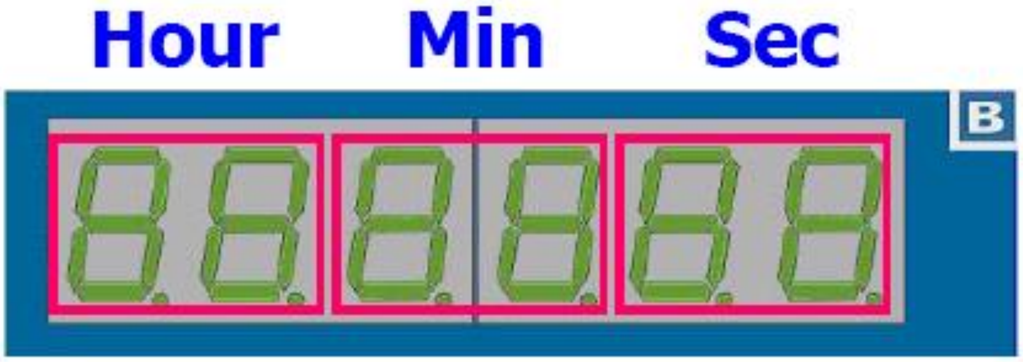
FPGA Lab-03

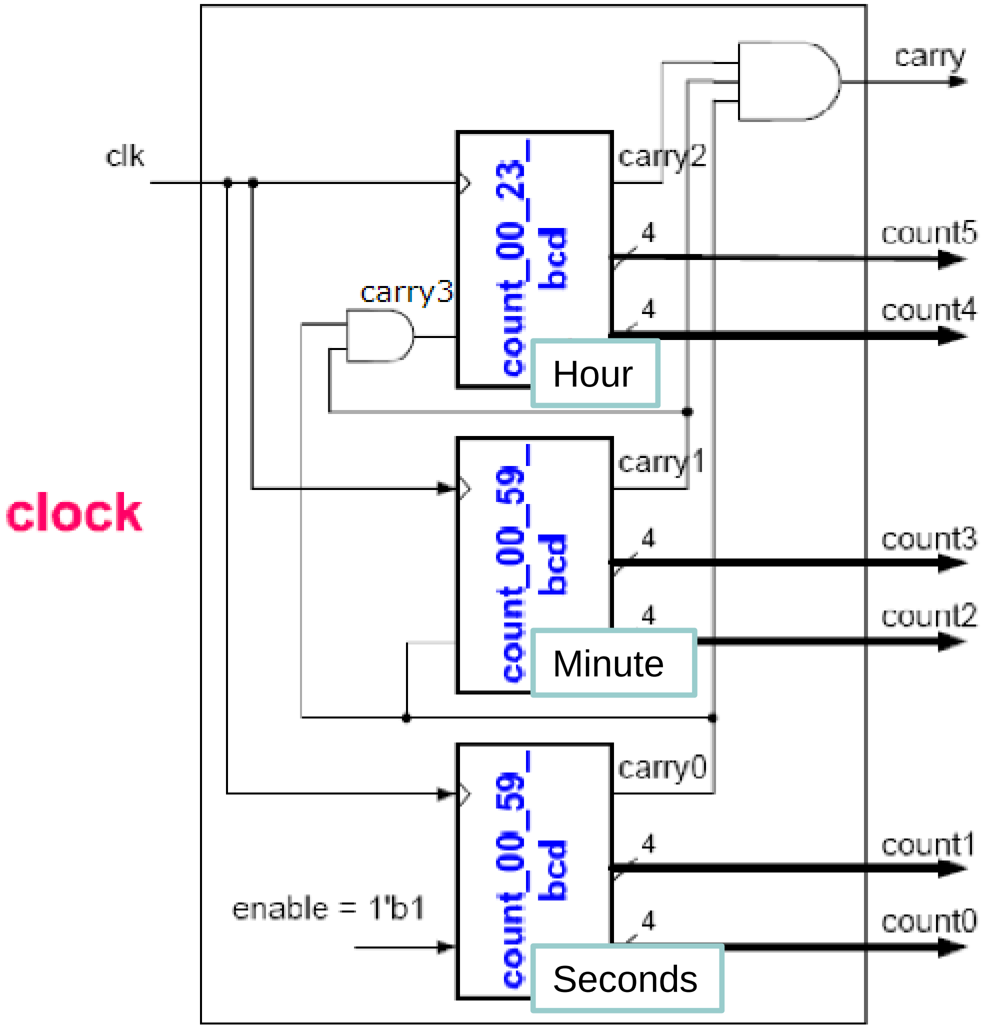
Application of seven-segment displays(2)--- Display a 24 hours clock

