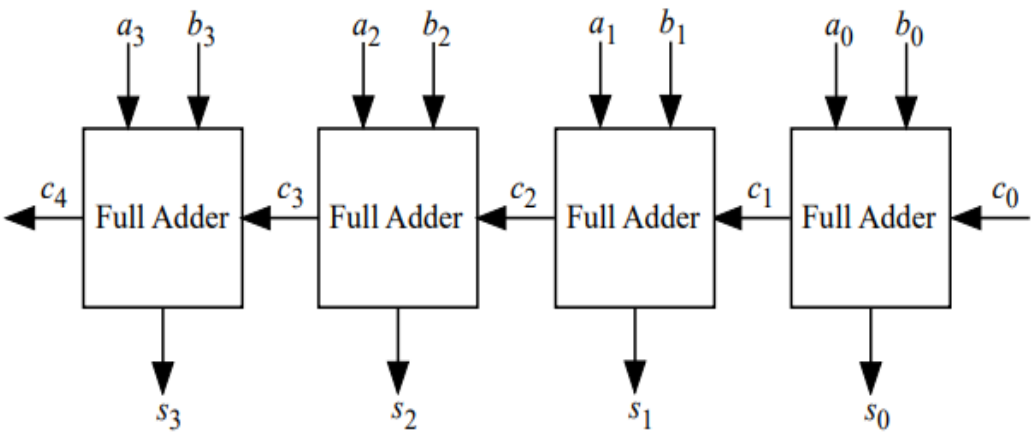
3rd Day

Ripple Carry Adder Using FA



Design

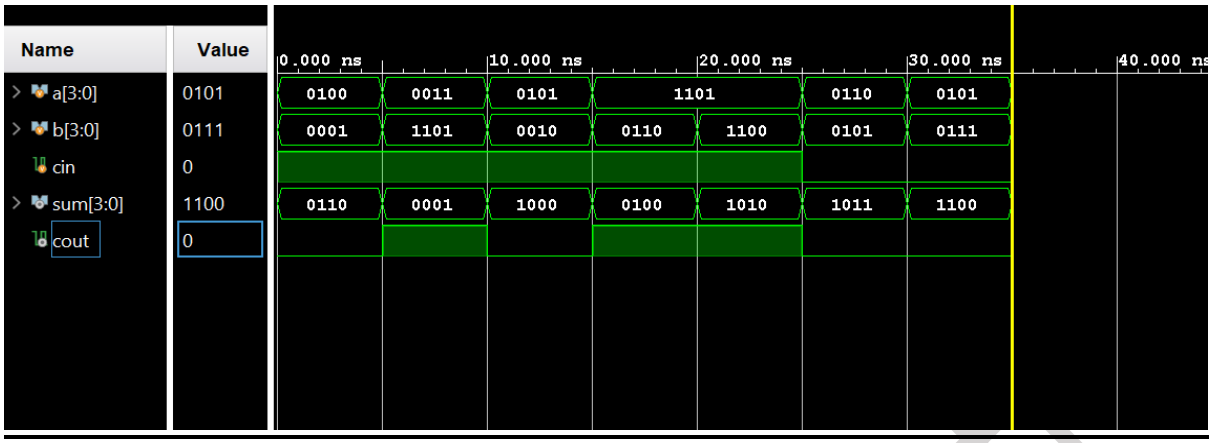
```
module full_adder(input a,b,cin,sum,cout);
assign sum=a^b^cin;
assign cout=a&b | b&cin | cin&a;
endmodule


module rca_4bit(input [3:0]a,b, input cin, output[3:0]sum,output cout);
wire [2:0]w;
full_adder m1(a[0],b[0],cin,sum[0],w[0]);
full_adder m2(a[1],b[1],w[0],sum[1],w[1]);
full_adder m3(a[2],b[2],w[1],sum[2],w[2]);
full_adder m4(a[3],b[3],w[2],sum[3],cout);
endmodule
```

Testbench

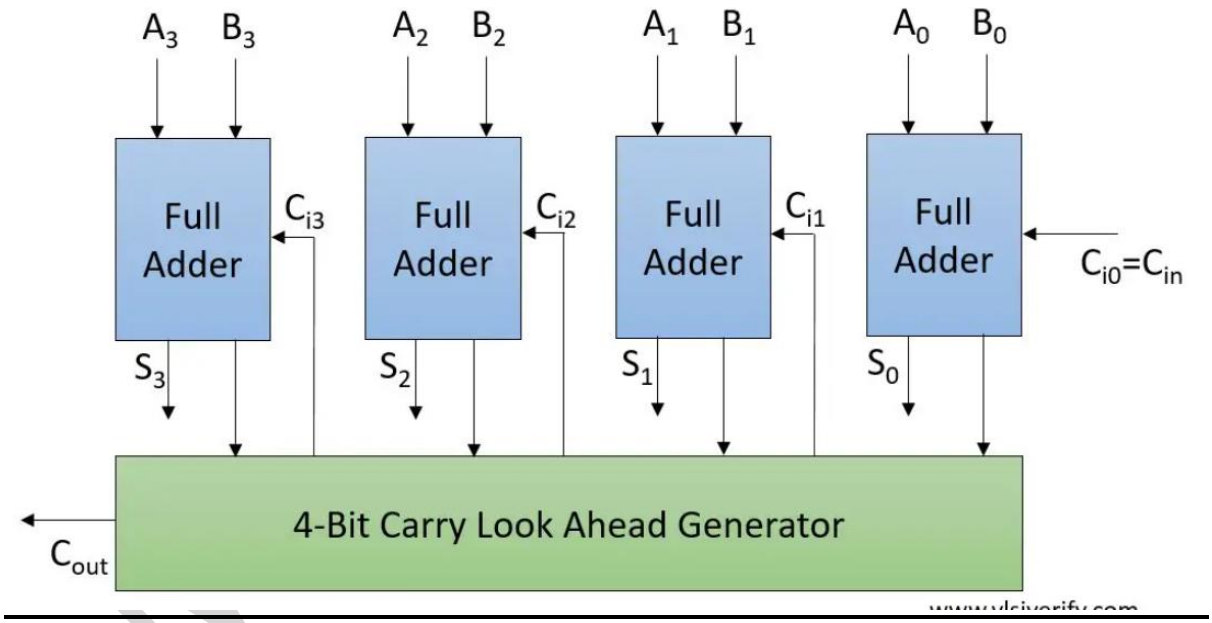
```
module rca_tb();
reg [3:0]a,b;
reg cin;
wire [3:0]sum;
wire cout;
rca_4bit dut(a,b,cin,sum,cout);
initial
begin
   $monitor("a=%b b=%b cin=%b sum=%b cout=%b",a,b,cin,sum,cout);
   repeat(7)
      begin
        a=$random;
        b=$random;
        cin=$random;
        #5;
      end
 $finish();
end
endmodule
```

Simulation

```
# run 1000ns
a=0100 b=0001 cin=1 sum=0110 cout=0
a=0011 b=1101 cin=1 sum=0001 cout=1
a=0101 b=0010 cin=1 sum=1000 cout=0
a=1101 b=0110 cin=1 sum=0100 cout=1
a=1101 b=1100 cin=1 sum=1010 cout=1
a=0110 b=0101 cin=0 sum=1011 cout=0
a=0101 b=0111 cin=0 sum=1100 cout=0
$finish called at time : 35 ns : File "C:/Users/manojmsd/100_days_of_RTL/100_days_of_RTL.srcs/sources_1/new/r
INFO: [USF-XSim-96] XSim completed. Design snapshot 'rca_tb_behav' loaded.
```



Carry Look ahead adder using FA



For each bit position i, we define:

- **Propagate:**

$$P_i = A_i \oplus B_i$$

  - If $P_i = 1$, it means carry will propagate from C_i to C_{i+1}.
- **Generate:**

$$G_i = A_i \cdot B_i$$

  - If $G_i = 1$, it means a new carry is generated at this bit position.

3 Sum Calculation S_i

Sum is computed using the formula:

$$S_i = P_i \oplus C_i$$

Since $P_i = A_i \oplus B_i$, this simplifies to:

$$S_i = (A_i \oplus B_i) \oplus C_i$$

Instead of waiting for carry to ripple, we directly compute carry using:

$$C_{i+1} = G_i + (P_i \cdot C_i)$$

Expanding further:

$$C_1 = G_0 + (P_0 \cdot C_0)$$

$$C_2 = G_1 + (P_1 \cdot C_1) = G_1 + (P_1 \cdot (G_0 + P_0C_0))$$

$$C_3 = G_2 + (P_2 \cdot C_2) = G_2 + (P_2 \cdot (G_1 + P_1G_0 + P_1P_0C_0))$$

$$C_4 = G_3 + (P_3 \cdot C_3) = G_3 + (P_3 \cdot (G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0))$$

This allows us to compute all carries simultaneously, making CLA much faster than Ripple Carry Adder.

Design

```
module fa(input a,b,cin,output sum,cout,p,g);
assign sum = a ^ b ^ cin;
assign cout = (a & b) | (b & cin) | (a & cin);
assign p = a ^ b;
assign g = a & b;
endmodule

module cla_adder(input [3:0]a,b,input cin,output [3:0]sum,output cout);
wire [3:0]p,g,c;
fa FA0 (a[0], b[0], cin, sum[0], , p[0], g[0]);
fa FA1 (a[1], b[1], c[1], sum[1], , p[1], g[1]);
fa FA2 (a[2], b[2], c[2], sum[2], , p[2], g[2]);
fa FA3 (a[3], b[3], c[3], sum[3], , p[3], g[3]);

assign c[0] = cin;
assign c[1] = g[0] | (p[0] & cin);
assign c[2] = g[1] | (p[1] & g[0]) | (p[1] & p[0] & cin);
assign c[3] = g[2] | (p[2] & g[1]) | (p[2] & p[1] & g[0]) | (p[2] & p[1] & p[0] & cin);
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] & cin);

endmodule
```

Testbench

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

   cla_adder dut (a, b, cin, sum, cout);
   initial
   begin
        $monitor("a=%b b=%b cin=%b | sum=%b cout=%b", a, b, cin, sum, cout);
         repeat(7)
            begin
            {a,b,cin}=$random;
             #5;
             end
          $finish;
    end
endmodule
```

Simulation

```
# run 1000ns
a=1001 b=0010 cin=0 | sum=1011 cout=0
a=0100 b=0000 cin=1 | sum=0101 cout=0
a=0000 b=0100 cin=1 | sum=0101 cout=0
a=0011 b=0001 cin=1 | sum=0101 cout=0
a=1000 b=0110 cin=1 | sum=1111 cout=0
a=1100 b=0110 cin=1 | sum=0011 cout=1
a=0011 b=0010 cin=1 | sum=0110 cout=0
$finish called at time : 35 ns : File "C:/Users/manojmsd/100_days_of_RTL/100_days_of_RTL.srcs/sources_1/new/cla_adder_tb.v" Line 41
INFO: [USF-XSim-96] XSim completed. Design snapshot 'cla_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:04 ; elapsed = 00:00:13 . Memory (MB): peak = 1609.766 ; gain = 0.000
```