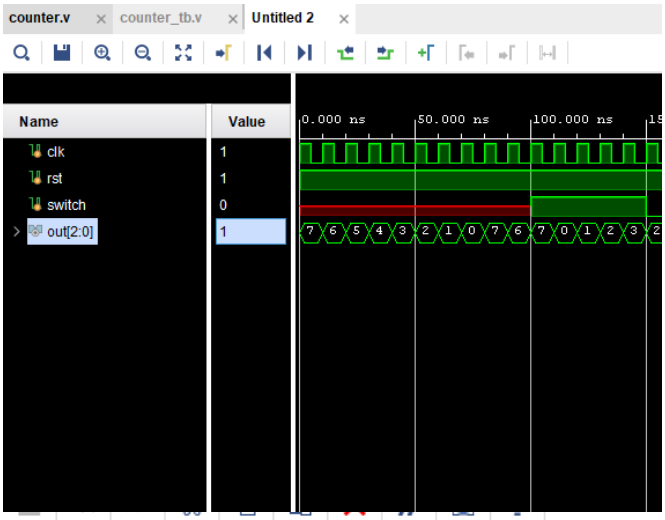
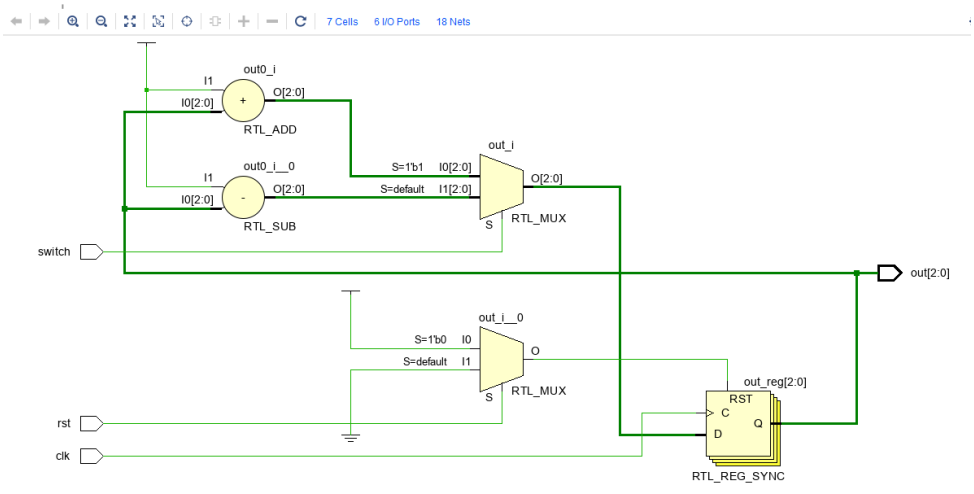
ASSINGMENT1

1) 0-7 aralığında anahtarı 0 yapınca geri 1 yapıncca ileriye doğuru sayan bir sayaç tarsarladım.