Lab

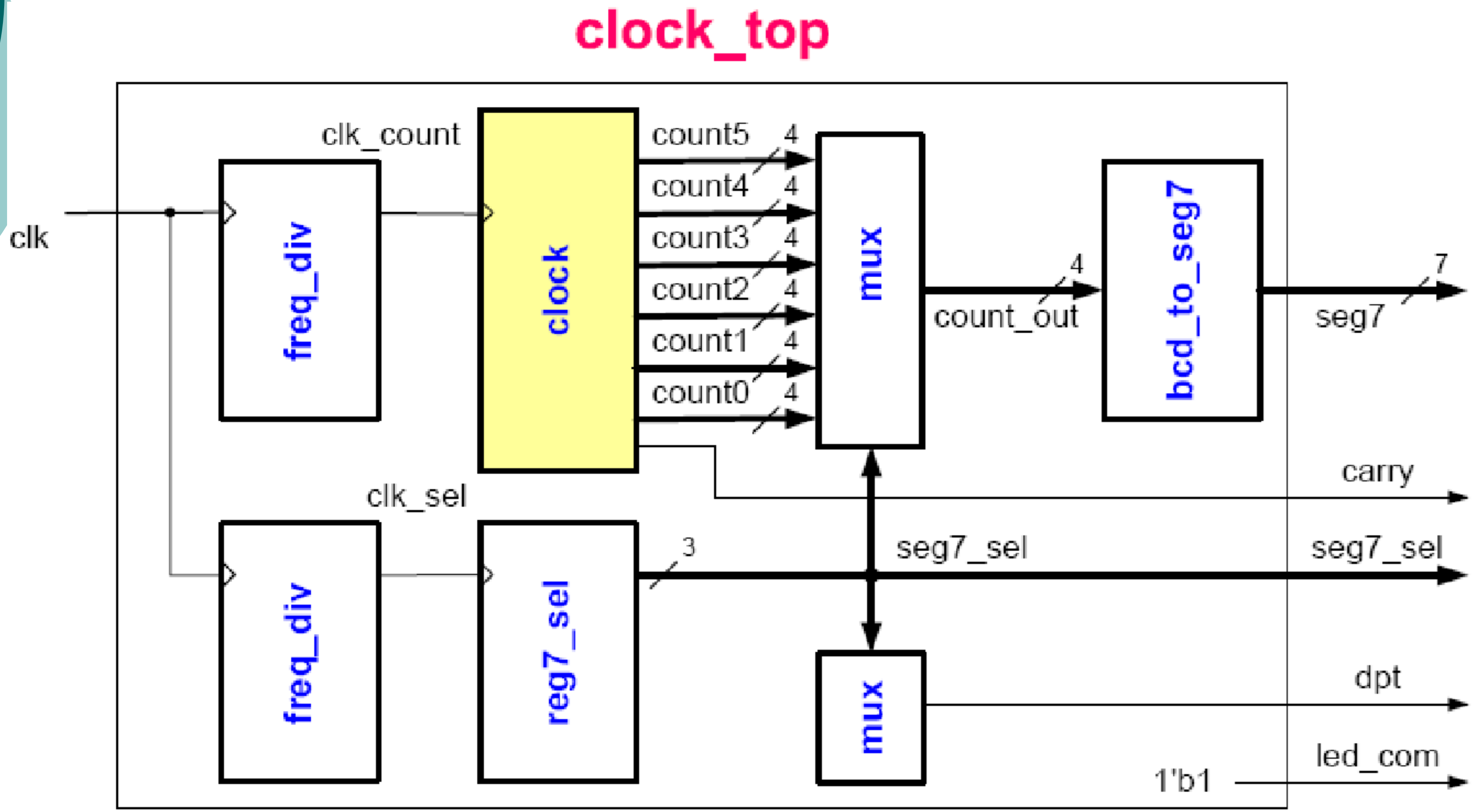
- Lab 1 : Design a counter from 00 to 59 。
- Lab 2 : Design a counter from 00-23 BCD Counter。
- Lab 3 : Design a 24 hours timer on display。



