Seung Ki Lee

CSE 3381 <Digital Logic Design N14 1172>

Post-Lab 04

1. Write Up

   I chose to implement the logic function 4-1 MUX, because I felt that from example 7, this was the best explained the behavior of this mux at varying inputs. I learnt how to alter the given top level Verilog code to work according to the designed input.

2. Source Code (mux41.v, counter.v, clkdiv.v, count8_top.v)

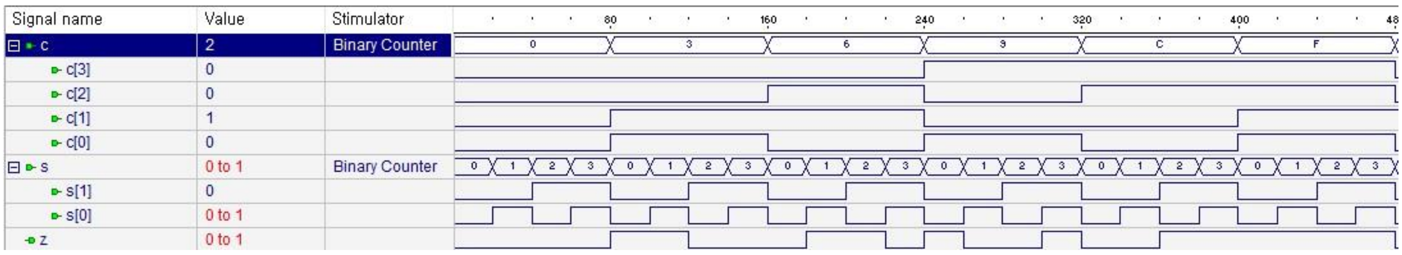
```
//: 4-to-1 MUX using logic equation

module mux41b (

input wire [3:0] c ,

input wire [1:0] s ,

output wire z

);

assign z = ~s[1] & ~s[0] & c[0]

| ~s[1] & s[0] & c[1]

| s[1] & ~s[0] & c[2]

| s[1] & s[0] & c[3];

endmodule


//Counter.v

module counter

#(parameter N = 8)

(input wire clr ,

input wire clk ,

output reg [N-1:0] q

);

// N-bit counter

always @(posedge clk or posedge clr)
```

```
begin
if(clr == 1)
q <= 0;
else
q <= q + 1;
end
endmodule

//clock divider
module clkdiv (
input wire clk ,
input wire clr ,
output wire clk190 ,
output wire clk25 ,
output wire clk3
);
reg [23:0] q;
// 24-bit counter
always @(posedge clk or posedge clr)
begin
if(clr == 1)
q <= 0;
else
q <= q + 1;
end
assign clk190 = q[17]; // 190 Hz
assign clk25 = q[0]; // 25 MHz
assign clk3 = q[0]; // 3 Hz
endmodule
```

```
// count8_top

module count8_top (

input wire mclk,

input wire [3:3] btn,

output wire [7:0] ld

) ;


wire clk3;


clkdiv U1

( .clk3(clk3),

.clr(btn[3]),

.clk(mclk)

);


counter

#( .N(8))

U2

( .clk(clk3),

.clr(btn[3]),

.q(ld[7:0])

);

Endmodule
```

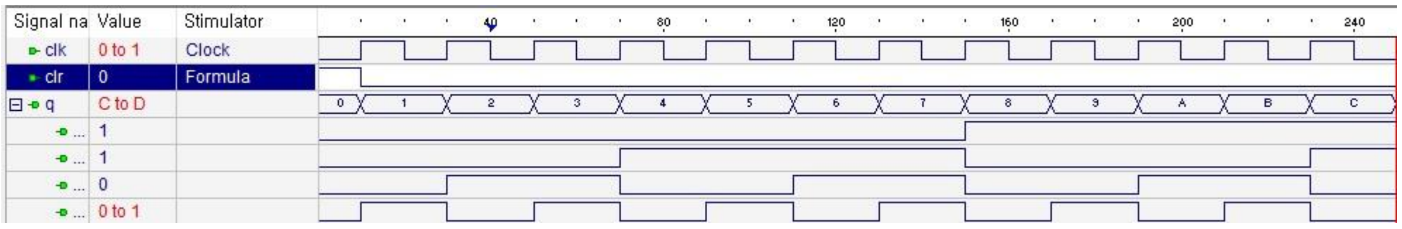
3. Simulation (mux41, counter)

MUX41



Signal name	Value	Stimulator
⊟ c	2	Binary Counter
c[3]	0	
c[2]	0	
c[1]	1	
c[0]	0	
⊟ s	0 to 1	Binary Counter
s[1]	0	
s[0]	0 to 1	
z	0 to 1	

Counter



4. Demo & Prelab Code

   Demo-Code: 5853

   Prelab-Code: 4585