PERFORMANCE ANALYSIS AND COMPARISON OF PARALLEL PREFIX ADDERS

Hima Bindu Challa[1*], Dr. S.V.R.K. Rao[2], Srujana Gollapalli[3]

[1]Scholar (Corresponding Author), [2]Professor & HOD, [3]Asst.Professor
Dept. of Electronics & Communication Engineering,
GIET(A), Rajahmundry, AP-533296, India

***Abstract* -** With the technology increasing by leaps and bounds, the concept of miniaturization took the front seat. The three main aspects power, area and delay are to be paid attention and balanced optimally. Adders are one of the basic and primitive digital circuits as they are used most of the higher level digital circuits. They also serve as the key elements in other arithmetic operations like subtraction, multiplication and division. As they say, change from the inner side is constructive, gate level and circuit level changes help in improving the overall performance of the digital system which is the eventual target of VLSI design. Quantum-dot Cellular Automata technique aids in bringing these gate level changes in BCD adders, where the logic gates like AND, OR & NAND are designed by this QCA technique. Various Parallel Prefix Adder structures such as Kogge-Stone, Brent-Kung and Ladner-Fischer Adders are discussed here which operate at higher speed than the conventional adders and at the same time doesn't compromise in terms of size and power consumption.

***Keywords* -** Parallel Prefix Adder, Quantum-dot Cellular Automata, Kogge-Stone, Brent-Kung, Ladner-Fischer.

I. INTRODUCTION

When discussed about VLSI digital circuits, adders are specially mentioned and discussed due to their application in most of the digital circuits like Digital Signal Processors, MAC Units and ALU units. Apparently, their structure, operation and miniaturization have to be paid ample importance. In the case of 1-bit addition, there is no carry propagation ad hence there is no delay. In the case of multi-bit addition, as the number of stages increases, the carry propagation time through all these stages also increases. Coming to our topic of "BCD addition"[1], it is a peculiar binary addition. In BCD addition, first the two 4-bit BCD numbers are added in binary format and then the sum is checked for correct BCD format. if the obtained sum is greater than binary 9, then a binary 6 is added to it to convert it into correct BCD form. Otherwise a binary 0 is added. Hence, the binary addition logic is accompanied by a correction logic for obtaining BCD sum. Hence, for the computation of efficient and quick BCD sum, we are in need of an efficient Binary adder. Designing of an effective adder structure which overcomes all the drawbacks of the previous adder structure isn't a cakewalk. The suggested adder has to possess many special abilities to stand out of all the existing structures. It should be error free and reliable enough to trust upon. Binary adders are of two types namely Serial adder and Parallel adder. Serial adders perform addition in a bit by bit fashion and due to this, computation time increases. Parallel adders perform addition in a parallel way by taking all the available inputs at a time. Several adder structures are proposed in the past such as Ripple carry adder, Carry select adder, Carry look-ahead adder etc,. In this paper, we took "QCA based BCD adder"[2] as the base of our work. Here we brief and compare the various binary adders, which are implemented using Parallel Prefix Technique for reducing the delay, power consumption and size of the chip than the prior ones like Ripple Carry Adder, Carry Look-Ahead Adder and Carry Save Adder.

II. BACKGROUND & RELATED WORK

The delay factor in binary adders is banked up on how quickly the carry moves from one bit to another bit. The binary adders and BCD adders have undergone many transformations owing to the development in the VLSI technology. One such method proposed is "Parallel Prefix adder"[3]. Previously, the ripple carry adders were used. But due to high delay factor, they cannot withstand the ever growing data rates. This is the origin of parallel prefix tree. Various new structures which implement n number adder architectures are proposed further out of which only few stand out. In 1973, "Kogge and Stone proposed a scheme which uses recursive doubling property and the properties of the prefix tree were determined by the minimum logic depth, structure and unity fan-out"[4]. The total time taken for output generation is $O(\log_2 N)$. In 1980, "Ladner and Fischer introduced minimum depth prefix graph"[5] with higher fan-out for driving larger capacitive loads. As a result, additional buffers have to be added to the circuit which increases the cost and delay in the circuit. It works based on the odd even algorithm and the prefix sum issue. Ling (1981) introduced a set of new carry generation equations where one propagate term is used to simplify the group generate function and thus reducing the burden on the first level prefix tree. "Brent and Kung (1982) presented a prefix computation graph"[6] which is equivalent to Kogge-Stone Adder in many terms but has got bigger internal circuitry and wirings. Next in 1987, "Han and Carlson came up with a new structure which is a blend of both Brent-Kung and Kogge-Stone adders"[7]. It can be called as the modified version of Kogge Stone adder. "Reto Zimmermann (1996) proposed a heuristic approach for prefix adder optimization using depth controlled compression and expansion"[8]. "Knowles (2001) presented a class of logarithmic adders with minimum depth by allowing the fan-out to grow"[9]. Later in 2005, "Dimitrakopoulos and Nikolos presented an innovative approach where one logic level of implementation can be avoided when compared to the traditional binary adders"[10].