Architecture model for clock



Architecture model for **clock_top**



Verilog Code -count_00_59_bcd.v

```
module count_00_59_bcd(clk, reset, enable, count1, count0, carry);
input clk, reset, enable;
output [3:0] count1, count0;
reg[3:0] count1, count0;
output carry;
wire carry = (count1 == 4'b0101 && count0 == 4'b1001) ? 1 : 0;
always@ (posedge clk or posedge reset)begin
if(reset) begin
count1 = 4'b0000; // 00
count0 = 4'b0000;
end
```

```
else if(enable == 1'b1) begin
if (count1 == 4'b0101 && count0 == 4'b1001) begin// 59
count1 = 4'b0000; // 00
count0 = 4'b0000;
end
else if(count0 == 4'b1001) begin
count0 = 4'b0000;
count1 = count1+ 1'b1;
end
else
count0 = count0+ 1'b1;
end
end
endmodule
```

Verilog Code -count_00_23_bcd.v

```
module count_00_23_bcd(clk, reset, enable, count1, count0, carry);
input clk, reset, enable;
output[3:0] count1, count0;
reg[3:0] count1, count0;
output carry;
~~~~~your code~~~~~~~
always@ (posedge clk or posedge reset)begin
if(reset) begin
~~~~~your code~~~~~~~
end
```

```
else if(enable == 1'b1) begin
if (~~~~~your code~~~~~~~) begin// 23
 ~~~~~your code~~~~~~~
end
else if(~~~~~your code~~~~~~~) begin
 ~~~~~your code~~~~~~~
end
else
 ~~~~~your code~~~~~~~
end
end
endmodule
```

Verilog Code -clock.v

```
module clock(clk, reset, enable, count5, count4, count3, count2, count1, count0, carry);
input clk, reset, enable;
output[3:0]count5, count4, count3, count2, count1, count0;
output carry;
wire[3:0]count5, count4, count3, count2, count1, count0;
wire carry, carry3, carry2, carry1, carry0;
assign carry = ~~your code~~~ ;
assign carry3 = ~~your code~~~ ;
count_00_23_bcd(clk, reset, ~~your code~~~);
count_00_59_bcd(clk, reset, ~~your code~~~);
count_00_59_bcd(clk, reset, ~~your code~~~);
endmodule
```

Verilog Code -clock_top.v

Main module

```
module clock_top(clk, reset, enable, seg7_sel, seg7_out,
dpt, carry, led_com);
input clk, reset, enable;
output[2:0]seg7_sel;
output[6:0] seg7_out;
output dpt, carry, led_com;
wire clk_count, clk_sel;
wire[3:0] count_out, count5, count4, count3, count2,
count1, count0;
assign led_com= 1'b1;
assign count_out= (seg7_sel == 3'b101 )? count0 :
(seg7_sel == 3'b100 )? count1 :
(seg7_sel == 3'b011 )? count2 :
(seg7_sel == 3'b010 )? count3 :
(seg7_sel == 3'b001 )? count4 : count5;
```

```
assign dpt= (seg7_sel == 3'b101 )? 1'b1 :
(seg7_sel == 3'b100 )? 1'b0 :
(seg7_sel == 3'b011 )? 1'b1 :
(seg7_sel == 3'b010 )? 1'b0 :
(seg7_sel == 3'b001 )? 1'b1 :1'b0;

freq_div#(21)(your code);
freq_div#(15)(your code);
clock(~~your code~~~ );
bcd_to_seg7(~~your code~~~ );
seg7_select#(?) (~~your code~~~ );
endmodule
```

Extra Question

- Create a 24-hour clock with accurate reading time.

(Use the 10MHz clock for the seconds)