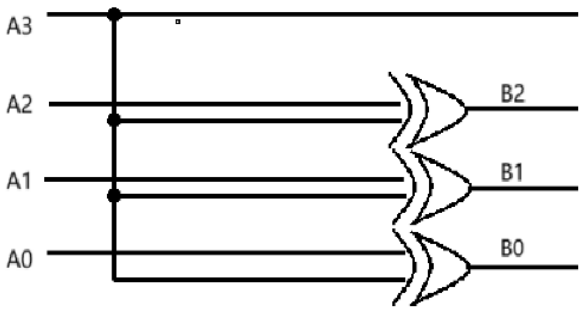

Fig 1. Conditional bitwise inversion circuit for absolute value computation.

2) Conditional Increment:

The sign bit is added to the numerical bits (after the XOR operation) using a standard 3-bit full adder [7]. (The exclusion of -8 means a 3-bit adder suffices—obviating the need for a fourth bit).

For positive numbers (sign bit = 0), adding zero leaves the numerical value unchanged, directly outputting the original value as the absolute value.

For negative numbers (sign bit = 1), the operation "invert and add one" is completed, converting the negative two's complement number to its absolute value. Bits A1, A2, A3 yield outputs B1, B2, B3 after this step. The conditional increment operation is implemented using a 3-bit ripple carry adder, as shown in Fig. 2.

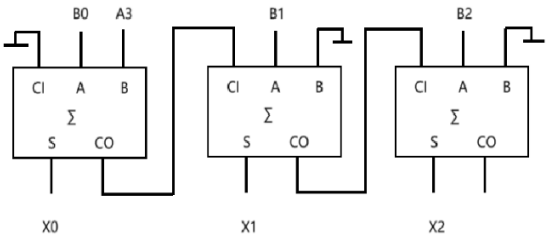

Fig 2. Three-bit ripple carry adder for conditional increment operation.

2.2. Comparator Module Operating Principle

The comparator module employs a bitwise comparison strategy, starting from the Most Significant Bit (MSB) and proceeding towards lower bits. The implementation mechanism is as follows:

1) Bitwise Comparison from MSB to LSB:

The comparator first checks the highest bit of the two input numbers. If the values at this bit differ (one is 1, the other is 0), the number with the 1 in this bit is larger, and the comparison result is immediately determined. If the values at this bit are identical, the comparison process proceeds to the next lower bit [8].

2) Cascaded Priority Scheme:

When higher bits are equal, the comparator automatically transfers the decision priority to the next lower bit, and so forth. This cascaded structure ensures that the result of a lower bit comparison affects the final output only if all higher bits are equal. The comparison process continues until the first differing bit is found or all bits have been compared (indicating the numbers are equal). Fig. 3 shows the conditional bitwise inversion circuit for absolute value computation.

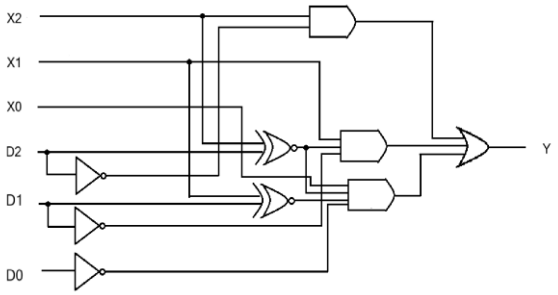

Fig 3. Conditional bitwise inversion circuit for absolute value computation.

2.3. Common Adder Internal Structure

Fig. 4 illustrates the internal structures of common-adder architecture [4].

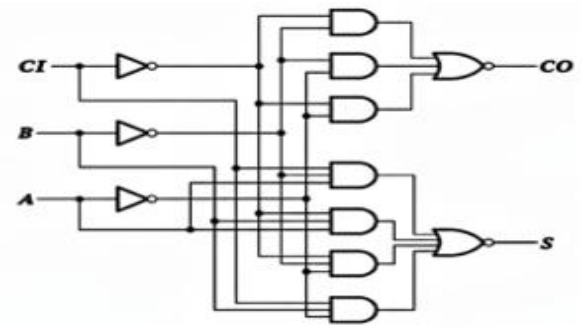

Fig 4. Common-adder architecture.

