East West University

Project

Semester	**:** Spring’21
Course Title	**:** Computer Architecture
Course Code	**:** CSE360
Section	**:** 04

Project Title: Suggest a high speed addition method and logic for 4-bit addition.

<u>Submitted to</u>

Surajit Das Barman

Senior Lecturer

Department of Computer Science & Engineering

<u>Submitted By</u>

Zubayar Mahatab Md Sakif	2018-1-60-105
Md. Ferdous	2018-1-60-098
Md Ashraful Alam Hridoy	2018-1-60-174

Date of Report Submission : 29-05-2021

Project Title : Suggest a high speed addition method and logic for 4-bit addition.

Objective: To improves speed and reduces delays in the circuit for 4-bit adder.

Description:

A 4-bit ripple-carry adder :

A ripple carry adder is simple, which allows for fast design time; however, the ripple-carry adder is relatively slow, since each full adder must wait for the carry-bit to be calculated from the previous full adder.

Delays in a circuit may be caused due to a variety of reasons. The primary shortcoming of a ripple carry adder is the fact that at every bit must wait for the result of the previous carry. This is a significant impediment for circuit speed. Each bit waits for the carry generated at the previous bit, in order to calculate the next sum. This causes delays in the circuit. In contrast, the carry look-ahead adder block, calculates the next carry bit with the sum. Due to this, the current bit already has the value of the carry, required to calculate the sum. This significantly improves speed, and reduces delays in the circuit.

A 4-bit carry look ahead adder

Carry look ahead is a digital circuit used for determining the carry bits used by the adder for addition without the wait for the carry propagation. It generates the carry bits for all the stages of the addition at the same time as soon as the input signal (Augend, addend, carry in) is provided.

The most critical component of this adder is the carry look ahead block. It performs the task of simultaneously calculating the carry, so that there need not be any delay in receiving carry from the previous bit. This enables faster computation and the inherent quality of reversible logic based circuits make it a lucrative option for circuit designers.

Cin

$C0 = A0B0 + Cin(A0 + B0)$

$= G0 + P0Cin$

$C1 = A1B1 + C0(A1 + B1)$

$= G1 + P1C0$

$= G1 + P1(G0 + P0Cin)$

$= G1 + P1 G0 + P1P0Cin$

C2 = A2B2 + C1(A2 + B2)

= G2 + P2C1

= G2 + P2(G1 + P1 G0 + P1P0Cin)

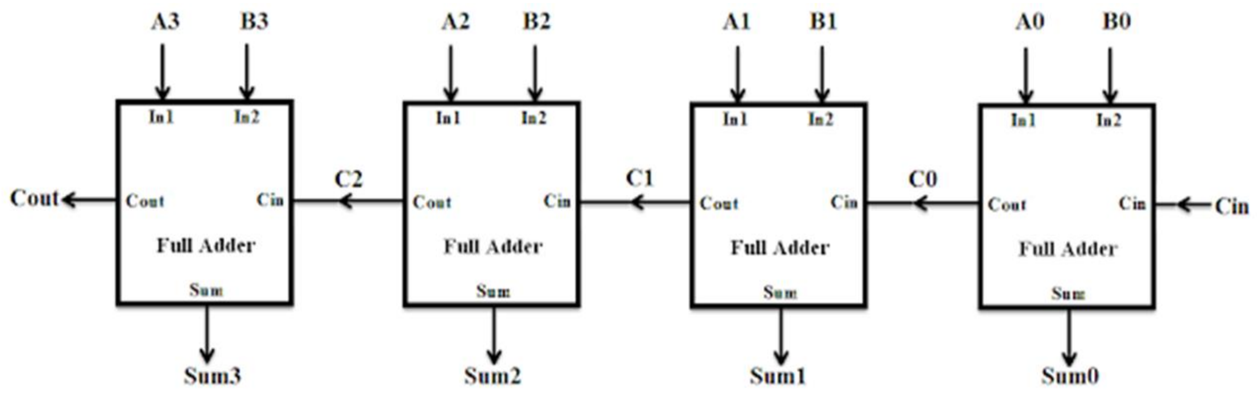
= G2 + P2 G1 + P2P1G0 + P2P1P0Cin

C3 = G3 + P3G2 + P3P2G1 + P3P2P1G0 + P3P2P1P0Cin

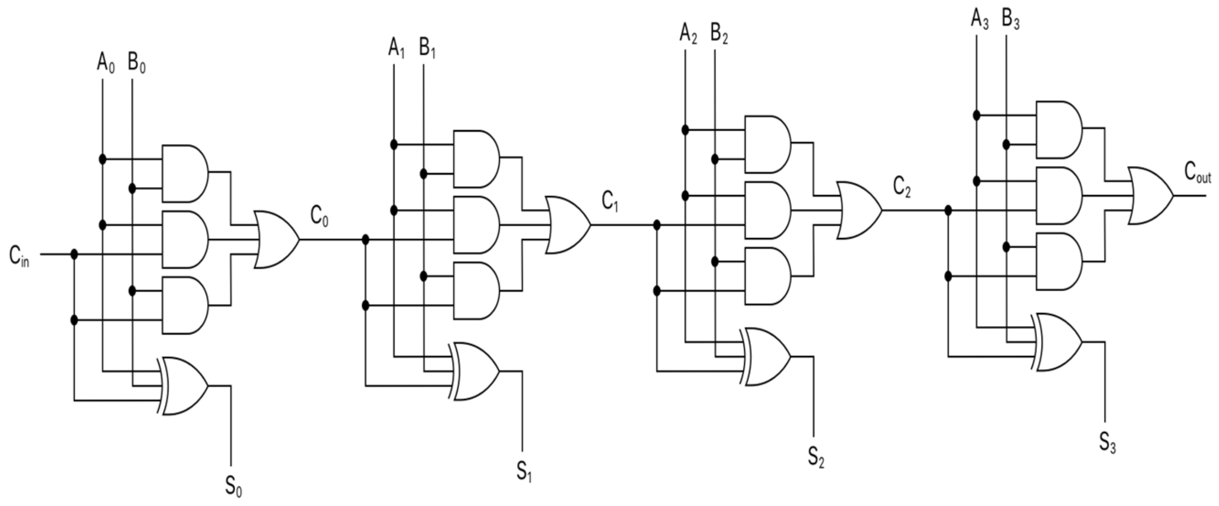
Design:

A 4-Bit Ripple Carry Adder

Block diagram:



Circuit diagram:



4-bit carry look ahead adder

Block diagram:

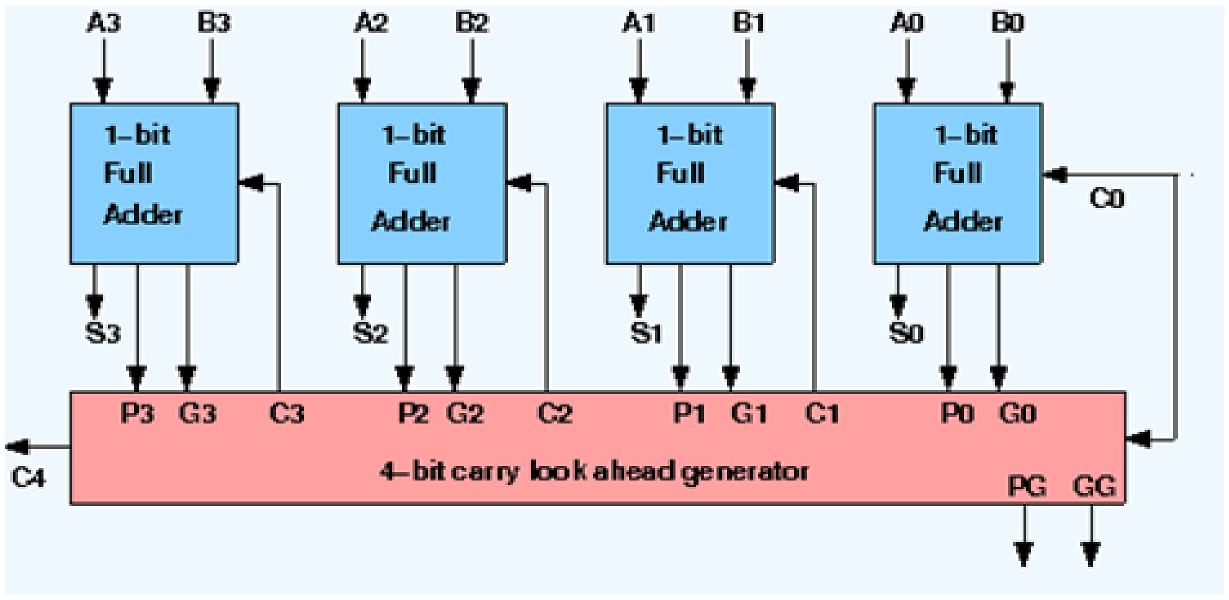
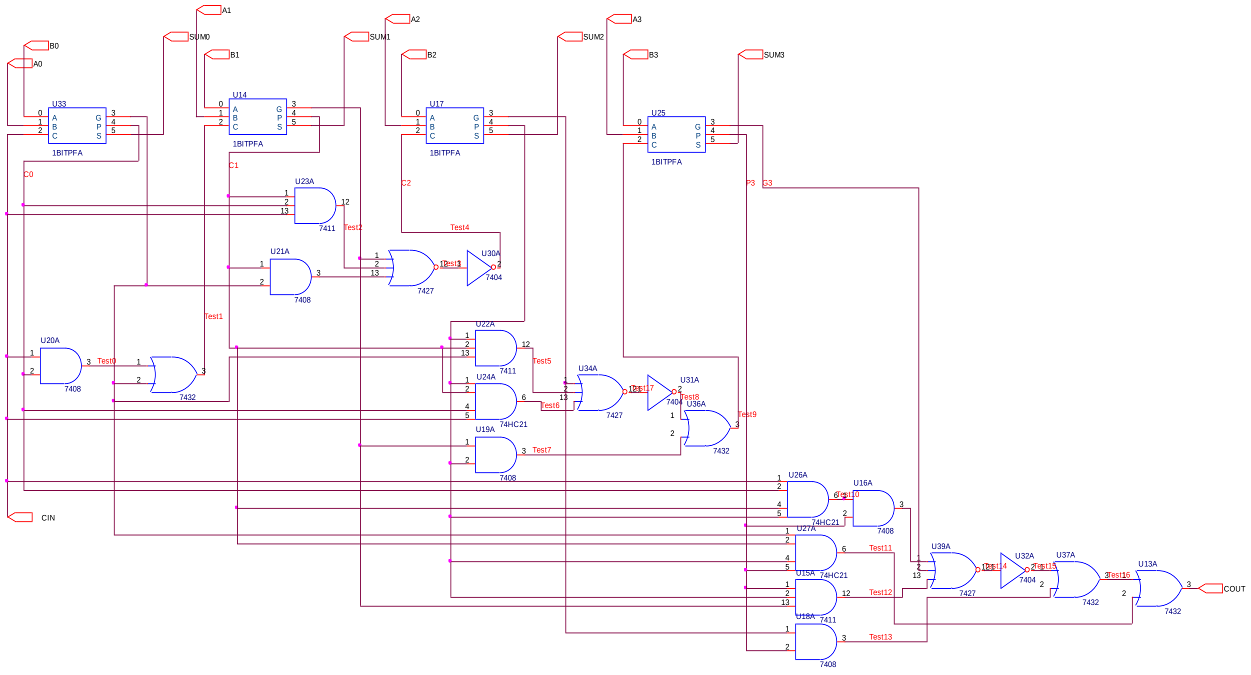

A3 B3 A2 B2 A1 B1 A0 B0
1-bit Full Adder
C0
S3 S2 S1 S0
P3 G3 C3 P2 G2 C2 P1 G1 C1 P0 G0
C4
4-bit carry look ahead generator
PG GG

