ASSIGNMENT: 2

ES 204: DIGITAL SYSTEMS

Prof. Joycee Mekie

Indian Institute of Technology, Gandhinagar

Aditya N. Mehta

22110017

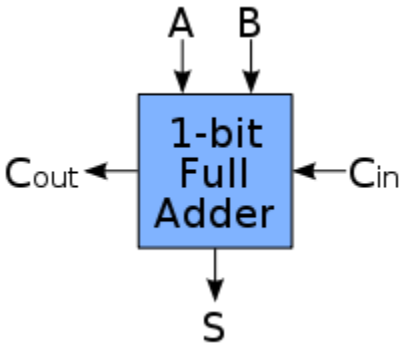
Hrriday V. Ruparel

22110099

Table of Contents

0) Creating a 1-bit full Adder

A) Diagram Representation:

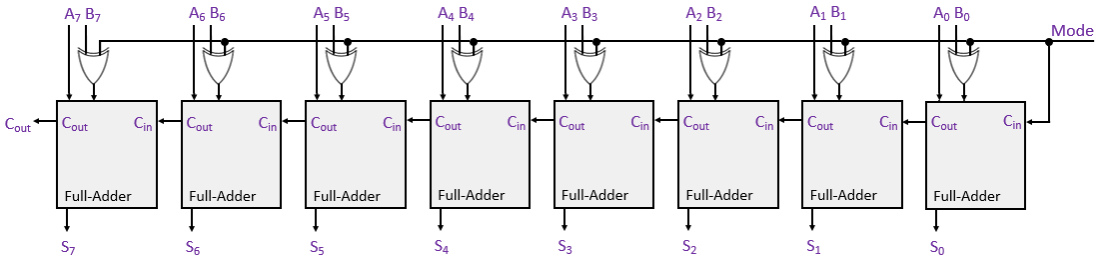


B) Verilog Code:

```
`timescale 1ns / 1ps

module FullAdder1Bit(A,B,Cin,Cout,Sum);
input A,B,Cin;
output Cout,Sum;

   and and1(f,A,B);
   and and2(g,B,Cin);
   and and3(h,Cin,A);
   or or1(Cout,f,g,h);

   and and4(i,A,~B,~Cin);
   and and5(j,~A,~B,Cin);
   and and6(k,~A,B,~Cin);
   and and7(m,A,B,Cin);

   or or2(Sum,i,j,k,m);

endmodule
```

1) Creating an 8-bit Adder/Subtractor

A) Diagram Representation:



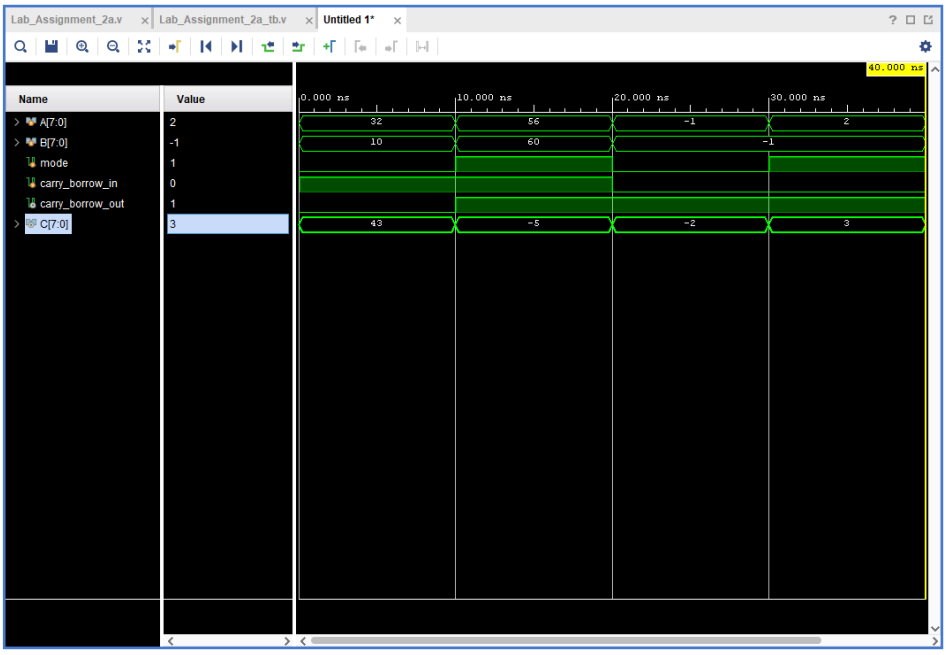
B) Verilog Code:

```
`timescale 1ns / 1ps

module Lab_Assignment_2a(A,B,carry_borrow_in,carry_borrow_out,C,mode);
input [7:0]A;
input [7:0]B;
input mode;
input carry_borrow_in;
output [7:0]C;
output carry_borrow_out;
wire [6:0]carry;
   xor x(Cin, mode, carry_borrow_in);
   xor x0(B0,B[0],mode);
   FullAdder1Bit inst0
(.A(A[0]),.B(B0),.Cin(Cin),.Cout(carry[0]),.Sum(C[0]));
   xor x1(B1,B[1],mode);
   FullAdder1Bit inst1
(.A(A[1]),.B(B1),.Cin(carry[0]),.Cout(carry[1]),.Sum(C[1]));
   xor x2(B2,B[2],mode);
   FullAdder1Bit inst2
(.A(A[2]),.B(B2),.Cin(carry[1]),.Cout(carry[2]),.Sum(C[2]));
   xor x3(B3,B[3],mode);
   FullAdder1Bit inst3
(.A(A[3]),.B(B3),.Cin(carry[2]),.Cout(carry[3]),.Sum(C[3]));
   xor x4(B4,B[4],mode);
   FullAdder1Bit inst4
(.A(A[4]),.B(B4),.Cin(carry[3]),.Cout(carry[4]),.Sum(C[4]));
   xor x5(B5,B[5],mode);
   FullAdder1Bit inst5
(.A(A[5]),.B(B5),.Cin(carry[4]),.Cout(carry[5]),.Sum(C[5]));
   xor x6(B6,B[6],mode);
   FullAdder1Bit inst6
(.A(A[6]),.B(B6),.Cin(carry[5]),.Cout(carry[6]),.Sum(C[6]));
   xor x7(B7,B[7],mode);
   FullAdder1Bit inst7
(.A(A[7]),.B(B7),.Cin(carry[6]),.Cout(Cout),.Sum(C[7]));
   xor x8(carry_borrow_out,Cout,mode);
endmodule
```

C) Test Beach Code:

```
`timescale 1ns / 1ps
/*
A = First Number
B = Second Number
Mode =  1 (for Subtraction)
        0 (for Addition)
*/
module Lab_Assignment_2a_tb();
reg [7:0]A;
reg [7:0]B;
reg mode;
reg carry_borrow_in;
wire carry_borrow_out;
wire [7:0]C;
Lab_Assignment_2a uut(A,B,carry_borrow_in,carry_borrow_out,C,mode);
   initial
   begin
   A = 8'd32; B = 8'd10; carry_borrow_in = 1; mode = 0;
   #10;
   A = 8'd56; B = 8'd60; carry_borrow_in = 1; mode = 1;
   #10;
   A = 8'd255; B = 8'd255; carry_borrow_in = 0; mode = 0;
   #10;
   A = 8'd2; B = 8'd255; carry_borrow_in = 0; mode = 1;
   #10;
   $finish();
   end
Endmodule
```

D) Simulation Waveform:



2) Gray Code Convertor

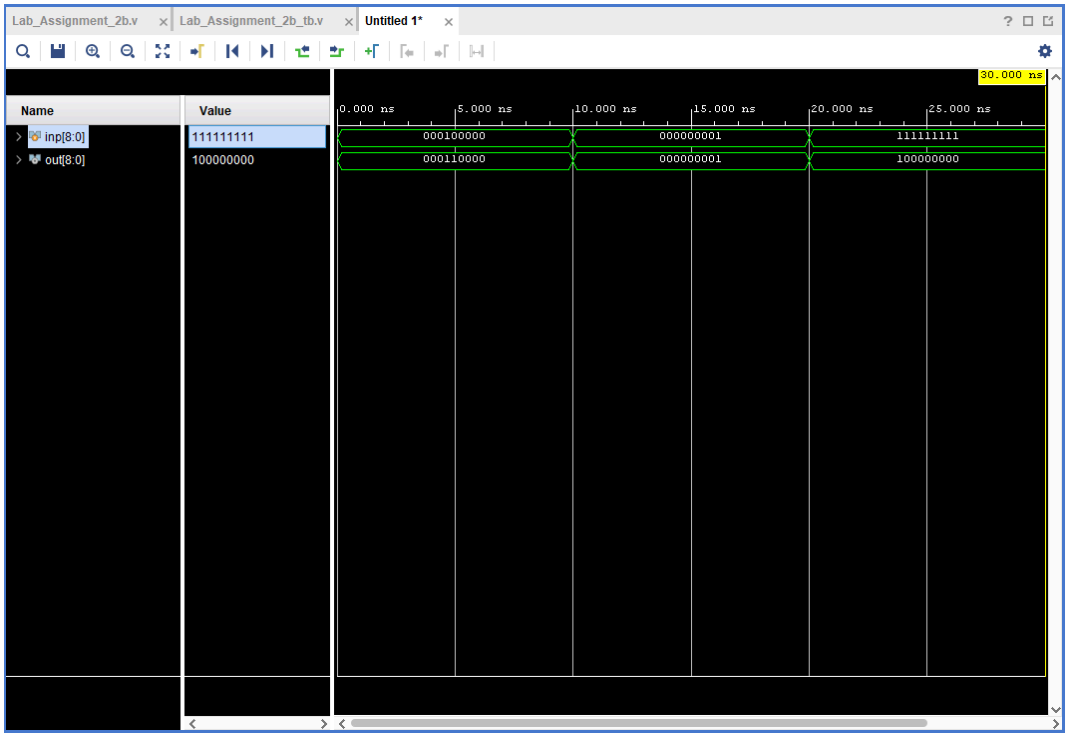
A) Verilog Code:

```
`timescale 1ns/1ps

module Lab_Assignment_2b(inp, out);
   input [8:0] inp;
   output reg [8:0] out;

 // inp[8] is the MSB
   always @(inp)
   begin

   out[8] <= inp[8];
   out[7] <= inp[7] ^ inp[8];
   out[6] <= inp[6] ^ inp[7];
   out[5] <= inp[5] ^ inp[6];
   out[4] <= inp[4] ^ inp[5];
   out[3] <= inp[3] ^ inp[4];
   out[2] <= inp[2] ^ inp[3];
   out[1] <= inp[1] ^ inp[2];
   out[0] <= inp[0] ^ inp[1];

   end
Endmodule
```

B) Test Bench Code:

```
`timescale 1ns/1ps

module Lab_Assignment_2b_tb();
reg [8:0]inp;
wire [8:0]out;

Lab_Assignment_2b uut(inp, out);

initial begin
//    $monitor($time, " inp = %d, out = %d", inp, out);
   inp = 9'd32; #10;
   inp = 9'd1; #10;
   inp = 9'd511; #10;
   $finish();

end
endmodule
```

C) Simulation Waveform:



3) Creating an 8-bit Adder/Subtractor

A) Verilog Code:

```
`timescale 1ns/1ps

module Lab_Assignment_2c(A, B, carry_borrow_in, mode, Gray_value);

input [7:0]A;
input [7:0]B;
input mode;
input carry_borrow_in;
output [8:0]Gray_value;
wire carry_borrow_out;
wire [7:0]C;

Lab_Assignment_2a add_sub (.A(A), .B(B),
.carry_borrow_in(carry_borrow_in),.carry_borrow_out(carry_borrow_out),
.C(C), .mode(mode));
Lab_Assignment_2b gray_convert(.inp({carry_borrow_out,C}),
.out(Gray_value));

endmodule
```

B) Test Bench Code:

```
`timescale 1ns / 1ps
/*
A = First Number
B = Second Number
Mode = 1 (for Subtraction)
       0 (for Addition)
*/
module Lab_Assignment_2c_tb();
reg [7:0]A;
reg [7:0]B;
reg carry_borrow_in;
reg mode;
wire [8:0]Gray_value;
Lab_Assignment_2c uut(A,B,carry_borrow_in,mode,Gray_value);
   initial
   begin
   A = 8'd32; B = 8'd10; carry_borrow_in = 1; mode = 0; #10;
   A = 8'd56; B = 8'd60; carry_borrow_in = 1; mode = 1; #10;
   A = 8'd255; B = 8'd255; carry_borrow_in = 0; mode = 0; #10;
   A = 8'd2; B = 8'd255; carry_borrow_in = 0; mode = 1; #10;
   $finish();
   end
endmodule
```

C) Simulation Waveform: