

Area Efficient Digital Comparator Design Using Pseudo NMOS and Pass Transistor Logic

Udayakumar M¹, Kumar P²

Department of Electronics and Communication Engineering, K. S. Rangasamy College of technology, Tiruchengode, Namakkal, Tamilnadu, India.

Department of Electronics and Communication Engineering, K. S. Rangasamy College of technology, Tiruchengode, Namakkal, Tamilnadu, India.

ABSTRACT- Comparators are one of the most key design elements for a large number of applications. This paper presents wide-range, area efficient and high-speed comparator using pseudo nMOS and pass transistor logic. The conventional CMOS design requires more number of transistors than pseudo nMOS and pass transistor logic. Hence the transistor count of a design can be reduced by using pseudo nMOS and pass transistor logic instead of conventional CMOS. The working operation of the comparator based on a novel scalable parallel prefix structure. The comparison starts from most significant bits (MSB) of each sequence, if the MSB bit of two different sequences is equal that time only the comparison operation moves towards the least significant bit (LSB). This method helps to minimize the dynamic power dissipation by eliminating unwanted transitions in a parallel prefix structure. Then the N -bit comparison result will be generated after $\lceil \log_4 M \rceil + \lceil \log_{16} M \rceil + 4$ CMOS gate delays. This comparator is designed by interconnecting CMOS gates with a maximum number of fan-in and fan-out of five and four, respectively, not depending on comparator bitwidth. This comparator plays a role for high speed and power efficiency for large bitwidth.

KEY WORDS- Bitwise competition logic, Pass transistor logic, pseudo nMOS logic.

I. INTRODUCTION

COMPARATORS are one of the most important design elements for a large number of applications such as scientific computation, test circuit applications [1] and general purpose processor components [2], [3]. The comparator logic design is straightforward; the wide use of comparators in high-performance systems having a great prominence on performance and power consumption optimizations.

The specialized arithmetic units are used for large comparisons, along with custom logic circuits. The comparator designs also use subtractors in the form of flat adder components, but this type of designs are commonly slow and area-demand, even when implemented using fast adders [4]. The comparator designs using a hierarchical prefix tree structure composed of 2-b comparators minimize comparison delays and improve scalability [9]. While comparing wide operands, the delay and area of these designs are high.

Pipelining is used to reduce power consumption and improve the speed and also minimize the switching transitions with respect to the actual input operands' bit values [7]. One comparator design uses all- N -transistor circuits for compensating high fan-in with high pipeline throughput [8]. The multiplexer based comparator architectures have two stages, the operation of the two stages be followed. The first stage has eight modules for 8-bit comparison, this module results are given as an input to the priority encoder. The second stage has 8-to-1 multiplexer to select the suitable result from the first stage [13]. In this architecture, two-phase domino clocking is used to execute both stages in one clock cycle. Here the design highly affected to race conditions when operations present on the rising and falling clock edges, this operation limits the operating speed and jitter margin.

Some comparator architectures save power by using novel ripple-based structures. This type of architectures incorporates wide-range ripple-carry adders [16]. Here the novel ripple-based structure eliminates unwanted computations dynamically. Compute-on-demand comparators compare a pair of binary numbers single bit at a time, moving from the MSB to the LSB. If the compared bits are equal it enables the comparison of the next bit else represents the final comparison decision. All bits of greater significance are equal then only the next comparison cell gets activated. Although these designs

reduce switching, they suffer from long worst case comparison delays for large worst case operands.

Some comparator architecture does not have any arithmetic operations. This type of architecture can reduce the long delays affected by bitwise ripple designs. The larger number can be found by determining which operand possesses the 1 bit at LSB after pre-encoding, before supplying the numbers to bitwise competition logic (BCL) structure [17]. The BCL structure dismemberment the operands into 8-bit blocks, after that the result for each block is input into a multiplexer to find the final comparison decision. So the BCL-based design has a low transistor count. So this design has the potential for low power consumption. In this BCL structure, the pre-encoder logic modules limit the maximum operating frequency being achieved. In addition, to enable the BCL unit some special control logic is needed. It switches dynamically in a synchronized fashion, thus considerably increase the power consumption and lowering the operating frequency.

