

Area Compressor and Carry Select Adder with Half-Sum and Half-Carry Method

P.Dhana Laxmi, B.Laxmi Supraja, D.Uma Maheswari

Assistant Professor in ECE Department in Sree Vahini Institute of Science and Technology, Tiruvuru

ABSTRACT: The basic idea of this work is to use half sum and half carry method instead of ripple carry adder (RCA) in the regular CSLA to achieve lower area and power consumption. The main advantage of this half sum and half carry logic comes from the lesser number of logic gates than the n-bit Full Adder (FA) structure. The work is further extended for an area-efficient carry select adder by sharing the common Boolean logic term. After Boolean simplification, we can remove the duplicated adder cells in the conventional carry select adder. Alternatively, we generate duplicate carry-out and sum signal in each single bit adder cell. By utilizing the half sum and half carry implementation delay can be overcomes the parallel architecture in the conventional carry select adder. In this way, the circuit area and Lut count can be greatly reduced and power delay product of the adder circuit can be also greatly lowered.

KEYWORDS: Low Area, Carry, Adder, Half-sum, Half-carry.

I. INTRODUCTION

The carry-ripple adder is composed of many cascaded single-bit full-adders. The circuit architecture is simple and area-efficient. However, the computation speed is slow because each full-adder can only start operation till the previous carry-out signal is ready. In the carry select adder, N bits adder is divided into M parts. Each part of adder is composed two carry ripple adders with cin_0 and cin_1, respectively. Through the multiplexer, we can select the correct output result according to the logic state of carry-in signal. The carry-select adder can compute faster because the current adder stage does not need to wait the previous stage's carry-out signal. The summation result is ready before the carry-in signal arrives; therefore, we can get the correct computation result by only waiting for one multiplexer delay in each single bit adder. In the carry select adder, the carry propagation delay can be reduced by M times as compared with the carry ripple adder. The carry select adder (CSLA) 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.

The carry-save adder (CSA) avoids carry propagation by treating the intermediate carries as outputs instead of advancing them to the next higher bit position, thus saving the carries for later propagation. The sum is a (redundant) digit carry-save number, consisting of the two binary numbers (sum bits) and (carry bits). A Carry-save adder accepts three binary input operands or, alternatively, one binary and one carry-save operand. It is realized by a linear arrangement of full-adders and has a constant delay.

The logic expression of carry selector given as:

$$2C + S = A_0 + A_1 + A_2$$
$$\sum_{i=1}^n 2^i c_i + \sum_{i=0}^{n-1} 2^i s_i = \sum_{j=0}^2 \sum_{i=0}^{n-1} 2^i a_{j,i}$$
$$2c_{i+1} + s_i = \sum_{j=0}^2 a_{j,i} ; \quad i = 0, 1, \dots, n-1$$

Where,

A0,A1,A2 three binary inputs

C is carry,S is sum,

C_{i+1} is carry out of the ith stage S_i=sum at the ith stag

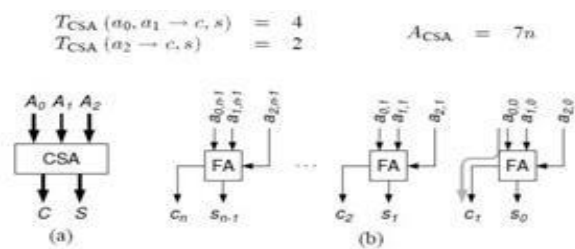
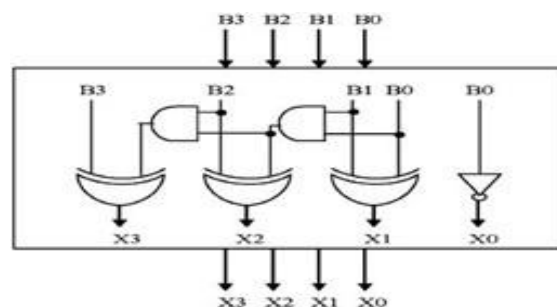


Fig.1: carry and adder

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. The CSLA 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 [1]. However, the CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input and, then the final sum and carry are selected by the multiplexers (mux).

The Square root (SQRT) CSLA has been chosen for comparison with the proposed design as it has a more balanced delay, and requires lower power and area. The AND, OR, and Inverter (AOI) implementation of an XOR gate is shown in Fig. 1. The gates between the dotted lines are performing the operations in parallel and the numeric representation of each gate indicates the delay contributed by that gate. The delay and area evaluation methodology consider all gates to be made up of AND, OR, and Inverter, each having delay equal to 1 unit and area equal to 1 unit. We then add up the number of gates in the longest path of a logic block that contributes to the maximum delay. The area evaluation is done by counting the total number of AOI gates required for each logic block. Based on this approach, the CSLA adder blocks of 2:1 mux, Half Adder (HA), and FA are evaluated and



listed in Table I.