Fig3: Carry Look ahead Adders

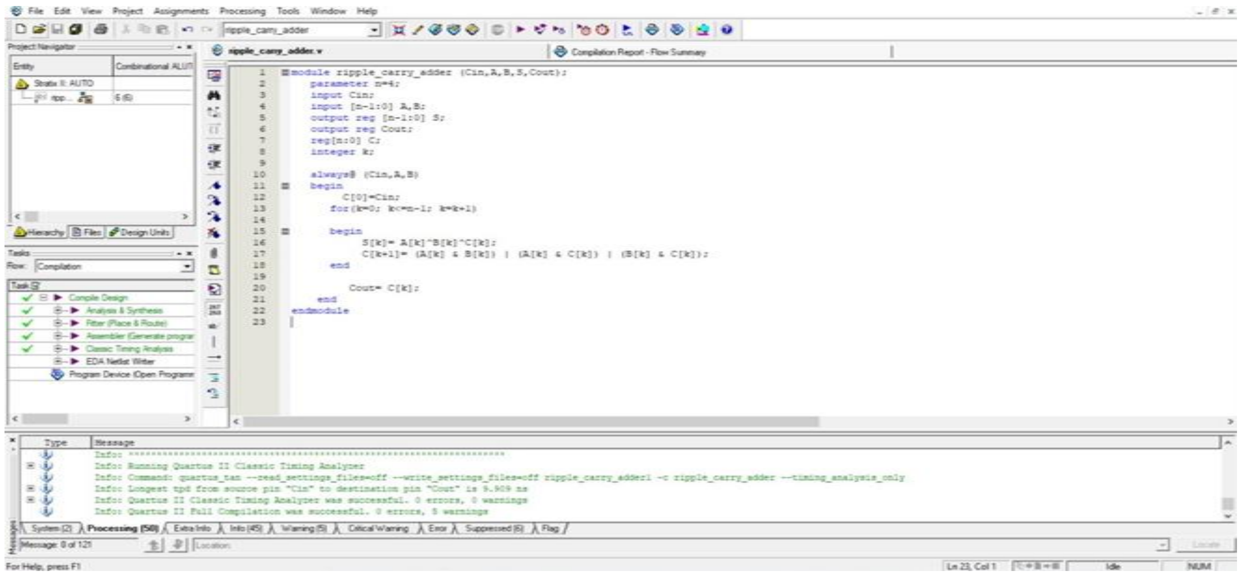
Circuit diagram:



Implementation:

A 4-bit ripple carry adder

Verilog code:

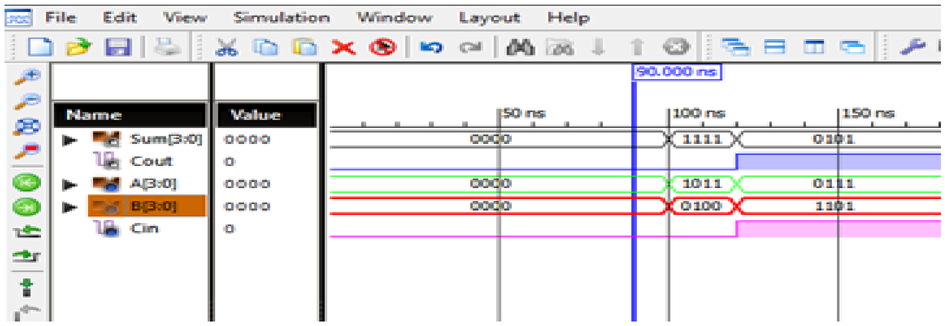


```
module ripple_carry_adder (Cin,A,B,S,Cout);
    parameter n=4;
    input Cin;
    input [n-1:0] A,B;
    output reg [n-1:0] S;
    output reg Cout;
    reg[n:0] C;
    integer k;

    always@ (Cin,A,B)
    begin
        C[0]=Cin;
      for(k=0; k<=n-1; k=k+1)

      begin
          S[k]= A[k]^B[k]^C[k];
          C[k+1]= (A[k] & B[k]) | (A[k] & C[k]) | (B[k] & C[k]);
      end

        Cout= C[k];
     end
 endmodule
```

```
Info: *******************************************************************
Info: Running Quartus II Classic Timing Analyzer
Info: Command: quartus_tan --read_settings_files=off --write_settings_files=off ripple_carry_adder1 -c ripple_carry_adder --timing_analysis_only
Info: Longest tpd from source pin "Cin" to destination pin "Cout" is 9.909 ns
Info: Quartus II Classic Timing Analyzer was successful. 0 errors, 0 warnings
Info: Quartus II Full Compilation was successful. 0 errors, 5 warnings
```

```
moduleripple_carry_adder (Cin,A,B,S,Cout);
parameter n=4;
inputCin;
input [n-1:0] A,B;
outputreg [n-1:0] S;
outputregCout;
reg[n:0] C;
integer k;

always@ (Cin,A,B)
begin
C[0]=Cin;
for(k=0; k<=n-1; k=k+1)

begin
        S[k]= A[k]^B[k]^C[k];
C[k+1]= (A[k] & B[k]) | (A[k] & C[k]) | (B[k] & C[k]);
end

Cout= C[k];
end
endmodule
```

Timing diagram



A 4-bit carry look ahead adder

Verilog code:

```
modulecarry_lookahead_adder (Cin,A,B,S,C);
parameter n=4;
inputCin;
input [n-1:0] A,B;
output [n-1:0] S;
output [n-1:0] C;
reg [n-1:0] G;
reg [n-1:0] P;
integer k;
always@ (A,B)
begin
for(k=0; k<=n-1; k=k+1)
begin
```

G[k]= (A[k]&B[k]);

P[k]= (A[k]^B[k]);

end

end

assign C[0]= G[0] | (P[0] &Cin);

assign C[1]= G[1] | (P[1] & G[0]) | (P[1] & P[0] &Cin);

assign C[2]= G[2] | (P[2] & G[1]) | (P[2] & P[1] & G[0]) | (P[2] & P[1] & P[0] &Cin);

assign C[3]= G[3] | (P[3] & G[2]) | (P[3] & P[2] & G[1]) | (P[3] & P[2] & P[1] & G[0]) | (P[3] & P[2] & P[1] & P[0] &Cin);

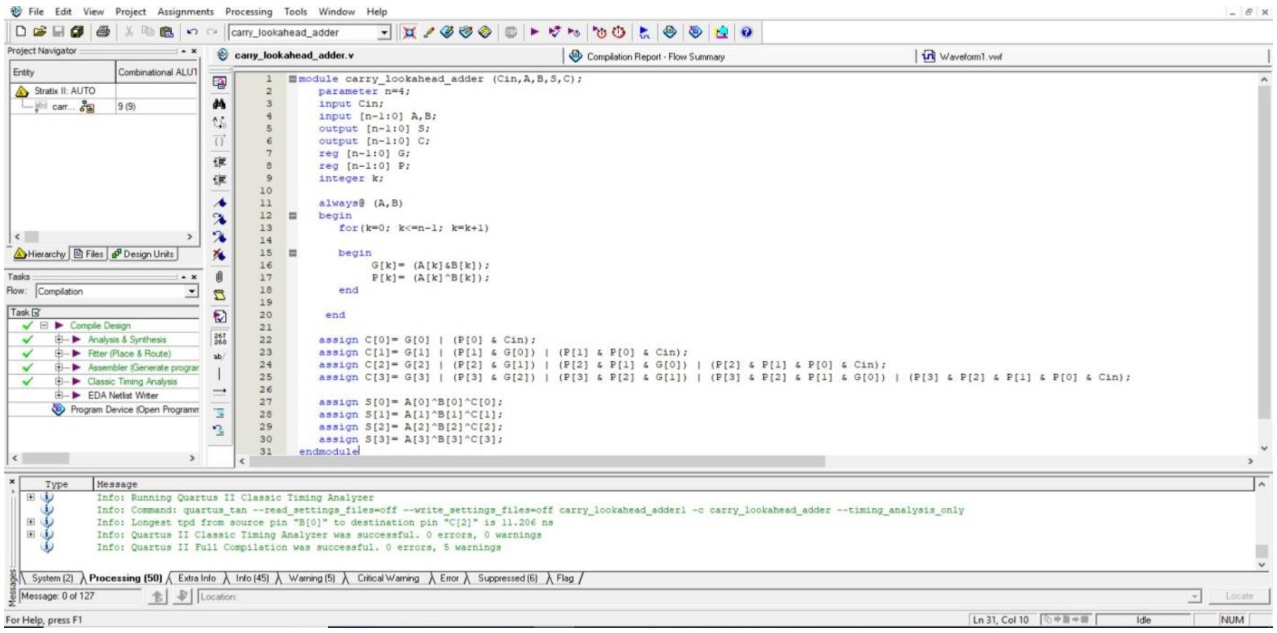
assign S[0]= A[0]^B[0]^C[0];

