Design of high speed hybrid carry select adder

[1]Shivani Parmar, [2]Kirat Pal Singh
[1, 2] Assistant Professor
[1, 2] Electronics and Communication Engineering Department
[1] Sachdeva Engineering College for Girls, Gharuan, Punjab, India
[2]SSET, Surya World University, Bapror, Rajpura, Punjab, India
Email – [1]kiratpal.singh@suryaworld.edu.in, [2]shivaniparmar03@gmail.com

***Abstract* - The paper describes the power and area efficient carry select adder (CSA). Firstly, CSA is one of the fastest adders used in many data-processing systems to perform fast arithmetic operations. Secondly, CSA is intermediate between small areas but longer delay Ripple Carry Adder (RCA) and a larger area with shorter delay carry look-ahead adder. Third, there is still scope to reduce area in CSA by introduction of some add-one scheme. In Modified Carry Select Adder (MCSA) design, single RCA and BEC are used instead of dual RCAs to reduce area and power consumption with small speed penalty. The reason for area reduction is that, the number of logic gates used to design a BEC is less than the number of logic gates used for a RCA design. Thus, importance of BEC logic comes from the large silicon area reduction when designing MCSA for large number of bits. MCSA architectures are designed for 8-bit, 16-bit, 32-bit and 64-bit respectively. The design has been synthesized at 90nm process technology targeting using Xilinx Spartan-3 device. Comparison results of modified CSA with conventional CSA show better results and improvements.**

Keyword - Adder, CSA, MCSA, delay, area.

I. INTRODUCTION

Performance of large digital circuits is dependent on the speed of circuits that form various functional units. Adders are one of the widely used in digital integrated circuit and system design. High speed adder is the necessary component in a data path, e.g. Microprocessors and a Digital signal processor. For adding two binary numbers there exists several adder structures based on very different design ideas. Thus if one need to implement an addition circuit one must decide which circuit is most appropriate for its planned application. There are many binary adder architecture ideas to be implemented in such applications. However, it has been difficult to do well both in speed and in area. The easiest type of parallel adder to build is a ripple carry adder, which uses a chain of one bit full adder to generate its output. The Ripple Carry Adder (RCA) gives the most compact design but takes longer computation time. The time critical applications use Carry Look-ahead scheme (CLA) to derive fast results but lead to increase in area. The Carry Select Adder (CSA) provides a compromise between small areas but longer delay Ripple Carry Adder (RCA) and a larger area with shorter delay Carry Look-Ahead Adder (CLA) [1].

In mobile electronics, reducing area and power consumption are key factors in increasing portability and battery life. Even in servers and desktop computers, power consumption is an important design constraint. Design of area- and power-efficient high-speed data path logic systems are one of the most substantial areas of research in VLSI system design. In digital adders, the speed of addition is limited by the time required to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated into the next position [3]. Among various adders, the CSA is intermediate regarding speed and area [2].

A. Adder Architectures [4] [5]

A two-operand adder is used not only when performing additions and subtractions, but also often employed when executing more complex operations like multiplication and division. Consequently, a fast and area efficient two-operand adder is essential. Two basic kinds of adders are: Half-Adder and Full Adder. A Half-adder (HA) is an adder that accepts two inputs and gives two outputs. A half adder consists of two logic gates. These are an AND gate, and an Exclusive OR gate. Full-adder (FA) performs the arithmetic sum of three bits. It consists of three inputs and two outputs .Three inputs involves two input bits plus an extra bit for an incoming carry. This is important for cascading adders together to create N-bit adders. A full-adder is made up of two HAs and an OR gate.

To add multiple inputs various types of Carry-Propagate Adders (CPA) are present. Among them RCA design occupies the small area but takes longer computing time. The delay of RCA is linearly proportional to number of input bits. For some input signals carry has to ripple all the way from least significant bit (LSB) to most significant bit (MSB). The propagation delay of such a circuit is defined as the worst case delay over all possible input patterns also called as critical path delay. The Carry Skip Adder (CSKA) uses a carry skip scheme to reduce the additional time taken to propagate the carry signal in RCA. Thus, CSKA is faster than RCA at the expense of a few simple modifications. The CLA offers a way to eliminate the ripple effect. For every bit, sum and carry is independent of the previous bits. CLA is faster than RCA but consumes large area. CLA is fast for a design having less input bits, for higher number bits it shows the worse delay [5]. In RCA every full adder has to wait for the incoming carry before an outgoing carry is generated. The CSA provide a way to get around this linear dependency is to anticipate both possible values of the carry input i.e. 0 and 1 and evaluate the result in advance. Once the real value of the carry is known the result can be easily selected with the help of a multiplexer stage. The CSA is intermediate between longer delay RCA and large area CLA.

978-1-4673-4529-3/12/$31.00 ©2012 IEEE

II. PREVIOUS DESIGN

The literature survey gives us an idea of the paper, which further need some preliminaries of Adders. An overview of these adder types is given below.