To reduce the number of transistors required for the previous designs, this paper influence pass transistor logic and pseudo nMOS logic cells to architect fast,scalable, wide-range, power-efficient and area efficient algorithmic comparators.

II.ARCHITECTURAL OVERVIEWOF THE COMPARATOR

The comparison resolution module in Fig.1 is a novel MSB-to-LSB parallel-prefix tree structure that performs a bitwise comparison of two *N*-bit operands *A* and *B*, denoted as $A_{N-1}, A_{N-2}, \dots, A_0$ and $B_{N-1}, B_{N-2}, \dots, B_0$, where the index range from *N*-1 for the MSB to 0 for the LSB. The comparison resolution module performs the bitwise comparison operation asynchronously from left to right, such that the computation of comparison logic is triggered only if all bits of greater significance are equal.

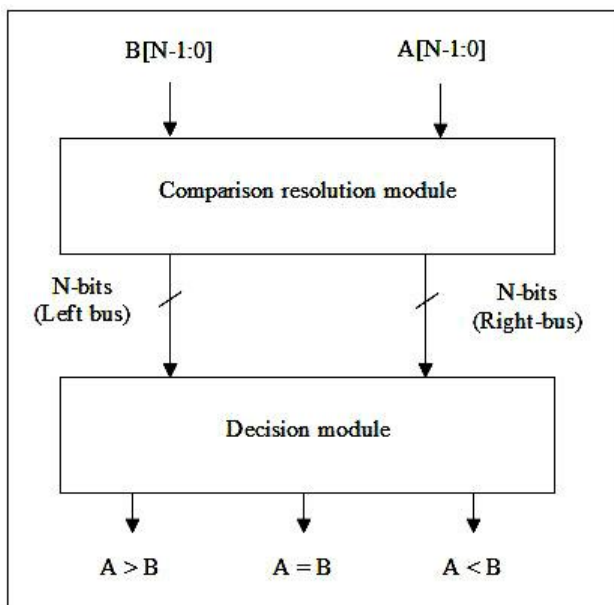


Fig. 1. Block diagram of proposed comparator architecture, consisting of a comparison resolution module connected to a decision module [5].

The bitwise comparison results are stored into two *N*-bit buses using parallel prefix structure. The two buses are the left bus and the right bus, each of which store the partial comparison result as each bit position is evaluated, such that

$$\text{if } A_x > B_x, \text{ then left}_x = 1 \text{ and right}_x = 0.$$

$$\text{if } A_x < B_x, \text{ then left}_x = 0 \text{ and right}_x = 1.$$

$$\text{if } A_x = B_x, \text{ then left}_x = 0 \text{ and right}_x = 0.$$

In this, to reduce switching activities, if the bitwise comparison is not equal, the bitwise comparison of all the bits of lower significance is terminated and all such positions are set to zero on both buses. Hence, there is never more than one high bit on either bus. The decision module uses NOR-NAND networks to output the final comparison decision based on all of the bits on the left bus producing the *Lbbit* and all of the bits on the right bus producing the *Rbbit*. If *LbRb*= 00, then *A* = *B*, if *LbRb*= 10 then *A* > *B*, if *LbRb*= 01 then *A* < *B*, and *LbRb*= 11 is not possible.

A 4-b comparison of input operands *A* = 1000 and *B* = 0101 is illustrated in Fig. 2. In the first step, a parallel prefix tree structure generates the encoded data on the left bus and right bus for each pair of corresponding bits from *A* and *B*. In this example, $A_3 = 1$ and $B_3 = 0$ encodes as $Lb_3 = 1$ and $Rb_3 = 0$. At this point, where the bits are unequal, the comparison terminates and a final comparison decision can be made based on the first bit evaluated.

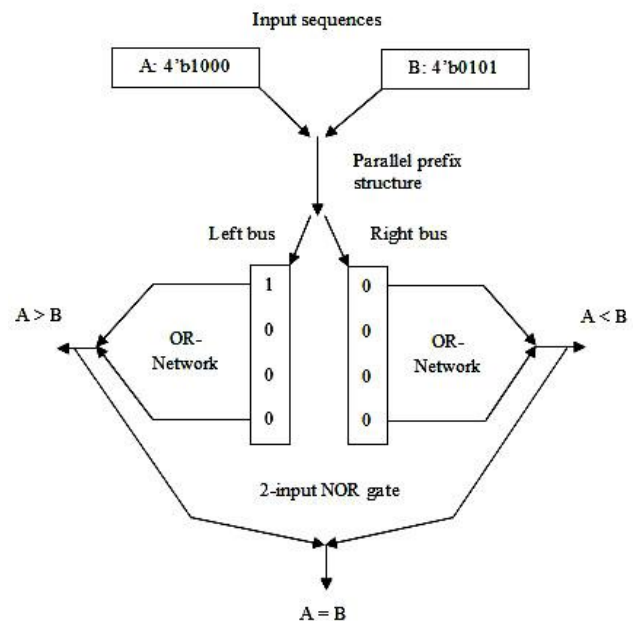


Fig. 2.Example 4-bit comparison.

The parallel prefix structure fixes all bits of lower significance of left and right bus to 0, regardless of the remaining bit values in the operands. In the second step, the OR-networks perform the bus OR-scans, resulting in 0 and 1, respectively, and the final comparison decision is *A* > *B*.

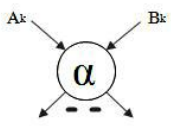

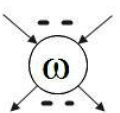
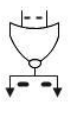
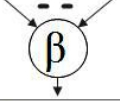
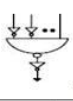
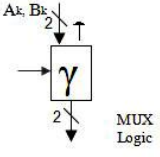
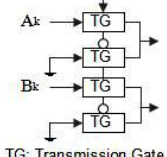
The structure is sectioned into five hierarchical prefixing sets, as shown in Fig. 3, with the associated symbolic representations in Tables I and II. Here each group performs a specific function. The output of each group is given as an input to the next group. At last the fifth group produces the output on the left bus and the right bus.

TABLE I
SYMBOLS AND DEFINITIONS [5].

Symbol (Cells)	Definition
N	Operand bitwidth
A	First input operand
B	Second input operand
R_b	Right bus result bit
L_b	Left bus result bit
Π	Bitwise AND
Σ	Bitwise OR
$CMP\{*\}$	Complement function of set *

In this design all cells are operating in parallel which increases the operating speed. This prefixing set structure bounds the components' fan-in and fan-out regardless of comparator bitwidth and eliminates heavily loaded global signals with parasitic components, thus improving the operating speed and reducing power consumption.

TABLE II
LOGIC GATE REPRESENTATIONS OF SYMBOLS USED IN Fig. 3 [6].

Cells	Logic Gate	Maximum Fan-in/Fan-out (Transistor counts)
		2/4 (6)
		4/4 (5)
		5/1 (16)
		3/2 (10)

III. DESIGN DETAILS OF THE COMPARATOR

The comparator design is based on using a novel parallel prefix tree.

The α -type cells in group 1 compare the N -bit operands A and B bit-by-bit. These cells provide a termination flag T_x to cells in groups 2 and 4, indicating whether the

computation should terminate. These cells compute (where $0 \leq x \leq N - 1$)

$$\alpha : T_x = A_x \oplus B_x. \tag{1}$$

The ω_2 -type cells in group 2 combine the termination flags for each of the four α -type cells from group 1 to limit the fan-in and fan-out to a maximum of four. If all the inputs to the ω_2 -type cells are 0s, then continue the comparisons for bits of lesser significance else comparison gets terminated, and then final decision can be made. For $0 \leq n \leq N/4 - 1$, there is a total of $N/4$ ω_2 -type cells, all functioning in parallel

$$\omega_2 : W_{2,m} = CMP(\sum_{i=4n}^{4n+3} T_i). \tag{2}$$

The ω_3 -type cells in Group 3 are very similar to ω_2 -type cells, but it has more logic levels, different inputs. A ω_3 -type cell doesn't provide any comparison functionality. These cells only limit the fan-in and fan-out irrespective of operand bitwidth. Group 3 provides functionality similar to group 2 to continue or terminate the bitwise comparison activity. If the comparison is terminated, group 3 signals group 4 to set the left bus and right bus bits to 0 for all bits of least significance. For $0 \leq n \leq N/4 - 1$, there is a total of $N/4$ ω_3 -type cells per level, with cell function as