Fig.2: 4-Bit BEC

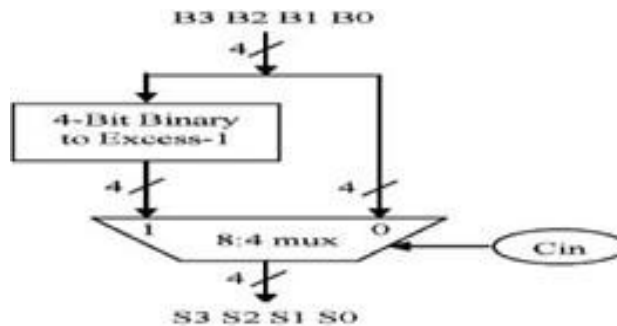


Fig.3: Basic function of the CSLA

As stated above the main idea of this work is to use BEC instead of the RCA with in order to reduce the area and power consumption of the regular CSLA. To replace the 4-bit RCA, an 4-bit BEC is required. A structure and the function table of a 4-b BEC are shown in Fig. 2 and Table II, respectively.

$$X0 = \sim B0$$

$$X1 = B0 \oplus B1$$

$$X2 = B2 \oplus (B0 \& B1)$$

$$X3 = B3 \oplus (B0 \& B1 \& B2).$$

illustrates how the basic function of the CSLA is obtained by using the 4-bit BEC together with the mux. 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. This produces the two possible partial results in parallel and the mux is used to select either the BEC output or the direct inputs according to the control signal Cin. The importance of the BEC logic stems from the large silicon area reduction when the CSLA with large number of bits are redesigned.

II.IMPLEMENTATION

A ripple carry adder (RCA) uses an easy style, however carries Propagation delay (CPD) is additional during this adder. Carry look-ahead and carry choose (CS) strategies are prompt to scale back the CPD of adders. a traditional carry choose adder (CSLA) is associate degree RCA configuration that generates a try of add words associate degreeed output-carry bits cherish carry inputs (Cin =0 and Cin = 1) and selects one out of every try for final sum and final-output-carry victimization the management signal Cin' a traditional CSLA has less CPD than an RCA. Few tries are created to avoid twin use of RCA in CSLA style. Kim and Kim used one RCA and one add-one circuit rather than 2 RCAs, wherever the add-one circuit is enforced employing a electronic device (MUX). Chang planned a square-root (SQRT)-CSLA to implement giant bit- width adders with less delay in an exceedingly SQRT CSLA, CSLAs with increasing size square measure connected in an exceedingly cascading structure.

The most objective of SQRT-CSLA style is to produce a parallel path for carry propagation that helps to scale back the adder delay. Ramkumar and Kittur prompt a binary to BEC-based CSLA. The BEC-based CSLA involves less logic resources than the standard CSLA, however it's marginally higher delay. A CSLA supported common Boolean algebra (CBL) is additionally planned. The CBL-based CSLA involves considerably less logic resource than the standard CSLA however it's longer CPD, that is nearly adequate that of the RCA. to beat this drawback, a SQRT-CSLA supported CBL was planned.

However, the CBL-based SQRT-CSLA model needs additional logic resource and delay than the BEC based SQRT-CSLA. Logic improvement mostly depends on handiness of redundant operations within the formulation,

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Special Issue , August 2019

International Conference on Recent Advances in Science, Engineering, Technology and
Management at Sree Vahini Institute of Science and Technology-Tiruvuru, Krishna Dist, A.P

wherever as adder delay principally depends on knowledge offered at the input. Within the existing style, logic is improved while not giving any thought to the information dependence. analysis created on logic operations concerned in typical and BEC-based CSLAs to check the information dependence and to spot redundant logicoperationssupported this analysis, a logic formulation planned for the CSLA.

The most contribution during this is logic formulation supported knowledge dependence and optimized carry generator (CG).

