Faculty of Engineering & Technology

Department of Electrical and Computer Engineering

ADVANCED DIGITAL SYSTEMS DESIGN

ENCS533

Project Advance-2020

Name: Bader Tawafsheh

ID: 1171214

Instructor: Dr. Abdullatif Abu Issa

Brief introduction and background

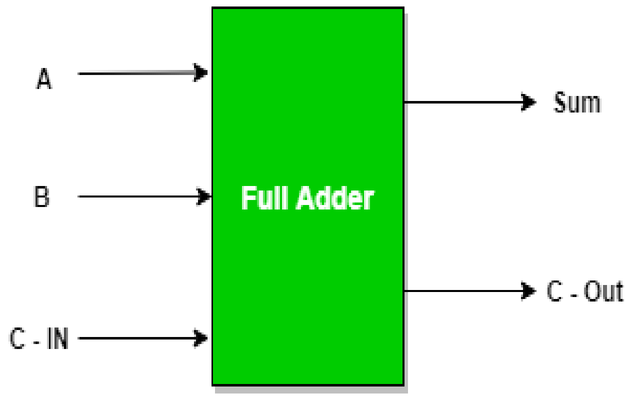
Background

Almost all applications use decimal data and spend most of their time in decimal arithmetic. Software implementation of decimal arithmetic is typically at least 100 times slower than binary arithmetic implemented in hardware. Therefore, hardware support for decimal arithmetic is required. In this paper, a high-speed binary coded decimal (BCD) adder is proposed. The 4-bit BCD adder comprises of two 4-bit full adders and a carry detection logic circuit in its conventional architecture. The two 4-bit full adders are basically Ripple carry adders where the carry output of one full adder cell is propagated onto the succeeding full adder cell at each computational stage. This forces each 1-bit full adder cell to wait for the carry propagation from its preceding cell to compute the sum. In the conventional design the Ripple carry adder is the protagonist of the logic design. With its simplified structure, the Ripple carry adder occupies less area and performs fast operation. But it is an irony that this fast operation is limited to lower order bits and finds itself unsuitable for higher multiple bit computations. The necessity of carry propagating from preceding stages serves the reason.

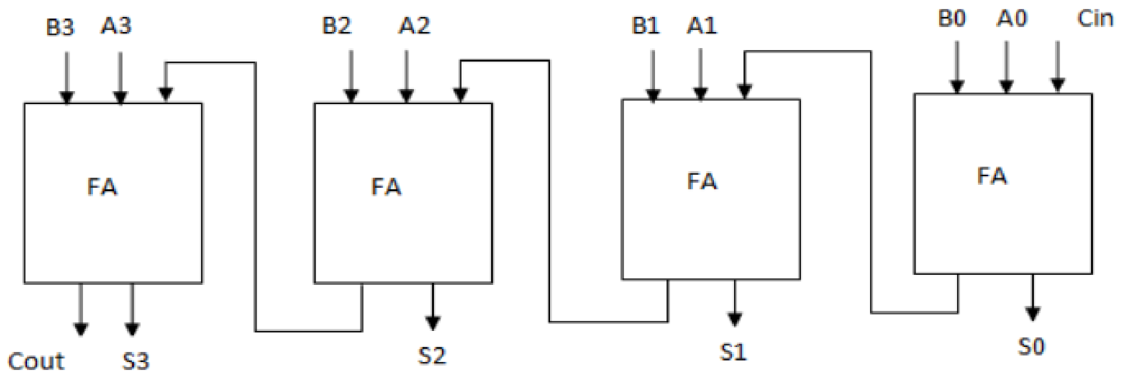
Introduction

-Adder: digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used in the arithmetic logic units or ALU. They are also used in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators and similar operations.

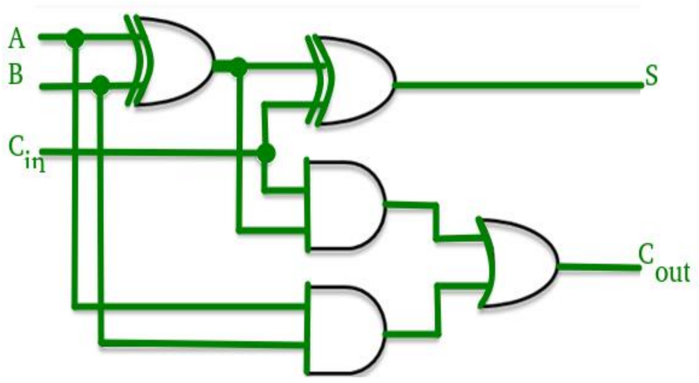
-Full Adder: Full Adder is the adder which adds three inputs and produces two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. The output carry is designated as C-OUT and the normal output is designated as S which is SUM.
A full adder logic is designed in such a manner that can take eight inputs together to create a byte-wide adder and cascade the carry bit from one adder to the another.

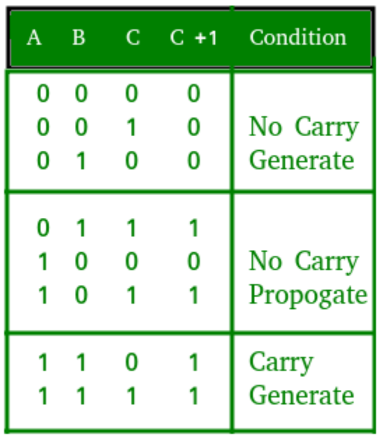


-Ripple-Carry Adder: Ripple Carry Adder the Ripple carry adder is the basic building block of the BCD adder. Constructing an n-bit ripple carry adder requires n full adder cells. Ripple carry adder in this design, LSB of the sum bit is produced by the LSB's of two numbers a0 and b0. Carry of these LSB bits is propagated to the next cell. c0 which is initially zero is applied as a third input to the full adder. This carry is computed at each adder cell and propagated to the succeeding stages.



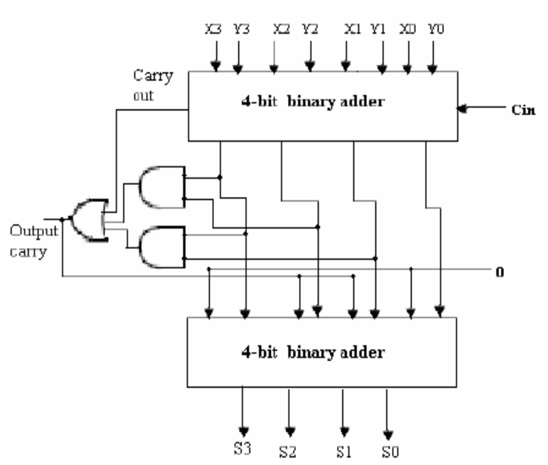
-Carry Look-Ahead Adder: A carry look-ahead adder reduces the propagation delay by introducing more complex hardware. In this design, the ripple carry design is suitably transformed such that the carry logic over fixed groups of bits of the adder is reduced to two-level logic. Let us discuss the design in detail. If we define two new binary variables **Pi = Ai XOR Bi**, **Gi = Ai AND Bi.** The output sum and carry can respectively be expressed as **Si = Pi XOR Ci**, **Ci+1 = Gi + PiCi**. Gi is called a carry generate, and it produces a carry of 1 when both Ai and Bi are 1, regardless of the input carry Ci. Pi is called a carry propagate, because it determines whether a carry into stage i will propagate into stage i + 1.





A	B	C	C +1	Condition
0	0	0	0	No Carry Generate
0	0	1	0	
0	1	0	0	
0	1	1	1	No Carry Propogate
1	0	0	0	
1	0	1	1	
1	1	0	1	Carry Generate
1	1	1	1	

-**BCD Adder**: A BCD adder is a circuit that adds two BCD digits in parallel and produces a sum digit also in BCD. A BCD adder must also include the correction logic in its internal construction. The two decimal digits, together with the input carry, are first added in the top 4-bit binary adder to produce the binary sum. When the output carry is equal to zero, nothing is added to the binary sum. When it is equal to one, binary 0110 is added to the binary sum using another 4-bit binary adder (bottom binary adder). The output carry generated from the bottom binary adder is ignored, since it supplies information already available at the output carry terminal



Design philosophy

I divided the work into 6 parts to make it easier.

First part, of the design was to implement the logical gates with the required delays, then save them to separate library called '**gates**', to make it easy and simple to use them in the coming stages.

Second part, I implement the adders (Full adders,4-bit Ripple adder and 4-bit look-ahead adder) Respectively, the implementation of Full adder is from logical gates (which in first part I defined) then the 4-bit ripple it was implemented by 4 Full adder and the last adder is a 4-bit look-ahead I defined some equations in introduction to use it for implementation .

Third part, to implement the BCD adder I need two 4-bit adder and the combinational circuit (it was implemented by 2 and gates and 1 or gate, sure the gates defined in the library in the first part) , The two 4-bit adder are inputs and also output enter combinational circuit , in this project the work divided into two stages the first stage is to implement BCD adder using 4-bit ripple adder and the second stage using 4-bit look-ahead adder .