assign S[1]= A[1]^B[1]^C[1];

assign S[2]= A[2]^B[2]^C[2];

assign S[3]= A[3]^B[3]^C[3];

endmodule



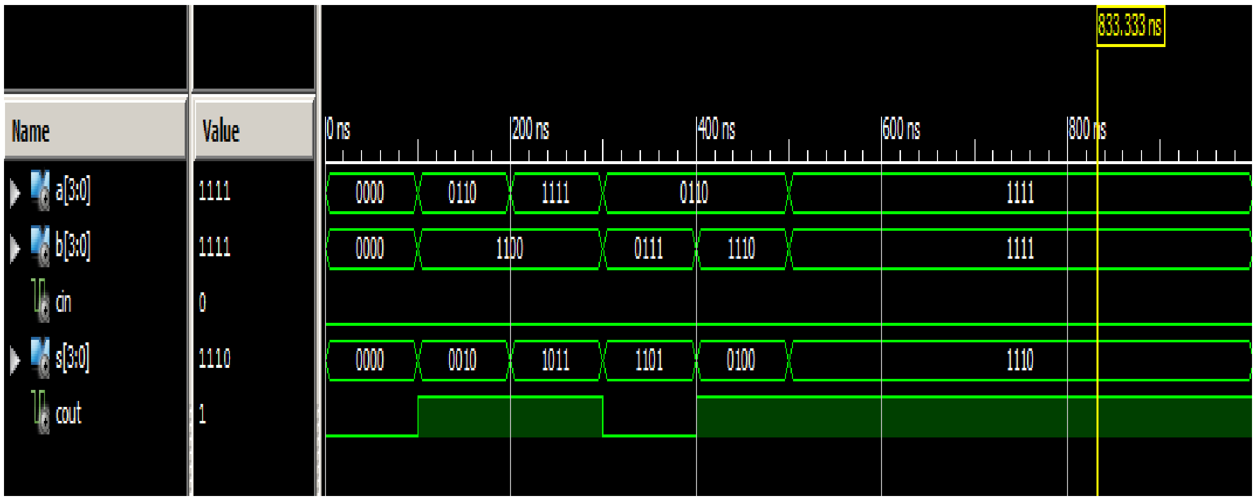
```
module carry_lookahead_adder (Cin,A,B,S,C);
   parameter n=4;
   input Cin;
   input [n-1:0] A,B;
   output [n-1:0] S;
   output [n-1:0] C;
   reg [n-1:0] G;
   reg [n-1:0] P;
   integer k;

   always@ (A,B)
   begin
      for(k=0; k<=n-1; k=k+1)

      begin
            G[k]= (A[k]&B[k]);
            P[k]= (A[k]^B[k]);
      end

    end

   assign C[0]= G[0] | (P[0] & Cin);
   assign C[1]= G[1] | (P[1] & G[0]) | (P[1] & P[0] & Cin);
   assign C[2]= G[2] | (P[2] & G[1]) | (P[2] & P[1] & G[0]) | (P[2] & P[1] & P[0] & Cin);
   assign C[3]= G[3] | (P[3] & G[2]) | (P[3] & P[2] & G[1]) | (P[3] & P[2] & P[1] & G[0]) | (P[3] & P[2] & P[1] & P[0] & Cin);

   assign S[0]= A[0]^B[0]^C[0];
   assign S[1]= A[1]^B[1]^C[1];
   assign S[2]= A[2]^B[2]^C[2];
   assign S[3]= A[3]^B[3]^C[3];
endmodule
```

```
Type  Message
      Info: Running Quartus II Classic Timing Analyzer
      Info: Command: quartus_tan --read_settings_files=off --write_settings_files=off carry_lookahead_adder1 -c carry_lookahead_adder --timing_analysis_only
      Info: Longest tpd from source pin "B[0]" to destination pin "C[2]" is 11.206 ns
      Info: Quartus II Classic Timing Analyzer was successful. 0 errors, 0 warnings
      Info: Quartus II Full Compilation was successful. 0 errors, 5 warnings
```

