4-Digit-Code: 6354

Seung Ki Lee

CSE 3381 <Digital Logic Design N14 1172>

Post-Lab 03

1. Write Up

   I chose to use alway(*) block and if statement because it is more intuitive depiction of how I understand the MUX. Especially as a beginner in pure verilog, I feel that visible and intuitive method is the best for understanding the logic. The required reading presented multiple ways of expressing one logic function and explained why each block worked.

2. Source Code (mux21.v, mux21_top.v, mux24.v)

   MUX21

```
//-----------------------------------------------------------------------------

`timescale 1 ns / 1 ps


//{{ Section below this comment is automatically maintained
//      and may be overwritten
//{module {mux21}}
module mux21 (
input wire a ,
input wire b ,
input wire s ,
output reg y
);
//}} End of automatically maintained section


// -- Enter your statements here -- //
always @(*)
 if(s == 0)
            y = a;
 else
```

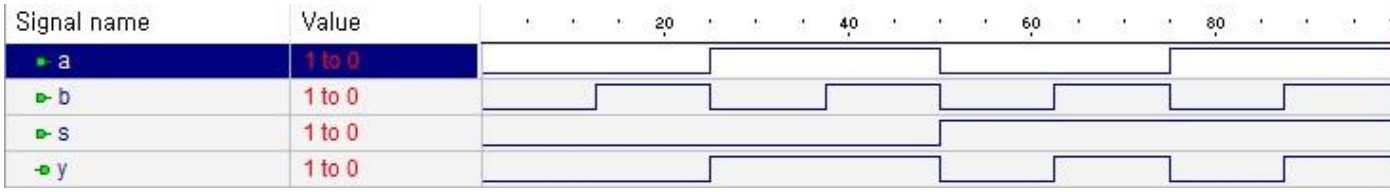
```
        y = b;

endmodule
```

MUX24

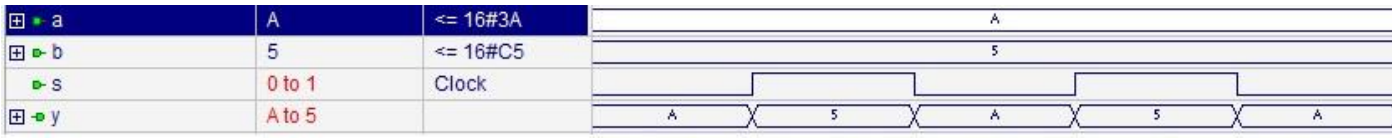
```
//----------------------------------------------------------------------------
`timescale 1 ns / 1 ps

//{{ Section below this comment is automatically maintained
//      and may be overwritten
//{module {mux24}}
module mux24 (
input wire [3:0] a,
input wire [3:0] b,
input wire s,
output reg [3:0] y
);
//}} End of automatically maintained section

// -- Enter your statements here -- //
always @(*)
 if(s == 0)
        y = a;
 else
        y = b;
endmodule
```

3. Simulation (mux21.v, mux24.v)

MUX21


Signal name
Value
20
40
60
80
a
1 to 0
b
1 to 0
s
1 to 0
y
1 to 0

MUX24


a
A
<= 16#3A
b
5
<= 16#C5
s
0 to 1
Clock
y
A to 5
A
5
A
5
A