SCHOOL OF ELECTRONICS ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY (KIIT)

VLSI LABORATORY REPORT

(EC-3095)

Submitted By

Name: Sayani Gorai **Roll No: 1804406**

Name: Debagnik Kar **Roll No: 1804373**

Name: Pallavi Mohini **Roll No: 1804395**

Name: Abhijeet Dey **Roll No: 1804351**

Section: ETC - 06

Semester: 6 TH SEM

Experiment Number-10 (open ended -2)

AIM: Design a 4 bit look ahead adder.

SOFTWARE REQUIRED:

EDA playground

THEORY:

The ripple-carry adder, its limiting factor is the time it takes to propagate the carry. The carry look-ahead adder solves this problem by calculating the carry signals in advance, based on the input signals. The result is a reduced carry propagation time. Consider the full adder circuit with corresponding truth table.

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

We define two variables as **'carry generate'** G_i and **'carry propagate'** P_i then,

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

The sum output and carry output can be expressed in terms of carry generate G_i and carry propagate P_i as

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

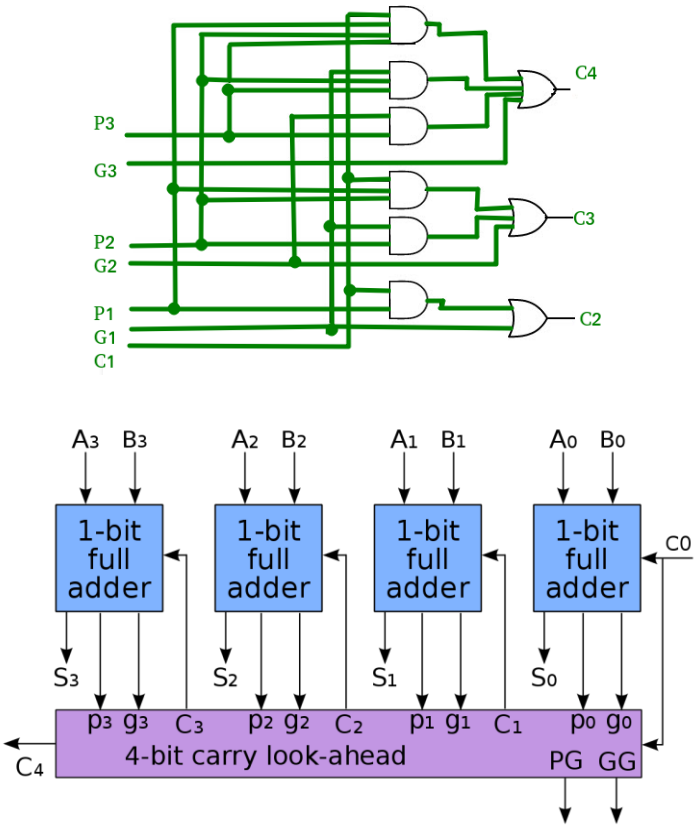
where G_i produces the carry when both A_i, B_i are 1 regardless of the input carry. P_i is associated with the propagation of carry from C_i to C_{i+1}.

The carry output Boolean function of each stage in a 4 stage carry look-ahead adder can be expressed as

$$C_1 = G_0 + P_0 C_{in}$$
$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_{in}$$
$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{in}$$
$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_{in}$$

From the above Boolean equations we can observe that C_4 does not have to wait for C_3 and C_2 to propagate but actually C_4 is propagated at the same time as C_3 and C_2. Since the Boolean expression for each carry output is the sum of products so these can be implemented with one level of AND gates followed by an OR gate.

The implementation of three Boolean functions for each carry output (C_2, C_3 and C_4) for a carry look-ahead carry generator shown in below figure.



