

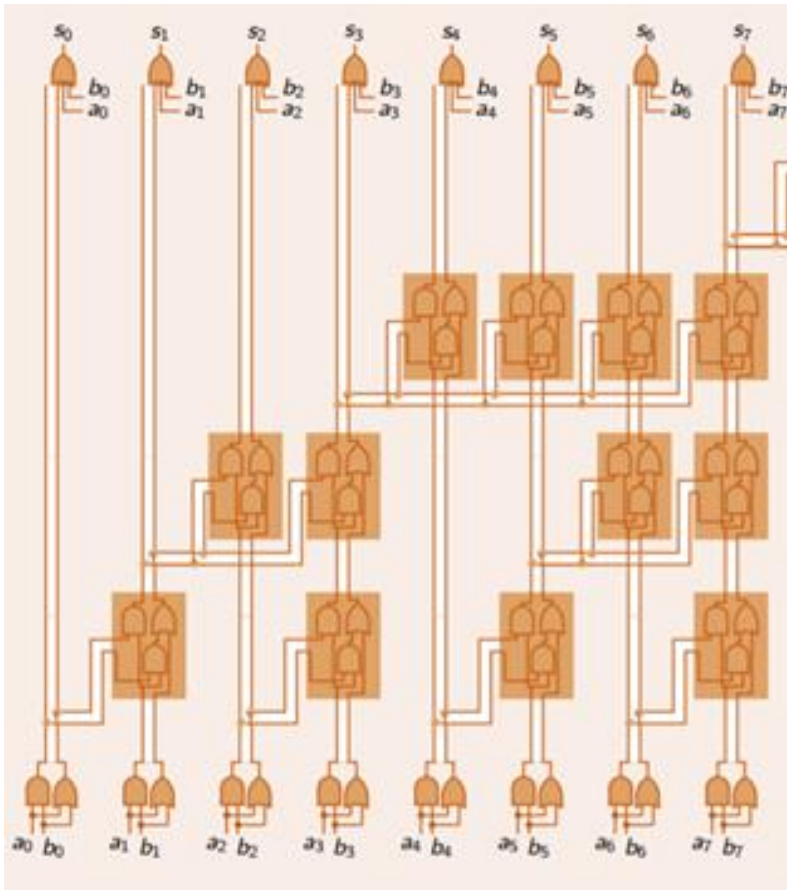
# **DDCO Mini Project**

## **Project Title: 8-bit Prefix Adder** **Section: D**

### **Batch Details**

Student Name	SRN
KAUSHAL SURESH	PES1UG20CS201
KESHAV MN	PES1UG20CS207
KIRAN SS	PES1UG20CS212
M B DHARVA	PES1UG20CS234

### **Circuit Diagram:**



## **MAIN CODE – IMPLEMENTATION FILE:**

### **prefix\_add.v:**

```
module and2 (input wire i0, i1, output wire o);
```

```
    assign o = i0 & i1;
```

```
endmodule
```

```
module or2 (input wire i0, i1, output wire o);
```

```
assign o = i0 | i1;
```

```
endmodule
```

```
module xor2 (input wire i0, i1, output wire o);
```

```
    assign o = i0 ^ i1;
```

```
endmodule
```

```
module xor3 (input wire i0, i1, i2, output wire o);
```

```
    wire temp;
```

```
    xor2 xor2_0 (i0, i1, temp);
```

```
    xor2 xor2_1 (i2, temp, o);
```

```
endmodule
```

```
module prop_gen(input wire x,y, output wire p,g);
```

```
    or2 o(x,y,p);
```

```
    and2 a(x,y,g);
```

```
endmodule
```

```
module carry(input wire p1,p0,g1,g0, output wire p, g);
```

```
    wire t;
```

```
    and2 a1(p1,p0,p);
```

```

    and2 a2(p1,g0,t);

    or2 o2(t,g1,g);

endmodule

module prefixAdd(input wire [7:0]a,b, input wire cin, output wire [7:0] S);
    wire p[7:0],g[7:0];
    wire cp[7:0],cg[7:0];
    wire cp21,cg21,cp43,cg43,cp54,cg54,cp65,cg65,cp64,cg64;

    prop_gen p_0(a[0],b[0],p[0],g[0]);
    prop_gen p_1(a[1],b[1],p[1],g[1]);
    prop_gen p_2(a[2],b[2],p[2],g[2]);
    prop_gen p_3(a[3],b[3],p[3],g[3]);
    prop_gen p_4(a[4],b[4],p[4],g[4]);
    prop_gen p_5(a[5],b[5],p[5],g[5]);
    prop_gen p_6(a[6],b[6],p[6],g[6]);
    prop_gen p_7(a[7],b[7],p[7],g[7]);

    assign cp[0] = 1'b0;
    assign cg[0] = 1'b0;

    carry c1(p[0],1'b0,g[0],cin,cp[1],cg[1]);
    carry c3_0(p[2],p[1],g[2],g[1],cp21,cg21);
    carry c5_0(p[4],p[3],g[4],g[3],cp43,cg43);
    carry c7_0(p[6],p[5],g[6],g[5],cp65,cg65);

