Carry-look aheadis the most important technique in the design of fast adders, especially large ones. In straightforward addition, e.g. in a ripple adder, the operational time is limited by the (worst-case) time allowed for the propagation of carries and is proportional to the number of bits added. So faster adders can be obtained by devising a way to determine carries before they are required to form the sum bits. Carry-look ahead does just this, and, in certain cases the resulting adders have an operational time that is independent of the operands word-length. A carry, Ci, is produced at bit-stage i if either one is generated at that stage or if one is propagated from the preceding stage. So a carry is generated if both operand bits are 1, and an incoming carry is propagated if one of the operand bits is 1 and the other is 0. Let Pi and Gidenote the generation and propagation, respectively, of a carry at stage i, Ai and Bi denote the two operands bits at that stage, and Ci-1 denote the carry into the stage. Then we have

Gi=AiBi

Pi = Ai^Bi

Ci = Gi + PiCi-1

and the sum can be written as Si== Pi^Ci-1 which allows the use of shared logic to produce Si and Pi.

C0 = G0 + P0Ci-1

C1 = G1 + P1P0C-1+ P1G0

.

.

.

Ci = Gi + Pi-1Gi-1+PiPi-1Gi-2+…+PiPi-1Pi-2…P0C-1

where Ci-1 is the carry into the adder. The equation for Ci states that there is a carry from stage i if there is a carry generated at stage i, or if there is a carry that is generated at stage i-1 and propagated through stage i or if , or if the initial carry-in, Ci-1, is propagated through stages 0,1,… i. The complete set, of equations show that, in theory at least, all the carries can be determined independently, in parallel, and in a time (three gate delays) that is independent of the number of bits to be added. The same is also therefore true for all the sum bits, which require only one additional gate delay.

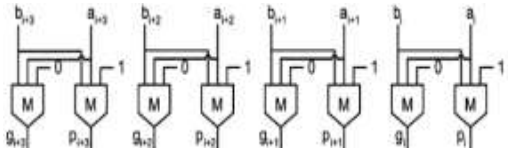

Figure 7: Generation Of Propagate and Generate Bits

Gi = AiBi

Pi = Ai^Bi

Compared with a ripple adder, as well as some of the other adders, a pure carry-look ahead adder has high logic costs. Furthermore, high fan-in and fan-out requirements can be problematic: the fan-out required of the Gi and Pi signals grows rapidly with n, as does the fan-in required to form Ci. For sufficiently large values of n, the high fan- in and fan-out requirements will result in low performance, high cost, or designs that simply cannot be realized.

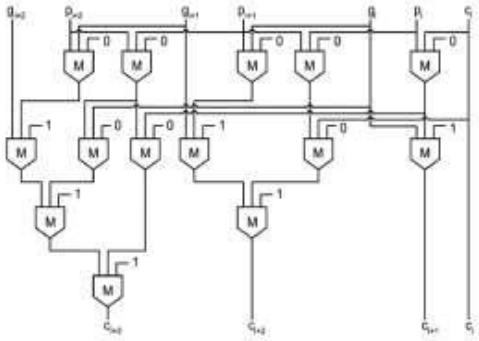

Figure 8: Carry Block

This carry block is cascaded with the propagate and generate block. So, that carry is obtained with thefollowing equation.

Ci = Gi + PiCi-1

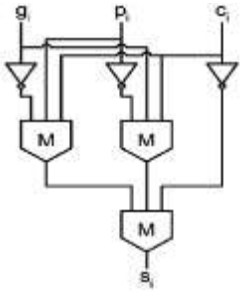

Figure 9: Sum Block

This sum block is cascaded with the above carry block to obtain the sum. The following equation gives the sum bit

Si = Pi ^ Ci-1.

6. Modified QCA Architecture

In Proposed System, majority logic is reduced for the reduction of complexity.Given three binary inputs, a, b, and c, the majority voting logic function can be expressed in terms of fundamental Boolean operators M(a, b, c)= ab + bc + ac.

Rule 1: If a, b and c are three binary inputs, then M(a, b, M(a, b, c)) = M(a, b, c).

Rule 2: Let a, b and c be three binary inputs. Then

$$M(a, b, \overline{M(a, b, c)}) = M(a, b, \overline{c}).$$

Rule3: Letf1,f2, and f3 be three Boolean functions such that f1 and f2 satisfy f1 f2 = f1 and f1 + f2 = f2. Then M(f1 ,f2 ,f3)= f1 + f2 f3.

