International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

A Novel Approach to Design a Scalable Comparator using QCA Based Parallel Prefix Tree

[1].Yammanuru Sruthi , [2].K.Pakeerappa ,
[1]*PG Student* ; [2]*m Tech,Asst.Proffesor,* **Chiranjeevi Reddy Institute Of Engineering And Technology, Ananthapuramu**

ABSTRACT

:COMPARATORS are the key design elements for a wide range of applications like scientific computation (graphics and image/signal processing),test circuit applications (jitter measurements, signature analyzers, and built-in self test circuits) and for general-purpose processor components (associative memories, load-store queue buffers, translation look-aside buffers, branch target buffers) and many other CPU argument comparison blocks .In this project a 32 bit comparator architectures is designed by using parallel prefix structure. This project evaluates the successful results as per requirement and specifications. In existing system ,the parallel prefix structure is designed for 32 bit architectures and the reports from the Xilinx tool concludes that for every bit range doubles the delay, memory. Quantum-dot cellular automata (QCA) are an attractive emerging technology suitable for the development of ultra dense low-power high-performance digital circuits. Efficient solutions have recently been proposed for several arithmetic circuits, such as adders, multipliers, and comparators. Nevertheless, since the design of digital circuits in QCA still poses several challenges, novel implementation strategies and methodologies are highly desirable. This paper proposes a new design approach oriented to the implementation of binary comparators in QCA. But

In the proposed design of my project, each and every element in the parallel prefix structure will be

changed by using QCA Technology and the obtained results will be compared with existed design for the same device specifications. By performing this modification in the architecture will leads to reduction in Power Consumption and in DELAY parameters.

Index Terms—Binary comparators, majority gates, quantum dot cellular automata (QCA).

I. INTRODUCTION Other comparator designs improve scalability and reduce comparison delays using a hierarchical prefix tree structure composed of 2-b comparators. These structures require log2 N comparison levels, with each level consisting of several cascaded logic gates. However, the delay and area of these designs may be prohibitive for comparing wide operands. The prefix tree structure's area and power consumption can be improved by leveraging two-input multiplexers (instead of 2-b comparator cells) at each level and generate-propagate logic cells on the first level (instead of 2-b adder cells), which takes advantage of one's complement addition. Using this logic composition, a prefix tree requires six levels for the most common comparison bit width of 64 bits, but suffers from high

power consumption due to every cell in the structure being active, regardless of the input operands' values. Furthermore, the structure can perform only "greater-than" or "less-than" comparisons and not equality. To improve the speed and reduce power consumption, several designs rely on pipelining and power-down mechanisms to reduce switching activity with respect to the actual input operands' bit values.

One design uses all-N transistor (ANT) circuits to compensate for high fan-in with high pipeline throughput. A 64-b comparator requires only three pipeline cycles using a multiphase clocking scheme. However, such a clocking scheme may be unsuitable for high-speed single-cycle processors because of several heavily loaded global clock signals that have high-power transition activity. Additionally, race conditions and a heavily constrained clock jitter margin may make this design unsuitable for wide-range comparators. An alternative architecture leverages priority-encoder magnitude decision logic with two pipelined operations that are triggered at both the falling and rising clock edges to improve operating speed and eliminate long dynamic logic chains. However, 64-b and wider comparators require a multilevel cascade structure, with each logic level consisting of seven nMOS transistors connected in series that behave in saturating mode during operation. This structure leads to a large overall conductive resistance, with heavily loaded parasitic components on the clock signal, which severely limits the clock speed and jitter margin. Other architectures use a multiplexer-based structure to split a 64-b comparator into two comparator stages: the first stage consists of eight modules performing 8-b comparisons and the modules' outputs are input into a priority encoder and the second stage uses an 8-to-1 multiplexer to select the appropriate result from the eight modules in the first stage.

