**//post lab (carry save adder)**

**//RA2111004010177** Program code

module carry\_save\_adder\_177(a,b, cin, sum, cout); input [3:0] a,b; input cin; output [3:0] sum; output cout; wire[3:0]p; wire c0; wire bp;

ripple g1(sum[3:0],c0,a[3:0],b[3:0],cin); propogate\_p p1(p,bp,a,b); mux\_2x1 g2(cout,c0,cin,bp); endmodule module propogate\_p(p,bp,a,b); input [3:0]a,b; output [3:0]p; output bp; assign p=a^b; assign bp=&p; endmodule

**//code for ripple:** module ripple\_177(sum, carry, a,b, cin); input [3:0] sum; input carry; input [3:0] a,b;

input cin; wire c1,c2,c3; fa f1(sum[0],c1,a[0],b[0],cin); fa f2(sum[1],c2,a[1],b[1],c1); fa f3(sum[2],c3,a[2],b[2],c2); fa f4(sum[3],carry,a[3],b[3],c3); endmodule

**//code for 2x1 mux:** module mux\_2x1(y, i0,i1,s); output y; input i0,i1,s; wire sbar; assign sbar=~s; assign y=(sbar&i0) | (s&i1); endmodule

//code for full\_adder module fa(sum, carry, a, b, cin); output sum; output carry; input a; input b; input cin; wire w1,w2,w3; ha h1(w1,w2,a,b); ha h2(sum,w3,cin,w1); or h3(carry,w3,w2); endmodule

**//code for half\_adder:**

module ha(sum, carry, a, b); output sum; output carry; input a; input b; xor g1(sum,a,b); and g2(carry,a,b); endmodule **//test bench:** module carry\_save\_177\_tb\_v;

// Inputs reg [3:0] a; reg [3:0] b; reg cin; // Outputs wire [3:0] sum; wire cout;

// Instantiate the Unit Under Test (UUT) carry\_save\_adder\_177 uut (

.a(a),

.b(b),

.cin(cin),

.sum(sum),

.cout(cout)

);

initial begin // Initialize Inputs a = 0;

b = 0; cin = 0;

// Wait 100 ns for global reset to finish

#100;a=4'b1010;b=4'b0000;cin=0;

#100;a=4'b0000;b=4'b1111;cin=0; // Add stimulus here end endmodule

**//RA2111004010177**

**//Output simulation waveform**:

