International Journal of Electrical Electronics & Computer Science Engineering
Volume 6, Issue 2 (April, 2019) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

A Carry Select Adder Design with Improved Performance

S Subha
Department of IT, SITE, VIT, Vellore, India
ssubha@rocketmail.com

Abstract:* Carry select adder is proposed in literature. This adder called traditional adder in this paper consists of ripple carry adders with partial sums calculated for input as the XOR of the inputs. Based on the carryin (C_{in}) at bit-i the sum and carryout are determined. The carryout is either AND or OR of the inputs based on carryin value of zero or one respectively. The proposed model initializes sum follows for any two bits. If the NOR of the inputs is one, the sum is zero. If the NAND of the inputs is zero the sum is zero. Else the sum is one. The carryout is determined based on carryin and input values. If the carryin is one, the sum is negated and the carryout is the OR of the inputs. Else if the carryin is zero, the carryout is the AND of the inputs. The proposed algorithm is simulated using Quartus2 toolkit for 16-bit input. A performance improvement of 6.97% with no change in area and power is observed for chosen parameters compared with traditional model described in the paper

***Keywords:* Boolean Logic, Carry Select Adder, Full Adder, Performance, Universal Gates.**

I. INTRODUCTION

The computer has three parts memory, central processing unit (CPU) and input/output systems. The CPU has arithmetic and logic unit (ALU) and control unit. The ALU performs arithmetic and logic functions. The arithmetic functions are addition, subtraction, multiplication and division. The addition operation involves finding the sum of two n-bit numbers in n-bit processor system. The full adder performs the addition of three bits a, b, cin and provides the sum and carryout (C_{out}). The truth table of full adder is given in Table1. From Table 1 it we derive the following equations.

$$Sum = A \oplus B \oplus C_{in} \tag{1}$$

$$C_{out} = A.B + B.C_{in} + C_{in}.A \tag{2}$$

A ripple carry adder adds two n-bit numbers sequentially bit by bit to give n-bit sum and carryout. Many adder designs with improved performance and energy savings are proposed in literature. Some of these designs are carry save adder, carry select adder, prefix adder etc.

The final sum is adjusted based on input carryin at each bit. The final sum of any bit position is calculated based on the input carry at run time. Carry select adder designs with improved area and energy savings are proposed in literature. The author in [1] proposes a carry select adder using parallel prefix blocks for reduced area. The authors in [2] propose a new carry select adder with sharing. The authors in [3] propose carry select adder that uses XOR to find the sum of the inputs, AND, OR gates to calculate the carryout based on the inputs, and multiplexers.

Table 1. Full Adder Truth Table

Input A	Input B	Input C_{in}	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

The authors in [4] propose carry select adder based on following principle. A carry select adder (CSA) can be implemented by using a single adder block and an add-one circuit instead of using dual adder blocks. The add-one circuit is based on "first" zero detection logic and a few multiplexers. In the modified CSA, one of the n-bit adder blocks is replaced by an add-one circuit consisting of fewer transistors. The authors in [5] propose carry select adder. The authors in [6] propose carry select adder with modification at transistor level to obtain improved area and power consumption The authors in [7] propose carry select adder that use incrementer circuit with modified Boolean expression to achieve performance. The performance analysis shows that the proposed architecture achieves three fold advantages in terms of delay-area-power. The autnors in [8] propose a new circuit to reduce area using add one logic. The authors in [9] propose carry select adder for power and area reduction. The authors in [10] propose carry select adder with single ripple carry adder with new MUX. The authors in [11] propose carry select adder design using the Kogge Stone parallel approach for improved area. A a new MUX design is proposed by authors in [12]. A new carry select adder with energy saving is proposed in [13].

This paper proposes carry select adder with the following logic. The proposed model initializes sum follows for any two bits. If the NOR of the inputs is one, the sum is zero. If the NAND of the inputs is zero the sum is zero. Else the sum is one. The carryout is determined based on carryin and input values. If the carryin is one, the sum is negated and the carryout is the OR of the inputs. Else if the carryin is zero, the carryout is the AND of the inputs. The proposed algorithm is simulated using Quartus2 toolkit for 16-bit input. An area improvement of 26% with comparable power comsumption with performance degradation of 2.36% is observed for the proposed model compared with traditional model.

The rest of paper is organized as follows. Section 2 gives motivation, section 3 gives proposed model, section 4 gives simulation, section 5 conclusion and references.

II. MOTIVATION

The sum of two bits is given by their XOR. The carry of two bits is the AND of the two bits. Consider two four bit numbers A and B. Let A = 1011 B = 0101. Let A =a[4..1] and B = b[4..1]. Consider the carry select adder to add these two numbers as described in [3]. The steps in this adder are as below.

1. Find the individual sums alone for the input. This gives the sum array as 1110.
2. Calculate the carryout of least significant bit as a[1] AND b[1]. This is equal to one. This is the carryin for bit position 2.
3. For i = 2 we have sum[2] = 1 carryin = 1 a[2] = 1 b[2] = 0. As carryin = 1 , sum[2] = !sum[2] = 0 and carryout = a[2] OR b[2] = 1.
4. For i=3 a[3] = 0 b[3] = 1 carryin = 1 sum[3] = 1. As carryin = 1, sum is negated to give sum[3] = 0 and carryout = OR of inputs = 1
5. For i=4 a[4] = 1 b[4] = 0 carryin = 1 and sum[4] = 1. As carryin is one, sum is negated to have sum[4] = 0 carryout = 1
6. The final answer is 10000
7. Stop

In the above logic, carryout is determined from carryin as follows. If carryin is one the carrout is the OR the inputs and if carryin is zero the carryout is the AND of the inputs.

The logic elements required for the above logic is given next. The initialization of sum in step 1 takes four XOR gates. The worst case takes four OR gates and four NOT gates. The AND gates usually occur in place of OR gates for calculation of carryout based on inputs.

Next consider the following algorithm. . Let A be a[4..1] and B be b[4..1]. The input is A = 1011 B = 0101

1. Start
2. The NAND of LSB is one. The sum is initialized to zero. For the LSB+1 bit, the NAND is one, NOR is zero. Hence the sum is one. For LSB+2 the same logic applies and the sum is initialized to one. For MSB again, the sum is initialized to one.
3. The caryin for LSB is zero. Hence ths sum is retained. The carryot is the AND of inputs which is one. For LSB+1 the carryin is the carryout of previous bit which is one. The sum is negated which becomes zero and the carryout is OR of the inputs which is one. For LSB+2 the carryin is one. The sum is negated which becomes zero and the carryout is OR of the inputs which is one. For the MSB the carryin is one. The sum becomes zero and the carryout is OR of the inputs which is one.
4. Hence the answer is 10000
5. Stop

It is assumed that carryout at step-i is the carryiin for step i+1. Decision is based on carryin and input values at each step.

The logic elements in the above algorithm is calculated next. For the input of four bits it takes four NOR and four NAND gates for initialization of sum in the worst case. For each bit in worst case one NOT gate is required and either of one OR gate is required. The AND gate replaces the OR logic if the tempcin is zero.

There is increase in number of logic elements. This is shown in Table 2. But as universal gates are used, the timing would be reduced. This is verified by simulating the proposed algorithm for sixteen bits as shown in Table 3.

Table 2. Comparison of Number of Logic Elements

Logic Gate	Traditional	Proposed
OR	4	4
AND	0	0
NOT	4	4
XOR	4	0
NOR	0	4
NAND	0	4

Table3. Simulation Results of 16bit Input

Parameter	Traditonal	Proposed	% Improvement
Area	31	32	0
Power (Mw)	6.3	6.3	0
Timing (Ns)	9.22	8.577	6.97397

III. PROPOSED MODEL

This section gives the proposed algorithm.

Algorithm **Modified Carry Select Adder**: Consider two n-bit numbers. The following algorithm calculates the sum of these numbers. Let the numbers be A and B.

Let A be a[1], a[2],…,a[n] and B be b[1], b[2],…,b[n]. The sum is the array sum[1..n] and carryout.

```
1.  Start
2.  For each of the n bits do steps 3-5
3.  If the NOR of the inputs is one the sum is zero
4.  If the NAND of the inputs is zero, the sum is zero
5.  Else the sum is one
6.   Put tempcin = 0 for LSB
7.  For i= LSB to MSB do steps 8-10
8.  If the tempcin = 0 put tempcout = AND of the inputs
9.  If the tempcin = 1 put sum[i] = !sum[i] and tempcout = OR of the inputs
10. Set tempcin = tempcout
11. The final result is (tempcin, sum array)
12. Stop
```

The logic elements in the above algorithm is calculated next. For the input of n bits it takes n NOR and n NAND gates for initialization of sum in the worst case. For each bit in worst case one NOT gate is required and either of one OR gate or one AND gate is required. The AND gate replaces the OR logic if the carryin (tempcin) is zero.