Similarly, other energy-efficient designs leverage schemes to reduce switching activity. Compute-on demand comparators compare two binary numbers one bit at a time, rippling from the most significant bit (MSB) to the least significant bit (LSB). The outcome of each bit comparison either enables the comparison of the next bit if the bits are equal, or represents the final comparison decision if the bits

are different. Thus, a comparison cell is activated only if all bits of greater significance are equal. Although these designs reduce switching, they suffer from long worst case comparison delays for wide worst case operands. To reduce the long delays suffered by bitwise ripple designs, an enhanced architecture incorporates an algorithm that uses no arithmetic operations. This scheme detects the larger operand by determining which operand possesses the leftmost 1 bit after pre-encoding, before supplying the operands to bitwise competition logic (BCL) structure. The BCL structure partitions the operands into 8-b blocks and the result for each block is input into a multiplexer to determine the final comparison decision. Due to this BCL-based design's low transistor count, this design has the potential for low power consumption, but the pre-encoder logic modules preceding the BCL modules limit the maximum achievable operating frequency. In addition, special control logic is needed to enable the BCL units to switch dynamically in a synchronized fashion, thus increasing the power consumption and reducing the operating frequency.

International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

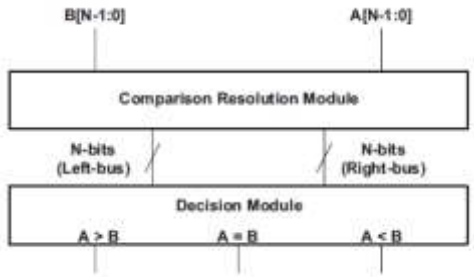


Fig.1. Block diagram of our comparator architecture, consisting of acomparison resolution module connected to a decision module.

To alleviate some of the drawbacks of previous designs (such as high power consumption, multi cycle computation, custom structures unsuitable for continued technology scaling, long time to market due to irregular VLSI structures, and irregular transistor geometry sizes), in this paper we leverage standard CMOS cells to architect fast, scalable, wide-range, and power-efficient algorithmic comparators with the following key features.

II. COMPARATOR ARCHITECTURAL OVERVIEW

The comparison resolution module in Fig. 1 (which depicts the high-level architecture of our proposed design) is a novel MSB-to-LSB parallel-prefix tree structure that performs bitwise comparison of two N-bit operands A and B, denoted as AN−1, AN−2, . . ., A0 and BN−1, BN−2, . . ., B0, where the subscripts range from N–1 for the MSB to 0 for the LSB. The comparison resolution module performs the bitwise comparison asynchronously from left to right, such that the comparison logic's computation is triggered only if all bits of greater significance are equal. The parallel structure encodes the bitwise comparison results into two N-bit buses, the left bus and the right bus, each of which store the partial comparison result as each bit position is evaluated, such that.

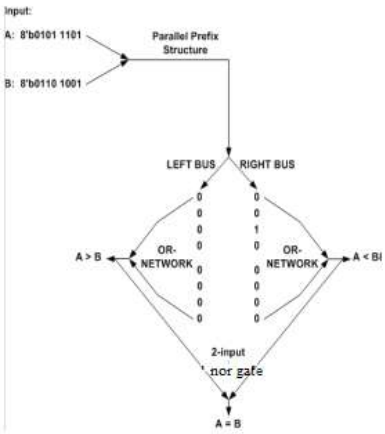


Fig. 2. Example 8-b comparison

if Ak > Bk, then leftk = 1 and rightk = 0

if Ak < Bk , then leftk = 0 and rightk = 1

if Ak = Bk , then leftk = 0 and rightk = 0.

In addition, to reduce switching activities, as soon as a bitwise comparison is not equal, the bitwise comparison of every bit of lower significance is terminated and all such positions are set to zero on both buses, thus, there is never more than one high bit on either bus. The decision module uses two OR-networks to output the final comparison decision based on separate OR-scans of all of the bits on the left bus (producing the L bit) and all of the bits on the right bus (producing the R bit). If LR = 00, then A = B, if LR = 10 then A > B, if LR = 01 then A < B, and LR = 11 is not possible. An 8-b comparison of input operands A = 01011101 and B = 01101001 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 the above example, A7 = 0 and B7

International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

= 0 encodes as left7 = right7 = 0, A6 = 1, and B6 = 1 encodes as left6 = right6 = 0, and A5 = 0 and B5 = 1 encodes left5 = 0 and right5 = 1. At this point, since the bits are unequal, the comparison terminates and a final comparison decision can be made based on the first three bits evaluated.

The parallel prefix structure forces all bits of lesser significance on each 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. We partition the structure into five hierarchical prefixing sets, as depicted in

hierarchical prefixing sets, as depicted in Fig. 3, with the associated symbol representations in Tables I and II, where each set performs a specific function whose output serves as input to the next set, until the fifth set produces the output on the left bus and the right bus.

The below symbols are usually used in implementation. Each symbol is represented by the corresponding logic gates. The symbol will perform the operation represented by the logic gate and maximum fan in and fan outs are indicated as 2/4 I.e., the maximum number of inputs are 2 and the maximum number of outputs are 4. These symbols are used to implement the several sets of operations.

Symbol (Cells)	Definition
N	Operand bitwidth
A	First input operand
B	Second input operand
R	Right bus result bit
L	Left bus result bit
$\prod$	Bitwise AND
$\sum$	Bitwise OR
T{*}	Logic function of cell type*
COMP{*}	Complement function of set*

TABLE II: SYMBOL NOTATION AND DEFINITIONS

All cells (components) within each set operate in parallel, which is a key feature to increase operating speed while minimizing the transitions to a minimal set of leftmost bits needed for a correct decision. This prefixing set structure bounds the components' fan-in and fan-out regardless of comparator bit width and eliminates heavily loaded global signals with parasitic components, thus improving the operating speed and reducing power consumption. Additionally, the OR-network's fan-in and fan-out is limited by partitioning the buses into 4-b groupings of the input operands, thus reducing the capacitive load of each bus.

2.3 COMPARATOR DESIGN DETAILS:

We partition the structure into five hierarchical prefixing sets, as depicted in Fig.2 with the associated symbol representations in Tables I , where as each set performs a exact function whose output serves as input to the next set, in hope of the fifth set produces the output on the left bus and the right bus Every part of cells components within each set operate in parallel were as it's a key feature to increase

International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

operating speed while minimizing the transitions to a minimal set of left most bits needed for a correct decision. This prefixing set structure bounds the components fan-in and fan-out regardless of comparator bit-width and eliminates heavily loaded global signals with parasitic components, thus improving the operating speed and reducing power consumption.

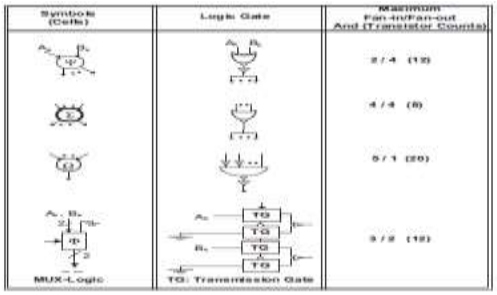


TABLE III: LOGIC GATE REPRESENTATIONS FOR SYMBOLS

In this section, we detail our comparator's design Figure 2, which is based on using a novel parallel prefix tree Tables I and II contain symbols and definitions. Each set or groups of cells that produce output and serve as inputs to the next set in the hierarchy, with the exception of set 1, the outputs serve as inputs to several sets. Set 1 compares the N -bit operands A and B bit-by-bit, using a single level of N Ψ Type cell. The Ψ type cells provide a Termination flag D*k* to cells in sets 2 and 4, indicating whether the computation should terminate. These cells compute (where $0 \le k \le N-1$)

Ψ:Dk = Ak ⊕ Bk.

Set 2 consists of Σ_2 -type cells, which combine the termination flags for each of the four $\Sigma_2\Psi$ -type cells from set1 (each $\geq_2$ -type cell combines the termination flags of one4-b partition) using NOR-logic to limit the fan-in and fan-out to a maximum of four. The -type cells either continue the comparison for bits of lesser significance if all four inputs are 0s, or terminate the comparison if a final decision can be made.

Set 3 consists of Σ_3 -type cells, which are similar to $\geq_2$-type cells, but can have more logic levels, different inputs, and carry different triggering points. A Σ_3 -type cell provides no comparison functionality; the cell's sole purpose is to limit the fan-in and fan-out regardless of operand bitwidth. To limit the Σ_3 -type cell's local interconnect to four, the number of levels in set 3 increases if the fan-in exceeds four. Set 3 provides functionality similar to set 2 using the same NOR logic to continue or terminate the bitwise comparison activity. If the comparison is terminated, set 3 signals set 4 to set the left bus and right bus bits to 0 for all bits of lower significance.

From left to right, the first four Σ_3 -type cells in set 3 combine the 4-b partition comparison outcomes from the one, two, three, and four 4-b partitions of set 2. Σ_3 -type cell has a fan-in of four, the number of levels in set 3 increases and set 3's fifth Σ_3 -type cell combines the comparison outcomes of the first 16 MSBs with a fan-in \

International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

9R-Type Cell	Input Driving 9R-Type Cell Output
Y_{15}	D_{15}
Y_{14}	$\overline{D}_{15}D_{14}$
Y_{13}	$\overline{D}_{15}\overline{D}_{14}D_{13}$
Y_{12}	$\overline{D}_{15}\overline{D}_{14}\overline{D}_{13}D_{12}$
Y_{11}	$C_{3,0}D_{11}$
Y_{10}	$C_{3,0}D_{11}D_{10}$
Y_9	$C_{3,0}D_{11}D_{10}D_9$
Y_8	$C_{3,0}D_{11}D_{10}D_9D_8$
Y_7	$C_{3,1}D_7$
Y_6	$C_{3,1}D_7D_6$
Y_5	$C_{3,1}D_7D_6D_5$
Y_4	$C_{3,1}D_7D_6D_5D_4$
Y_3	$C_{3,2}D_3$
Y_2	$C_{3,2}D_3D_2$
Y_1	$C_{3,2}D_3D_2D_1$
Y_0	$C_{3,2}D_3D_2D_1D_0$

TABLE IV:OUTCOME OF Ω -TYPE CELLS IN SET 4 FOR A 16-B COMPARISON

Set 4 consists of Ω type cells, whose outputs control the select inputs of Φ-type cells (two-input multiplexors) in set 5,which in turn drive both the left bus and the right bus.

$$\Omega : Y_k = C_{3,\lfloor k/4 \rfloor - 1} D_k \prod_{i=4\lfloor K/4 \rfloor - 1}^{k-1} \overline{D}_i.$$

2.4 BASIC ARCHITECTURE OF 16 BIT COMPARATOR USING PARALLEL PREFIX TREE

For an Ω type cell and the 4-b partition to which the cell belongs, bitwise comparison outcomes from set 1 provide information about the more significant bits in the cell's Ω type cells, Set 5 consists of N Φ -type cells (two-input, 2-b-wide multiplexers). One input is (AK, Bk) and the other is hardwired to "00." The select control input is based on the Ω type cell output from set 4. We define the 2-b as the left-bit code (AK) and the right-bit code (Bk), where all left-bit codes and all right-bit codes combine to form the left bus and the right bus, respectively. The Φ-type cells compute (where $0 \le k \le N-1$).

$$\Phi : F_k^{1,0} = Y_k \times M_k + \overline{Y}_k \times (00).$$

The output $F_k^{1,0}$ denotes the "greater-than," "less-than," or "equal to" final comparison decision.

$$F_k^{1,0} \begin{cases} 00, & \text{for } A_k = B_k \\ 01, & \text{for } A_k < B_k \\ 10, & \text{for } A_k > B_k. \end{cases}$$

