International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.4, No.2, February 2015 DOI:10.15693/ijaist/2015.v4i2.41-45

Design and Analysis of a Digital Clock Using Carry Select Adder

S.JahabarSathic, *Student, National College of Engineering,*.
K.Murugan, Assistant Professor, *National College of Engineering*.

***Abstract*—The logic operations involved in conventional carry select adder (CSLA) and binary to excess-1 converter (BEC)-based CSLA to study the data-dependency and to identify redundant logic operations. Eliminated all the redundant logic operations present in conventional CSLA and proposed a new logic formulation for CSLA. In the proposed scheme, the carry-select operation is scheduled before the calculation of final-sum, which is different from the conventional approach. Bit-patterns of two anticipating carry-words (corresponding to Cin=0 and 1) and fixed Cin bits are used for logic optimization of carry select and generation units. An efficient CSLA design is obtained using optimized logic units. The proposed CSLA design involves significantly less area and delay than the recently proposed BEC-based CSLA. Due to small carry-output delay, the proposed CSLA design is a good candidate for square-root (SQRT)-CSLA.**

***Index Terms*—CLSA (carry select adder), BEC (binary to excess converter),SQRT (square root carry select adder).**

I. INTRODUCTION

Highspeed data path logic systems are one of the most substantial areas of research in VLSI system design. The speed of addition is limited by the time required to propagate a carry through the adder in digital adders. 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 the CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry. Design of high speed data path logic systems are one of the most substantial research area in VLSI system design. High-speed addition and multiplication has always been a fundamental requirement of highperformance processors and systems.

The major speed limitation in any adder is in the production of carries and many authors have considered the addition problem. The basic idea of the proposed work isusing n-bit binary to excess-1 code converters (BEC) to improve the speed of addition. The detailed structure and function of BEC. This logic can be implemented with any type of adder to further improve the speed. The proposed 16, 32 and 64-bit adders are compared in this paper with the conventional fast

The improved performance of the CSA with BEC logic through custom design and layout. The final stage CPA constitutes a dominant component of the delay in the parallel multiplier. Signals from the multiplier partial products summation tree do not arrive at the final CPA at the same time. This is due to the fact that the number of partial-product bits is larger in the middle of the multiplier tree. Due to un-even arrival time of the input signals to the final CPA, the selection of the ASIC Implementation of Modified Faster Carry Save Adder 54 final adder is an important work in parallel multipliers. Therefore decrease in carry propagation delay will result in major enhancement of the speed of the adder and multiplier. An overview of the 4-bit binary to excess-1 logic is provided. Section 3 deals with the proposed modified carry save adder (MCSA) architecture.

Among the myriad of aggressive techniques, carry select adder (CSLA) has been an eminent technique in the space-time tug-of-war of CPA design. It exhibits the advantage of logarithmic gate depth as in any structure of the distant-carry adder family. Conventionally, CSLA is implemented with dual ripple-carry adder (RCA) with the carry-in of 0 and 1, respectively. Depending on the configuration of block length, CSLA is further classified as either linear or square root. The basic idea of CSLA is anticipatory parallel computation. Although it can achieve high speed by not waiting for the carry-in from previous sub-block before computation can begin, they consume more power due to doubling the amount of circuitry needed to do the parallel addition of which half of the speculative computations will be redundant. Digital Adders are the core block of DSP processors.

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 in reference1. 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 Cin=0 and Cin=1, then the final sum

adders such as carry save adder (CSA) and carry look ahead adder.

and carry are selected by the multiplexers (MUX). The basic idea of this work is to use Binary to Excess-1 converter (BEC) instead of RCA with Cin=1 in the regular CSLA to achieve lower area and power consumption in ref 2-4 The main advantage of this BEC logic comes from the lesser number of logic gates than the n-bit Full Adder (FA) structure. The details of the BEC logic are discussed. Carry-ripple adder (CRA) is the simplest approach.