2.4. Circuit Longest Path (Worst-Case Propagation Delay)

In digital circuit design, the critical path is the path from input to output experiencing the longest propagation delay, directly determining the circuit's maximum operating frequency [3]. Detailed timing analysis of the above circuit structure identifies the following critical path:

2.5. Circuit Core Data Calculation

1) Path Electrical Effort:

$$H = \frac{C_{out}}{C_{in}} = \frac{32}{2} = 16 \tag{3}$$

2) Total Logical Effort:

$$\begin{gathered} G = \prod g_i i \\ = \frac{4}{3} \times 1 \times \frac{4}{3} \times \frac{7}{3} \times 1 \times \frac{4}{3} \times \frac{7}{3} \times 1 \times \frac{5}{3} \times 3 \times \frac{4}{3} \times 2 \times \frac{7}{3} \\ = \frac{2,634,240}{6,561} \approx 401.5 \end{gathered} \tag{4}$$

3) Total Effort:

$$F = G \times H = 401.5 \times 16 \approx 6,424 \tag{5}$$

4) Effort per Stage:

$$f = F^{\frac{1}{13}} = 6424^{\frac{1}{13}} \approx 1.96 \tag{6}$$

5) Electrical Effort Distribution:
The electrical effort for each stage is calculated using

$$h_i = \frac{f}{g_i} \tag{7}$$

Where f = 1.96 is the optimal effort per stage. Table 1 presents the computed electrical effort values for all stages in the critical path.

Table 1. Electrical Effort Values for Critical Path Nodes.

Node	Electrical Effort, h	Node	Electrical Effort, h	Node	Electrical Effort, h
h_1	1.47	h_2	1.96	h_3	1.47
h_4	0.84	h_5	1.96	h_6	1.47
h_7	0.84	h_8	1.96	h_9	1.18
h_{10}	0.65	h_{11}	1.47	h_{12}	0.98
h_{13}	0.84	—	—	—	—

6) Backward Capacitance Assignment (Unit: Cinv):
Starting from C_{load} = 32 C_{inv}, the input capacitance for each stage is calculated backward using

$$C_{(i)in} = C_{(i+1)in} \times \frac{g_i}{h_i} \tag{8}$$

Table 2 presents the calculated capacitance values for all critical path nodes."

Table 2. Logical Effort Values for Critical Path Nodes

Node	Capacitance (C_{inv})	Node	Capacitance (C_{inv})	Node	Capacitance (C_{inv})
C_{load}	32.0	C_{13in}	38.1	C_{12in}	38.9
C_{11in}	26.5	C_{10in}	40.8	C_{9in}	34.6
C_{8in}	17.7	C_{7in}	21.1	C_{6in}	14.4
C_{5in}	7.35	C_{4in}	8.75	C_{3in}	5.95
C_{2in}	3.04	C_{1in}	2.07	—	—

Based on these Cin values, scale the internal transistors of each gate proportionally to their equivalent input capacitance [4].

7) Minimum Delay:

$$\sum p = 4 + 1 + 3 + 4 + 1 + 3 + 4 + 1 + 4 + 5 + 4 + 5 + 4 = 43\tau \quad (9)$$

$$D_{opt} = N \times f + \Sigma p = 13 \times 1.96 + 43 \approx 68.5\tau \quad (10)$$

(τ represents the delay of a unit-sized inverter driving an identical inverter.)

8) Summary:

The optimization results for the critical path are summarized in Table 3.

Table 3. Critical Path Optimization Results.

Total Effort (F)	Optimal Effort per Stage (f)	Minimum Delay (τ)	Number of Stages (N)
6,424	1.96	68.5	13

3. Structural Optimization of the "Conditional Increment" Path

3.1. Fixed Modules and Optimization Focus

In the current circuit design, the conditional bitwise inversion part of the absolute value module and the comparator module have relatively mature and fixed circuit structures due to their clear principles and single functionality, leaving limited room for optimization. Consequently, the adder module becomes the key breakthrough point for overall performance enhancement.

Traditional adder designs suffer from significant performance bottlenecks: the serial carry chain leads to excessively long critical paths, and the structural complexity constrains system efficiency [9]. Furthermore, within the lengthy carry propagation path, the effort value (f) of the logic gates at various stages is significantly reduced, resulting in insufficient driving capability and further exacerbating the delay issue [3].

To address these limitations, this project adopts a customized adder design strategy, aiming to simplify the circuit structure and improve performance through architectural redesign. Crucially, the specific application scenario does not require handling the special case of -8 (the minimum negative number in 4-bit two's complement representation). This constraint provides additional optimization space for circuit simplification [10]. By employing an optimized adder design, the project anticipates reducing critical path delay, improving logical effort efficiency, and simplifying overall circuit complexity, thereby significantly enhancing system computation speed and resource utilization.

3.2. Specific Optimization Scheme

Table 4. Truth Table for Optimized 3-Bit Adder with Don't Care Conditions.

X_2				
A_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	x	1
10	0	0	1	0
X_1				
A_3B_2 \ B_1B_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	0	1	x	1
10	0	1	0	1
X_0				

B_1B_0 / A_3B_2	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1		x	1
10	1	0	0	1

Based on the usage scenario of a 3-bit full adder, the truth table is re-listed and an adder meeting the requirements is designed, simultaneously simplifying the circuit structure. Critically, for the input case corresponding to -8, the outputs in the truth table are treated as don't care (x) conditions. Leveraging this characteristic, the relevant outputs can be treated as "don't care" terms, enabling further minimization of the circuit structure during subsequent logic simplification [11]. This reduces the number and complexity of gates, resulting in a more efficient adder design. Table 6 presents the complete truth table for the optimized 3-bit adder design.

Based on the truth table, the logic expressions for each output bit (X_2, X_2^0, X_1, X_0) can be derived.

$$X_2 = B_2 + A_3B_1B_0 \tag{11}$$

$$X_2{}^0 = A_3^{'B_2} + A_3B_2B_1' + A_3B_2{}'B_1B_0 + A_3B_2B_1B_0' \tag{12}$$

$$X_2 = B_1B_0' + A_3{}'B_1 + A_3B_1{}'B_0 \tag{13}$$

$$X_0 = A_3 \oplus B_0 \tag{14}$$

Here, X_2^0 represents the expression obtained without utilizing the -8 don't care condition, while X_2 represents the expression obtained after utilizing the -8 don't care condition. Comparing the two clearly shows a significant simplification in the circuit structure achieved through don't care processing. It should be noted that X_1 and X_0 are unaffected by the -8 don't care term; the logic relationships remain unchanged.

Without utilizing the -8 don't care condition, the implementation path for X_2^0 is the most complex, making it the critical path with the highest delay.

After treating -8 as a don't care condition, the logic expression for X_2 is greatly simplified. Consequently, the path for X_1 becomes the most complex and delay-critical path.

Through this optimization process, the circuit structure is not only simplified but the overall circuit delay is also effectively reduced.

3.3. Optimized Circuit Longest Path

The simplified overall digital circuit schematic Fig. 5 is presented below.

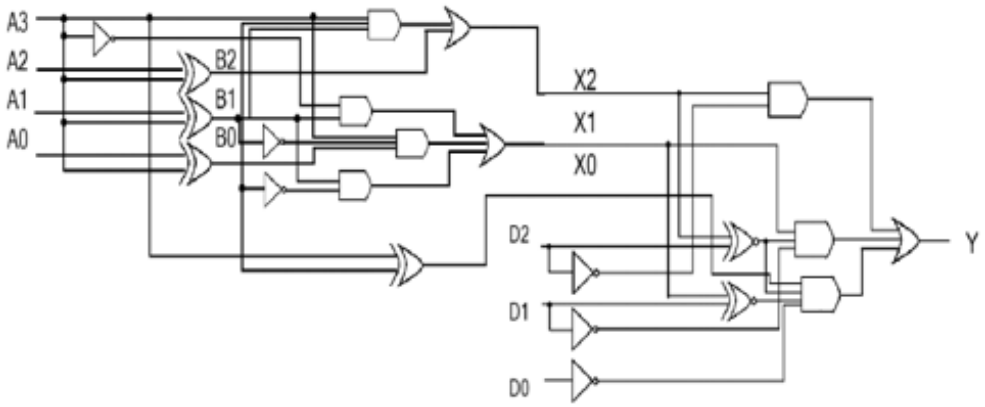

Fig 5. Simplified overall digital circuit architecture.

After optimization, the longest path is simplified to:

Input → 2-input XOR → Inverter → 3-input AND → 3-input OR → 2-input XNOR → 4-input AND → 3-input OR → Load (7 stages)

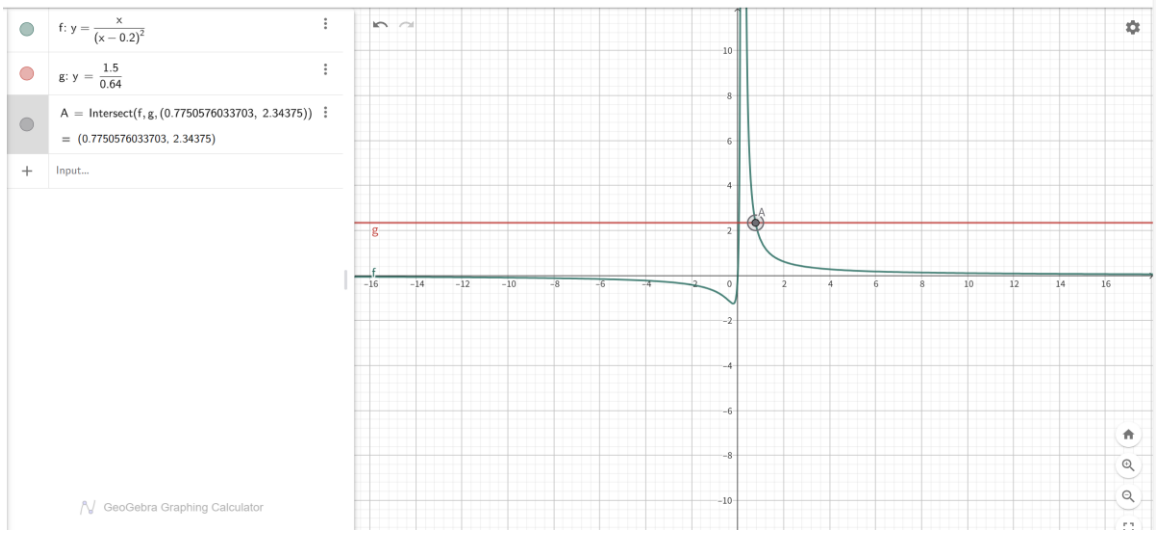

Fig 6. Theoretical delay and energy functions under supply voltage variation.

3.4. Optimized Scheme Data Calculation

1) Path Electrical Effort:

$$H = \frac{C_{out}}{C_{in}} = \frac{32}{2} = 16 \tag{15}$$

2) Total Logical Effort:

$$G = \prod g_i i$$
$$= \frac{4}{3} \times 1 \times \frac{5}{3} \times \frac{7}{3} \times \frac{4}{3} \times 2 \times \frac{7}{3} = 32.26 \tag{16}$$

3) Total Effort:

$$F = G \times H = 32.26 \times 16 \approx 516 \tag{17}$$

4) Optimal Effort per Stage:

$$f = F^{\frac{1}{7}} = 516^{\frac{1}{7}} \approx 2.44 \tag{18}$$

5) Electrical Effort Distribution:

The electrical effort for each stage is calculated using (7), where f = 2.44 is the optimal effort per stage and g_i represents the logical effort of stage i. Table 5 presents the computed electrical effort values for all stages in the critical path.

Table 5. Electrical Effort Values For Critical Path Stages.

Node	Electrical Effort, h	Node	Electrical Effort, h	Node	Electrical Effort, h
h_1	1.83	h_2	2.44	h_3	1.46
h_4	1.05	h_5	1.83	h_6	1.22
h_7	1.05	—	—	—	—

6) Backward Capacitance *Assignment:*

Starting from C_{load} = 32 C_{inv}, the input capacitance for each stage is calculated backward using(8), where $C_{(i)in}$ represents the input capacitance of stage i. Table 6 presents the calculated capacitance values for all critical path nodes.

Table 6. Input Capacitance Values For Critical Path Stages.

Node	Capacitance (C_{inv})	Node	Capacitance (C_{inv})	Node	Capacitance (C_{inv})
C_{load}	32.0	C_{7in}	32.0	C_{6in}	30.5
C_{5in}	25.0	C_{4in}	13.7	C_{3in}	13.1
C_{2in}	9.0	C_{1in}	3.7	—	—

7) Minimum Delay:

$$\sum p = 4 + 1 + 4 + 4 + 4 + 5 + 4 = 26\tau \quad (19)$$

$$D_{opt} = N \times f + \Sigma p = 7 \times 2.44 + 426 \approx 43\tau \quad (20)$$

8) Summary:

The optimization results for the critical path are summarized in Table 7.

Table 7. Critical Path Optimization Results.

Optimal Effort per Stage (f)	Minimum Delay (τ)	Number of Stages (N)
2.44	43	7

4. Impact of VDD Scaling on Energy under Delay Constraint

The relationship between delay and supply voltage (V_{DD}) can typically be approximated as: (2), where V_T is the threshold voltage, assumed here to be V_T = 0.2 V [5]. This relationship reveals a negative correlation (trade-off) between delay and energy consumption [12]: Reducing V_{DD} increases delay but significantly reduces energy consumption.