A. CONVENTIONAL CSLA AND ITS LOGIC FORMULATION:

The conventional CSLA consists of one sum and carry generation unit and sum and carry selection unit as shown in Fig. 4. Sum and carry generation unit can be composed of two ripples carry adders one with carry input zero and other with carry input one as shown in fig.4. Where n is the adder bit width. An n-bit RCA can be composed using half-sum generator (HSG), half- carry generator (HCG), full-sum generator (FSG), full- carry generator (FCG) (shown in fig.4). The RAC-I and RAC-2 generates n-bit sum (S_0 and S_1) and carry out ($C0_{out}$ and $C1_{out}$) corresponds to input-carry ($C_{in}=0$ and $C_{in}=1$) respectively.

Logic expression of RAC-I are givenas,

$$S^0_0(i) = A(i) \oplus B(i) \quad (1a)$$

$$C^0_0(i) = A(i).B(i) \quad (1b)$$

$$S^0_1(i) = s^0_0(i) \oplus c^0_1(i-1) \quad (1c)$$

$$C^0_1(i) = c^0_0(i) + s^0_0(i).c^0_1(i-1) \quad (1d)$$

$$C^0_{out} = c^0_1(n-1) \quad (1e)$$

Logic expression of RCA-2 are givenas,

$$S^1_0(i) = A(i) \oplus B(i) \quad (2a)$$

$$C^1_0(i) = A(i) . B(i) \quad (2b)$$

$$S^1_1(i) = s^1_0(i) \oplus c^1_1(i-1) \quad (2c)$$

$$C^1_{out} = c^1_1(n-1) \quad (2e)$$

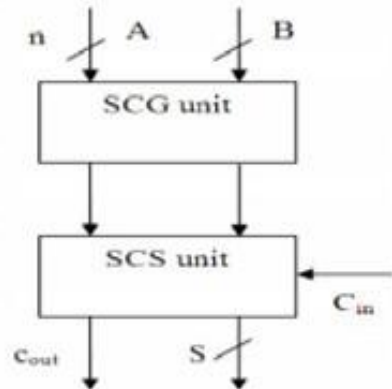


Fig.3.structure of conventional CSLA

Here, $s_o(i)$, $c_o(i)$, $s_o(i)$, $c_o(i)$ are out of (HSG), HCG, FSG, FCG respectively and c_{out} final carry-out of n-bit RAC-1 $s_o(i)$, $c_o(i)$, $s_o(i)$, $c_o(i)$ are out of HSG, HCG, FSG FCG. From the above logic expression, it is clear that (1a)-(2a) and (1b)-(2b) are identical. The HSG and HCG units can be removed from RCA-2 to have an optimized design for CSLA; here the HSG and HCG of RAC-1 can be shared to construct RCA-2. Based on this have used an add-one circuit instead of RCA-2 for the design of CSLA. A BEC based circuit is used to construct CSLA.

III. PROPOSED CSLA DESIGN

The proposed CSLA structure is as shown in Fig.4. It is composed of one half-sum generation (HSG) unit, one fullsum generation (FSG) unit, one carry- generation (CG) unit, and one carry-selection (CS) unit. The CG unit composed of two units namely CG0 and CG1 Corresponding to input-carry '0' and '1', respectively. Input to the HSG unit is two n-bit operands A and B and outputs are half-sum (HS) word s_o and half-carry (HC) word c_o of width n-bit each. CG unit receives both s_o and c_o from HSG unit and gives two n-bit full-carry words c_o , and c_1 , corresponds to carry-input '0' and '1', respectively. The carry selection unit selects final carry based on the C_{in} from two anticipated carry words C_o and c_1 . If $C_{in} = 0$ then it selects C_o ; otherwise it selects c_1 . C_{out} is the MSB of c_1 and remaining (n-1) LSBs of C_o are XORed with (n-1) MSBs of half-sum (s_o) in the FSG unit to obtain final-sum. The proposed logic formulation for the CSLA is given by

$$s_o(i) = A(i) \oplus B(i) \quad (4a)$$

$$c_o(i) = A(i).B(i) \quad (4b)$$

$$c_1^0(i) = c_1^0(i-1).s_o(i) + c_o(i) \text{ for } (c_1^0(0) = 0) \quad (4c)$$

$$c_1^1(i) = c_1^1(i-1).s_o(i) + c_o(i) \text{ for } (c_1^1(0) = 1) \quad (4d)$$

$$C(i) = c_1^0(i) \text{ if } (c_{in} = 0) \quad (4e)$$

$$C(i) = c_1^1(i) \text{ if } (c_{in} = 1) \quad (4f)$$

$$C_{out} = c_1(n-1) \quad (4g)$$

A.PERFORMANCE COMPARISON BETWEEN DIFFERENT TYPES OF CSLAS:

Area and delay of the overall design is calculated by using the following relations:

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Special Issue , August 2019

International Conference on Recent Advances in Science, Engineering, Technology and
Management at Sree Vahini Institute of Science and Technology-Tiruvuru, Krishna Dist, A.P

$$\text{Area} = n.N_n + o.N_o + a.N_a \quad (5a)$$

$$\text{Delay} = n_n.T_n + n_o.T_o + n_a.T_a \quad (5b)$$

Where (Tn' To, Ta) and (n, o, a), represents the delay and area of (NOT, OR, AND) gates respectively. (N", No, Na) and (nn, no, n.) represents number of gate count and critical path of (NOT, OR, AND) gates of the total design. In this design each gate is made by using 2-input AND, 2-input OR, and NOT. Two inputs XOR is composed of two AND, one OR, and two NOT gates. Area and delay of AND, OR, NOT gates which is taken from the synopsis Armenia Educational Department 90-nm standard cell library datasheet for theoretical estimation of area and delay of the design (shown in table-I).

Each gate in the design is realized by using AND-OR- NOT gates. 2-input XOR gate is realized by using two 2-input AND gates, two NOT gates and one OR gate. Area-Power product (APP) and Area-Delay-Power product of conventional CSLA, BEC-based CSLA, CBL-based CSLA. From that graph it is clear that ADP of CBL-based CSLA is much more than other type of CSLA, as number of bits increased ADP of CBL-based CSLA differ more from its counterpart. ADPP of Conventional CSLA is increased more than that of Prop. CSLA

IV. RESULTS

Existing RTL Proposed model

Design Statistics	
# IOs	: 26
Cell Usage :	
# BELS	: 42
# INV	: 7
# LUT2	: 35
# FlipFlops/Latches	: 43
# FD	: 17
# FDR	: 7
# FDS	: 19
# Clock Buffers	: 1
# BUFGP	: 1
# IO Buffers	: 25
# IBUF	: 17
# OBUF	: 8

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Special Issue , August 2019

International Conference on Recent Advances in Science, Engineering, Technology and
Management at Sree Vahini Institute of Science and Technology-Tiruvuru, Krishna Dist, A.P

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	25	5888	0%
Number of Slice Flip Flops	43	11776	0%
Number of 4-input LUTs	42	11776	0%
Number of bonded IOBs	26	372	6%
Number of GCLKs	1	24	4%

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	18	5888	0%
Number of 4-input LUTs	31	11776	0%
Number of bonded IOBs	9	372	13%

Design Statistics

IOs : 49

Cell Usage :

BELS : 32

LUT3 : 29

LUT4 : 2

MUXF5 : 1

IO Buffers : 49

IBUF : 33

OBUF : 16

Comprasion Table:

	Proposed	Existing
No of slices	18	25
No of 4-LUT	31	43
No of flipflops	0	42
Delay(ns)	22	53



ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 6, Special Issue , August 2019

**International Conference on Recent Advances in Science, Engineering, Technology and
Management at Sree Vahini Institute of Science and Technology-Tiruvuru, Krishna Dist, A.P**

REFERENCES

1. A. P. Chandrakasan, N. Verma, and D. C. Daly, "Ultralow-power electronics for biomedical applications," *Annu. Rev. Biomed. Eng.*, vol. 10, pp. 247–274, Aug. 2008.
2. O. J. Bedrij, "Carry-select adder," *IRE Trans. Electron. Comput.*, vol. EC-11, no. 3, pp. 340–344, Jun. 1962.
3. Y. Kim and L.-S. Kim, "64-bit carry-select adder with reduced area," *Electron. Lett.*, vol. 37, no. 10, pp. 614–615, May 2001.
4. Y. He, C. H. Chang, and J. Gu, "An area-efficient 64-bit square root carryselect adder for low power application," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2005, vol. 4, pp. 4082–4085.
5. B. Ramkumar and H.M. Kittur, "Low-power and area-efficient carry-select adder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 2, pp. 371–375, Feb. 2012.
6. I.-C. Wey, C.-C. Ho, Y.-S. Lin, and C. C. Peng, "An area-efficient carry select adder design by sharing the common Boolean logic term," in *Proc. IMECS*, 2012, pp. 1–4.
7. S. Manju and V. Sornagopal, architecture of carry select adder design by common Boolean logic," in *Proc. VLSI ICEVENT*, 2013, pp. 1–5.
8. B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.
9. B.K. Mohanty, S K Patel "Area –Delay - Power Efficient Carry Select Adder" *IEEE Transactions on circuits and systems – II*, Vol 61, No. 6 2014, pp-418-422.