A. Adders

Adders are extensively used in processing units such as the ALU's (Arithmetic Logic Unit) or in DSP (Digital Signal Processing) applications. Before going further, the detail discussion for basic adders is as under:

A two-operand adder is used not only when performing additions and subtractions, but also often employed when executing more complex operations like multiplication and division. Consequently, a fast and area efficient two-operand adder is essential. Two basic kinds of adders are:

1) Half-Adder (HA)

A half-adder (HA) is an adder that accepts two inputs and gives two outputs. The two inputs are the two single bit binary values. The two outputs represent the sum and carry. A half adder consists of two logic gates, an AND gate, and an Exclusive OR gate. Equations for sum and carry for H.A are given as

$$\text{sum} = A \oplus B$$

$$\text{carry} = A.B$$

A diagram of a half adder is shown below in Figure 1. It has two inputs A and B and two outputs are Sum and Carry.

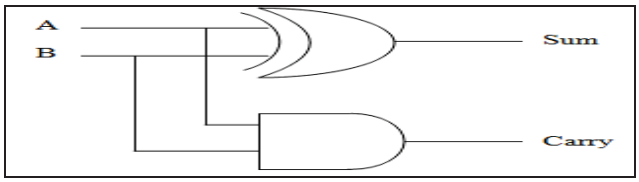

Figure 1. Half Adder using XOR and AND gate [6]

2) Full-Adder (FA)

The main difference between a HA and a FA is that a FA performs the arithmetic sum of three bits. It consists of three inputs and two outputs .Three inputs involve two input bits A and B plus an extra bit for an incoming carry C_{in}. Two outputs are Sum and carry out C_{out}. This is important for cascading adders together to create N-bit adders. A full-adder is made up of two HAs and an OR gate. A diagram below shows how a full adder is connected.

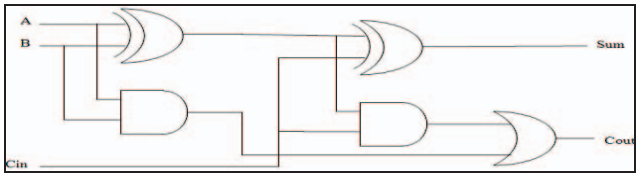

Figure 2. Full Adder using two HAs and an OR gate [6]

The most straightforward implementation of a parallel adder for two operands A_{n-1}, A_{n-2} . . . ,A_o and B_{n-1}, B_{n-2} .....,B_o is through the use of N basic full adders units. The FA is a combinational digital circuit implementing the binary addition of three bits through the following Boolean equations:

$$\text{Sum} = A + B + C_{in}$$

$$C_{out} = A.B + B.C_{in} + A.C_{in}$$

Where symbol $\oplus$ is the exclusive-or operation (XOR), and A.B is the AND operation, and A+B is the OR operation.

B. Carry-Propagate Adders (CPA)

1) Ripple Carry Adder (RCA) [5]

Ripple Carry Adder for N-bit numbers is implemented by joining N full adders as shown in Figure 3, where N=4. This is called a RCA, since the carry signal "ripple" from the least significant bit position to the most significant bit position. The carry of this adder traverses longest path called worst case delay path through N stages. The propagation time through the RCA is calculated using following mathematical equation:

$$t_{adder} = N - 1\, t_{carry} + t_{sum}$$

where, t_{carry} is time taken for the carry to ripple from C_{in} to C_{out} , t_{sum} is the time taken to produce the sum, N is the total number of bits.

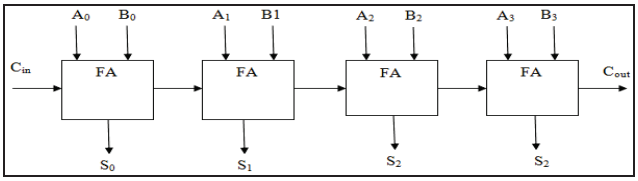

Figure 3. Block diagram of Ripple Carry Adder [5]

For some input signals carry has to ripple all the way from LSB to MSB. The propagation delay of such a circuit is defined as the worst case delay over all possible input patterns also called as critical path delay. Main drawback of RCA is delay. As the N-bit length increases, the delay path increases proportionally. The delay is then proportional to number of bits in the input words N.