III. CONVENTIONAL ADDERS

Coming to the adder structures, primarily two adders were proposed for single bit and multi-bit addition. They are Half adder and Full adder respectively. The carry propagation problem arises only in full adder as it sums more than two bits at a time. The sum and carry expressions of Full adder are given below

$$Si = ai \text{ (xor) } bi \text{ (xor) } ci \qquad (1)$$

$$Ci+1 = (ai . bi) +(ai . ci) + (bi . ci) \qquad (2)$$

When one has to perform multi-bit operation, these full adders/half adders are connected in cascade fashion so that the carry out of one stage becomes the carry in of the next stage. In this way there are few adders which can be used to add n number of bits. Those adders which were us prior to Parallel Prefix adders are mentioned here.

A. Ripple Carry Adder

In the structure of Ripple Carry Adders, the full adder circuits are joined to one another in such a way that the carry out of each block acts as the carry in for the next full adder. In this way, several full adders can be connected to form a n-bit "ripple carry adder"[11]. Since all the adder logics are joined together, every adder has to wait for it's feed of carry in and then only it can furnish the sum bit and carry bit. So the first carry in at the first level has to travel through all the given levels to generate the terminal carry of the adder. This accounts to a lot of delay in the circuit. Carry propagation and sum generation of a simple 4-bit ripple carry adder are shown below

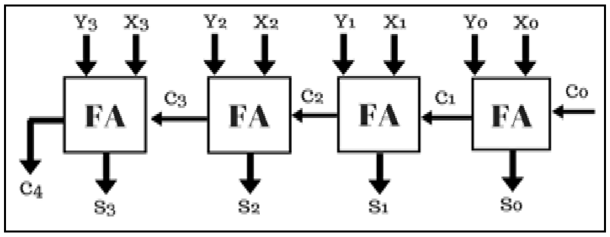

Figure1: A 4-bit Ripple Carry Adder[11]

B. Carry Select Adder

"It is a simple but rather fast adder than the ripple carry adder and has got the logical depth of O(√n)[12]". Here two additions are implemented. One addition operation is done by taking the carry in as zero and the another addition operation is performed by considering the carry in as one. Then finally the result is selected with the aid of the multiplexer. As the multiplexer circuits occupy more space, it becomes more spacious to accommodate higher order multiplexers and is not economical. A primitive 4-bit carry select adder is depicted down

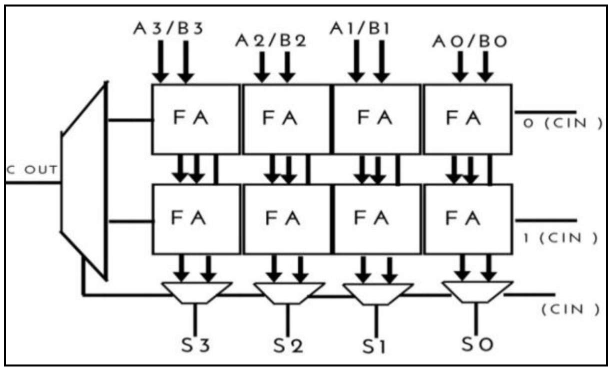

Figure2: A 4-bit Carry Select Adder[11]

C. Carry Look-Ahead Adder

It accommodates a different logic to predict the higher order carrys so that the higher order full adder blocks need not wait for the carry in to propagate till them. Two terms called Generate and Propagate are used here. These two are used to compute the higher order carrys. The logic expressions for the Propagate and Generate are as follows

$$G = A_i(and)B_i \qquad (3)$$

$$P = A_i(xor)B_i \qquad (4)$$

Then with the help of generate and propagate terms the carry is furnished as follows