However, the carrylook ahead adder (CLA) and its fast version, the parallel prefix CLA, is the selected scheme for time critical applications with a considerable cost in terms of silicon area and power dissipation. The CSA provides a compromise between a RCA and a CLA adder. Due to rapidly growing system on chip industry, not only the faster units but also smaller area and less power has become a major concern for designing very large scale integration (VLSI) circuits. Digital circuits make use of digital arithmetic's. Among various arithmetic operations, multiplication is one of the fundamental operation used and is being performed by an added. There are many ways to build a multiplier each providing tradeoff between delays and other characteristics, such as area and energy dissipation.

II. CARRY SELECT ADDER

The carry select adder generally consists of two ripple carry adders and a multiplexer. Adding two n-bit numbers with a carry select adder is done with two adders (therefore two ripple carry adders) in order to perform the calculation twice, one time with the assumption of the carry being zero and the other assuming one. After the two results are calculated, the correct sum, as well as the correct carry, is then selected with the multiplexer once the correct carry is known. The carry select adder partitions the adder into several groups, each of which performs two additions in parallel. Therefore, two copies of ripple-carry adder act as carry evaluation block per select stage. One copy evaluates the array chain assuming the block carry in is zero, while the other assumes it to be one.

The conventional n bit CSLA consists of one n/2-bit adder for the lower half of the bits and two n/2-bit adders for the upper half of the bits. Of the two latter adders, one performs the addition with the assumption that Cin=0, whereas the other does this with the assumption that Cin=1. Using a multiplexer and the value of carry out that is propagated from the adder for the n/2 least significant bits, the correct value of the most significant part of the addition can be selected.

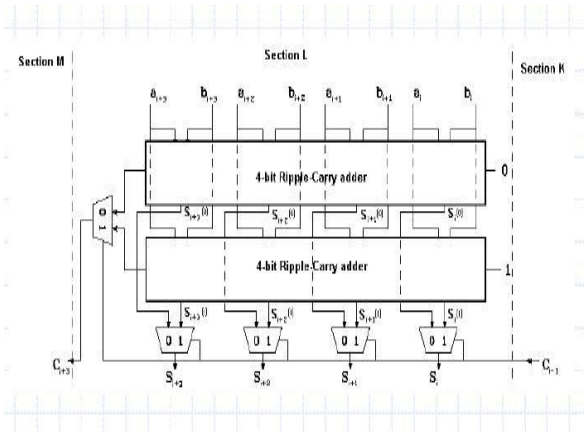

Fig1 Carry Select Adder

Although this technique has the drawback of increasing the area, it speeds up the addition operation. A 16-bit carry select adder with a uniform block size of 4 can be created with three of these blocks and a 4-bit ripple carry adder. Since carry-in is known at the beginning of computation, a carry select block is not needed for the first four bits. The delay of this adder will be four full adder delays, plus three MUX delays. Two 4-bit ripple carry adders are multiplexed together, where the resulting carry and sum bits are selected by the carry-in. Since one ripple carry adder assumes a carry-in of 0, and the other assumes a carry-in of 1, selecting which adder had the correct assumption via the actual carry-in yields the desired result.

The carry select adder comes in the category of conditional sum adder. Conditional sum adder works on some condition. Sum and carry are calculated by assuming input carry as 1 and 0 prior the input carry comes. When actual carry input arrives, the actual calculated values of sum and carry are selected using a multiplexer.

III. PROPOSED CARRY SELECT ADDER

A. Proposed Carry Select Adder

The proposed CSLA is based on the logic formulation given in bellow Fig, and its structure is shown in Fig.3.1 it consists of one HSG unit, one FSG unit, one CG unit, and one CS unit. The CG unit is composed of two CGs (CG0 and CG1) corresponding to input-carry '0' and '1'. The HSG receives two n-bit operands (A and B) and generate half-sum word s0 and half-carry word c0 of width n bits each. Both CG0 and CG1 receive s0 and c0 from the HSG unit and generate two n-bit full-carry words c01 and c11 corresponding to input-carry '0' and '1', respectively.