$$\omega_3 : W_{3,m} = CMP(\sum_{i=0}^n W_{2,i}). \tag{3}$$

Group 4 consists of β -type cells. The outputs of these cells control the selection line of γ -type cells in group 5. The comparison outcomes from group 1 gives information about the more significant bits to the β -type cells, which compute ($0 \leq x \leq N - 1$)

$$\beta : Z_k = W_{3, \lfloor x/4 \rfloor} T_x \prod_{i=4\lfloor x/4 \rfloor - 1}^{x-1} \bar{T}_i. \tag{4}$$

In the β -type cells the number of inputs increases from left to right in each partition, finishes with a fan-in of five. Thus, the β -type cells in group 4 determine whether group 5 propagates the bitwise comparison codes. The N γ -type cells in group 5 produce the outputs to left and right bit bus. One input of γ -type cell is (A_x, B_x) and the other is hardwired to "00." The select control input is based on the β -type cell output from group 4. The 2-bits can be defined as the left-bit code (A_x) and the right-bit code (B_x). All left-bit codes and all right-bit codes combine to form the left bus and the right bus, respectively. The output of γ -type cells $f_x^{1,0}$ denotes the "greater-than," "less-than," or "equal to" final comparison decision

$$f_x^{1,0} \begin{cases} 00, & \text{for } A_x = B_x \\ 01, & \text{for } A_x < B_x \\ 10, & \text{for } A_x > B_x \end{cases} \tag{5}$$

Essentially, the 2-b code $f_x^{1,0}$ can be realized by OR-ing all left bits and all right bits separately, as shown in the decision module (Figs. 1 and 2), using an OR-gate network in the form of NOR-NAND gates yielding a more optimum gate structure

$$LG_{1,j}^1 = \sum_{x=4j}^{4j+3} f_x^1(6).$$

$$RG_{1,j}^0 = \sum_{x=4j}^{4j+3} f_x^0(7).$$

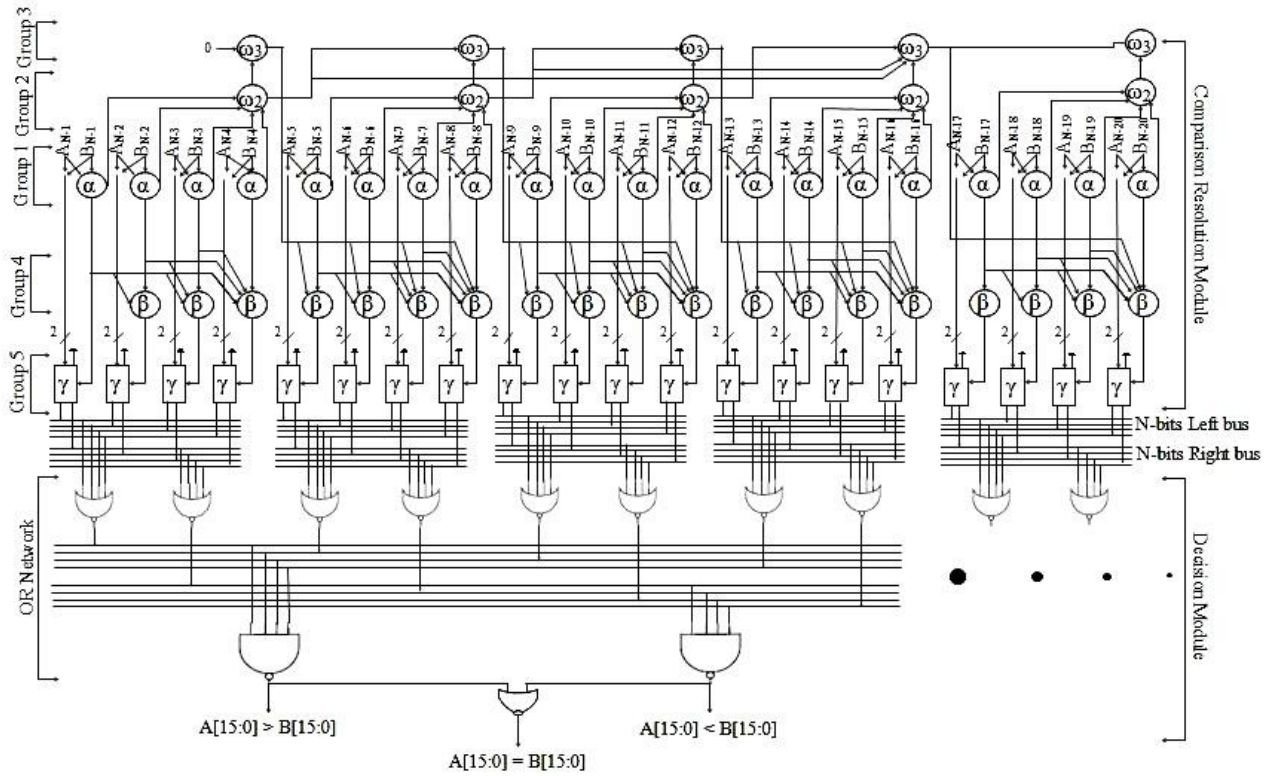


Fig. 3. Implementation details of the comparison resolution module (groups 1 through 5) and the decision module [5].

The power “1” and “0” in (6) and (7) denote the addition of the left and right bits, respectively, and the suffix “1” denotes the first level of OR-logic in the decision module that receives data directly from group 5.

IV. RESULTS AND DISCUSSION

This section analyzes the area (in number of transistors), operating speed, and power requirements of our proposed comparator architecture and calculates the number of logic levels required for an N -bit comparator based on simple logic gates.

A. Area Calculation

The total number of cells required is derived and use Table 4.1 to translate the cell counts into transistors for an N -bit comparator. Based on (1)–(7), the number of E_{CRM} cells required for the comparison resolution module and the numbers of E_{DM} cells in the decision module is, respectively

$$E_{CRM} = (N \times \alpha) + \left(\frac{N}{4} \times \omega_2\right) + (\lceil \log_{16} N \rceil \times \frac{N}{4} \times \omega_3) + (N \times \beta) + (N \times \gamma). \quad (8)$$

$$E_{DM} = 2 \sum_{x=1}^{\lceil \log_{16} N \rceil} \lceil N^x / 4 \rceil \quad (9).$$

Table III shows the total number of cells and the required number of levels per group for various comparator

bitwidths based on (8) and (9). The cell counts in Table III, along with the number of transistors per cell type (Table II), allow us to derive the total number of transistors for various bitwidths (Table IV). The results show an approximate linear growth in comparator size as a function of bitwidth.

B. Input-Output Delay

This section analyzes the critical path delay of the proposed comparator with N -bit inputs. The delay $DELAY_{CRM}$ for the comparison resolution module is

TABLE III
TOTAL NUMBER OF CELLS AND CIRCUIT LEVELS IN EACH GROUP FOR VARIOUS COMPARATOR BITWIDTHS.

Size of the Comparator	Group 1		Group 2		Group 3		Group 4		Group 5	
	Cells	Levels	Cells	Levels	Cells	Levels	Cells	Levels	Cells	Levels
16-b	16 α	1	4 ω_2	1	4 ω_3	1	16 β	1	16 γ	1
32-b	32 α	1	8 ω_2	1	8 ω_3	1	32 β	1	32 γ	1
64-b	64 α	1	16 ω_2	1	16 ω_3	1	64 β	1	64 γ	1

TABLE IV
TOTAL NUMBER OF TRANSISTORS FOR VARIOUS COMPARATOR BITWIDTHS.

