ISSN: 2278-4632
Vol-12 Issue-06 No.03 June 2022

DESIGN OF HIGH SPEED VLSI ARCHITECTURE FOR BCD ADDER USING QCA MAJORITY GATES

Goddu Vamsinaidu Research Scholar, Sri Sivani College of Engineering, Chilakapalem, Andhra Pradesh 532402
Dr. V.Suryanarayana Associate Professor, Sri Sivani College of Engineering, Chilakapalem, Andhra Pradesh 532402

ABSTRACT

Majority-of-three (MAJ) gates may be realized natively by a variety of new ways. Although numerous MAJ gates can utilize upgraded gates There is Directly derived from the physical qualities of new approaches, such as exclusive-OR (XOR) gates, is a very efficient XOR architecture. We covered three-input exclusive-OR multi-digit binary coded decimal (BCD) adder concepts (XOR3),square root carry choose binary adder, and MAJ gates in this short. BCD adders are used in financial, commercial, and industrial computers. The designs were implemented using QCA technology. The suggested logic representations achieve high speed, and the proposed architecture was generated and simulated using Verilog HDL in Xilinx ise.

Index terms: QCA technology, majority gate, 3input xor gate and BCD architecture.

I. INTRODUCTION

High-performance digital circuits might be created in the future using QUANTUM-DOT cellular automata (QCA) technology. The sub modules for arithmetic are among the most studied architectures among the logic circuits lately introduced. Non-elementary digital modules are cleverly built by mixing In the QCA technology, only inverters and majority gates are biologically accessible (MGs). Despite logic function. Applications in business, finance, and the Internet need to be more accurate. decimal arithmetic has gotten a lot of attention lately. The mistakes resulting

from the translation In these circumstances, the distinction between binary and decimal number representations could no longer be maintained, hence some contemporary microprocessors have hardware decimal arithmetic units in their cores, which are compliant with the IEEE 754–2008 standard. To optimize performance and area behaviour, such digital circuits need correct techniques at both the logic and layout levels.

This brief presents a unique method for designing QCA-based n-digit (with n 1) BCD adders that can outperform current equivalents in terms of computational performance without losing occupied space or cell count. Figure 1 shows some of the benefits. The BCD adders' structure is shown in. accomplished by combining a novel logic technique with specially built QCA modules.

Optimisation of the latter takes into account the fact When a carry is formed by summing the operands' The most important sum digit is found by starting with the least important digits, moving them to the next n 2-digit locations, and then absorbing them at the end.

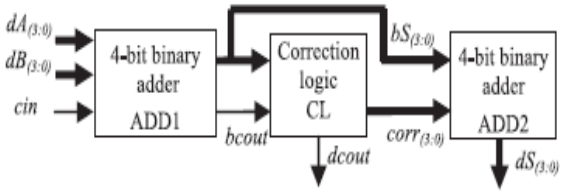

Fig. 1. Structure of the BCD adders.

The innovative a carry signal is generated, propagated, and absorbed by a one-digit decimal adder and power consumption than its quicker equivalents reported in. Unlike any earlier work, we expanded our efforts to include the creation of 2D QCA-BCD adder. The 2-digit sum computation takes 18 clock phases, and the circuit measures 2.74 m2.

VLSI technology has come a long way. These advancements, however, may slow down in the future. The connection issue and power dissipation are two of the most significant technical constraints for this slowness.

As more devices are crammed into a smaller space, it becomes harder to get rid of the heat made during a switching cycle. This wears down the chip. Wire resistance and capacitance limit the growth of interconnections as gadgets do. This causes a bottleneck in the wiring. QCA is a good replacement for silicon technology. QCA-based circuits are good because they are fast, reliable, and use little power. QCA circuits also benefit from high parallel processing. The universal gates, XOR and XNOR

In digital logic, they may be used in a variety of ways. Circuits such as parity generators and checkers may benefit from their usage in the design and development process, as well as circuits for error detection and repair. Not only that, but since these two gates are capable of realizing all circuits, they have long had a particular place in the hearts of physicists.

Until now, whenever these gates have been built using QCA, they have been constructed first using Some of the most often used digital logic gates are the AND, OR, and NOT gates and then using QCA. In this study, we suggest using the unique properties of QCA gates to create XOR and XNOR gates. It can be shown that when digital logic gates are employed, the number of gates needed is smaller.

II. LITERATURE SURVEY

"Cost-efficient decimal adder design in quantum-dot cellular automata," by W. Liu, L. Lu, E. E. Swartzlander, and M. ONeill. Integrated Circuit (IC) Technology is evolving at a rapid pace to improve circuit performance and density small systems. For the last four decades, conventional CMOS technology has played a critical role in digital processing. However, scaling CMOS devices has posed some difficulties in recent years. QCA is a unique nano electronic technology that has been recognized as a prospective nano electronic technology for overcoming the constraints of the transistor-based CMOS technology. This has led to its identification as a promising nano electronic technology. An effective architecture for a BCD adder in QCA technology is shown here. This architecture makes use of an area-optimized QCA full adder to provide a new concept of IC design in a way that is both effective and optimal. The suggested BCD adder circuit optimizes cost function, layout space, and QCA cells, which results in an improvement over past designs of BCD adder circuits. By C. S. Lent and P. D. Tougaw: device architecture for computing with quantum dots." Quantum-dot cellular automata are a computer paradigm based on interacting quantum dots (QCA). We demonstrate how quantum-dot cell arrays may be utilized to conduct important calculations. A novel adiabatic switching paradigm is created that allows for timed control, removes metastability issues, and allows for a pipelined design.

III. HIGH SPEED VLSI ARCHITECTURE FOR BCD ADDER USING QCA MAJORITY GATES

New computer paradigms are being used to drive forward computational efficiency as CMOS technology hits its physical limit. New approaches occur as a result of either improving the operation of existing devices or creating whole new devices employing new materials Quantum-dot cellular automata (QCA), nano magnet logic, spin-wave devices, and two-dimensional polarity-controllable transistors are some of the things that are being worked on the outcome being carefully researched.

These developing approaches, unlike traditional CMOS Boolean logic, may intrinsically implement majority-of-three (MAJ) basic gates. M(x, y, z) Only if two of the three arguments are true does this method return true. Disjunctive and conjunctive normal forms may both be used to express it:

$$M(x, y, z) = xy + xz + yz = (x + y)(x + z)(y + z). \quad (1)$$

Because xy = M(x, y, 0) and x + y = M(x, y, 0), (x, y, 1). Developing approaches can be used to show compact exclusive-OR (XOR) and exclusive-NOR (XNOR) gates, as well as a MAJ gate. It's important to remember that XOR/XNOR gates in CMOS need complicated designs or a lot of gates. These gates, on the other hand, may be built with fewer components using new approaches.

ISSN: 2278-4632
Vol-12 Issue-06 No.03 June 2022

According to recent research, QCA designs with upgraded gates (XOR/XNOR) outperform designs with just native MAJ gates. As a result, it's critical to take advantage of MAJ and XOR/XNOR gate-based arithmetic circuit designs.

Decimal arithmetic relies on the binary coded decimal (BCD) adder. Majority logic-based BCD adders have been the subject of a slew of investigations. Computational architectures and logical representations based on Boolean function adders are common. RCAs are simple but inefficient carry-propagation structures for multi-digit BCD adders. As a possible solution to this issue, carry-flow and carry-look ahead computing architectures are discussed here.

The physical layout's design complexity is proportional to the logic representations of BCD adders. Rewriting Boolean functions, for example, to fully use the MAJ and reduce the use of constant inputs, may result in considerable area/delay gains. The literature, on the other hand, has paid little attention to BCD adder designs with increased gates. According to this research, The main contribution is a hybrid logic representation for an n-digit (n-1) BCD adder that uses XOR3 and MAJ basic gates for three inputs.

Because X(x, y, 0) = xy and X(x, y, 1) = x y, the three-input XOR3 function X(x, y,z) = x yz can be turned into a two-input XOR/XNOR by making any one input constant. X(x, y,z) = (x y) z = (x y) As a result, the suggested designs may be advantageous to technologies that favour MAJ and XOR3 (XOR/XNOR) primitives.