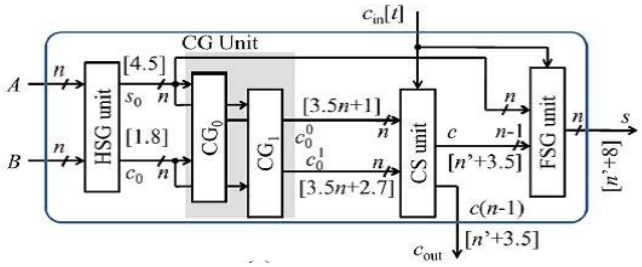

Fig2 Proposed CS Adder Design

B. HSG Unit:

The logic diagram of the HSG unit is shown in Fig.3 It consist of Combinational Logical AND Gate and OR Gate.

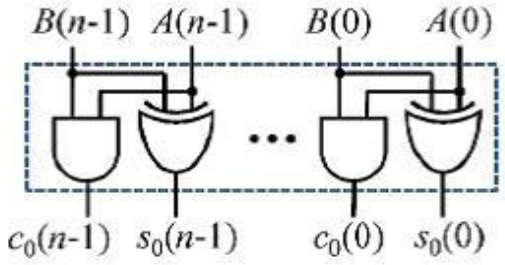

Fig3 HSG Unit

C.CG0 & CG1 Unit:

The logic circuits of CG0 and CG1 are optimized to take advantage of the fixed input-carry bits. The optimized designs of CG0 and CG1 are shown in Fig.4 It Consist of logical or and logical and gates.

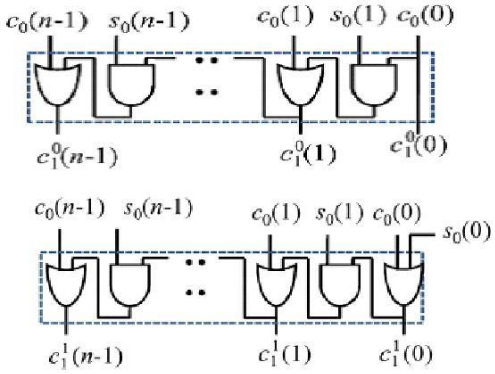

Fig4 CG0 & CG1 Unit

D. CS Unit:

CS Unit is referred as Carry Select unit and it consist of combination of logical OR and Logical AND gates. The CS unit selects one final carry word from the two carry words available at its input line using the control signal cin. It selects c01 when cin = 0; otherwise, it selects c11. The CS unit can be implemented using an n-bit 2-to-l MUX. However, we find from the truth table of the CS unit that carry words c01 and c11 follow a specific bit pattern. If c01 (i) = '1', then c11 (i) = 1, irrespective of s0(i) and c0(i), for $0 \leq i \leq n-1$. This feature is used for logic optimization of the CS unit. The optimized design of the CS unit is shown in Fig. 3(e), which is composed of n AND–OR gates. The final carry word c is obtained from the CS unit. The MSB of c is sent to output as cout, and (n − 1) LSBs are XORed with (n − 1) MSBs of half-sum (s0) in the FSG [shown in Fig.3.4] to obtain (n − 1) MSBs of final-sum (s). The LSB of s0 is XORed with cin to obtain the LSB of s.

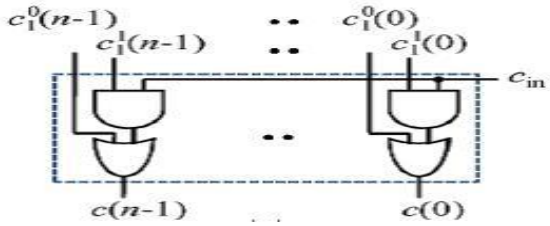

Fig5 CS Unit

E. FSG Unit:

FSG Unit Referred as final-sum generation unit. It consists of Combination of logical XOR gates. It performs Sum operation.

