Assignment-2

EE2003: Computer Organization

Name: Aayush Patel

Roll No.: EE21B003

Question (P15)

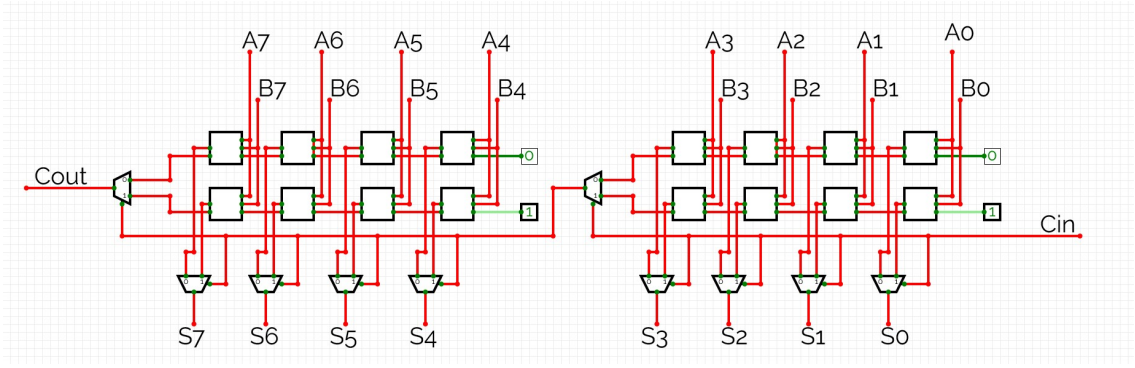
Design a 8-bit carry select adder module in Verilog.

The module should have two 8-bit inputs (A and B) and a 1-bit input Cin. It should output an 8-bit sum Sum and a 1-bit carry-out Cout.

Use two 4-bit ripple carry adders and a multiplexer to implement the carry select adder.

Carry Select Adder

I will be using four 4bit-ripple carry adders and 4 multiplexers. The design we will follow is provided below.



(In the image I have shown different muxes for each bit, but in Verilog I made the mux for 4-bits.)

Source Code

```
`timescale 1ns / 1ps
//MULTIPLEXER
module mux2to1 (
  input [3:0]A,
  input [3:0]B,
  input Sel,
  output [3:0]Y
);
assign Y = (Sel) ? B : A;
endmodule

//RIPPLE CARRY ADDER
module ripple_carry_adder (
  input [3:0] A,
  input [3:0] B,
  input Cin,
  output [3:0] Sum,
  output Cout
);

wire [3:0] rippleSum;
wire C0, C1, C2, C3;
```

```
// Calculate the Sum and Carry-out for each bit
assign rippleSum[0] = A[0] ^ B[0] ^ Cin;
assign C0 = (A[0] & B[0]) | (Cin & (A[0] ^ B[0]));
assign rippleSum[1] = A[1] ^ B[1] ^ C0;
assign C1 = (A[1] & B[1]) | (C0 & (A[1] ^ B[1]));
assign rippleSum[2] = A[2] ^ B[2] ^ C1;
assign C2 = (A[2] & B[2]) | (C1 & (A[2] ^ B[2]));
assign rippleSum[3] = A[3] ^ B[3] ^ C2;
assign C3 = (A[3] & B[3]) | (C2 & (A[3] ^ B[3]));

// Output Sum and final Carry-out
assign Sum = rippleSum;
assign Cout = C3;
endmodule

//CARRY SELECT ADDER
module carry_select_adder (
  input [7:0] A,
  input [7:0] B,
  input Sel,
  output [7:0] Sum,
  output Cout
);

wire [3:0] A0, B0, A1, B1;
```

```
wire [3:0] Sum00, Sum01, Sum10, Sum11;
wire Cout00, Cout01, Cout10, Cout11, Cout0, Cout1; // Carry-out from the 4-bit adders and select signal

// Split input A and B into two 4-bit parts each
assign {A1, A0} = A;
assign {B1, B0} = B;

// Four 4-bit ripple carry adders
ripple_carry_adder rca0 (
  .A(A0),
  .B(B0),
  .Cin(1'b0), // Carry-in is 0 for the first adder
  .Sum(Sum00),
  .Cout(Cout00)
);

ripple_carry_adder rca1 (
  .A(A0),
  .B(B0),
  .Cin(1'b1), // Carry-in is 1 for the second adder
  .Sum(Sum01),
  .Cout(Cout01)
);
```

```
ripple_carry_adder rca2 (
  .A(A1),
  .B(B1),
  .Cin(1'b0), // Carry-in is 0 for the first adder
  .Sum(Sum10),
  .Cout(Cout10)
);

ripple_carry_adder rca3 (
  .A(A1),
  .B(B1),
  .Cin(1'b1), // Carry-in is 1 for the second adder
  .Sum(Sum11),
  .Cout(Cout11)
);

// Create multiplexers to select the appropriate Sum and Cout
mux2to1 mux_sum0 (
  .A(Sum00),
  .B(Sum01),
  .Sel(Sel),
  .Y(Sum[3:0])
);

mux2to1 mux_sum1 (
```

```
  .A(Sum10),
  .B(Sum11),
  .Sel(Cout0),
  .Y(Sum[7:4])
);

mux2to1 mux_cout0 (
  .A(Cout00),
  .B(Cout01),
  .Sel(Sel),
  .Y(Cout0)
);

mux2to1 mux_cout1 (
  .A(Cout10),
  .B(Cout11),
  .Sel(Cout0),
  .Y(Cout1)
);

Endmodule
```

Testbench

```
`timescale 1ns / 1ps

module testbench;

 reg [7:0] A, B;
 reg Cin;
 wire [7:0] Sum;
 wire Cout;

 carry_select_adder uut (
  .A(A),
  .B(B),
  .Sel(Cin),
  .Sum(Sum),
  .Cout(Cout)
 );

 initial begin
  $dumpfile("carry_select_adder.vcd");
  $dumpvars(0, testbench);
```

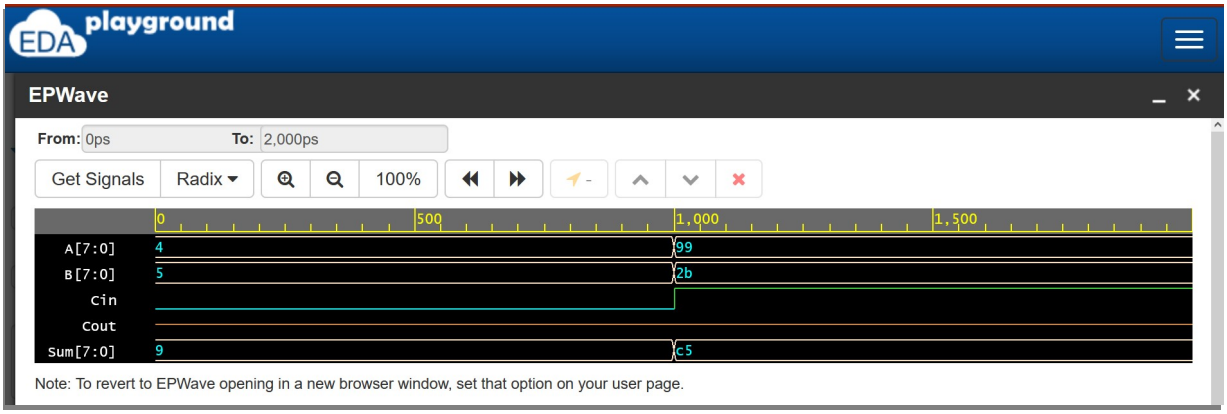
```
    // Test case 1: A = 4, B = 5, Cin = 0
    A = 4;
    B = 5;
    Cin = 0;
    #1 $display("Test Case 1: A = %d, B = %d, Cin = %d, Sum = %d, Cout = %d", A, B, Cin, Sum, Cout);

    // Test case 2: A = 153, B = 43, Cin = 1
    A = 153;
    B = 43;
    Cin = 1;
    #1 $display("Test Case 2: A = %d, B = %d, Cin = %d, Sum = %d, Cout = %d", A, B, Cin, Sum, Cout);

    $finish;
  end
endmodule
```

Waveform – Output

I have used “EDA playground” to generate the simulations/waveforms.



(4+5+0=9)

And

(153+43+1=197 i.e 99+2b+1=c5)