Essentially, the 2-b code $F_k^{1,0}$ can be realized by OR-ing all left bits and all right bits separately, as shown in the decision module (Figs. 2 and 3), using an OR-gate network in the form of NOR-NAND gates yielding a more optimum gate structure

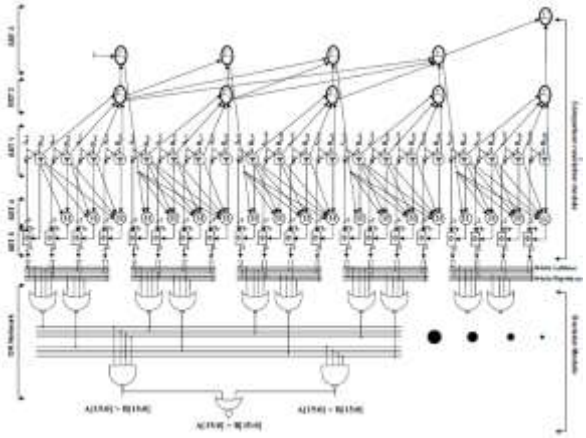


Fig 2.2. Implementation details for the comparison resolution module (sets 1 through 5) and the decision module.

We define the 2-b as the left-bit code (Ak) and the right-bit code (Bk), where all left-bit codes and all right-bit codes combine to form the left

International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

bus and the right bus, respectively. The Φ-type cells compute (where $0 \leq k \leq N-1$). From left to right, the first four ∑3-type cells in set 3 combine the 4-b partition comparison outcomes from the one, two, three, and four 4-b partitions of set 2. Since the fourth ∑3-type cell has a fan-in of four, the number of levels in set 3 increases and set 3's fifth ∑3-type cell combines the comparison outcomes of the first 16 MSBs with a fan-in of only two and a fan-out of one.

In comparison resolution module four sets are used and each set performs different gate operations. In set1 the XOR operation is performed with A and B inputs and the output of the gates is D which is 32 down to 0. Set2 perform the NOR operation. The set1 output is given as input for set2 and each gate has 4 inputs with one output. Set3 is similar to set2 (XOR operation). The inverted inputs are applied to NAND gate and its output is also inverted. In decision module set5 performs the multiplexer operation.

3.PROPOSED METHOD:

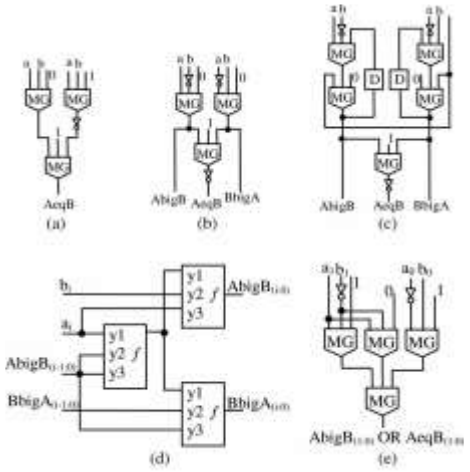
3.1 Quantum-Dot Cellular Automata (Qca)

. It takes great advantage of a physical effect: the Coulomb force that interacts between electrons. There also exists an alternative implementation that uses magnetic fields, but

this practical course will not cover magnetic QCA for now. Though it is still difficult to produce and operate with these devices under typical temperature conditions, simulations predict promising numbers, like theoretical clock rates of several THz.

3.2 QCA BASED COMPARATOR

There are several QCA designs of comparators in the literature. A 1-bit binary comparator receives two bits *a* and *b* as inputs and establishes whether they are equal, less than or greater than each other. These possible states are represented through three output signals, here named A_{eq} *B*, A_{big} *B*, B_{big} *A*, that are asserted, respectively, when $a = b$, $a > b$, and $a < b$. Full comparators are those that can separately identify all the above cases, whereas non-full comparators recognize just one or two of them. As an example, the comparator designed in and depicted in Fig. 5(a) can verify only whether $a = b$. Conversely, the circuits shown in Fig. 5.1(b) and (c), proposed , are full comparators. The latter also exploits two 1-bit registers *D* to process *n*-bit operands serially from the least significant bit to the most significant one.