$$C_{i+1} = G_i + [P_i \, . \, C_i] \qquad (5)$$

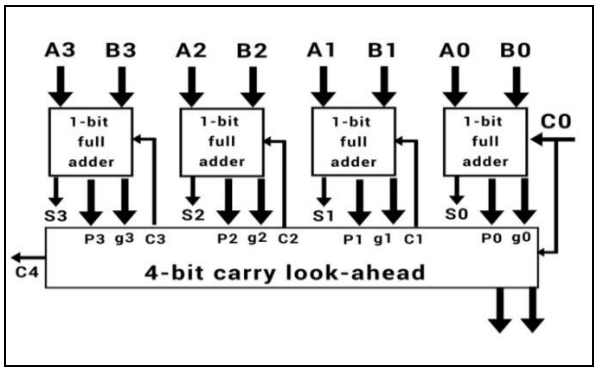

Figure3: A Carry Look-Ahead Adder[13]

The application of this carry look ahead adder becomes difficult as the number of stages increases. This is because the number of logic gates in the expression C_{i+1} increases, which in turn increases the complexity and the space consumption.

IV. PARALLEL PREFIX ADDER STRUCTURE

Coming to our concept of "Parallel Prefix Adders"[14], it's operation is similar to Carry Look-ahead adder but it differs in the way the carry is generated and propagated. This is an age old technique which can be implemented even in the modern days for its effectiveness.

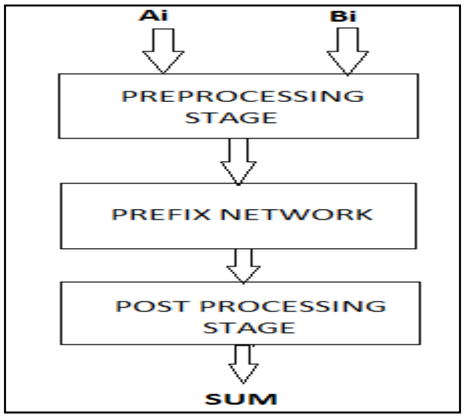

Figure5: Structure of Parallel Prefix Adder

A. Pre-Computation Stage

In this stage, the individual generate and propagate signals of all the available input bits are calculated. These propagate and generate terms are the primitive terms that are used for the calculation of the higher order carrys. They are mainly seen in the carry look ahead structures.

$$G_i = A_i.B_i \quad (6)$$

$$P_i = A_i(xor)B_i \quad (7)$$

B. Prefix Network Stage:

Here, the most crucial operation of carry merging is performed.As mentioned prior, the terms Propagate and Generate play a key role in the computation of the sum. As a part of this, group generate and group propagate signals are computed.

$$G_{(i:j)} = G_{(i:k)} + \left(G_{(k-1:j)}.P_{(i:k)}\right) \quad (8)$$

$$P_{(i:j)} = P_{(i:k)}.P_{(k-1:j)} \quad (9)$$

The group generate and propagate terms are furnished by using two operators called as "Black cell and the Gray cell"[15]. The majority gates and full adder structures of the QCA method consume a lot of space and the power consumption also increases. So the majority gate logic is replaced by the PPA structure's Black cells and Gray cells. The outputs of this prefix network stage are the passed to the post processing stage/post computation stage.

C. Post-Computation Stage

In this stage, the final sum and carry are calculated with the aid of Ex-OR gates, AND gates and OR gates. The method of computation of sum and carry is given below

$$SUM = P_i(xor)C_{i-1} \quad (10)$$

$$C_{i-1} = [P_i(and)C_{in}]or\ G_i \quad (11)$$

Based on this structure of PPA, Few topologies are proposed

1. Kogge Stone Adder
2. Brent Kung Adder
3. Ladner and Fischer Adder
4. Han Carlson Adder

Kogge Stone Adder:

Kogge-Stone adder is proposed by "M.Kogge and Harold.S.Stone"[16]. Every stage in the Kogge stone adder has got a fan out of two. It is an attractive approach for high speed applications. But the cost is what all matters. Also the power consumption also increases. The overall lag in the circuit is given by $\log_2$ n. The total number of computation nodes in this tree structure is given by [(n)($\log_2$ n)- n+1]. The total time taken by a Kogge and Stone adder to generate the output is O($\log_2$N). The property of idempotency has found it's application here for limiting the lateral fan out. "This is because there, is a massive overlap between the prefix sub-terms being pre-computed"[5]. The prefix graph of a 32-bit Kogge Stone Adder is given below.