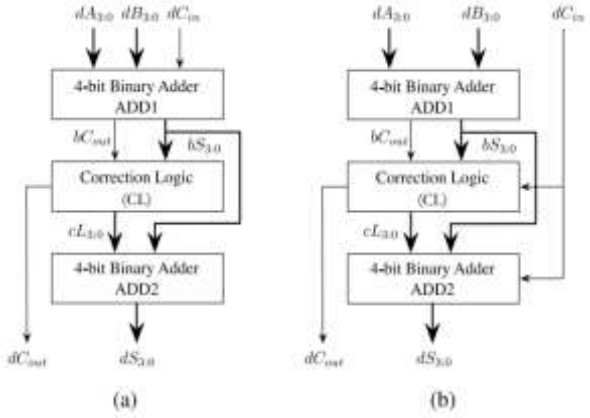

Fig. 2. BCD adder block diagrams (a) conventional design and (b) suggested design.

The ADDER BCD ADDER Figure 2(a) shows Two 4-bit binary adders (ADD1 and ADD2) plus a correction logic (CL) circuit make up the RCA-BCD adder. The ADD1 adds dA3:0, dB3:0, and d Cin to make bS3:0, the binary sum, and b Cout, the output carry. If bS3:0 = 10, the CL circuit is used to turn the BCD sum output into a positive number. It makes the cL3:0 and d Cout carry signals for the decimal output.

If d Cout = 1, the cL3:0 is (0110)2, else it is (0000)2. To get the decimal total dS3:0, the ADD2 adds bS3:0 and cL3:0. A carry-look ahead structure is presented in for computing the output carries, as illustrated in Fig. 1(b), and it was used in multi-digit BCD adder designs with parallel implementation In addition, a new structure was added to the CL circuit, with d Cin functioning as an extra input. As a consequence, the d Cout latency for multi-digit PBA-BCD adders has been greatly decreased.

In this project, the XOR-Majority Graph (XMG) logic model, which was created in, is explained first. Then, the designs of 1-digit and multi-digit BCD adders based on RCA and PBA are shown one by one. XMGs are an XML document type. The primitives of the XMG are the MAJ, the XOR with two inputs, and the inverter. In XMGs, instead of two-input XORs, we employ XOR3s in our designs. To distinguish the XM

Gs, The three MAJ and XOR inputs might be recast as a new extended XMG labelled XMG. As an example, changing (x-y-z) z to (x-y-z) would reduce XMG's number of XOR nodes from 2 to 1.Because inverters are expensive in developing approaches, the introduction of XOR3 opens up new possibilities for inversion optimization Because two-input XOR expressions, as demonstrated in (2), can only realise one inverter, two-input XOR expressions can only realise one inverter.

$\bar{x} \oplus \bar{y} = x \oplus \bar{y}$.

ISSN: 2278-4632
Vol-12 Issue-06 No.03 June 2022

$$\overline{x \oplus y} = x \oplus \bar{y} = \bar{x} \oplus y$$
$$= 0 \oplus \bar{x} \oplus y = 1 \oplus x \oplus y \quad (2)$$

By rewriting the equation as an XOR3, the inversion is reduced to constant input, resulting in a low-cost implementation. RCA-to-BCD Converter Adder S and cout are defined as follows for the 1-bit full adder with a, b, and cin binary inputs.

$$s = a \oplus b \oplus c_{in} = \mathbb{X}(a, b, c_{in}),\ c_{out} = \mathbb{M}(a, b, c_{in}) \quad (3)$$

To describe a 1-bit complete adder, we just need two operations: one MAJ and one XOR3. For logic representation. An example of an ADD1 circuit that uses four MAJ and four XOR3 operations may be seen in the (4).

$$G_0 = \mathbb{X}(dA_0, dB_0, dC_{in}) \quad H_0 = \mathbb{M}(dA_0, dB_0, dC_{in})$$
$$G_1 = \mathbb{X}(dA_1, dB_1, H_0) \quad H_1 = \mathbb{M}(dA_1, dB_1, H_0)$$
$$G_2 = \mathbb{X}(dA_2, dB_2, H_1) \quad H_2 = \mathbb{M}(dA_2, dB_2, H_1)$$
$$G_3 = \mathbb{X}(dA_3, dB_3, H_2) \quad bC_{out} = \mathbb{M}(dA_3, dB_3, H_2) \quad (4)$$