International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015

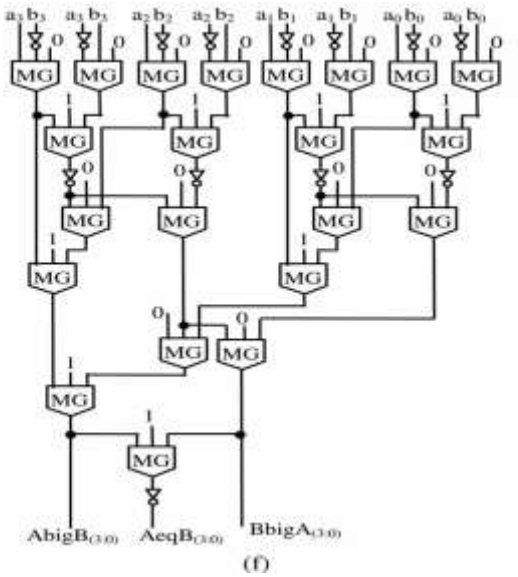


Fig 3.1 QCA based comparator presented in:(a),(b),(c),(d), (e), (f) .

The design described in exploits a tree-based (TB) ar-chitecture and exhibits a delay that in theory logarithmically increases with *n*. The 2-bit version of such designed compara-tor is illustrated in Fig. 5.1(e) Also the full comparator proposed in exploits a TB ar-chitecture to achieve high speed. As shown in Fig. 5.1(f), where 4-bit operands are assumed, one instance of the 1-bit comparator presented it is used for each bit position. The intermediate results obtained in this way are then further processed through a proper number of cascaded 2-input OR and AND gates imple-mented by means of MGs having one input permanently set to 1 and 0, respectively. Analyzing existing QCA implementations of binary compara-tors it can be observed that they were designed directly mapping the basic Boolean functions consolidated for the CMOS logic designs to MGs and inverters, or ULGs. Unfortunately, in this way the computational capability offered by each MG could be underutilized . As a consequence, both the complexity and the overall delay of the resulting QCA designs could be increased in vain.

3.3 NOVEL QCA COMPARATORS

The first proposed comparator exploits a cascade-based (CB) architecture. To explain better how the overall computation is performed, the schematic diagram illustrated in Fig. 3 is provided. It shows a possible implementation of a 32-bit comparator based on the proposed theory. Following the criterion illustrated in Fig. 3, an *n*-bit CB full comparator designed as proposed here uses: *n*/3 instances of T1 and/or T2; *n*/3 cascaded instances of T4 through which the signals *A*big*B*(*n*−1:0) and *B*big*A*(*n*−1:0) are computed; and one instance of C2, needed to compute also *A*eq*B*(*n*−1:0). Circles visible in Fig. 5.2 indicate the additional clock phases that have to be inserted on wires to guarantee the correct synchronization of the overall design. The CB full comparator was designed for operands word lengths ranging from 2 to 32 and using, for n > 2, the split criterion summarized in Table I. Obviously, alternative splits could be used.