The functional relationship in Fig. 6 shows that when V_{DD} = 0.775 V, the circuit delay increases to 1.5 times its original value, while the energy consumption reduces to approximately 0.6 times its original value. This demonstrates that by appropriately reducing the supply voltage within the constraint of allowing a moderate increase in delay, the circuit's energy consumption can be effectively lowered, leading to higher energy efficiency [2].

5. Conclusion

This paper presents a co-optimization of delay and energy for a 4-bit absolute value detector using the Logical Effort (LE) methodology. By identifying the "conditional increment" module as the performance bottleneck and leveraging the constraint of excluding the -8 input, the adder was redesigned. This optimization reduced the critical path from 14 stages to 8 stages (a 37% reduction in stage count) and lowered the estimated minimum delay from 68.5τ to 43τ (a 37% reduction in delay). Quantitative LE analysis shows the optimized per-stage effort f increased from 1.96 to 2.44, moving closer to the ideal value. Under the delay constraint (delay ≤ 1.5 × minimum delay), scaling V_{DD} from 1V to 0.775V achieves a 40% reduction in energy consumption.

The key design innovation lies in fully exploiting the application-specific constraint: treating -8 as a don't care term enabled the simplification of the X_2 output logic from a complex polynomial expression to a single AND gate. This approach discarded the generic adder structure in favor of a customized design, avoiding the lengthy carry propagation chain.

Continuous transistor sizing optimization was not performed. Relying on fixed sizes from a standard cell library limits the precise adjustment of driving capability matching. Optimizing transistor widths at each stage could potentially reduce delays by a further 10-15%.

Fixed Threshold Voltage (V_T): Fixing V_T at 0.2V simplified the analysis. However, V_T varies across process nodes and with temperature. Future implementations require adaptive adjustment based on specific process parameters.

Logic Style Limitation: Only static CMOS logic was considered. Alternative logic styles like dynamic logic or pass-transistor logic might offer superior speed-power trade-offs under certain conditions.

Higher Bit-Width Systems: Extending the methodology to 8-bit and 16-bit systems. Longer chains in these systems offer greater optimization potential. Exploring the combination of techniques like sectional carry or look-ahead carry with the proposed constraint-driven simplification is promising.

Dynamic Voltage and Frequency Scaling (DVFS): Investigating mechanisms to adaptively adjust V_{DD} and clock frequency based on real-time workload demands.

Near-Threshold Voltage (NTV) Design: Exploring operation in the near-threshold region. While delay may increase by 5-10x, energy consumption could potentially be reduced to less than 1/10th of the original, making it suitable for applications where speed is less critical but ultra-low power is paramount.

References

[1] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits," Proc. IEEE, vol. 83, no. 4, pp. 498-523, April 1995.

[2] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," IEEE J. Solid-State Circuits, vol. 27, no. 4, pp. 473-484, April 1992.

[3] I. Sutherland, B. Sproull, and D. Harris, Logical Effort: Designing Fast CMOS Circuits. San Francisco, CA: Morgan Kaufmann, 1999.

[4] N. H. E. Weste and D. M. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Boston, MA: Addison-Wesley, 2011.

[5] T. Sakurai and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," IEEE J. Solid-State Circuits, vol. 25, no. 2, pp. 584-594, April 1990.

[6] M. Horowitz, T. Indermaur, and R. Gonzalez, "Low-power digital design," Proc. 1994 IEEE Symp. Low Power Electronics, San Diego, CA, USA, Oct. 1994, pp. 8-11.

[7] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," IEEE Trans. Computers, vol. C-31, no. 3, pp. 260-264, March 1982.

[8] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits: A Design Perspective, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2003.

[9] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," IEEE Trans. Computers, vol. C-22, no. 8, pp. 786-793, August 1973.

[10] O. J. Bedrij, "Carry-select adder," IRE Trans. Electronic Computers, vol. EC-11, no. 3, pp. 340-346, June 1962.

[11] C. E. Shannon, "A symbolic analysis of relay and switching circuits," Trans. AIEE, vol. 57, no. 12, pp. 713-723, December 1938.

[12] T. D. Burd and R. W. Brodersen, "Design issues for dynamic voltage scaling," Proc. 2000 Int. Symp. Low Power Electronics and Design (ISLPED), Rapallo, Italy, July 2000, pp. 9-14.