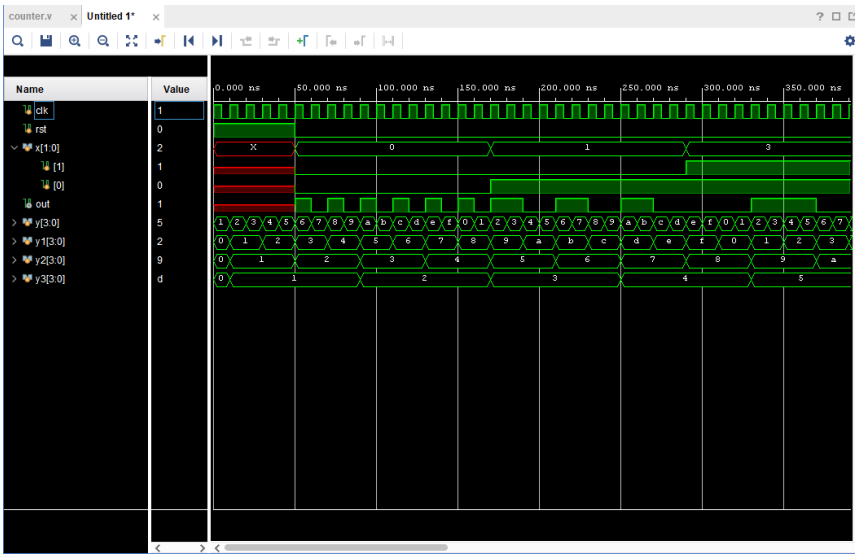


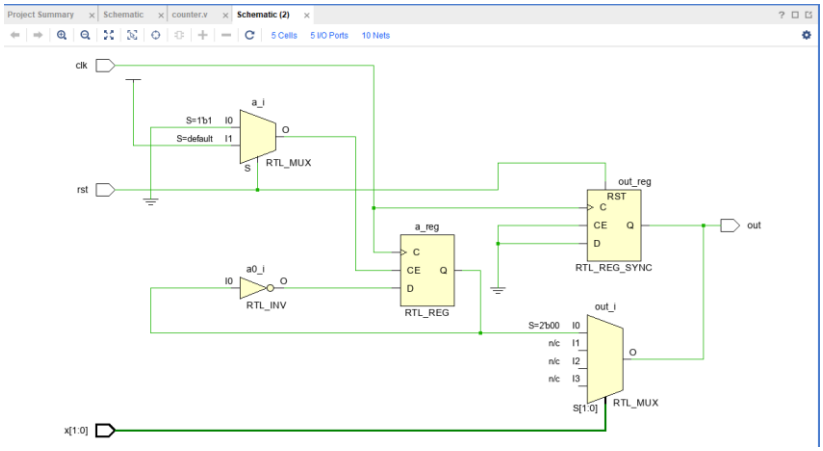

```
module counter(
    input clk, rst,switch,
    output reg [2:0] out= 3'b0
);

// File and modules names must be the same!!

always @(posedge clk) begin
    if (~rst)
        out <= 0;
    else begin
    if (switch) begin

        out <= out + 1;
        end
        else begin
        out<= out-1;
        end
    end
end
endmodule
```



2) Sayaçların hızını ayarlayabileceğimiz bir clock divider tasarladım.

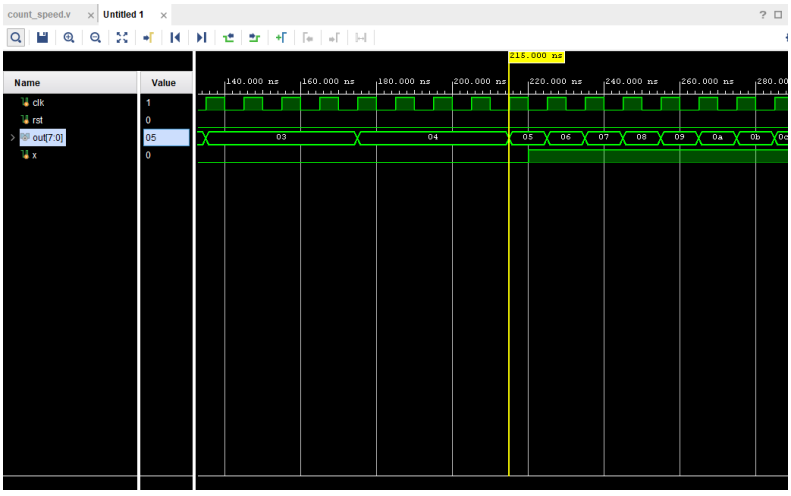
```
module clkdvd (


  input  clk,
  input [1:0] x,

  output reg out


);

reg a,b,c,d;

reg [3:0]y=0,y1=0,y2=0,y3=0 ;


  always@ (*) begin

  case(x)

  2'b00:    out=a;
  2'b01:    out=b;
  2'b11:    out=c;
  2'b10:    out=d;

  endcase

  end



  always @ (posedge clk) begin
   y <= y+1;
   if ((y %2)==1) begin
    a  = 1;
   end
   else begin
    a = 0;
   end
  end

 always @ (posedge a) begin
   y1 <= y1+1;
   if ((y %4)==1)  begin
    b = 1;
   end else begin
    b = 0;
   end
  end

  always @ (posedge b) begin
   y2 <= y2+1;
   if ((y %8)==1)  begin
    c = 1;
   end else begin
    c = 0;
   end
```

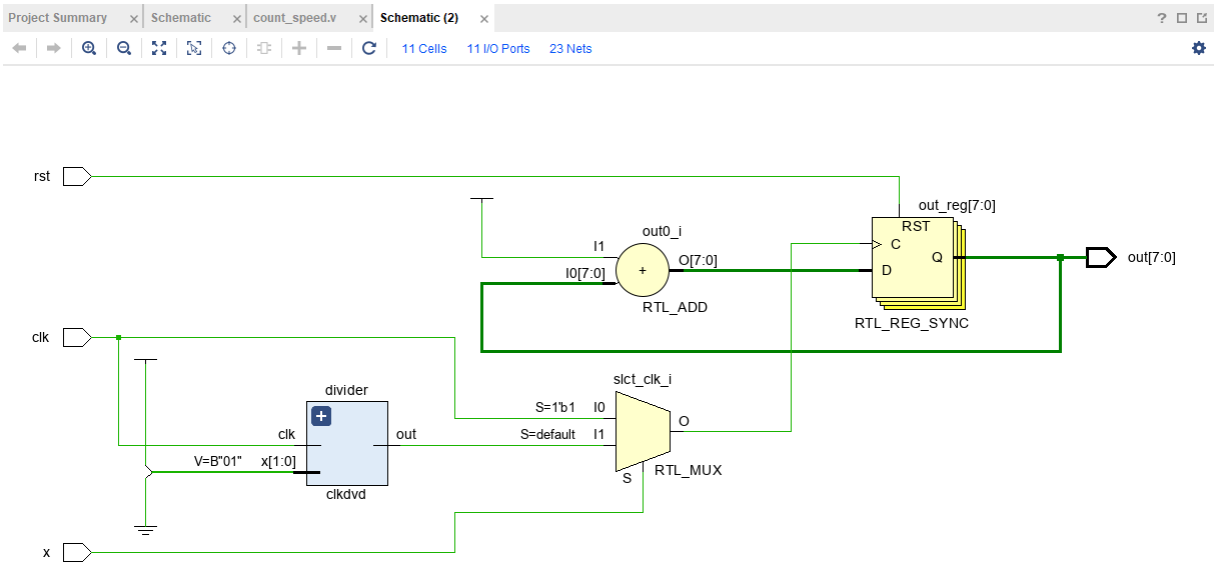




3) Sayacı hızlandırıp yavaşlatabileceğim bir count speed tasarladım.

```
module count_speed (

  input clk, rst, x,
   output reg [7:0]out
);
 wire slow_clk,slct_clk;
clkdvd divider(.out(slow_clk),.x(2'b01),.clk(clk));

 assign slct_clk= x? clk: slow_clk;


always @(posedge slct_clk) begin
    if (rst) begin
      out <= 8'b0;
    end else begin
     out <= out + 1;
  end
  end


endmodule
```

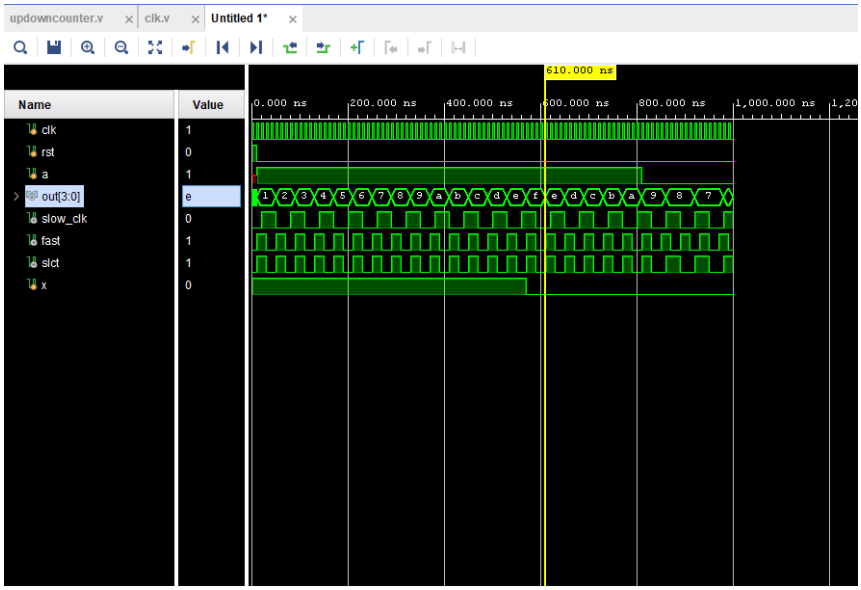


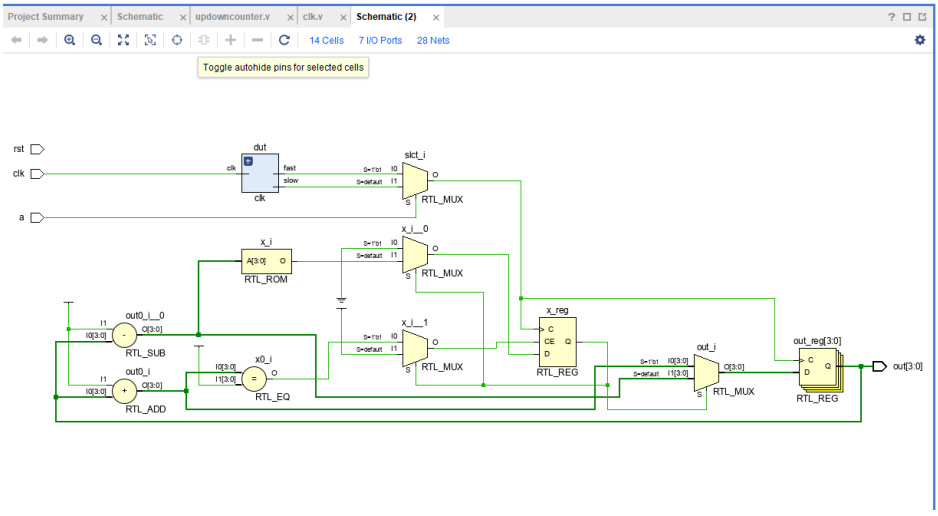


4)

a) Otomatik bir şekilde ileriye ve geriye sayabilen bir sayaç tasarladım.

```
module updowncounter(
  input clk,rst,a,
  output reg [3:0] out=4'b0000
);
 reg x=1;
 wire slow_clk,fast,slct;
clk dut (.slow(slow_clk),.clk(clk),.fast(fast));

assign slct = a? fast:slow_clk ;

always @(posedge slct) begin

/*if(rst)
out<=0;*/

      if (x) begin
       out = out + 1;
        if(out == 15)begin
           x = !x;
          end
       end
   else begin

      out = out - 1;
      if(out == 0)begin

        x = !x;

         end
      end
```



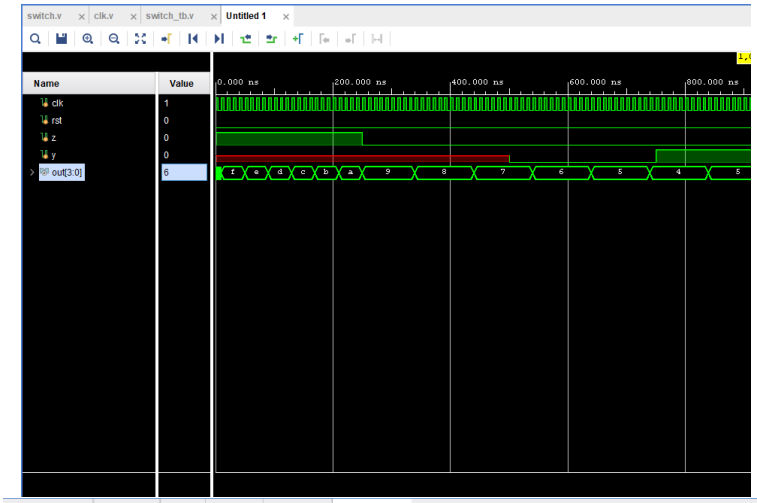


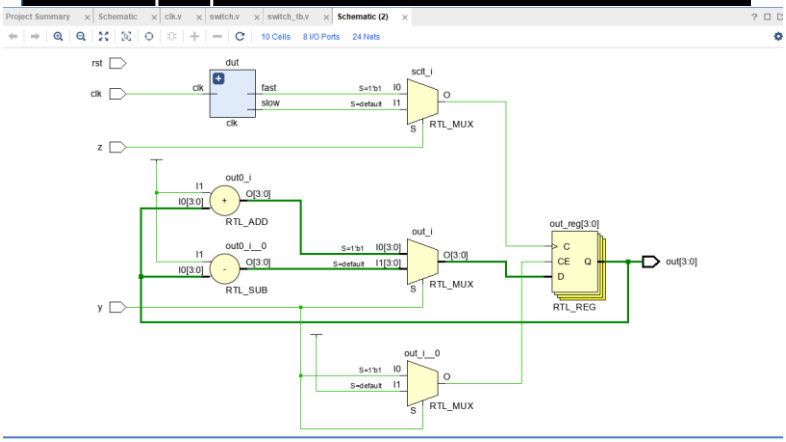
b) Bir anahtar yardımı ile sayacın hızını değiştirebileceğim bir modül tasarladım.

