A survey report on carry save adder

September 30, 2018

Ritika Kumari
Roll no. 39/CSE/17069/267
Email id: ritikakumari1302@gmail.com

Abstract

This paper presents a technology-independent design and simulation of a modified architecture of the Carry-Save Adder. This architecture is shown to produce the result of the addition fast and by requiring a minimum number of logic gates. Binary addition is carried out by a series of XOR, AND and Shift-left operations. These operations are terminated with a completion signal indicating that the result of the addition is obtained. Because the number of shift operations carried out varies from 0 to n for n-bit addends, a behavioral model was developed in which all the possible addends having 2- to 15-bits were applied. A mathematical model was deducted from the data and used to predict the average number of shift required for standard binary numbers such as 32, 64 or 128-bits. 4-bit prototypes of this adder were designed and simulated in both synchronous and asynchronous modes of operation.

1 Introduction

n digital computer arithmetic, addition and subtraction are the most basic core operations, especially for digital signal processing applications. The other two fundamental operations li ke multiplication and division are too performed using addition and subtraction and hence they play a very important role in processors like microprocessors, microcontrollers and digital signal processors. The fast adders can be used to speed up the ar ithmetic operations in a processor. There are numerous ways available to perform addition but with different trade - offs. Some are good in producing low power at the cost of area and some are best in offering performance (i.e., high speed) at the cost of po wer and/or area. Arithmetic operations are essential building blocks in any system; either the system can be designed based on a processor, or an FPGA/ASIC. The data path circuitries in a microprocessor, Multiply - Accumulate (MAC) operations in a digital signal processor and high speed integrated circuits in communication systems are the few application examples which would perform arithmetic operations especially using regular full adders. The k - bit ripple - carry adder is the most simplest adde r structure, which adds two words having k - bits. The delay of the RCA is very high since the carry is propagated (i.e., rippled) to the full adder stages to produce a final sum. If the value of k is very large, then the delay would be more. These RCAs are not suitable in situations where the speed data processing is involved. So the demand to design fast adders grows rapidly to meet the current high speed integrated circuits trend CSA is a kind of adder with low propagation delay, but instead of adding two input numbers to a single sum output, it adds three input numbers to an output pair of numbers. When its two outputs are then summed by a traditional carry - lookahead or ripple carry adder , we received the sum of all three inputs. In particular, the propagation delay of a CSA is not affected by the width of vectors being added. Each full adders output S is connected to corresponding output bit of one output, and its out put Cout is connected to the next higher output bit of the second output; the lowest bit of the second output is fed directly from the carry - saves Cin input

The carry save addition of 2 N-bit numbers results in two (N + 1)-bit numbers being produced, the virtual carry (VC) and the virtual sum (VS).But after getting VC and VS you still have to add the two values together with a convectional adder to get your final result, so only adding 2 numbers is pointless.Take this example, lets say one carry-save addition takes k*T ms, where k = number of N - bit numbers being added, and a convectional adder takes 5T ms to add 2 numbers (regardless of bit width), then if:

1) you added 2 numbers, then

$Time(Carry - Save) = 2T + 5T = 7T$

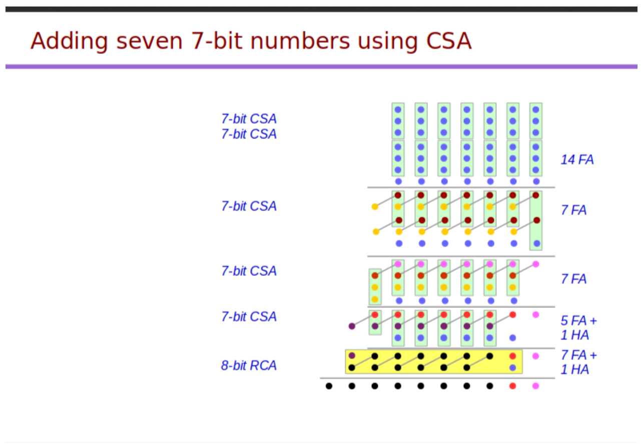
$Time(ConvectionalAdder) = 5T$

2) you added 3 numbers, then

$Time(Carry - Save) = 3T + 5T = 8T$

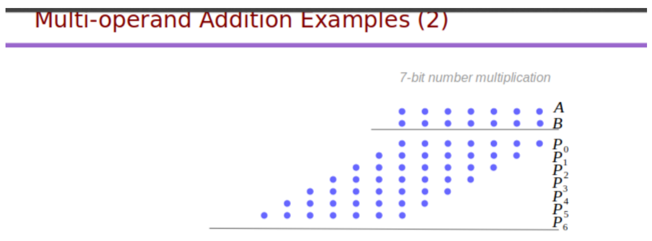
$*Time(ConvectionalAdder) = 5T + 5T = 10T$

So carry-save addition is only useful if you have at least 3 operands to add.


Adding seven 7-bit numbers using CSA

2 MULTI-OPERAND ADDITION USING CARRY SAVE ADDERS

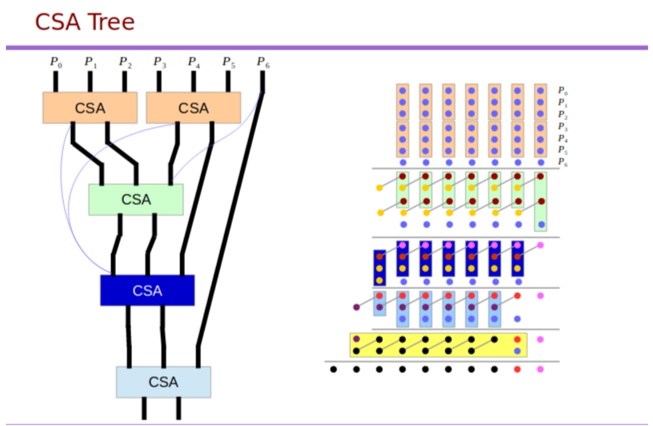
The addition of more than two numbers would be done using carry - save adders based on the fact that a full - adder has 3 inputs and obtains 2 outputs and hence the full adder becomes the basic building block . The addition of four 4 - bit numbers, four 8 - bit numbers, four 16 - bit numbers, four 32 - bit numbers and four 64 - bit numbers are ex emplified using CSA principle in this study. Addition of four 4 - bit numbers A 0 - 3 , B 0 - 3 , C 0 - 3 and D 0 - 3 are the four 4 - bit numbers which are used for addition. . The carry from each stage is propagated down instead in the same stage and hence reduce the overall delay. These are known as carry save stages and varies depend on the number of operands. Finally, the ripple - carry adder is used to obta in the final sum at the cost of delay.


Multi-operand Addition Examples (2)

3 FINAL SUM COMPUTATION METHODS OF CSA

Multi - operand addition using CSA has two important stages as carry - save stages and final stage to compute the overall sum. This section exemplifies the final stage using ripple - carry adder, carry look - ahead adder and carry select adder to determine the final output sum. Sum output using RCA In this study, the final sum output using RCA is referred in this study as CSA - RCA. The carry from the previous stages are considered here and propagated within the last stage to find the final sum output valu e.

Sum output using CLA The last stage is replaced with carry look - ahead adder circuit and is referred to as CSA - CLA. The carry out is calculated in advance based on the propagate and generate terms instead of depending on the previous ca rry. Since the rippling effect is reduced, this circuit should offer better performance compared with the RCA combination. . The VHDL portion of a CLA stage with the addition of four 8 - bit numbers is shown i n Figure 11. If a 0 = b 0 = 1 (i.e., g 0 = 1), then the c1 is computed as 1 based on the generate term. If p 0 = 1 and c 0 = 1, then the c1 is computed as 1 based on the propagate term. The CLA circuit is constructed based on the following equation (3). C i +1 = (A i B i)C i + A i B i = P i C i + G i (3) For a 4 - bit CLA circuit, these carry terms can be computed as,



4 References

4.1 1. FPGA IMPLEMENATION OF HIGH SPEED AND LOWPOWER CARRY SAVE ADDER VS.Balaji , Har Narayan Upadhyay

4.2 2. https://www.quora.com/How-should-I-design-a-carry-save-adder-circuit-so-that-I-can-make-it-as-fast-and-compact-as-possible

4.3 3. http://www.geoffknagge.com/fyp/carrysave.shtml

4.4 4. Behrooz Parhami, "Computer Arithmetic", Oxford Press, 2000, pp131

4.5 5. Performance Analysisof Different Bit Carry Look Ahead Adder Using VHDL Environment Rajender Kumar,Sandeep DahiyaSES,BPSMV,Khanpur Kalan, Gohan, Sonipat ,Haryana International Journal of Engineering Science and Innovative Technology (IJESIT) Volume 2, Issue 4, July 2013

4.6 6. Modified energy efficient carry save adder Benisha bennet, s. maflin 2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]

4.7 7. Design of carry save adder using transmission gate logic J.Princy Joice Dept of ECE/Sathyabama University, Chennai/Tamilnadu/India M.Anitha Dept of ECE/Sathyabama University, Chennai/Tamilnadu/India Mrs.I.Rexlin Sheeba Assistant Professor, Dept of ECE/Sathyabama University, Chennai/Tamilnadu/India INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT] ISSN: 2394-3696 VOLUME 2, ISSUE 1 JAN-2015