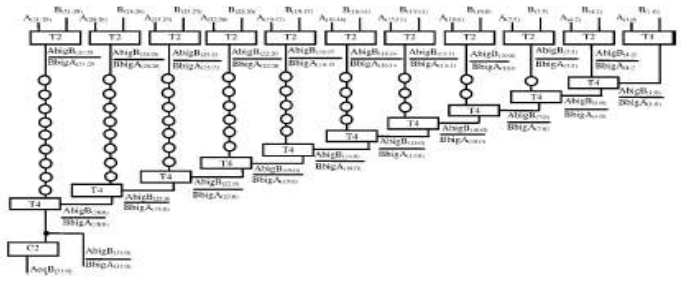
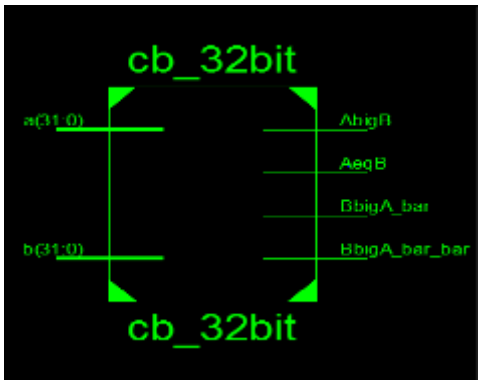


Fig 5.2 Novel 32-bit CB full comparator

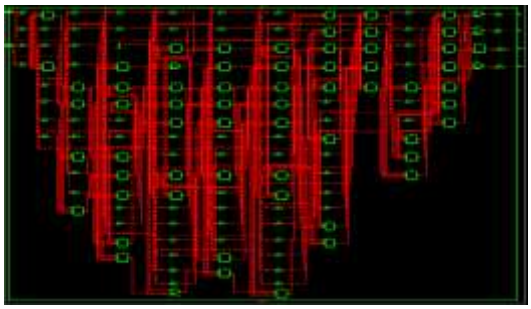
4 IMPLEMENTATION AND RESULTS

4.1 RTL SCHEMATIC:

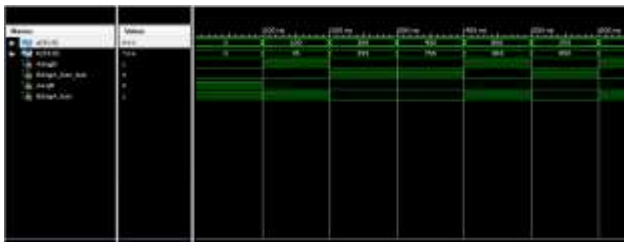
International Journal of Research
Available at http://internationaljournalofresearch.org/
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 01 Issue 09
October 2015



4.2 Technology schematic:



4.3 Waveform:



4.4 Comparision table:

	No of 4 input LUT'S Used	Delay(ns)	Power(mw)
EXISTING	118	14.383ns	0.9625mw
PROPOSED	61	16.091ns	0.4978mw

5.CONCLUSION

A new methodology useful to design binary comparators in QCA has been presented. It is based on innovative formulations that allow increased speed performances and reduced overall sizes to be achieved with respect to the existing competitors. The novel comparators split the received *n*-bit inputs into a proper number of 2- and 3-bit sub words that are processed in parallel through 2- and 3-bit comparators designed by applying theorems demonstrated here.Thanks to the basic logic and layout strategies adopted, a 32-bit CB full comparator designed as described in this paper exhibits a delay of only 3 + (3⁄4) clock cycles, occupies an active area of 2.66 μm^2 , and achieves an area-delay product less than 10. When the alternative TB architecture presented here is ex-ploited, the delay is further reduced to 2 + (3⁄4) clock cycles; the active area is ~2.9 μm^2 , whereas the area-delay product is less than 8.

6.REFERENCES

[1] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernestein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, 1993.

[2] M. T. Niemer and P. M. Kogge, "Problems in designing with QCAs: Layout = timing," *Int. J. Circuit Theory Appl.*, vol. 29, pp. 49–62, 2001.

[3] G. H. Bernstein,A. Imre,V.Metlushko,A.Orlov, L. Zhou, L. Ji,G. Csaba, and W. Porod, "Magnetic QCA systems," *Microelectron. J.*, vol. 36, pp. 619–624, 2005.

[4] J.Huang and F. Lombardi, *Design and Test of DigitalCircuits by Quantum-Dot Cellular Automata*. Norwood, MA, USA: Artech House, 2007.