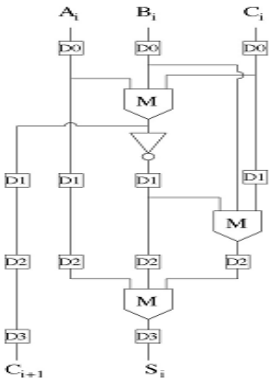

Figure 10: Full Adder Realization Using Three Majority Gates and One Inverter; Numbered D-Latches Enable Delay Determination

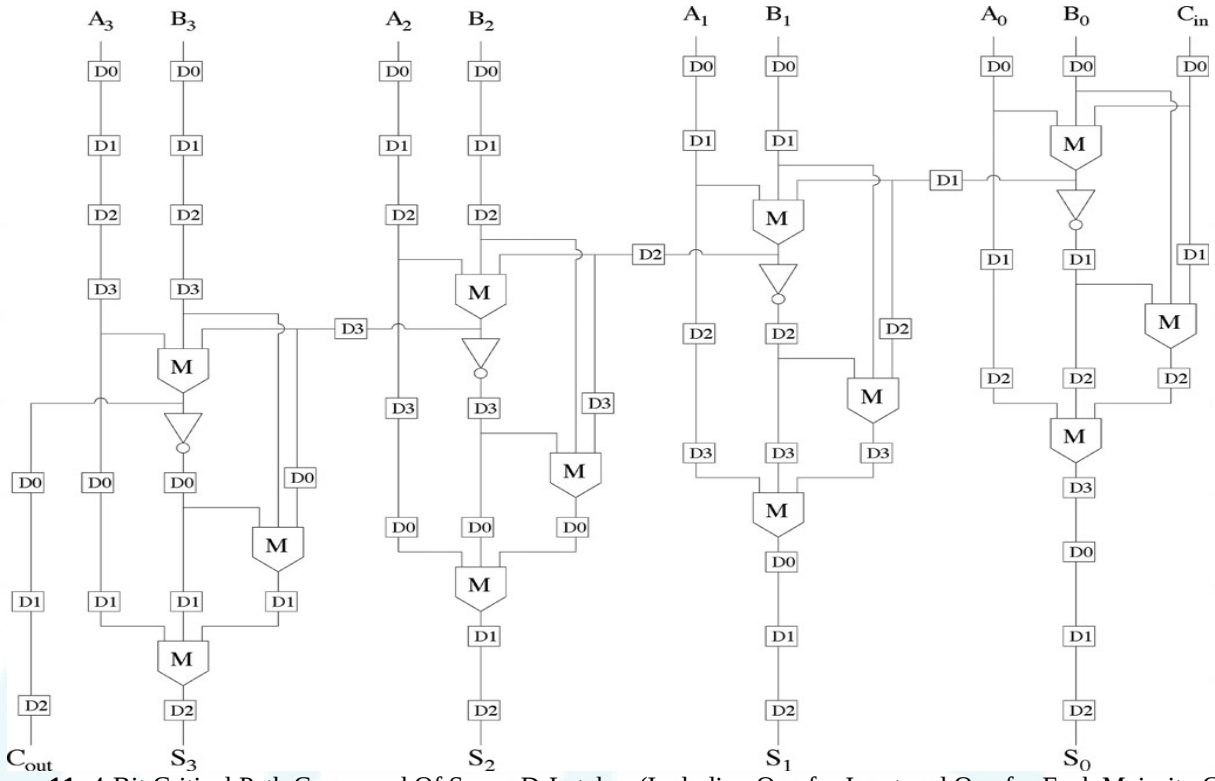

Figure 11: 4-Bit Critical Path Composed Of Seven D-Latches (Including One for Input and One for Each Majority Gate

7. Conclusion

A new adder designed in QCA was presented. It achieved speed performances higher than all the existing QCA adders, with an area requirement comparable with the cheap RCA and CFA demonstrated in.The novel adder operated in the RCA fashion, but it could propagate a carry signal through a number of cascaded MGs significantly lower than conventional RCA adders. In addition, because of the adopted basic logic and layout strategy, the number of clock cycles required for completing the elaboration was limited. A 128-bit binary adder designed as described in this brief exhibited a delay of only seventeen clock cycles, occupied an active area of 32.25 μm2, andachieved an ADP of only 548.25.

8. Acknowledgement

My wishes to acknowledge Mrs.P.VINITHA mam, Lecturer who guided us for the conference paper preparation. Thank you mam.

References

[1] StefaniaPerri, Pasquale Corsonello, andGiuseppe Cocorullo, “Area-Delay EfficientBinary Adders in QCA”, 2013 IeeeTransactions On Very Large ScaleIntegration (Vlsi) Systems
[2] C. S. Lent, P. D. Tougaw, W. Porod, and G.H. Bernestein, “Quantum cellularautomata,” Nanotechnology, vol. 4, no. 1,pp. 49–57, 1993
[3] H. Cho and E. E. Swartzlander, “Adderdesign and analyses for quantum-dotcellular automata,” IEEE Trans.Nanotechnol., vol. 6, no. 3, pp. 374–383,May 2007
[4] H. Cho and E. E. Swartzlander, “Adder andmultiplier design in quantum-dot cellularautomata,” IEEE Trans. Comput., vol. 58,no. 6, pp. 721–727, Jun. 2009
[5] W. Liu, L. Lu, M. O’Neill, and E. ESwartzlander, Jr., “Design rules for quantum-dot cellular automata,” in Proc. IEEE Int. Symp. Circuits Syst., May 2011, pp. 2361–2364