Size of the Comparator	Number of Transistors					
	Group 1	Group 2	Group 3	Group 4	Group 5	Total
16-b	16 \times 6	4 \times 5	4 \times 5	16 \times 16	16 \times 10	552
32-b	32 \times 6	8 \times 5	4 \times 5	32 \times 16	32 \times 10	1084
64-b	64 \times 6	16 \times 5	4 \times 5	64 \times 16	64 \times 10	2148

$$DELAY_{CRM} = D_{group1} + D_{group2} + D_{group3} + D_{group4} + D_{group5} \quad (10).$$

All terms, except the third, on the right-hand side of (10) entail a single gate delay D_U , resulting in

$$DELAY_{CRM} = D_U + D_U + (\lceil \log_{16}(N) \rceil) D_U + D_U + D_U = 4D_U + (\lceil \log_{16}(N) \rceil) D_U \quad (11).$$

The delay $DELAY_{DM}$ for the decision module's NOR-NAND gate network is

$$DELAY_{DM} = (\log_4(N)) D_U \quad (12).$$

The total comparator delay $DELAY_T$ from input to output for an N -bit comparator is

$$D_T = 4D_U + (\lceil \log_{16}(N) \rceil) D_U + (\lceil \log_4(N) \rceil) D_U \quad (13).$$

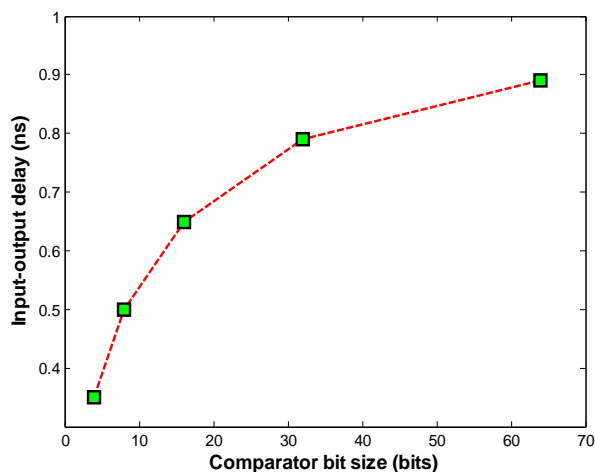


Fig. 4. Maximum input-output delay versus input bitwidth.

The maximum total input-to-output delay versus input bitwidth for the proposed comparator is shown in Fig. 4. The graph (Fig.4) shows that the delay is increased linearly with respect to the comparator bit size. This

comparator offers a 33% speed advantage over the design in [15] and 59% speed advantage over the design in [13].

V. RESULT COMPARISON

To evaluate the functionality and performance of this comparator, the complete design is simulated with various inputs using the T-Spice simulator with 0.15 μ m-TSMC digital CMOS technology. The worst case delay of this comparator is calculated by turn on the maximum number of cells, including all the lower significant cells. Here the objective is to increase the operating speed and reduce the transistor counts. The maximum measured cell delay was

0.0847 ns for the β -type cell with a maximum fan-in of five and a maximum fan-out of one. This comparator is evaluated with conventional CMOS comparator design, whose structures represent recently proposed topologies and circuits targeted for high-speed operation and power savings. Simulation results for 64-b comparator and reported results for conventional CMOS comparator design and proposed comparator design are shown in Table V. The maximum total input-to-output delay versus input bitwidth for our comparator is shown in Fig. 4. The simulation results closely match the analytical model in Table IV.

TABLE V
RESULTS COMPARISON BETWEEN EXISTING AND PROPOSED DESIGN.

Comparator Type (Technology)	Transistor Count	Maximum Input-Output Delay
Salehet <i>al.</i> [5] (150nm)	4000 (64-bit)	0.86ns
Proposed comparator (150 nm)	2210 (64-bit)	0.89ns

VI. CONCLUSION

This work proposed a high-speed, low-power and area-efficient comparator using parallel prefix structure consisting of the comparison resolution module and the decision module. These modules are designed using simple logic gates with a maximum fan-in of 5 and fan-out of 4, independent of the input bitwidth. The cells used in this comparator are designed using pass transistor logic and pseudo nMOS logic. These logic styles reduce the number of transistors required in the design. Leveraging the parallel prefix tree structure for the comparator design is novel in that this design performs the comparison operation from the MSB to the LSB, using parallel operation. In this design, all cells are locally interconnected, which avoids the need for large cell drivers, thus balancing all cells to a uniform transistor size. This comparator is designed in Tanner EDA tool using 180nm digital CMOS technology and simulated using T-Spice simulation tool.