Block diagram of 4 bit look ahead adder

CODES:-

1. Design code for 4 bit look ahead adder

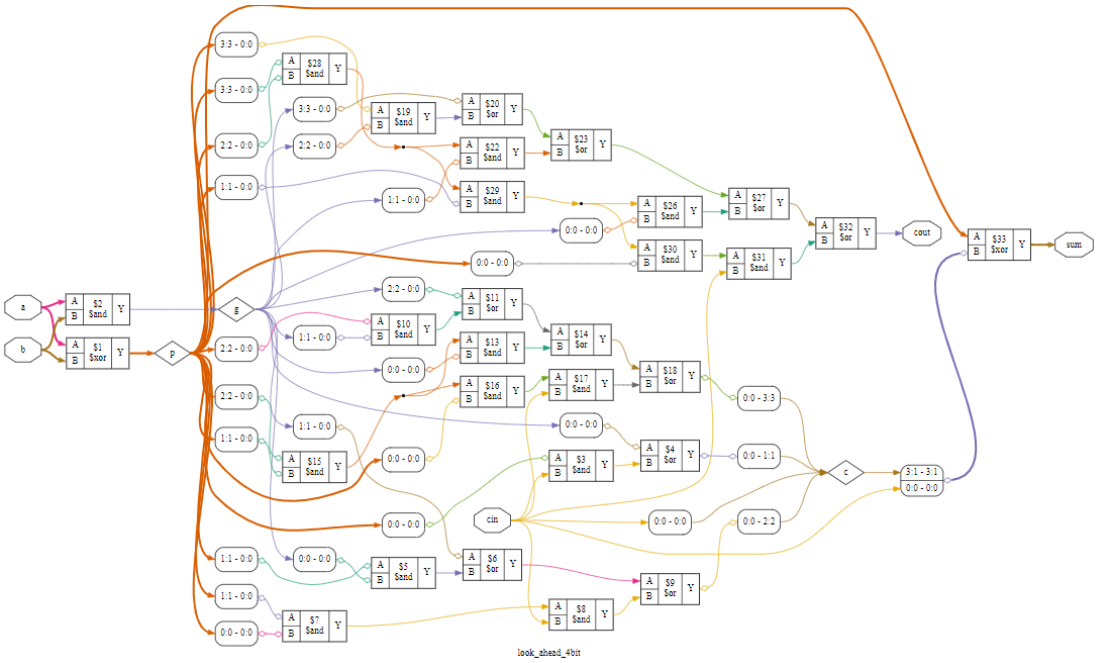
```
// Code your design here
//4-bit Look Ahead Adder

module look_ahead_4bit(a,b, cin, sum,cout);
input [3:0] a,b;
input cin;
output [3:0] sum;
output cout;

wire [3:0] p,g,c;

assign p=a^b;//propagate
assign g=a&b; //generate

//carry=gi + Pi.ci

assign c[0]=cin;
assign c[1]= g[0]|(p[0]&c[0]);
assign c[2]= g[1] | (p[1]&g[0]) | p[1]&p[0]&c[0];
assign  c[3]=  g[2]  |  (p[2]&g[1])  |  p[2]&p[1]&g[0]  |
p[2]&p[1]&p[0]&c[0];
assign  cout=  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]&c[0];
assign sum=p^c;
endmodule
```

2. Testbench code for 4 bit look ahead adder

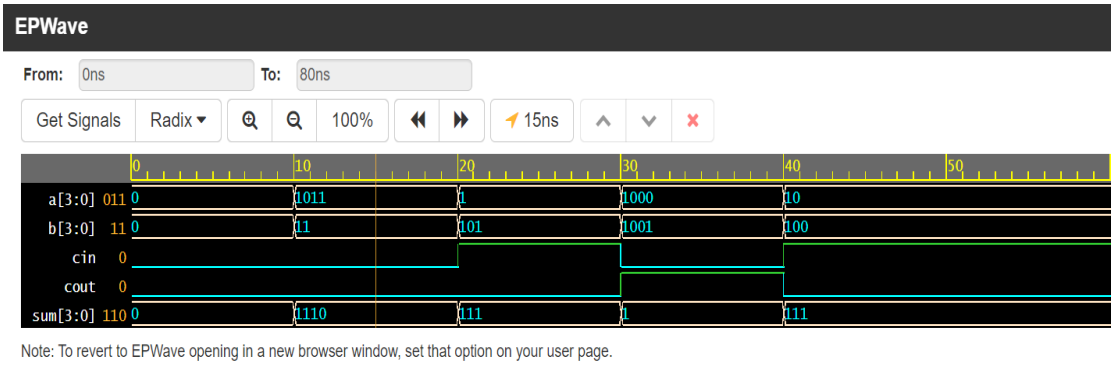
```
module look_ahead_4bit_tb;
  reg [3:0] a,b;
reg cin;
  wire [3:0] sum;
wire cout;

look_ahead_4bit
g1(.a(a), .b(b),.cin(cin),.sum(sum),.cout(cout));

initial begin
  $dumpfile("dump.vcd");
  $dumpvars;
```

```
  a=0; b=0; cin=0;
  #10 a=4'b1011; b=4'b0011; cin=1'b0;
  #10 a=4'b0001; b=4'b0101; cin=1'b1;
  #10 a=4'b1000; b=4'b1001; cin=1'b0;
  #10 a=4'b0010; b=4'b0100; cin=1'b1;
end
endmodule
```

OUTPUTS:-



schematic diagram of 4 bit look-ahead adder



Output waveform of 4 bit look ahead adder

Conclusion:

In this experiment we have successfully designed a 4 bit look ahead adder by writing code in EDA playground and simulated the outputs by writing the testbench code for it.