[5] W. Liu, L. Lu, M. O'Neill, and E. E. Swartzlander Jr., "Design rules for quantum-dot cellular automata," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Rio De Janeiro, Brazil, May 2011, pp. 2361–2364.

[6] K. Kim, K. Wu, and R. Karri, "Towards designing robust QCA architectures in the presence of sneak noise paths," in *Proc. IEEE Design, Automation Test Eur. Conf. Exhib. (DATE)*,Munich, Germany,Mar. 2005, pp. 1214–1219.
[7] K. Navi, M. H. Moaiyeri, R. F. Mirzaee, O. Hashemipour, and B. M. Nezhad, "Two new low-power full adders based on majority-not gates," *Microelectron. J.*, vol. 40, pp. 126–130, 2009.
[8] H. Cho and E. E. Swartzlander Jr., "Adder design and analyses for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 374–383, May 2007.
[9] H. Cho and E. E. Swartzlander Jr., "Adder and multiplier design in quantum-dot cellular automata," *IEEE Trans. Comput.*, vol. 58, no. 6, pp. 721–727, Apr. 2009.
[10] V. Pudi and K. Sridharan, "Efficient design of a hybrid adder in quantumdot cellular automata," *IEEE Trans. VLSI Syst.*, vol. 19, no. 9, pp. 1535–1548, Jul. 2011.
[11] M. Gladshtein, "Quantum-dot cellular automata serial decimal adder," *IEEE Trans. Nanotechnol.*, vol. 10, no. 6, pp. 1377–1382, Nov. 2011.
[12] V. Pudi and K. Sridharan, "Low complexity design of ripple carry and Brent-Kung adders in QCA," *IEEE Trans. Nanotechnol.*, vol. 11, no. 1, pp. 105–119, Jan. 2012.
[13] S. Perri and P. Corsonello, "New methodology for the design of efficient binary circuits addition inQCA," *IEEE Trans.Nanotechnol.*, vol. 11, no. 6, pp. 1192–1200, Nov. 2012.
[14] V. Pudi and K. Sridharan, "New decomposition theorems on majority logic for low-delay adder designs in quantum dot cellular automata," *IEEE Trans. Circuits Syst. II: Exp. Brief*, vol. 59, no. 10, pp. 678–682, Oct. 2012.
[15] H. Cho and E. E. Swartzlander Jr., "Serial parallel multiplier design in quantum-dot cellular automata," in *Proc. IEEE Symp. Comput. Arithmetic*, 2007, pp. 7–15.
[16] S. W. Kim and E. E. Swartzlander Jr., "Parallel multipliers for quantumdot cellular automata," in *Proc. IEEE Nanotechnol. Mater. Devices Conf.*, 2009, pp. 68–72.
[17] S. W. Kim and E. E. Swartzlander Jr., "Multiplierswith coplanar crossings for quantum-dot cellular automata," in *Proc. IEEE Int. Conf. Nanotechnol.*, 2010, pp. 953–957.
[18] W. Liu, L. Lu, M. O'Neill, and E. E. Swartzlander Jr., "Montgomery modular multiplier design in quantum-dot cellular automata using cut-set retiming," in *Proc. IEEE Int. Conf. Nanotechnol.*, 2010, pp. 205–210.
[19] L. Lu, W. Liu, M. O'Neill, and E. E. Swartzlander Jr., "QCA systolic matrix multiplier," in *Proc. IEEE Annu. Symp. VLSI*, 2010, pp. 149–154.
[20] J. D.Wood andD. Tougaw, "Matrix multiplication using quantum-dot cellular automata to implement conventional microelectronics," *IEEE Trans. Nanotechnol.*, vol. 10, no. 5, pp. 1036–1042, Sep. 2011.
[21] L. Lu, W. Liu, M. O'Neill, and E. E. Swartzlander Jr., "QCA systolic array design," *IEEE Trans. Comput.*, vol. 62, no. 3, pp. 548–560, Mar. 2013.