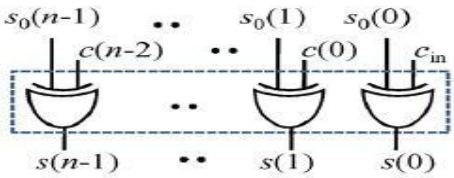

Fig6 FSG Unit

IV. SIMULATION RESULTS AND ANALYSIS

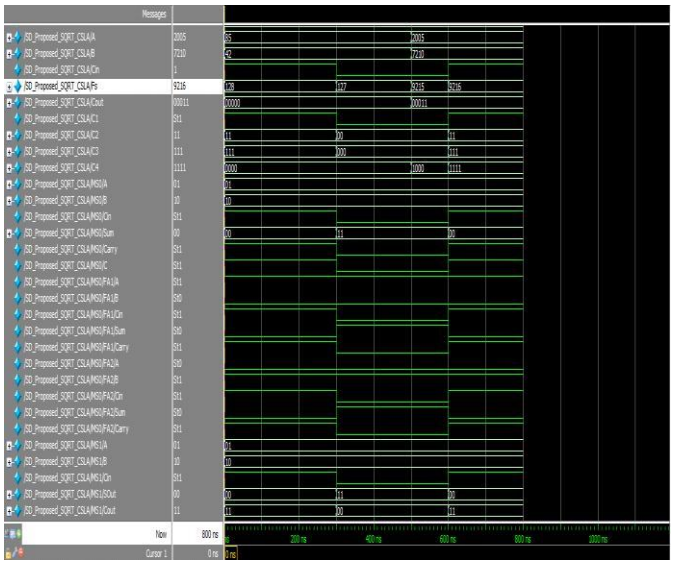

Fig6 Proposed SQRT CSLA n=16

Snapshot is nothing but every moment of the application while running. It gives the clear elaborated of application. It will be useful for the new user to understand for the future step. In this snapshot represented by proposed square root carry select adder simulation using sixteen bits. The proposed carry select adder design to four modules. Half Generation Unit (HSG), Carry Generation Zero and Carry Generation One unit (CG0&CG1) , Carry Select Unit ,Final Sum Generation Unit(FSG). The input for A, B, Cin and output for Sum and carry.

V. DIGITAL CLOCK

we have built a digital clock with 12 hour count time. The clock runs from 00:00 to 11:59 and then back to 00:00. Our display has four digits, two digits for minutes and two for hour. The specialty of this clock is that it has very low power consumption and condensed layout.

We know that 60 seconds equal to 1 minute and 60 minutes equal to 1 hour. Hence the minute section is drived by second section and hour section by the minute section. Each of the minute and second section has been designed to give a count from 00 to 59 after which it resets to 00. and the hour section to give a count from 00 to 11 hours after which it resets to 00. For each cycle of 00 to 59 in second section the minute section increases its count by 1. Similarly for each cycle of 00 to 59 in minute section the hour section increases it's count by 1. In this way when the clock reaches 11hrs. 59mins. 59secs. each of the section resets to 00 giving us a display 00.00.00 popularly known as the 0th hour.