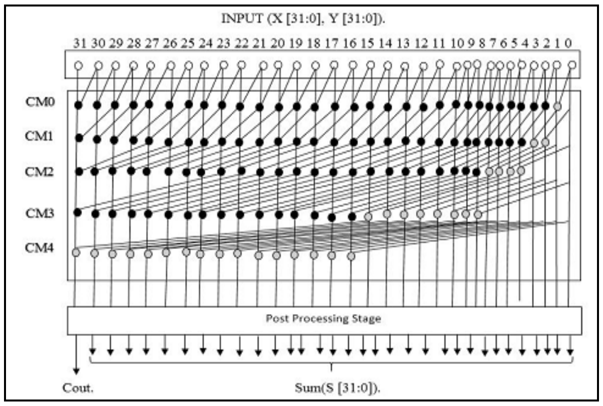

Figure 6: A 32-bit Kogge-Stone Adder

Brent Kung Adder:

This adder is put forward by two persons Brent and Kung in the year 1982. "Brent Kung Adder"[6] is mainly used when the input is bigger/have more number of bits. Brent Kung adder has got a very long cynical path and the circuit structure doesn't support a fast operation. A feasible adder structure for the binary addition is possible, if and only if the carrys are furnished at every power of two bit positions as suggested by Brent and Kung. The wiring and complexity of this adder is very much less than Kogge Stone Adder. The delay and number of nodes are given below

$$Delay = (2*\log_2 n)-2 \quad (12)$$

$$Nodes = (2*n)- 2 - \log_2 n \quad (13)$$

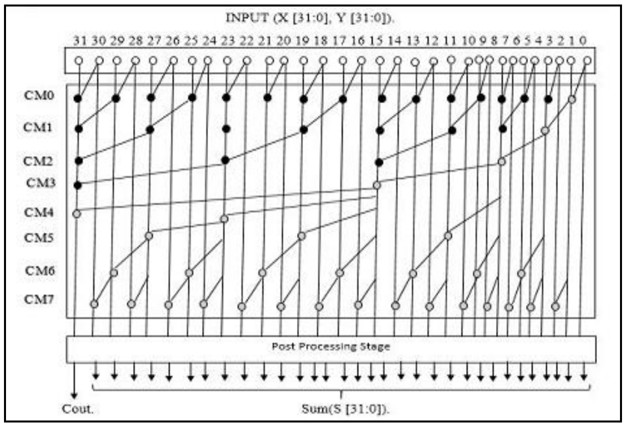

Figure 7: A 32-bit Brent Kung Adder

If there are N number of input bits, then the total number of logic levels in a Brent Kung Adder is given by 2($\log_2$ N-1).

Ladner-Fischer Adder:

This adder was proposed by "R.Ladner and M.Fischer"[14] in the year 1980. Here every node in the circuit performs sum of two numbers. "By using this structure, one can select a trade-off between the circuit depth and the number of nodes"[12]. The pre-computation and post computation stages are same here as of the carry look ahead adder. As mentioned prior, the number of bits is divided into two parts for ease of computation. Further this half number of blocks are also divided into two halves and the process goes on.

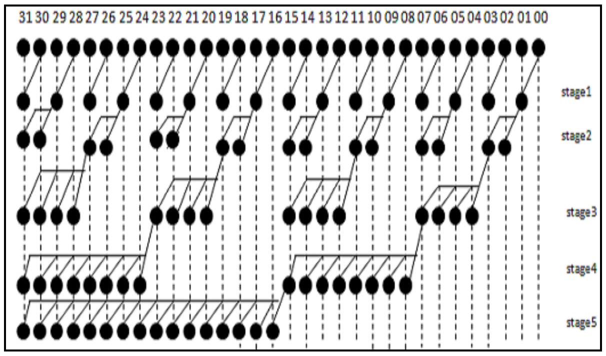

Figure 8: A 32-bit Ladner-Fischer Adder

This can be assumed in terms of a reverse tree structure. So finally the area occupied also gets reduced. There is no limit for fan out here. This leads to more delay in the circuit and thus slowing down it's operation.

Han Carlson Adder:

"Han Carlson adder"[17] is a blend of both Brent Kung adder and Kogge Stone adder. It was suggested in the year 1987 by Han and Carlson. A Brent Kung adder action can be seen in the first stage. In the next three stages Kogge Stone adder is implemented. Wire lengths of this adder are very less than that of the Kogge Stone adder. The even bits are operated by carry merge operation and the odd number bits undergo carry propagation operation. The delay in the circuit is given by [($\log_2$ n)+1]. The computation hardware complexity is [n/2 ($\log_2$ n)].

