**Digital Signal Design Lab**

Lab CEL-442

Lab Journal: 9



Name: Omar

Class: BCE-06

Enrollment No:01-132182-024

**Lab # 9**

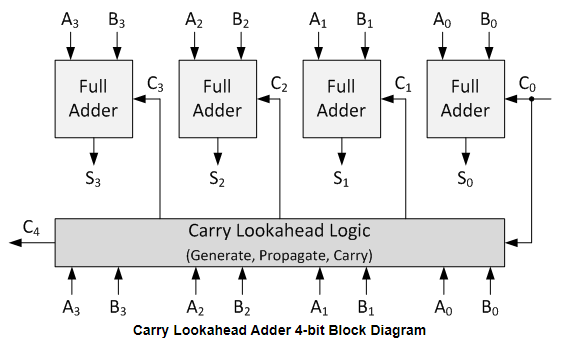
**Carry Lookahead Adder (CLA)**

**Objective**  This lab is aimed at a Carry Look Ahead Adder and will implement in Verilog HDL.

**Introduction**

**Carry Lookahead Adder**

* A Carry Lookahead (Look Ahead) Adder is made of a number of full adders cascaded together.
* It is used to add together two binary numbers using only simple logic gates. The figure below shows 4 full-adders connected together to produce a 4-bit carry lookahead adder.
* Carry lookahead adders are similar to Ripple Carry Adder. The difference is that carry lookahead adders are able to calculate the Carry bit before the Full Adder is done with its operation.
* This gives it an advantage over the Ripple Carry Adder because it is able to add two numbers together faster.
* The drawback is that it takes more logic.



**Task 01**

* **Implement 4-bit Carry Lookahead Adder in Verilog.**

**Module:**

`timescale 1ns / 1ps

module LAB\_9(

input [3:0]a,

input [3:0]b,

input cin,

output [3:0]sum,

output cout,

wire p0,p1,p2,p3,

wire g0,g1,g2,g3,

wire c1,c2,c3,c4

);

assign p0 = a[0] ^ b[0];

assign p1 = a[1] ^ b[1];

assign p2 = a[2] ^ b[2];

assign p3 = a[3] ^ b[3];

assign g0 = a[0] & b[0];

assign g1 = a[1] & b[1];

assign g2 = a[2] & b[2];

assign g3 = a[3] & b[3];

assign c1 = (p0 & cin) | g0;

assign c2 = (p0 & p1 & cin) | (p1 & g0) | g1;

assign c3 = (p0 & p1 & p2 & cin) | (p1 & p2 & g0) | (p2 & g1) | g2;

assign c4 = (p0 & p1 & p2 & p3 & cin) | (p1 & p2 & p3 & g0) | (p2 & p3 & g1) | (p3 & g2) | g3;

assign sum[0] = p0 ^ cin;

assign sum[1] = p1 ^ c1;

assign sum[2] = p2 ^ c2;

assign sum[3] = p3 ^ c3;

assign cout = c4;

endmodule

**Stimulus Module:**

`timescale 1ns / 1ps

module LAB\_9\_TEST;

// Inputs

reg [3:0] a;

reg [3:0] b;

reg cin;

// Outputs

wire [3:0] sum;

wire cout;

// Instantiate the Unit Under Test (UUT)

LAB\_9 uut (

.a(a),

.b(b),

.cin(cin),

.sum(sum),

.cout(cout)

);

initial begin

// Initialize Inputs

a = 1100;

b = 1010;

cin = 0;

// Wait 100 ns for global reset to finish

#100;

a = 0000;

b = 1111;

cin = 0;

#100;

a = 1111;

b = 1111;

cin = 0;

#100;

a = 0011;

b = 1010;

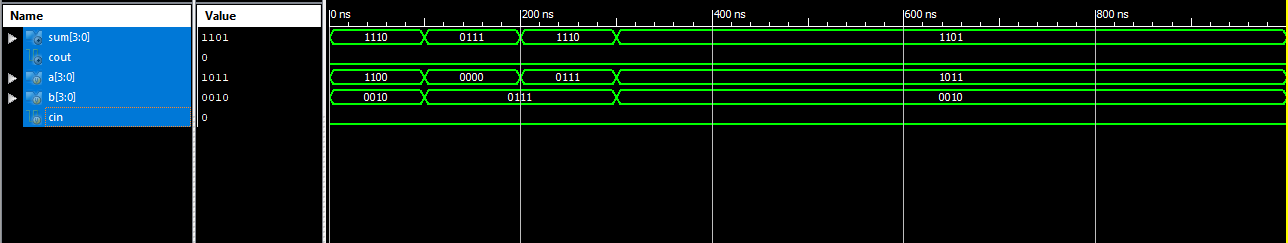
cin = 0;

#50;

end

endmodule

**Output:**

****

**Task 02**

* **Implement 16-bit Carry Lookahead Adder in Verilog.**

**Module:**

module CLA1(a,b,c0,s,cout);

input [7:0]a,b;

input c0;

output [7:0]s;

output cout;

wire [7:0]p,g;

wire c1,c2,c3,c4,c5,c6,c7;

assign p[0]=a[0]^b[0];

assign p[1]=a[1]^b[1];

assign p[2]=a[2]^b[2];

assign p[3]=a[3]^b[3];

assign p[4]=a[4]^b[4];

assign p[5]=a[5]^b[5];

assign p[6]=a[6]^b[6];

assign p[7]=a[7]^b[7];

assign g[0]=a[0]&b[0];

assign g[1]=a[1]&b[1];

assign g[2]=a[2]&b[2];

assign g[3]=a[3]&b[3];

assign g[4]=a[4]&b[4];

assign g[5]=a[5]&b[5];

assign g[6]=a[6]&b[6];

assign g[7]=a[7]&b[7];

assign c1=p[0]&c0|g[0];

assign c2=p[0]&p[1]&c0|p[1]&g[0]|g[1];

assign c3=p[0]&p[1]&p[2]&c0|p[1]&p[2]&g[0]|p[2]&g[1]|g[2];

assign c4=p[0]&p[1]&p[2]&p[3]&c0|p[1]&p[2]&p[3]&g[0]|p[2]&p[3]&g[1]|p[3]&g[2]|g[3];

assign c5=p[4]&(p[0]&p[1]&p[2]&p[3]&c0|p[1]&p[2]&p[3]&g[0]|p[2]&p[3]&g[1]|p[3]&g[2]|g[3])|g[4];

assign c6=p[5]&(p[4]&(p[0]&p[1]&p[2]&p[3]&c0|p[1]&p[2]&p[3]&g[0]|p[2]&p[3]&g[1]|p[3]&g[2]|g[3])|g[4])|g[5];

assign c7=p[6]&(p[5]&(p[4]&(p[0]&p[1]&p[2]&p[3]&c0|p[1]&p[2]&p[3]&g[0]|p[2]&p[3]&g[1]|p[3]&g[2]|g[3])|g[4])|g[5])|g[6];

assign cout=p[7]&(p[6]&(p[5]&(p[4]&(p[0]&p[1]&p[2]&p[3]&c0|p[1]&p[2]&p[3]&g[0]|p[2]&p[3]&g[1]|p[3]&g[2]|g[3])|g[4])|g[5])|g[6])|g[7];

assign s[0]=p[0]^c0;

assign s[1]=p[1]^c1;

assign s[2]=p[2]^c2;

assign s[3]=p[3]^c3;

assign s[4]=p[4]^c4;

assign s[5]=p[5]^c5;

assign s[6]=p[6]^c6;

assign s[7]=p[7]^c7;

endmodule

**Stimulus Module:**

`timescale 1ns / 1ps

module y1;

// Inputs

reg [7:0] a;

reg [7:0] b;

reg c0;

// Outputs

wire [7:0] s;

wire cout;

// Instantiate the Unit Under Test (UUT)

CLA1 uut (

.a(a),

.b(b),

.c0(c0),

.s(s),

.cout(cout)

);

initial begin

// Initialize Inputs

a = 10;

b = 5;

c0 = 0;

// Wait 100 ns for global reset to finish

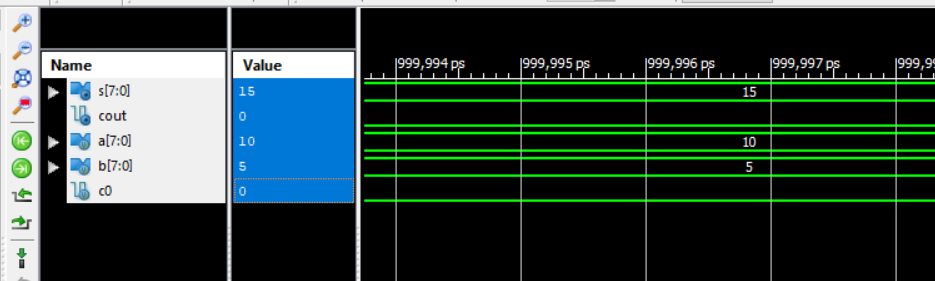
#100;

// Add stimulus here

end

endmodule

**Output:**



**Conclusion: -**

In this lab we learned about a Carry Look Ahead Adder and will implement in Verilog HDL