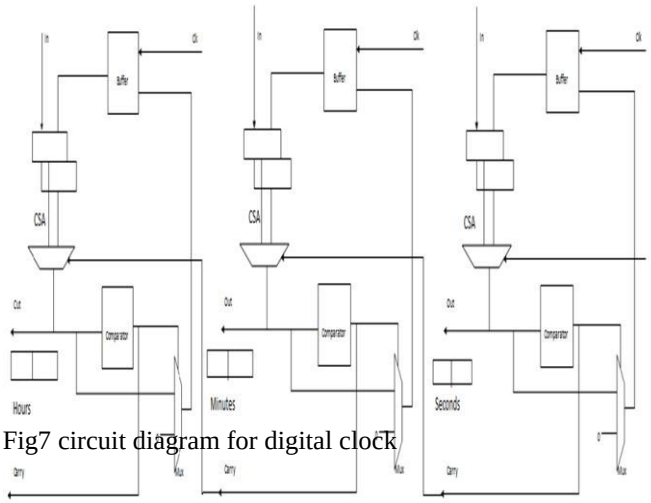

Fig7 circuit diagram for digital clock

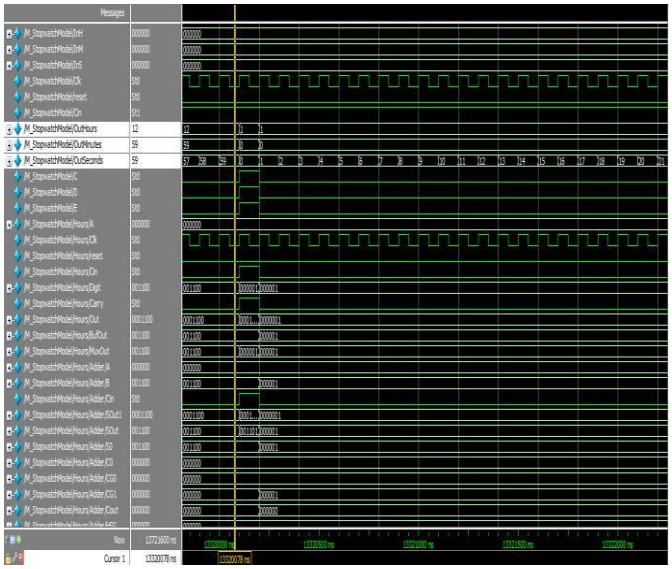

Fig8 Digital clock output

VI. CONCLUSION

The logic operations involved in the conventional and BEC-based CSLAs to study the data dependence and to identify redundant logic operations. Eliminated all the redundant logic operations of the conventional CSLA and proposed a new logic formulation for the CSLA. The CS operation is scheduled before the calculation of final-sum, which is different from the conventional approach. Carry words corresponding to input-carry '0' and '1' generated by the CSLA based on the proposed scheme follow a specific bit pattern, which is used for logic optimization of the CS unit. Fixed input bits of the CG unit are also used for logic

optimization. Based on this, an optimized design for CS and CG units are obtained. Using these optimized logic units, an efficient design is obtained for the CSLA. The proposed CSLA design involves significantly less area and delay than the recently proposed BEC-based CSLA. Due to the small carry output delay, the proposed CSLA design is a SQRT adder implemented 16 Bit Proposed adders. In future we can develop converted into 32 bit Adders and analyzed. This adder will be done by using our Verilog.

REFERENCES

[1]. Bedrij O.J, (1962) "Carry-select adder," IRE Trans. Electron.
Comput.vol. EC-11, no. 3, pp.340–344.

[2]. Chandrakasan A.P, Daly D.C, and Verma N, (2008) "Ultralow-power electronics forbiomedical applications," Annu. Rev. Biomed.
Eng., vol. 10, pp. 247– 274.

[3]. Chang C.H, Gu J, and He Y, (2005) "An area-efficient 64-bit square root carry select adderfor low power application," in Proc. IEEE Int. Symp. Circuits Syst., vol. 4, pp. 4082–4085.

[4]. Ho C.C, Lin Y.S, Peng C.C, and Wey I.C, (2012) "An area-efficient carry select adder design by sharing the common Boolean logic term," in Proc. IMECS, pp. 1–4.

[5]. Kim L.-S, and Kim Y, (2001) "64-bit carry-select adder with reduced area," Electron. Lett.,vol. 37, no. 10, pp. 614–615.

[6]. Kittur H.M, and Ramkumar B, (2012) "Low-power and area-efficient carry-select adder," IEEE Trans. Very Large Scale Integr.
(VLSI) Syst., vol. 20, no. 2, pp. 371–375.

[7]. Manju S, and Sornagopal V, (2013) "An efficient SQRT architecture of carry select adder design by common Boolean logic," in Proc. VLSI ICEVENT, pp. 1–5.

[8]. Parhami B, (2010) Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed. New York, NY, USA: Oxford Univ. Press.

[9]. Parhi K.K, (1998) VLSI Digital Signal Processing. New York, NY, USA:Wiley.

Authors Profile

S.JahabarSathic received the **B.E.** degree in electronics and communication engineering from the national College of Engineering, Tirunelveli, Anna University,Chennai, India, in 2013.Currently doing **M.E.** in electronics and communication engineering (Applied Electronics) in National College of Engineering, Tirunelveli, Anna University, Chennai, India. His research interest includes VLSI.

K.Murugan received the **B.E.** degree in electronics and communication engineering from the P.S.N.A College of Engineering and Technology, Dindugal, Anna University, Chennai, India, in 2001, and received the **M.E. degree** in electronics and communication engineering (Applied electronics) in Mohamed Sathak Engineering College, Kilakarai, Anna University, Thirchy, India. His Ph.D work is going on Information and communication, Anna University, Chennai. His research area is VLSI.