```

```
carry c2(p[1],cp[1],g[1],cg[1],cp[2],cg[2]);  
carry c3_1(cp21,cp[1],cg21,cg[1],cp[3],cg[3]);  
carry c6_0(p[5],cp43,g[5],cg43,cp54,cg54);  
carry c7_1(cp65,cp43,cg65,cg43,cp64,cg64);
```

```
carry c4(p[3],cp[3],g[3],cg[3],cp[4],cg[4]);  
carry c5_1(cp43,cp[3],cg43,cg[3],cp[5],cg[5]);  
carry c6_1(cp54,cp[3],cg54,cg[3],cp[6],cg[6]);  
carry c7_2(cp64,cp[3],cg64,cg[3],cp[7],cg[7]);
```

```
xor3 s0(a[0],b[0],cin,S[0]);  
xor3 s1(a[1],b[1],cg[1],S[1]);  
xor3 s2(a[2],b[2],cg[2],S[2]);  
xor3 s3(a[3],b[3],cg[3],S[3]);  
xor3 s4(a[4],b[4],cg[4],S[4]);  
xor3 s5(a[5],b[5],cg[5],S[5]);  
xor3 s6(a[6],b[6],cg[6],S[6]);  
xor3 s7(a[7],b[7],cg[7],S[7]);
```

```
endmodule
```

## TESTBENCH FILE CODE:

### tb\_prefix\_add.v:

```
module tb_prefix;
reg [7:0] t_a,t_b;
reg t_cin;
wire [7:0] t_S;
initial begin $dumpfile("dump.vcd"); $dumpvars(0, tb_prefix); end
prefixAdd pra (.a(t_a), .b(t_b), .cin(t_cin), .S(t_S));
initial
begin
    t_a [7:0] = 8'b00000000; //0
    t_b [7:0] = 8'b00101001; //41
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b01100100; //100
    t_b [7:0] = 8'b00011000; //24
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b00010100; //20
    t_b [7:0] = 8'b10110010; //178
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b00100001; //33
    t_b [7:0] = 8'b01001011; //75
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b01100100; //100
    t_b [7:0] = 8'b00110010; //50
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b01011010; //90
    t_b [7:0] = 8'b00101000; //40
    t_cin = 1'b0;
    #5
    t_a [7:0] = 8'b10110001; //177
    t_b [7:0] = 8'b00110110; //54
    t_cin = 1'b0;
```

```
#5
t_a [7:0] = 8'b01011010; //90
t_b [7:0] = 8'b00111100; //60
t_cin = 1'b0;
#5
t_a [7:0] = 8'b00011000; //24
t_b [7:0] = 8'b00101011; //43
t_cin = 1'b0;
#10
t_a [7:0] = 8'b00000000;
t_b [7:0] = 8'b00000001;
t_cin = 1'b0;
end
endmodule
```

## Output:

