Seung Ki Lee

CSE 3381 <Digital Logic Design N14 1172>

Post-Lab 05

1. Write Up & Prelab Code

   In this lab I've learnt how to alter the parameters of code I already know to solve a new problem. Learning to change to clk inputs enhanced my understanding of what each command does. Drawing out the k-map from 7segment decoder was good exercise, but I wish we'd gone a bit more into the ways different columns interact. The logic functions from K-map helped me visualize where the Verilog code came from.

   Prelab-Code: 8830

2. Source Code (hex7seg.v, hex7seg_top.v, mux7seg.v, x7seg.v, x7seg_top.v, clkdiv.v, counter.v)

   Hex7seg.v

```
module hex7seg (
input wire [3:0] x ,
output reg [6:0] a_to_g
);
always @(*)
case(x)
0: a_to_g = 7'b0000001;
1: a_to_g = 7'b1001111;
2: a_to_g = 7'b0010010;
3: a_to_g = 7'b0000110;
4: a_to_g = 7'b1001100;
5: a_to_g = 7'b0100100;
6: a_to_g = 7'b0100000;
7: a_to_g = 7'b0001111;
8: a_to_g = 7'b0000000;
9: a_to_g = 7'b0000100;
'hA: a_to_g = 7'b0001000;
'hb: a_to_g = 7'b1100000;
```

```
'hC: a_to_g = 7'b0110001;
'hd: a_to_g = 7'b1000010;
'hE: a_to_g = 7'b0110000;
'hF: a_to_g = 7'b0111000;
default: a_to_g = 7'b0000001; // 0
endcase
endmodule
```

Hex7seg_top

```
module hex7seg _top (
input wire [3:0] sw ,
output wire [6:0] a_to_g ,
output wire [3:0] an ,
output wire dp
);
assign an = 4'b0000; // all digits on
assign dp = 1; // dp off
hex7seg D4 (.x(sw),
.a_to_g(a_to_g)
);
endmodule
```

MUX44

```
module mux44 (
input wire [15:0] x ,
input wire [1:0] s ,
output reg [3:0] z
);
always @(*)
case(s)
0: z = x[3:0];
1: z = x[7:4];
```

```
2: z = x[11:8];
3: z = x[15:12];
default: z = x[3:0];
endcase
endmodule
```

X7seg

```
module x7seg (
input wire cclk,
input wire clr,
input wire [15:0] x,
output wire [6:0] a_to_g,
output wire [3:0] an
);
wire nq0;
wire nq1;
wire [3:0] digit;
wire [1:0] q;
assign nq1 = ~(q[1]);
assign nq0 = ~(q[0]);
assign an[0] = q[0] | q[1];
assign an[1] = nq0 | q[1];
assign an[2] = q[0] | nq1;
assign an[3] = nq0 | nq1;
hex7seg U1
( .a_to_g(a_to_g),.x(digit));

mux44 U2
( .s({q[1:0]}),.x(x),.z(digit));

counter
```

```
#( .N(2)) U3
( .clk(cclk),.clr(clr),.q(q[1:0]));
endmodule
```

X7seg_top

```
module x7seg_top (
input wire mclk,
input wire [3:3] btn,
output wire dp,
output wire [6:0] a_to_g,
output wire [3:0] an
);
wire clk190;
wire [15:0] x;
assign x = 'hC5E;
assign dp = 1;
clkdiv U1
( .clk3(clk3),
.clr(btn[3]),
.clk(mclk)
);
x7seg U3
( .a_to_g(a_to_g),
.an(an),
.cclk(clk3),
.clr(btn[3]),
.x(x)
);
endmodule
```

Clkdiv

```
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[1]; // 3 Hz
endmodule
```

Counter

```
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

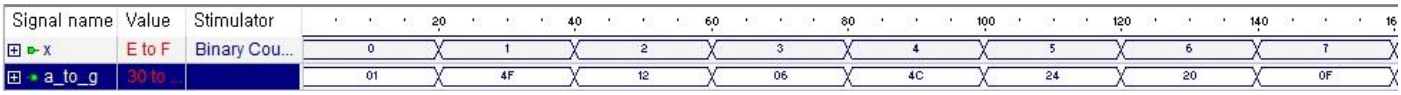
q <= q + 1;

end

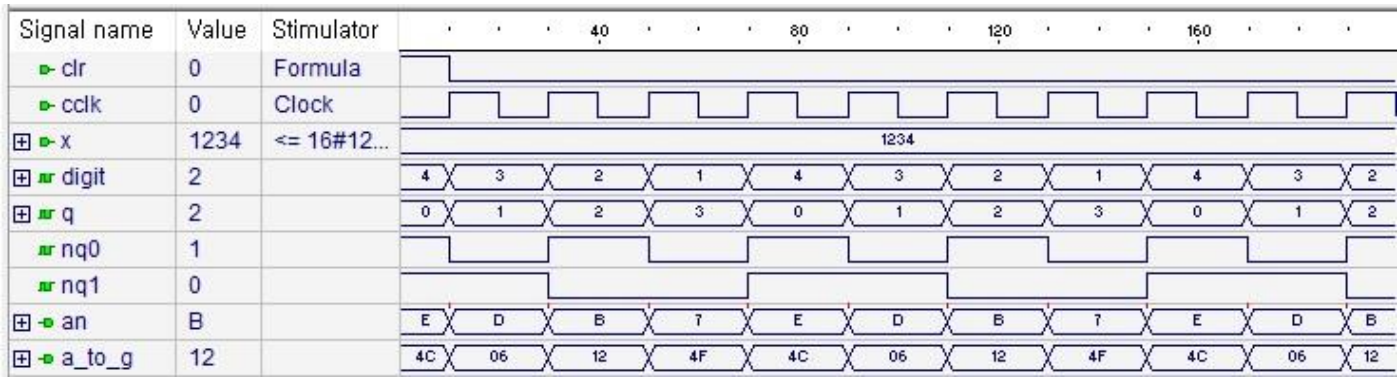
endmodule

3. Simulation (Example 9, Example 10)

Example 9



Example 10



4. Demo Code

Demo9-Code: 2432

Demo10-Code: 2373