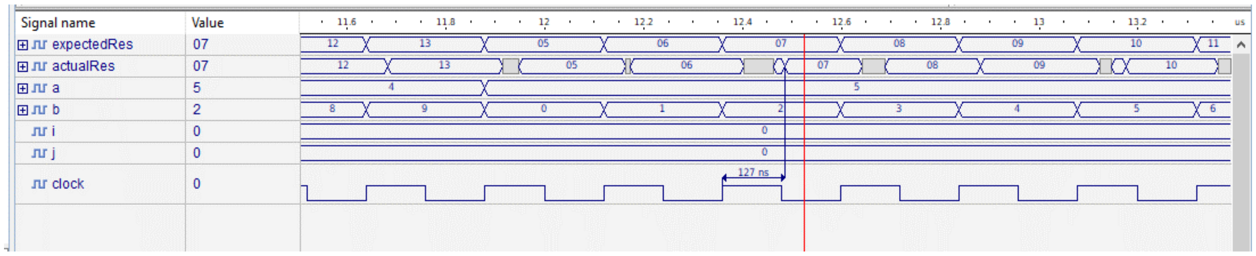
Fourth part, was to make 2 registers (group of d-flip flops), first register takes 8-bit input to BCD adder and the second register 5-bit output of BCD adder

Fifth part, was to implement the whole system that contains a BCD adder (done in third part) and 2 Registers (done in fourth part). In addition, to obtain the max delay of BCD adders, by trying the inputs, so the clock of the synchronous BCD was based on that.

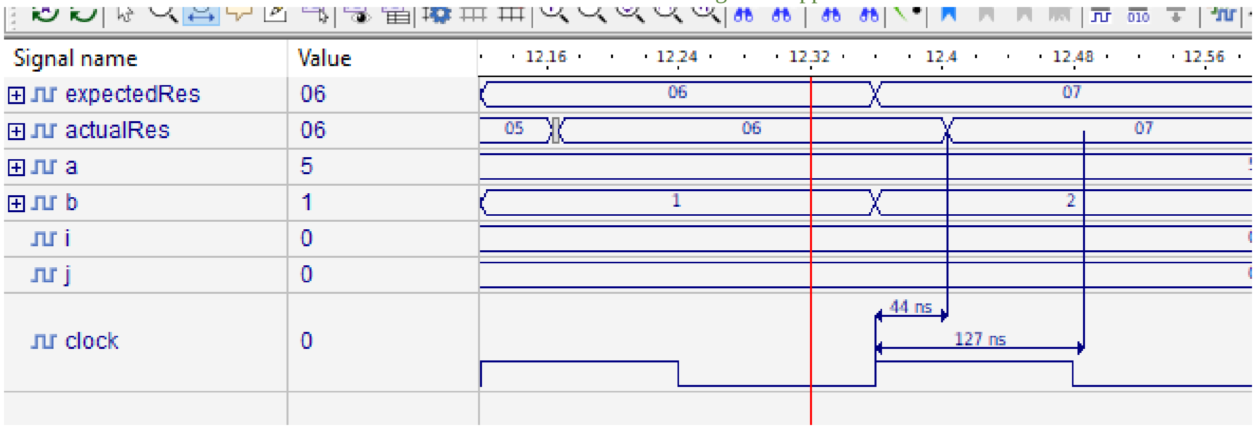
Final part, in verification step we test the possible states that our circuit will handle, the possible inputs are first number start from 0 to 9 and the same for the second, that means there is 10*10 possible states. for each state we calculate the expected output and get the output from our circuit then test if the expected is match the actual. then print to file the inputs, out circuit output, expected output and if its matches or not.
after end the 100 states verification will stop.

Results

The max delay was obtained from the BCD using 4-bit ripple adder was around (130ns) when the inputs were (0101) and (0010), and for the BCD using 4-bit carry look ahead adder it was around (45ns) for the same input. For whole system , "BCD adder with registers" I can't obtained the best clock to get correct results.



Simulation for BCD adder using 4-bit ripple



Simulation for BCD adder using 4-bit look-ahead

Conclusion and Future works

According to results we get in verifications step I found that the carry look ahead adder is faster than the ripple adder because ripple adder uses gates that have grater delay than the look-ahead adder use.
Look ahead adder using more gated to implement but it faster than ripple that using less number of gates.

Note: the codes are in an attachment file which I will attach with this report