2) Carry Skip Adder (CSKA) [4]

Since the C_{in}-to-C_{out} represents the longest (critical) path in the RCA. The carry is accelerated by choosing the value of Ai and Bi in such a way that Carry-Propagate Pi signals within a group of bits is Pi(i=0,1,2,3)=1. If all the Pi signals within a group are ones i.e. Pi = 1, then the condition exists for the carry to skip the entire group. If Pi(i=0, 1, 2, 3) =1, then C1=Cin for the 1st stage, Pi (i=4,5 ,6 ,7)=1, then C2 =C1 for the 2nd stage, Pi(i=8, 9, 10, 11) =1, then C3 =C2 for the 3rd stage, Pi(i=12 , 13, 14, 15)=1, then C_{out}=C3 for the 4th stage. The CSKA divides the words to be added into groups of equal size of M-bits (where M=4). The basic structure of an N-bit CSKA is shown in Figure 4, where N=16. Within the group, carry propagates in a ripple-carry fashion. The worst case delay path t_{adder} is carry generated at least significant bit position, ripples through the Mbit positions, skip over (N/M - 2)bypass stages in the middle generating extra delays due to multiplexers t_{bypass}, and is consumed at the last bit position without generating an output carry C_{out} through M-1bit positions of most significant stage.

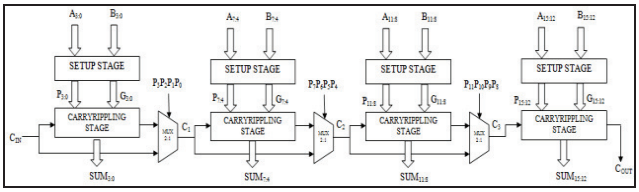

Figure 4. Block Diagram of Carry Skip Adder (CSKA)

The propagation time through the CSKA is calculated using following mathematical equation:

$$t_{adder} = t_{setup} + Mt_{carry} + NM - 1t_{bypass} + M - 1t_{carry} + t_{sum}$$

where, t_{setup} is the fixed delay of the setup stage to produce propagate and generate signals, t_{carry} is time taken for the carry to ripple through a full adder, t_{bypass} is delay of the multiplexer in bypass stage, t_{sum} is time taken to produce the sum, M is the number of stages and N/M is number of bits per stage. Thus, CSKA is faster than RCA at the expense of a few simple modifications for N >8. The delay is still linearly dependent on the size of the adder N.

3) Carry Look-Ahead Adder (CLA) [7]

In architectures of adders RCA and CSKA rippling effect of the carry is still present. A significant speed improvement in the implementation of a parallel adder was introduced by a Carry-Look-Ahead-Adder (CLA) developed by Weinberger.

In CLA sum and carry is independent the previous bit by using following related equations

$$Gi = A_i, B_i : \quad Pi = (A_i \oplus B_i i)$$

$$C_{i+1} = G_i + P_i, C_i$$

$$C_1 = G_0 + P_0, C_0$$

$$C_2 = G_1 + P_1, C_1 = G_1 + P_1, G_0 + P_1, P_0, C_0$$

$$C_3 = G_2 + P_2, G_1 + P_2, P_1 G_0 + P_2, P_1, P_0, C_0$$

$$C_4 = G_3 + P_3, G_2 + P_3, P_2, G_1 + P_3, P_2, P_1, G_0 + P_3, P_2, P_1, P_0, C_0$$

& so on.

Sum will be calculated as

$$S_i = A_i \oplus B_i \oplus C_i = P_i \oplus C_i$$

The rippling effect has been effectively eliminated in CLA as the dependency C_{out} on previous stages is completely eliminated by expanding the above equation. Thus the addition time is independent of number of bits in CLA. The block diagram of 16-bit CLA is shown in Figure 5, here N is the total number of bits, and M is the number of bits per stage. 16-bit CLA is divided into 4 stages; carry of each stage is generated by group carry look-ahead generator block.

Setup stages generate P and G signals for all the stages. Carry for individual stage is generated by carry look-ahead generator block of each stage. As we increase the number of bits in the CLA i.e. the word size, the complexity increases because the number of gates increases in calculating the carry expression Ci+1, thus the area increases. It is one of the fastest adder architecture, and allows significant design trade-offs to be made in terms of latency, area and power.

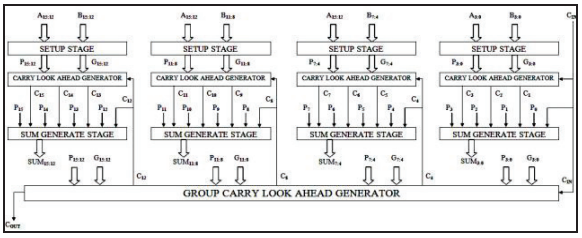

Figure 5. Block Diagram of Carry Look-Ahead Adder (CLA)

4) Carry Select Adder (CSA)

In RCA every full adder has to wait for the incoming carry before an outgoing carry is generated. One way to get around this linear dependency is to anticipate both possible values of the carry input i.e. 0 and 1 and evaluate the result in advance. Once the real value of the carry is known the result can be easily selected with the help of a simple multiplexer stage.

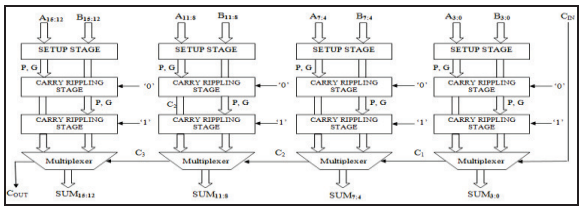

Figure 6. Block Diagram of Carry Select Adder [5]

A 16-bit CSA is constructed by dividing into 4 stages i.e. N=16 total number of bits, M=4 number of bits per stage, (N/M = 4) and chaining such four equal length blocks as shown in Figure 6. CSA has less delay as compared to RCA due to the anticipation of both possible values of C_{in} in advance, and as the multiplexers are used for the selection of C_{in}, so the area increases as compared to RCA.

The propagation time through the Carry Select Adder is calculated using following mathematical equation:

$$t_{adder} = t_{setup} + Mt_{carry} + (N/M)t_{mux} + t_{sum}$$

where, t_{setup} is delay of the setup stage to produce propagate and generate signals, t_{carry} is the time taken by the carry to ripple through a length of the stage M t_{carry} , t_{mux} is the delay of the multiplexer stage and t_{sum} is sum of time.

C. Add-One Circuits [7]

The conventional Carry Select Adder consists of two 4-bit adder blocks, one for C_{in} = 0and the other for C_{in}=l. If the results for C_{in} =0is known, the result for Cin=l can be found by adding one to the result for C_{in}=0. Thus, an add-one circuit can replace the 4-bit adder block for C_{in}=l. With an efficient design of an add-one circuit, the power and area of CSA can be reduced.

There are many proposed add-one circuits, among them we are discussing one proposed by et.al. Yajuan He. The logic for implementing add-one circuit is given below:

Assume $S^0 = (S_{n-10}, S_{n-2}{}^0, \ldots, S_0{}^0)$ and $S^1 = (S_{n-1}{}^1, S_{n-2}{}^1, \ldots, S_0{}^1)$ are the sum outputs of these two copies of RCA with block carry-in $C_{-1}{}^0 = 0$ and $C_{-1}{}^1 = 1$, respectively. The add one circuit proposed by Chang mitigates the resource overhead of CSL by replacing one copy of the RCA by

$S^1 = S^0 + 1$

Let k denote the position that the first bit of "0" is detected in Si 0, starting from LSB. Then, from

$$\prod_{i=0}^{k} S \prod_{i=0}^{k} S_i\, 0, k \in [0, n-1]$$

We have

$S_l{}^1 = (S_t{}^0)$ 't∈ [0, k],

$S_t{}^1 = S_t{}^0$t∈ [k+1, n-1]

From the above derivation, the add-one circuit is in essence, based on a "first" zero detection logic. It generates S1by inverting each bit in S0 starting from the LSB until the first zero is encountered as shown in Figure 7.

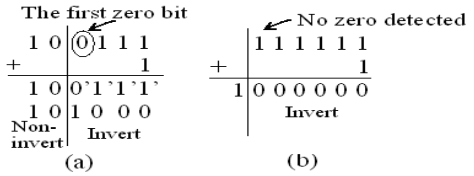

Figure 7. Examples for the first zero detection logic [7]

However, if no zero is detected in S^0 as illustrated in Figure 7, i.e., k ∉ [0, n-1], S^1 = (1, $(S_{n-1}{}^0)$', $((S_{n-2}{}^0)$', ….., $((S_0{}^0)$'). In other words, the carry-out signal for the add-one circuit is one if and only if all the sum outputs from the n-bit block are one.

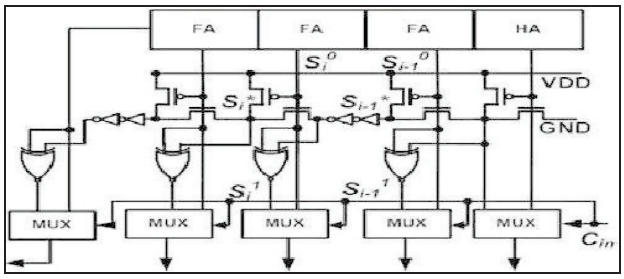

Figure 8. CSA adder with single RCA and add-one circuit

As all sums equal one, the first zero detection circuit generates one at the final node. For all the other cases, it generates a zero carry-out. As oppose to using dual RCAs in conventional CSL, the architecture of contemporary CSL adder comprises a single RCA, a first zero detection and selective complement add-one circuit, and a carry-select multiplexer circuit [7], as shown in Figure 8.

D. *Binary to Excess -1 Converter (BEC)*

BEC [5] is a circuit used to add 1 to the input numbers .A circuit of 4-bit BEC and the truth table is shown in Figure 9 and Table 1 respectively. The goal of addition is achieved using BEC together with a multiplexer (mux) shown in Figure 10, one input of the 8:4 mux gets as it input (B3, B2, B1, and B0) and another input of the mux is the BEC output.