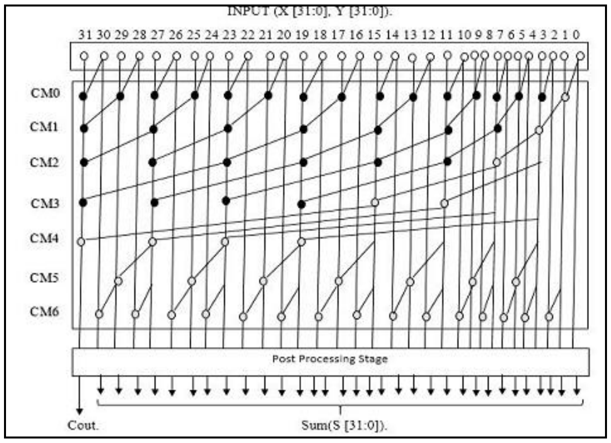

Figure 9: A 32-bit Han Carlson Adder

V. SIMULATION RESULTS & DISCUSSION

The above are the famous PPA Adder structures. "Quantum dot Cellular Atomata"[18](QCA) technique is also used for the designing of BCD Adders previously. Alongside with it Clocking Schemes are implemented, where the entire clock pulse is divided into four stages namely Switch, Hold, Release and Relax. "2D Clocking technique"[19] is mainly used for this purpose where the stages are divided and operations are carried out to reduce the overall delay. The generate and propagate terms are computed by using "Majority Gates"[20] here. The Simulation Results of the QCA Based Adder when they were run with the help of Xilinx ISE Design Suite 14.5 are shown below

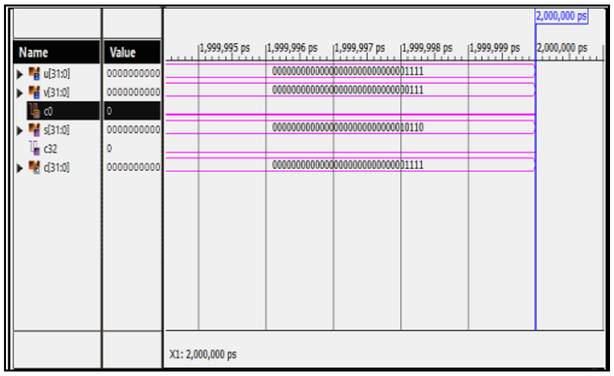

Figure 10: Simulation results of QCA based Adder

By implementing the latest PPA technique, the worst case delay can be highly reduced even though it can be eliminated completely. The utilization of LUTs and IOBs are also increased. The simulation results of the PPA Adder are given below. In this way, few PPA Adders are proposed to overcome the previous adder structure. The various Parallel Prefix Adders are compared in terms delay, performance, computational nodes and fan-out and are presented here in tabular form. These figures prove that the PPA Adders are any day better and preferable when compared to the basic old fashioned adders.

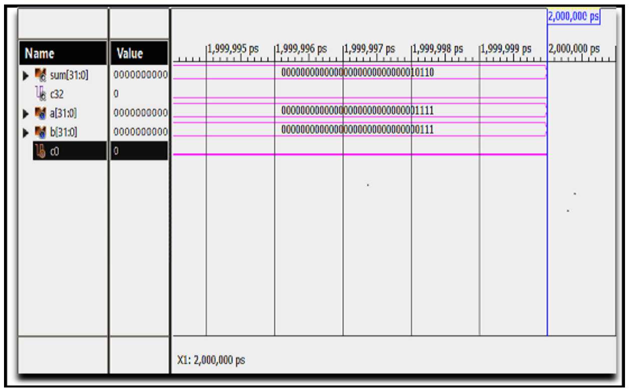

Figure 11: Simulation Results of PPA based Adder

Table.1 Comparison of Power, Delay and Power-Delay product

Adder Name	**Adder Power (µW)**	**Delay (ns)**	**Power-Delay Product**
Kogge Stone	350.2907	0.89	311.7587
Brent Kung	197.0751	1.29	254.2268
Han Carlson	233.8426	0.07	250.2115
Ladner Fischer	211.6786	1.13	239.1968

VI. CONCLUSION

Finally it can be concluded that the Parallel Prefix Adder is better than many other conventional adder structures. The decimal adder provides a higher logical depth which proves about the critical logic operations that take place inside. The delay of the existing QCA based BCD adder is 43.142ns which is reduced to 32.981ns in the suggested novel adder. This adder design can be further exploited with the help of upcoming technologies enriched with enhanced logics in such a way to provide much less area consumption. Improvements in logic with the help of innovative current day technologies may aid in providing less delay along with more efficiency.

