Lab Assignment – 2

Mumuksh Jain – 22110160

Nishant Kumar – 22110170

Code for a full adder circuit:

```
`timescale 1ns / 1ps


module fulladder(a,b,c,sum,co,sel);
    input a,b,c,sel;
    output co,sum;
    //to select negation of b when doing subtraction
        and(q,~sel,b);
        and(r,sel,~b);
        or(s,q,r);
    //finding out carry/borrow out
        and(e,a,s);
        and(f,s,c);
        and(g,c,a);
        or(co,e,f,g);
    //finding out sum/difference
        xor(h,a,s);
        xor(sum,h,c);


endmodule
```

Code for converting into Gray:

```
`timescale 1ns / 1ps


module graycode(
```

```
  input [7:0] a,
  input c,
  output [8:0]out,
  input sel
);

or(out[8],c,c);
xor(out[7],c,a[7]);
xor(out[6],a[7],a[6]);
xor(out[5],a[6],a[5]);
xor(out[4],a[5],a[4]);
xor(out[3],a[4],a[3]);
xor(out[2],a[3],a[2]);
xor(out[1],a[2],a[1]);
xor(out[0],a[1],a[0]);

endmodule
```

Code for instantiation of modules:

```
`timescale 1ns / 1ps

module addsub1(
  input [7:0] a,
  input [7:0] b,
  input c,
  output co,
  output [7:0] sum,
  input sel,
  output [8:0] outp
);
```

```
  wire c1, c2, c3, c4, c5, c6, c7,c8;

      fulladder a1(.a(a[0]), .b(b[0]), .c(sel), .sum(sum[0]), .co(c1),.sel(sel));
      fulladder a2(.a(a[1]), .b(b[1]), .c(c1), .sum(sum[1]), .co(c2),.sel(sel));
      fulladder a3(.a(a[2]), .b(b[2]), .c(c2), .sum(sum[2]), .co(c3),.sel(sel));
      fulladder a4(.a(a[3]), .b(b[3]), .c(c3), .sum(sum[3]), .co(c4),.sel(sel));
      fulladder a5(.a(a[4]), .b(b[4]), .c(c4), .sum(sum[4]), .co(c5),.sel(sel));
      fulladder a6(.a(a[5]), .b(b[5]), .c(c5), .sum(sum[5]), .co(c6),.sel(sel));
      fulladder a7(.a(a[6]), .b(b[6]), .c(c6), .sum(sum[6]), .co(c7),.sel(sel));
      fulladder a8(.a(a[7]), .b(b[7]), .c(c7), .sum(sum[7]), .co(c8),.sel(sel));
      assign co = (sel == 1'b1) ? ~c8 : c8;
      graycode a9(.a(sum),.c(co),.out(outp),.sel(sel));
endmodule
```

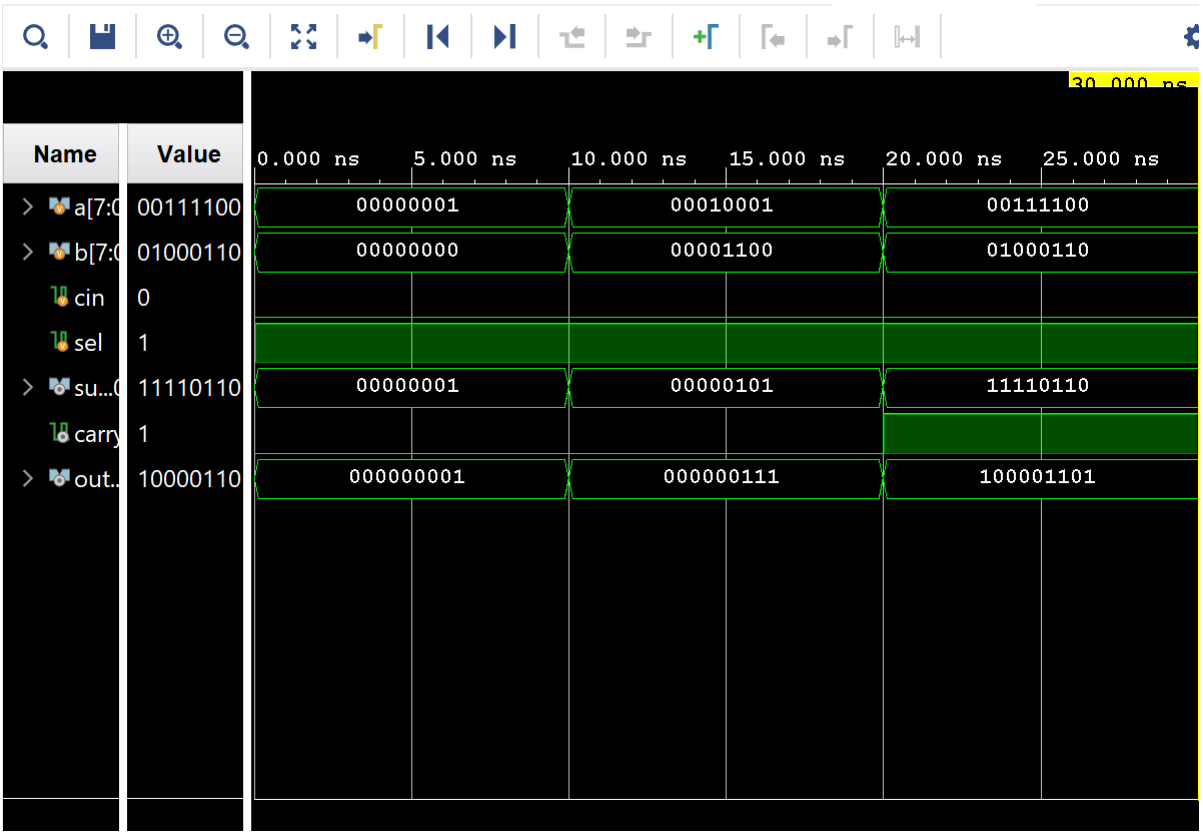
Testbench:

```
`timescale 1ns / 1ps

module addsub1_tb();
reg [7:0]a,b;
reg cin;
reg sel;
wire [7:0]sum;
wire carry;
wire [8:0]outp;

addsub1 uut(a, b, cin, carry, sum, sel, outp);

initial
begin
```

```
sel=1; // if 1, then subtraction and if 0, then adder
a=1; b=0; cin=0;
#10;
a=17; b=12; cin=0;
#10;
a=60; b=70; cin=0;
#10;
$finish();
end
endmodule
```

Simulation:

Here sel=1, that means subtraction is happening.



If sel=0, it means adder circuit so below is the simulation