The suggested decimal carry out d Cout is written as

$$dC_{out} = \mathbb{M}(bC_{out}, G_3 + bC_{out}, \mathbb{M}(G_1, G_2, G_3)). \quad (5)$$

We discovered the identification via expansion and simulation (M(a, b, c), c + d, d) = M(a + b, c + d, d) is valid, As a result, we may rewrite (5) as

$$dC_{out} = \mathbb{M}(bC_{out}, G_3 + bC_{out}, G_1 + G_2). \quad (6)$$

Because the ADD2 section adds bS3:0 and cL3:0 to get the decimal total dS3:0, we may represent it in bits as G3G2G1G0 + 0dCoutdCout0 = dS3dS2dS1dS0, which necessitates another 4-bit adder. The following are the expressions:

$$dS_0 = G_0$$
$$dS_1 = \mathbb{X}(G_1, dC_{out}, 0)$$
$$dS_2 = \mathbb{X}(G_2, \mathbb{M}(dC_{out}, \bar{G}_1, 0), 0)$$
$$dS_3 = \mathbb{X}(G_3, \mathbb{M}(dC_{out}, \mathbb{M}(G_1, G_2, 1), 0), 0) \quad (7)$$

Note that in dS3, M(G1, G2, 1) = G1 + G2 appears in (6) but not in (5), indicating that we may reuse this signal to save area.

PBA-BCD Adder

Although the suggested BCD adder improves the area, there is no substantial improvement in the delay. The creation of the carry signals is critical for a multi-digit BCD adder design with great performance The RCA-BCD adder's d Cout has a long critical path, and the d Cin from the previous digit's d Cout controls the delay for the next digit. In the CL section, a MAJ-based circuit is utilised to calculate d Cout by taking into account the input d Cin. The PBA-BCD adder is supported by this design type. Furthermore, the d C[3:0] signal was derived by subtracting d Cout from d Cout.

$$dC_{[3:0]} = [0, dC_{out}, dC_{out}, 0]. \quad (8)$$

We use the same CL to create the BCD adder structure in Fig. 1(b). We used XMG representations to redesign the remaining two 4-bit binary adders, ADD1 and ADD2. The logic formulas for the ADD1 section are presented in (9), with d Cin removed.

$$G_0 = \mathbb{X}(dA_0, dB_0, 0) \quad H_0 = \mathbb{M}(dA_0, dB_0, 0)$$
$$G_1 = \mathbb{X}(dA_1, dB_1, H_0) \quad H_1 = \mathbb{M}(dA_1, dB_1, H_0)$$
$$G_2 = \mathbb{X}(dA_2, dB_2, H_1) \quad H_2 = \mathbb{M}(dA_2, dB_2, H_1)$$
$$G_3 = \mathbb{X}(dA_3, dB_3, H_2) \quad bC_{out} = \mathbb{M}(dA_3, dB_3, H_2) \quad (9)$$

The mechanics of producing the dCout signal for the CL circuit may be found in. The phrases are rewritten as follows:

$$K_1 = \mathbb{M}(bC_{out}, \mathbb{M}(1, bC_{out}, G_3), \mathbb{M}(G_1, G_2, G_3))$$
$$K_2 = \mathbb{M}(1, K_1, \mathbb{M}(0, G_0, G_3))$$
$$dC_{out} = \mathbb{M}(dC_{in}, K_1, K_2) \quad (10)$$

The d Cin signal is incorporated in the logic formulas for the ADD2 section as:

ISSN: 2278-4632
Vol-12 Issue-06 No.03 June 2022

$$dS_0 = \mathbb{X}(G_0, dC_{in}, 0)$$
$$dS_1 = \mathbb{X}(G_1, dC_{out}, \mathbb{M}(0, G_0, dC_{in}))$$
$$dS_2 = \mathbb{X}(G_2, dC_{out}, \mathbb{M}(G_1, dC_{out}, \mathbb{M}(0, G_0, dC_{in})))$$
$$dS_3 = \mathbb{X}(G_3, \mathbb{M}(G_2, dC_{out}, \mathbb{M}(G_1, dC_{out}, \mathbb{M}(0, G_0, dC_{in}))), 0). \quad (11)$$

PROPOSED BINARY ADDER

The ripple carry adder generates the carry out bit by rippling the incoming carry created in the preceding step after receiving the carry-in (Cin) bit. Since a result, the pace of RCA is reduced, as each stage's Cout is reliant on the previous stage's Cout or the current stage's Cin. The linear dependence issue of Cout on Cin is solved by assuming that Cin has just two potential values: '0' and '1'. The partial Sum (S) and Cout are created in parallel utilising these potential Cin values.

The final total and carry are then chosen using a multiplexer depending on the actual carry input received. Although this feature helps to reduce calculation time, the usage of a redundant adder circuit reduces the area efficiency. Carry Select Adder is made up of two rca blocks. In this research, we suggested a square root carry save adder to lower the optimum latency.

Square Root Carry Select Adder

The block size in Square Root Carry Select Adder (SRCSA) may be changed. For the sake of simplicity, the whole analysis is omitted here, however a 16-bit adder may be made utilising block sizes of 2-2-3-4-5 rather than the uniform block size of four (as done before). This break-up is great when the Full-Adder delay matches the MUX delay. The suggested SRCSA adder for 16 bits is shown in Fig. 3, where the inputs are A and B, the carry-in is represented as Cin, and the outputs are marked by sum (S) and carry-out (C) (Cout).

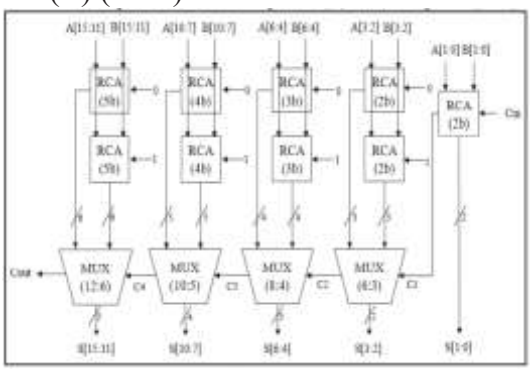

Fig. 3. 16-bit SRCSA

IV.RESULTS

RTL SCHEMATIC:- The register transfer level (RTL) schematic denotes the architecture's blueprint and is used to compare perfect architecture that we must create from the intended architecture. The hdl language is used to transform the architecture's description or summary into the functioning summary using a coding language like verilog or vhdl. The internal connection blocks are even specified in the RTL schematic for easier analysis. Below is a schematic representation of the design's RTL implementation.

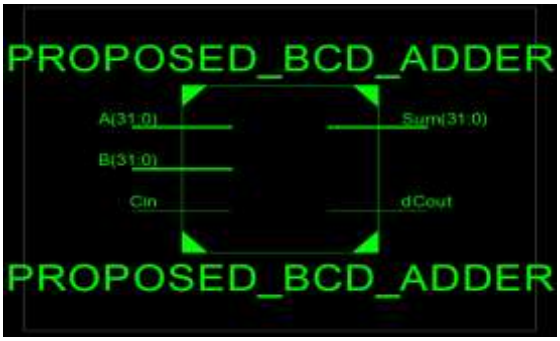

Fig4: RTL Schematic of Proposed BCD adder

ISSN: 2278-4632
Vol-12 Issue-06 No.03 June 2022

TECHNOLOGY SCHEMATIC With the LUT area parameter being used to estimate architecture design in VLSI, this diagram shows the technology's architecture in LUT format. The FPGA's LUTs, which are square units, represent the code's memory allocation.

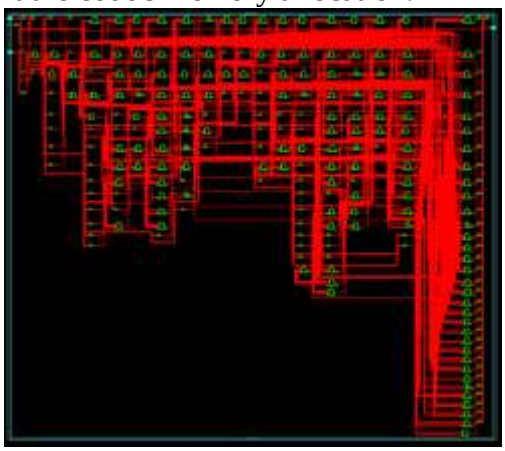

Fig5: View Technology Schematic of proposed BCD adder

SIMULATION:-

Unlike the schematic, which only verifies the connections and blocks of a circuit, a simulation verifies the circuit's workings. Waveforms are the only form of output that can be viewed in the simulation window because it is launched by shifting from implementation to simulation. Since it can support multiple radix number systems, this is a useful feature.

Fig6 :Simulated Waveforms of proposed BCD Adder

PARAMETERS:-

Consider that in VLSI, the factors considered are area, delay, and power; using these metrics It's possible to make comparisons between several designs. In this paper speed is considered here. XILINX 14.7 is used to acquire the parameter, while verilog is used as the HDL language.

Parameter	Existed BCD Adder	Proposed BCD Adder
Delay (ns)	39.669	25.380

Table1 : parameter comparison

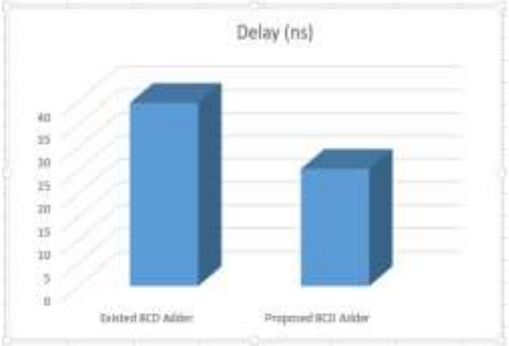

Fig7: delay comparison bargraph

V. CONCLUSION

In upcoming computer nanotechnologies, majority logic is commonly employed. The implementation of arithmetic functions using majority logic has received a lot of attention. In this short, One and multi-digit BCD adder expressions are created using better logic representations of three-input XOR and majority operations in QCA Designer and we have gotten good delay results. When compared to the greatest designs currently available, our proposed BCD adder has less delay product improvement for the 32-digit scenario.

REFERENCES

[1] W. Liu, L. Lu, E. E. Swartzlander, and M. OŠNeill, "Cost-efficient decimal adder design in quantum-dot cellular automata," in Proc. IEEE Int. Symp. Circuits Syst., Seoul, South Korea, 2012, pp. 1347–1350.
[2] M. F. Cowlishaw, "Decimal floating-point: Algorism for computers," in Proc. IEEE Symp. Comput. Arithmetic, Santiago de Compostela, Spain, 2003, pp. 104–111.
[3] IEEE Standard for Floating-Point Arithmetic, IEEE Standard 754-2008, pp. 1–70, 2008.
[4] C. S. Lent and P. D. Tougaw, "A device architecture for computing with quantum dots," Proc. IEEE, vol. 85, no. 4, pp. 541–557, Apr. 1997.
[5] G. Csaba, A. Imre, G. H. Bernstein, W. Porod, and V. Metlushko, "Nanocomputing by field-coupled nanomagnets," IEEE Trans. Nanotechnol., vol. 1, no. 4, pp. 209–213, Dec. 2002.
[6] A. Khitun and K. L. Wang, "Nano scale computational architectures with spin wave bus," Superlattices Microstruct., vol. 38, no. 3, pp. 184–200, 2005.
[7] G. V. Resta et al., "Doping-free complementary logic gates enabled by two-dimensional polarity-controllable transistors," ACS Nano, vol. 12, no. 7, pp. 7039–7047, 2018.
[8] D. E. Knuth, The Art of Computer Programming, Volume 4A: Combinatorial Algorithms, Part 1. Boston, MA, USA: Pearson Educ., 2011.
[9] S. Ahmed and S. F. Naz, "Notice of violation of IEEE publication principles: Design of cost efficient modular digital QCA circuits using optimized XOR gate," IEEE Trans. Circuits Syst. II, Exp. Briefs, early access, Oct. 12, 2020, doi: 10.1109/TCSII.2020.3030180.
[10] L. Wang and G. Xie, "A novel XOR/XNOR structure for modular design of QCA circuits," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 67, no. 12, pp. 3327–3331, Dec. 2020