switch.v × clk.v × switch_tb.v × Untitled 1 ×

C:/Users/Can/Desktop/homework_1/proj1_v/project_1_4b/project_1_4b.srcs/sources_1

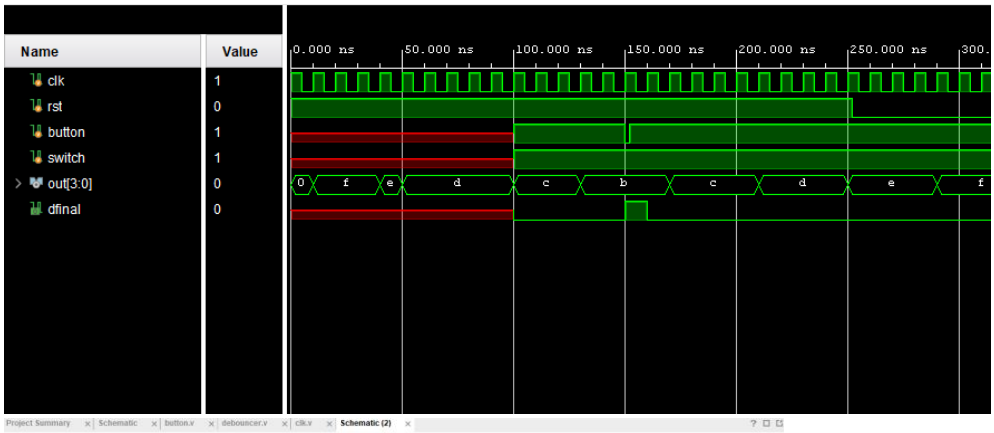
```
module switch (
  input clk,rst,z,y,
  output reg [3:0] out=4'b0000
);


wire sclt,slow,fast,counter;

clk dut (.slow(slow), .fast(fast),.clk(clk));
assign sclt = z? fast: slow;


always@ (posedge sclt) begin
if (y) begin
    out = out + 1;


   end
else begin
   out <= out - 1;


  end
end




endmodule
```

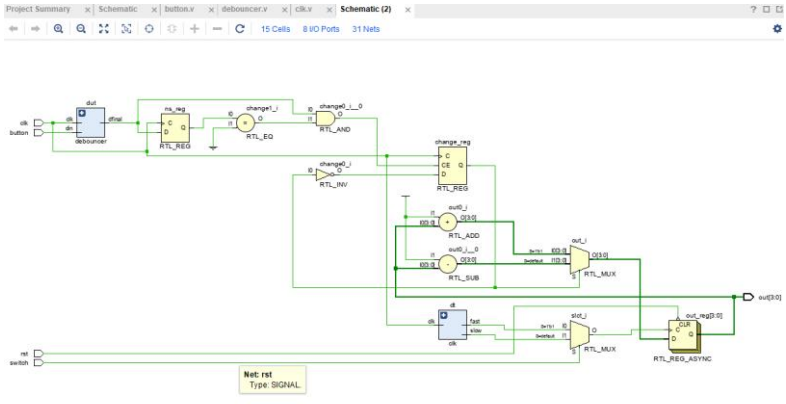




c) Hızını bir anahtar yardımı ile ileri ve geri saymasını da bir buton yardımı ile değiştirebileceğimiz bir modül tasarladım.

```
module button (

  input clk,rst, button,switch,
  output reg [3:0] out=4'b0

    );
  reg change=0,ns=0;
  wire slct,fast,slow,d_button_syc;

debouncer dut (.din(button),.clk(clk),.dfinal(d_button_syc));

clk_divider_main dt(.clk(clk),.slow(slow),.fast(fast));

assign slct = switch? fast:slow;

always@ (posedge clk) begin

                ns <= d_button_syc;

        if(d_button_syc==1 && ns==0)begin

            change<=!change;
   end
        end

always@ (posedge slct, negedge rst) begin
        if (~rst) begin

             out <= 0;
        end
     else begin
        if (change) begin
            out <= out+1;

        end

        else begin
            out <= out-1;
       end
    end
 end


endmodule
```





d) 4. Madde için yapmış olduğum modüle bir start-stop butonu daha ekledim.

```
            ns <= d_button_syc;
        if(d_button_syc==1 && ns == 0)begin

            change<=!change;
    end
        end

        always@ (posedge clk) begin

            ss <= start_stop_syc;

        if(start_stop_syc ==1 && ss == 0)begin

            start<=!start;
    end
        end

always@ (posedge slct, negedge rst) begin


        if (~rst) begin

            out <= 0;

            end
     else begin
        if (start) begin
            out<=out;
        end
        else begin
                if (change) begin

                 out <= out+1;

                 end

                else begin
                out <= out-1;
                end
            end
      end
end
```