تمرین اول

توجه: انجام این تمرین به‌صورت انفرادی است.

۱- الف) یک کدگشای[1] ۲ در ۴ را با زبان برنامه‌نویسی وریلاگ با دستورات تخصیص پیوسته[2] توصیف کنید و برای آن یک ماژول بسترآزمون (ماژول تحریک) نوشته که تمامی حالات را در ابزار شبیه‌سازی ModelSim مورد آزمون قرار دهد.

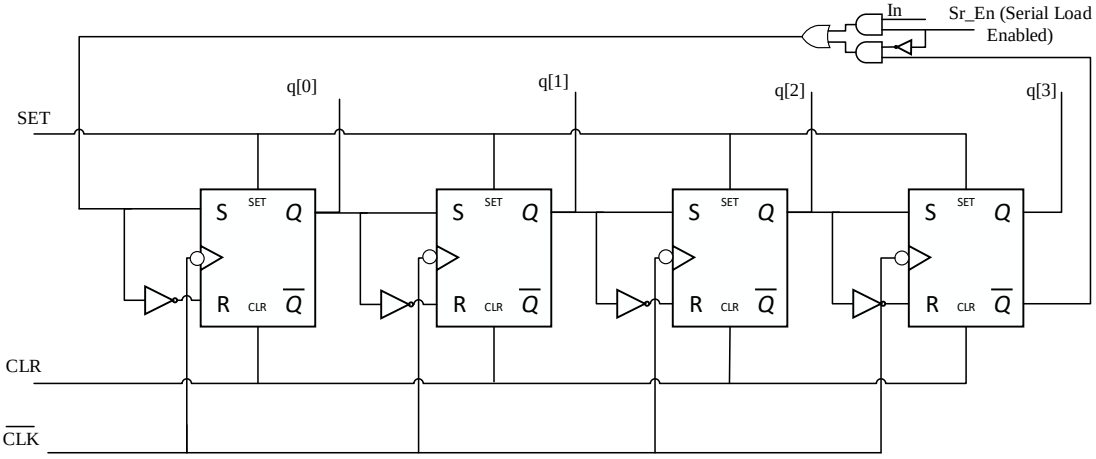
ب) سپس با استفاده از این کدگشای ۲ در ۴ یک کدگشای ۳ در ۸ طراحی کنید. برای این کدگشا یک ماژول بسترآزمون نوشته که تمامی حالات را در ابزار شبیه‌سازی ModelSim مورد آزمون قرار دهد.

۲- الف) سخت‌افزار زیر را در وریلاگ توصیف کنید (ورودی‌های SET و CLR به‌صورت ناهم‌زمان هستند).

ب) خروجی‌های ممکن سخت‌افزار زیر را مشخص کنید.

پ) برای آن یک ماژول بسترآزمون[3] (ماژول تحریک[4]) نوشته که تمامی حالات را در ابزار شبیه‌سازی ModelSim مورد آزمون قرار دهد.

توجه: سیگنال‌های ورودی و خروجی در توصیف وریلاگ با نام سخت‌افزار یکسان باشند.



۳- الف) کد وریلاگ زیر ضرب دو عدد علامت‌دار ۸ بیتی را توصیف می‌کند. برای آن یک ماژول بسترآزمون نوشته و آن را در ابزار ModelSim شبیه‌سازی کنید. سپس نتایج شبیه‌سازی را به ۳ روش (چاپ در خروجی، نمایش موج‌ها و فایل vcd) گزارش کنید.

ب) درصورتی‌که مدار دارای اشکال است، ورودی آزمونی که اشکال را تشخیص می‌دهد، مشخص کنید؛ اگر نیست، توضیح دهید چگونه از صحت مدار خود اطمینان یافتید.

---

[1] Decoder
[2] Continues Assignment
[3] Testbench
[4] Stimulus Block

```
`define width 8
`timescale 1ns/1ps

module mult (p, x, y);

  parameter width=`width;
  parameter N = `width/2;
  input[width-1:0]x, y;
  output[width+width-1:0]p;
  reg [2:0] cc[N-1:0];
  reg [width:0] pp[N-1:0];
  reg [width+width-1:0] spp[N-1:0];
  reg [width+width-1:0] prod;
  wire [width:0] inv_x;
  integer kk,ii;

  assign inv_x = {~x[width-1],~x}+1;

  always @ (x or y or inv_x)
  begin
    cc[0] = {y[1],y[0],1'b0};
    for(kk=1;kk<N;kk=kk+1)
      cc[kk] = {y[2*kk+1],y[2*kk],y[2*kk-1]};

    for(kk=0;kk<N;kk=kk+1) begin
      case(cc[kk])
        3'b001 , 3'b010 : pp[kk] = {x[width-1],x};
        3'b011 : pp[kk] = {x,1'b0};
        3'b100 : pp[kk] = {inv_x[width-1:0],1'b0};
        3'b101 , 3'b110 : pp[kk] = inv_x;
        default : pp[kk] = 0;
      endcase
      spp[kk] = $signed(pp[kk]);
      for(ii=0;ii<kk;ii=ii+1)
        spp[kk] = {spp[kk],2'b00};
    end //for(kk=0;kk<N;kk=kk+1)

    prod = spp[0];
    for(kk=1;kk<N;kk=kk+1)
      prod = prod + spp[kk];
  end

  assign p = prod;
```

```
endmodule
```