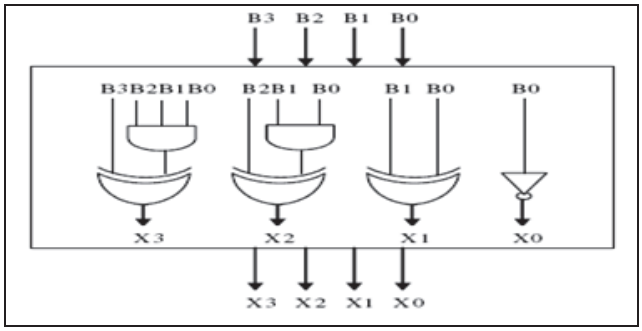

Figure 9. 4-bit Binary to Excess-1 code Converter

This gives the two partial results in parallel and the muxes are used to select either BEC output or the direct inputs according to the control signal C_{in}. The Boolean expressions of 4-bit BEC are listed below, (Note: functional symbols, ~ NOT, & AND, ^ XOR).

X0 = ~B0
X1 = B0 (1)^ B1
X2 = B2 ^(B0 & B1)
X3 = B3 ^ (B0 & B1 & B2)

TABLE 1 TRUTH TABLE OF 4-BIT BINARY TO EXCESS-1 LOGIC

Binary Logic $B_0B_1B_2B_3$	Excess-1 Logic $X_0X_1X_2X_3$
0 0 0 0	0 0 0 1
0 0 0 1	0 0 1 0
0 0 1 0	0 0 1 1
0 0 1 1	0 1 0 0
0 1 0 0	0 1 0 1
0 1 0 1	0 1 1 0
0 1 1 0	0 1 1 1
0 1 1 1	1 0 0 0
1 0 0 0	1 0 0 1
1 0 0 1	1 0 1 0
1 0 1 0	1 0 1 1
1 0 1 1	1 1 0 0
1 1 0 0	1 1 0 1
1 1 0 1	1 1 1 0
1 1 1 0	1 1 1 1
1 1 1 1	0 0 0 0

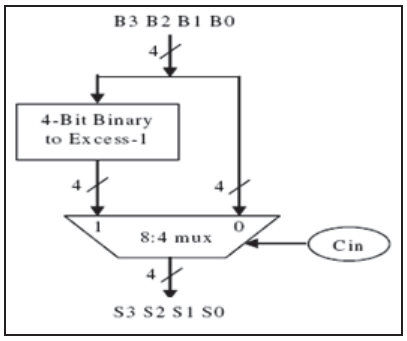

Figure 10. 4-bit Binary to Excess-1 logic with 8:4 multiplexer

The main idea of this work is to use BEC instead of the RCA with C_{in}=1 in order to reduce the area and power consumption of the conventional CSA. To replace the N-bit RCA, an N+1 bit BEC is required. The importance of BEC logic comes from the large silicon area reduction when designing MCSA for large number of bits.

III. CONVENTIONAL CARRY SELECT ADDER (CSA) DESIGN

CSA is one of the fastest adders used in many data-processing processors to perform fast arithmetic functions. The CSA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum [8].

A CSA consists of a number of blocks each of which run a pair of ripple adder chains in parallel. The N-bit slices of a carry select adder are divided into k blocks of same length. CSA works on the following principle: Two additions are performed in parallel; each block is evaluated conditionally with block carry-in value 0 and 1. When the block carry-in value to a block has been assigned its final value, it is used to select the sum bits from one of the two conditional blocks. At this point in time, the block carry-out can also be evaluated, which in turn selects the sum

bits and carry out of the next block. The 4-bit adder block is RCA. In case of dual RCA based CSA the additional cost of carry select adder over ripple carry adder is the duplicate chain and select logic 25 which is [4(n)]. The critical path for a carry-select adder is the longer of the carry chain in the largest block and the carry select chain. [3] Figure 11 shows the conventional 16-bit regular CSA which consists of two RCAs in each block, one for C_{in} (block) = 0 and the other for C_{in} (block) =1, having four groups of same size. And the size of each group is 4-bit wide.

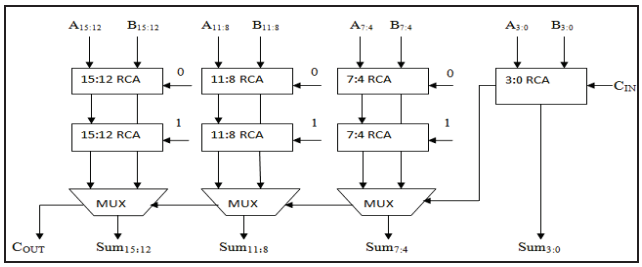

Figure 11. 16-bit Conventional Carry Select Adder (CSA)

In the same way conventional CSA architectures are developed for 8-bit, 16-bit, 32-bit and 64-bit respectively. Conventional CSA is still area-consuming due to the dual ripple carry adder structure.

IV. MODIFIED CARRY SELECT ADDER (MCSA) DESIGN

A Modified Carry Select-Adder (MCSA) design is proposed, which make use of single RCA and Binary to Excess-1 Converter (BEC) instead of using dual RCAs to reduce area and power consumption with small speed penalty. The reason for area reduction is that, the number of logic gates used to design a BEC is less than the number of logic gates used for a RCA design. Thus, 31 importance of BEC logic comes from the large silicon area reduction when designing MCSA for large number of bits. The MCSA architecture for 16-bit is shown in Figure 12.

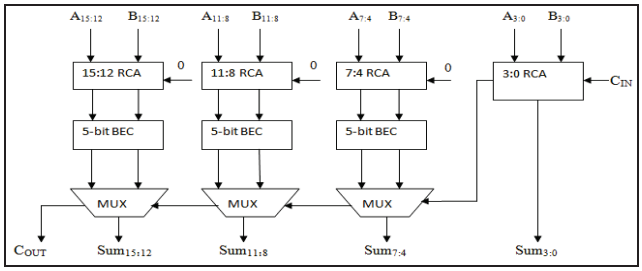

Figure 12. 16-bit Modified Carry Select Adder (MCSA)

As to replace the N bit RCA, an N+1 bit BEC is used, so in above designed architecture of MCSA, the 4-bit RCA is used in each block and thus the BEC used is of 5-bit wide. In the similar way MCSA architectures are designed for 8-bit, 16-bit, 32-bit and 64-bit. Conventional CSA is still area-consuming due to the dual ripple carry adder structure. The metrics for measurement of performance of a circuit are area, power and delay. So, after designing MCSA for 8-bit, 16-bit, 32-bit and 64-bit its area, power and delay are analyzed. The results so obtained are then compared with the results of conventional CSA.

V. VERIFICATION AND SYNTHESIS

The complete Design is modeled in Pure VHDL. The syntax of the RTL design is checked using Xilinx tool. For functional verification of the design is modeled in Hardware descriptive language. The design is verified both at a block level and top level Design. Test cases for the block level are generated in VHDL by both directed and random way.

The complete design along with all timing constraints, area utilization and optimization options are described using synthesis report. The adder design is synthesized at Spartan-3 (XC3S200-5ft256), 90nm process technology.

A. *AREA Utilization Analysis*

The area utilization summary of the 8-bit, 16-bit, 32-bit and 64-bit CSA and MCSA is briefly shown in Table 2 and Table 3. The area synthesis report shows that with the Modified CSA has lesser area as compared with CSA. The graphical representation of the area utilization is shown in Figure 13. It can be seen that in MCSA number of slices and 4 input LUTs are lesser than that of conventional CSA.

TABLE 2 AREA UTILIZATION SUMMARY OF CONVENTIONAL CSA

Area utilization	CSA			
	8-bit	*16-bit*	*32-Bit*	*64-bit*
No. of Slices	12	27	57	117
No. of 4 I/P LUT's	21	47	99	203
No. of Bonded IOBs	28	52	100	196
No. of GCLKs	1	1	1	1

TABLE 3 AREA UTILIZATION SUMMARY OF MODIFIED CSA

Area utilization	MCSA			
	8-bit	*16-bit*	*32-Bit*	*64-bit*
No. of Slices	10	21	44	88
No. of 4 I/P LUT's	18	39	81	163
No. of Bonded IOBs	28	52	100	196
No. of GCLKs	1	1	1	1

The resultant values of area utilization (number of slices LUT's) of modified carry select adder shows better results because the basic programmable part of FPGA is the slice that contain four LUTs (Look up tables) and eight flip flops. Some of the slice can use their LUT's as distributed RAM.

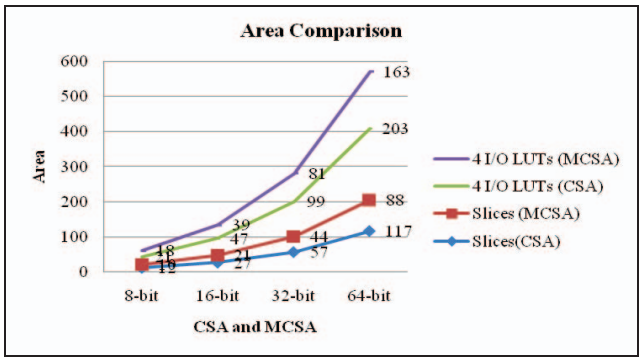

Figure 13. Graphical representation shows area comparison between conventional and modified CSA

B. *Performance Analysis*

The performance of proposed MCSA is analyzed and compared against the conventional CSA designs. The number of gates used in the design indicates the area of design. The power consumption is measured in terms of

total power and dynamic power. The speed of the adder is estimated by the delay involved in the design.

TABLE 4 AREA UTILIZATION SUMMARY OF CONVENTIONAL CSA

Parameters	CSA			
	8-bit	*16-bit*	*32-Bit*	*64-bit*
Max. Attainable Frequency(MHz)	69.14	50.47	32.76	19.25
Delay (ns)	14.462	19.813	30.517	51.924
Power Consumption(mW)	94.48	94.63	95.01	95.49
Area/Gates (AOI)	200	480	1040	2160

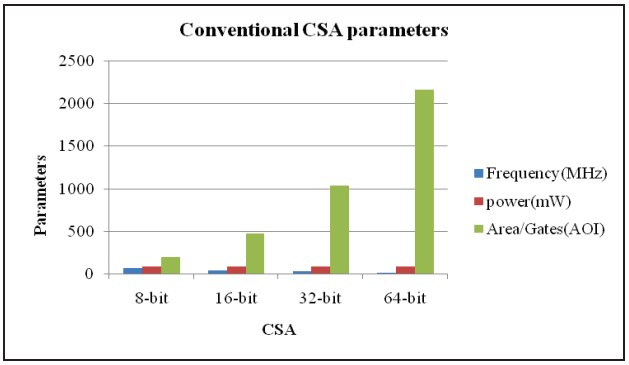

Figure 14. Graphical representation shows performance comparison of conventional CSA

Table 4 gives a comparison of the area, power consumption and delay of CSA for 8-bit, 16-bit, 32-bit and 64-bit. The graphical representation of performance parameters are shown in Figure 14.

Table 4 and 5 gives a clear comparison of the area, power consumption and delay of CSA and MCSA for 8-bit, 16-bit, 32-bit and 64-bit. It can be seen from Table 5 that area and power consumption of MCSA is less than that of conventional CSA, whereas delay is more in MCSA. This shows that area and power consumption of MCSA is reducing at the cost of small decrease in speed. As the number of gates used in the design of MCSA are fewer than the conventional CSA. The reduced number of gates of the MCSA offers a great advantage in the reduction of area and total power consumption.

The graphical representation of MCSA performance parameters are shown in Figure 15.

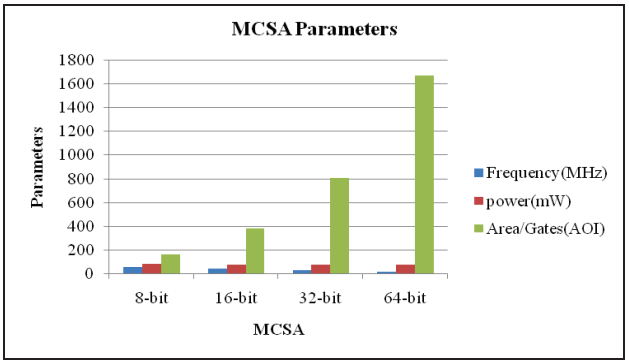

Figure 15. Graphical representation shows performance comparison of modified CSA

The comparison results of CSA and MCSA shows that there is a tradeoff between maximum clock frequency and power consumptions. As the frequency increases the power consumption decreases.

TABLE 5 AREA UTILIZATION SUMMARY OF CONVENTIONAL CSA

Parameters	MCSA			
	8-bit	*16-bit*	*32-Bit*	*64-bit*
Max. Attainable Frequency(MHz)	60.12	46.30	30.52	18.51
Delay (ns)	16.63	21.596	32.65	54.01
Power Consumption(mW)	86.92	81.38	79.81	79.25
Area/Gates (AOI)	167	381	809	1665

VI. APPLICATIONS OF ADDERS

Addition is by far the most fundamental arithmetic operation. It has been ranked the most extensively used operation among a set of real-time digital signal processing benchmarks from application-specific DSP to general purpose processors [3]. The other important applications of adders are as follows:

- Data paths in Microprocessors.
- Digital Adders are the core block of DSP processors.
- Extensively used in processing units such as the Arithmetic Logic Unit (ALU's).
- Forming dedicated integer and/or floating-point units.
- In Multiply-accumulate (MAC) structures.
- For address calculation and flag generation purposes in controllers.
- Digital Signal processing.
- High speed Integrated circuits
- In application-specific ICs.

VII. CONCLUSION

We proposed the efficient modified Carry Select Adder (CSA) of 8-bit, 16-bit, 32-bit and 64-bit word length by using a single Ripple Carry Adder (RCA). The selection of ripple carry adder gives the specifications by accurate resource estimation. The high speed carry select adder performs binary addition pervasive in FPGA applications. Modified carry select adder shows performance and resource improvements as compared with conventional carry select adder. The power consumption and area of modified CSA get reduced from 64-bit to 8-bit in comparison with conventional CSA. The frequency of conventional CSA is better than modified CSA. This paper proposes a scheme which reduces the delay, area and power than conventional CSA. The overall improvement in Modified CSA shows better results in terms of area power and delay. Hence, proposed modified CSA is being used for power and area efficient devices.

FUTURE WORK

In this paper, MCSA architecture is designed for 8-bit, 16-bit, 32-bit and 64-bit word size and results are evaluated for parameters viz., area, power and speed. This work can be further extended for higher number of bits. New architectures can be designed and developed to overcome the speed limitation of MCSA architecture. In addition the BEC scheme can also be implemented in Square root Carry Select Adder to reduce area and power.

REFERENCES

[1] Kuldeep Rawat, Tarek Darwish and Magdy Bayoumi, "A low power and reduced area Carry Select Adder", 45th Midwest Symposium on Circuits and Systems, vol.1, pp. 467-470, March 2002.

[2] Youngjoon Kim and Lee-Sup Kim, "64-bit carry-select adder with reduced area", Electronics Letters, vol.37, issue 10, pp.614-615, May 2001.

[3] B.Ramkumar, Harish M Kittur and P.Mahesh Kannan, "ASIC implementation of Modified Faster Carry Save Adder", European Journal of Scientific Research, vol.42, pp.53-58, 2010.

[4] Israel Koren, "Computer Arithmetic Algorithm", A.K .Peters, 2nd edn, 2002.

[5] J. M. Rabaey, "Digital Integrated Circuits- A Design Perspective", New Jersey, Prentice-Hall, 2001.

[6] M.Moris Mano, "Digital Design", Pearson Education, 3rd edition, 2002.

[7] Yajuan He, Chip-Hong Chang and Jiangmin Gu, "An area efficient 64-bit square root Carry-Select Adder for low power applications", IEEE International Symposium on Circuits and Systems,vol.4, pp.4082-4085, May 2005.

[8] Behnam Amelifard, Farzan Fallah and Massoud Pedram, "Closing the gap between Carry Select Adder and Ripple Carry Adder: a new class of low-power high-performance adders", Sixth International Symposium on Quality of Electronic Design, pp.148-152. April 2005.

[9] Yuke Wang, C. Pai, and Xiaoyu Song, ''The design of hybrid Carry-Lookahead/ Carry–Select Adders'', IEEE transaction on Circuits and Systems II: Analog and Digital Processing, vol.49, pp.16-24, January 2002.

[10] Youngjoon Kim and Lee-Sup Kim, "A low power carry select adder with reduced area", IEEE International Symposium on Circuits and Systems, vol.4, pp.218-221, May 2001.

[11] T.-Y. Chang and M.-J. Hsiao,"Carry-Select Adder using single Ripple-Carry Adder", Electronics letters, vol.34, pp.2101-2103, October 1998.

[12] D. J. Kinniment, "An evaluation of asynchronous addition", IEEE transaction on very large scale integration (VLSI) systems, vol.4, pp.137-140, March 1996.

[13] Hiroyuki Morinaka, Hiroshi Makino, Yasunobu Nakase, Hiroaki Suzuki and Koichiro Mashiko, "A 64bit Carry Look-ahead CMOS Adder using Modified Carry Select",Proceeding of IEEE on Custom Integrated Circuits Conference, pp.585-588, May 1995.

[14] June Wang, Zhongde Wang, G.A. Jullien and W.C. Miller, "Area-time analysis of Carry Lookahead Adders using enhanced multiple output domino logic", IEEE International Symposium on Circuits and systems, vol.4, pp.59-62, June 1994.

[15] Akhilesh Tyagi, "A reduced-area scheme for Carry-Select Adders", IEEE transaction on Computers, vol. 42, pp.1163-1170, October 1993.

[16] David Jeff Jackson and Sidney Joel Hannah, "Modelling and Comparison of Adder Designs with Verilog HDL", 25th Southeastern Symposium on System Theory, pp.406-410, March 1993.

[17] Akhilesh Tyagi, "A Reduced Area Scheme for Carry-Select Adders", IEEE International Conference on Computer design, pp.255-258, Sept 1990.

[18] Belle W.Y.Wei and Clark D.Thompson, "Area-Time Optimal Adder Design", IEEE transactions on Computers, vol.39, pp. 666-675, May1990.

[19] Richard P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders", IEEE transactions on Computers, vol.c-31, pp.260-264, March 1982.

[20] O. J. Bedrij, "Carry-Select Adder", IRE transactions on Electronics Computers, vol.EC-11, pp. 340-346, June1962.

[21] Neil H.E.Weste and K.Eshraghian, "Principle Of CMOS VLSI designs: a system perspective", (Addison-Wesley, 1998), 2nd edn, 1998.

[22] Nhon T.Quach and Michael J.Flynn, "High –speed addition in CMOS", IEEE transaction on Computers, Vol.41, pp.1612-1615, December 1992.

[23] Huey Ling, "High-speed binary adder", IBM journal on Research and Development, vol.25, May 1981.

[24] Sarabdeep Singh, Dilip Kumar, "Design of Area and Power Efficient Modified Carry Select Adder", International Journal of Computer Applications, Vol. 33, No. 3, pp.14-18, November 2011.