IV. SIMULATIONS

The proposed model was simulated using Quartus2 Toolkit. The simulation parameters are listed in Table 4.

Table 4. Simulation Parameters

Parameter	Value
Processor Family	MaxV
Package	FBGA
Pins	324
Speed Grade	Fastest
Device Selected	Auto by Fitter

The proposed model described in algorithm Modified Carry Select Adder is compared with the traditional carry select model. The results are shown in Table 3. As seen from the Table3 performance improvement of 6.97% with no change in area and power is observed.

V. CONCLUSION

A carry select adder design is proposed in this paper. The proposed model initializes sum for any two bits. If the NOR of the inputs is one, the sum is zero. If the NAND of the inputs is zero the sum is zero. Else the sum is one. The carryout is determined based on carryin and input values. If the carryin is one, the sum is negated and the carryout is the OR of the inputs. Else if the carryin is zero, the carryout is the AND of the inputs. The proposed algorithm is simulated using Quartus2 toolkit for 16-bit input. A performance improvement of 6.97% with no change in area and power is observed for chosen parameters compared with traditional model described in the paper.

VI. REFERENCES

[1] A Tyagi , A Reduced Area Scheme for Carry Select Adder, IEEE Transactions on Computers, Vol. 42, pp. 1163-1170, October 1993.

[2] B. Amelifard, F. Fallah and M. Pedram (2005), 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.

[3] I-Chyn Wey, Cheng-Chen Ho, Yi-Sheng Lin, Chien-Chang Peng (2012), An Area-efficient carry select adder design by sharing the common boolena terms, Proc. of the International Multiconference of Engineers and Computer Scientists, 2012, pp. 1091-1094.

[4] K. Rawat, T. Darwish, and M. Bayoumi , A low power and reduced area carry select adder, The 45th Midwest Symposium on Circuits and Systems., vol. 1, 2002, pp. 467-470.

[5] M. Vinod Kumar Naik , Mohammed Aneesh Y , Design of carryselectadder for low-power and high speed VLSI applications, IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), 2015, pp. 1 – 4**.**

[6] R Hemima. ; Gnana Suji.C Chrisjin , Design of 4 bit low power carry select adder, International Conference on Signal Processing, Communication, Computing and Networking Technologies , 2011, pp.685 – 688.

[7] Samiappa Sakthikumaran1, S. Salivahanan, V. S. Kanchana Bhaaskaran, V. Kavinilavu, B. Brindha and C. Vinoth , A Very Fast and Low Power Carry Select Adder Circuit, 3rd International Conference on Electronics Computer Technology, Volume: 1, 2011, pp. 273 – 276**.**

[8] Sun yan, zhang xin, Jin xi, Low-power Carry Select Adder Using Fast All-one Finding Logic, Institute of Microelectronics Department of Physics, USTC, Hefei, Anhui, 230026.

[9] T. Abhiram ; T. Ashwin ; B. Sivaprasad ; S. Aakash ; J. P. Anita , Modified carryselectadder for power and area reduction International Conference on Circuit ,Power and Computing Technologies (ICCPCT) , 2017, pp. 1-8.

[10] T.Y. Chang, M.J. Hsiao(1998), Carry-select adder using single ripple-carry adder, Electronic Letters, Vol. 34, pp. 2101-2103, October 1998.

[11] U. Sajesh Kumar ; K. K. Mohamed Salih ; K. Sajith , Design and implementation of CarrySelectAdder without using multiplexers, pp. 1-5.

[12] Youngioon Kim and Lee-Sup Kim (2001), 64-bit carry-select adder with reduced area, Electronics Letters, vol. 37 No. 10, pp. 614-615, May 2001.

[13] Y. Chen, H. Li. Roy and K. C. Kok , Cascaded carry-select adder: a new structure CSA for low-power design, Proceedings of the 2005 International Symposium on Low Power Electronics and Design, 2005, pp. 115 - 118.