REFERENCES

- [1] H. Chan and G. W. Roberts, "A jitter characterization system using a component-invariant Vernier delay line," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 1, pp. 79–95, Jan. 2004.
- [2] H. Suzuki, C. H. Kim, and K. Roy, "Fast tag comparator using diode partitioned domino for 64-bit microprocessor," *IEEE Trans. Circuits Syst. I*, vol. 54, no. 2, pp. 322–328, Feb. 2007.
- [3] V. Ponomarev, G. Kucuk, O. Ergin, and K. Ghose, "Energy efficient comparators for superscalar datapaths," *IEEE Trans. Comput.*, vol. 53, no. 7, pp. 892–904, Jul. 2004.
- [4] S. Abdel-Hafeez, "Single rail domino logic for four-Phase clocking scheme," U.S. Patent 6 265 899, Oct.20, 2001.
- [5] S. Abdel-Hafeez, A. Gordon-Ross and B. Parhami, "Scalable digital CMOS comparator using parallel prefix tree", *IEEE transactions on very large scale integration (VLSI) systems*, vol. 21, no. 11, Nov.2013.
- [6] S.-W. Cheng, "A high-speed magnitude comparator with small transistor count," in *Proc. IEEE Int. Conf. Electron., Circuits, Syst.*, vol. 3. Dec. 2003, pp. 1168–1171.
- [7] W. Belluomini, D. Jamsek, A. K. Nartin, C. McDowell, R. K. Montoye, H. C. Ngo, and J. Sawada, "Limited switch dynamic logic circuits for high-speed low-power circuit design," *IBM J. Res. Develop.*, vol. 50, nos. 2–3, pp. 277–286, Mar.–May 2006.
- [8] C.-C. Wang, C.-F. Wu, and K.-C. Tsai, "1 GHz 64-bit high-speed comparator using ANT dynamic logic with two-phase clocking," in *IEE Proc.-Comput. Digit. Tech.*, vol. 145, no. 6, pp. 433–436, Nov. 1998.
- [9] J. E. Stine and M. J. Schulte, "A combined two's complement and floating-point comparator," in *Proc. Int. Symp. Circuits Syst.*, vol. 1. 2005, pp. 89–92.
- [10] C.-C. Wang, P.-M. Lee, C.-F. Wu, and H.-L. Wu, "High fan-in dynamic CMOS comparators with low transistor count," *IEEE Trans. Circuits Syst. I*, vol. 50, no. 9, pp. 1216–1220, Sep. 2003.
- [11] N. Maheshwari and S. S. Sapatnekar, "Optimizing large multiphase level-clocked circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 18, no. 9, pp. 1249–1264, Nov. 1999.
- [12] C.-H. Huang and J.-S. Wang, "High-performance and power-efficient CMOS comparators," *IEEE J. Solid-State Circuits*, vol. 38, no. 2, pp.254–262, Feb. 2003.
- [13] H.-M. Lam and C.-Y. Tsui, "A mux-based high-performance single-cycle CMOS comparator," *IEEE Trans. Circuits Syst. II*, vol. 54, no. 7, pp. 591–595, Jul. 2007.
- [14] Frustaci, S. Perri, M. Lanuzza, and P. Corsonello, "Energy-efficient single-clock-cycle binary comparator," *Int. J. Circuit Theory Appl.*, vol. 40, no. 3, pp. 237–246, Mar. 2012.
- [15] S. Perri and P. Corsonello, "Fast low-cost implementation of single clock-cycle binary comparator," *IEEE Trans. Circuits Syst. II*, vol. 55, no. 12, pp. 1239–1243, Dec. 2008.
- [16] J. D. Bruguera and T. Lang, "Multilevel reverse most-significant carry computation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 6, pp. 959–962, Dec. 2001.
- [17] J.-Y. Kim and H.-J. Yoo, "Bitwise competition logic for compact digital comparator," in *Proc. IEEE Asian Solid-State Circuits Conf.*, Nov. 2007, pp. 59–62.
- [18] L. Yo-Sheng, C. Wu, C. Chang, R. Yang, W. Chen, J. Liaw, and H. Diaz, "Leakage scaling in deep submicron CMOS for SoC," *IEEE Trans. Electron. Devices*, vol. 49, no. 6, pp. 2034–1041, Jun. 2002.