VII. REFERENCES

[1]. Schmookler, M.S. and Weinberger, A., International Business Machines Corp, 1971. Improved decimal adder for directly implementing bcd addition utilizing logic circuitry. U.S. Patent 3,629,565.

[2] Lent, C.S., Tougaw, P.D., Porod, W. and Bernstein, G.H., 1993. Quantum cellularautomata. Nanotechnology, 4(1), p.49.

[3] Beaumont-Smith, A. and Lim, C.C., 2001. Parallel prefix adder design. In Computer Arithmetic, 2001. Proceedings. 15th IEEE Symposium on (pp. 218-225). IEEE.

[4] Poornima, N. and Bhaaskaran, V.K., 2014, January. Power-Delay Optimized 32 Bit Radix-4, Sparse-4 Prefix Adder. In 2014 Fifth

International Conference on Signal and Image Processing (ICSIP) (pp. 201-205). IEEE.

[5] Knowles, S., 1999. A family of adders. In Computer Arithmetic, 1999. Proceedings. 14th IEEE Symposium on (pp. 30-34). IEEE.

[6] Brent, R.P. and Kung, H.T., 1982. A regular layout for parallel adders. IEEE transactions on Computers, (3), pp.260-264.

[7] Roy, S., Choudhury, M., Puri, R. and Pan, D.Z., 2014. Towards optimal performance-area trade-off in adders by synthesis of parallel prefix structures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 33(10), pp.1517- 1530.

[8] Zhu, H., Cheng, C.K. and Graham, R., 2005, January. Constructing zero- deficiency parallel prefix adder of minimum depth. In Design Automation Conference, 2005. Proceedings of the ASP-DAC 2005. Asia and South Pacific (Vol. 2, pp. 883-888). IEEE.

[9] Ramanathan, P. and Vanathi, P.T., 2009. A novel power delay optimized 32-bit parallel prefix adder for high speed computing. International Journal of Recent Trends in Engineering, 2(6).

[10] Basha, M.M., Ramanaiah, K.V. and Reddy, P.R., Modified Reverse Converter Design With intervention of Efficacious Parallel Prefix Adders.

[11] Cuccaro, S.A., Draper, T.G., Kutin, S.A. and Moulton, D.P., 2004. A new quantum ripple-carry addition circuit. arXiv preprint quant- ph/0410184.

[12] Chandrakasan, A.P., Sheng, S. and Brodersen, R.W., 1992. Low-power CMOS digital design. IEICE Transactions on Electronics, 75(4), pp.371-382.

[13] Doran, R.W., 1988. Variants of an improved carry look-ahead adder. IEEE Transactions on Computers, 37(9), pp.1110-1113.

[14] Ladner, R.E. and Fischer, M.J., 1980. Parallel prefix computation. Journal of the ACM (JACM), 27(4), pp.831- 838.

[15] Harris, D. and Sutherland, I., 2003, November. "Logical effort of carry propagate adders.In Signals, Systems and Computers", 2004. Conference Record of the Thirty- Seventh Asilomar Conference on (Vol. 1, pp. 873-878). IEEE.

[16] Kogge, P.M. and Stone, H.S., 1973. A parallel algorithm for the efficient solution of a general class of recurrence equations. IEEE transactions on computers, 100(8), pp.786- 793.

[17] Esposito, D., De Caro, D., Napoli, E., Petra, N. and Strollo, A.G.M., 2015. Variable Latency Speculative Han-Carlson Adder. IEEE Trans. on Circuits and Systems, 62(5), pp.1353-1361.

[18] Cho, H. and Swartzlander, E.E., 2007. Adder designs and analyses for quantum-dot cellular automata. IEEE Transactions on Nanotechnology, 6(3), pp.374-383.

[19] Vankamamidi, V., Ottavi, M. and Lombardi, F., 2006, June. Clocking and cell placement for QCA. In Nanotechnology, 2006. IEEE-NANO 2006. Sixth IEEE Conference on (Vol.1, pp. 343- 346). IEEE.

[20] Imre, A., Csaba, G., Ji, L., Orlov, A., Bernstein, G.H. and Porod, W., 2006. Majority logic gate for magnetic quantum-dotcellular automata. Science, 311(5758), pp.205- 208.