Results Analysis:

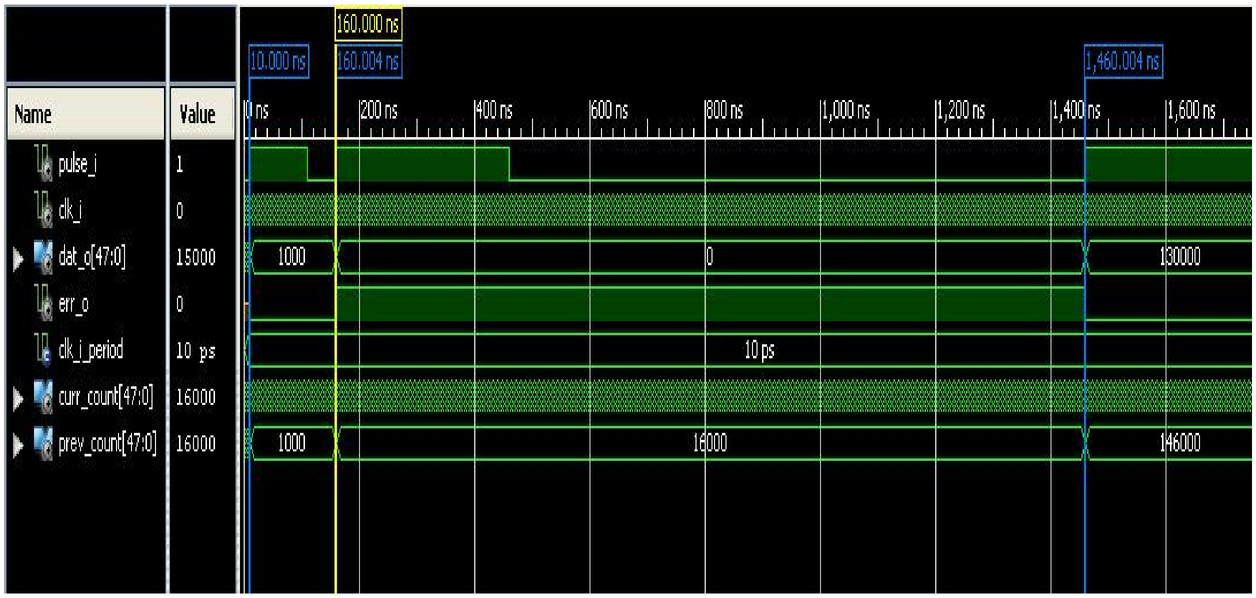
A 4-bit ripple carry adder

Timing diagram :



A 4-bit carry look ahead adder

Timing diagram :



Advantages :

- Carry look ahead circuit is a digital circuit.
- This circuit significantly improves speed, and reduces delays in the circuit.
- A 4-bit carry look ahead adder is 3 percent faster from a 4-bit ripple carry adder.

Conclusion :

- As we have discussed in the ripple carry adder that the adder takes a time delay to compute sum because the carry was propagated through all the gates from input to output.
- To increase the speed of the addition we need to provide the carry bits used by these adders without the wait for the preceding additions is known as Carry look ahead logic.

Bibliography:

1. https://allaboutfpga.com/4-bit-ripple-carry-adder-vhdl-code/
2. https://www.ques10.com/p/17073/explain-carry-look-ahead-adder-and-its-advantage-1/
3. https://allaboutfpga.com/4-bit-ripple-carry-adder-vhdl-code/
4. file:///C:/Users/HP/Desktop/Different-types-of-adder.pdf
5. file:///C:/Users/HP/Desktop/4e70453fb4234608b56d1b952671c829882c.pdf