ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ

BÁO CÁO THÍ NGHIỆM 4

LAB 4: THỰC HIỆN HỆ TỔ HỢP VÀ HỆ TUẦN TỰ CƠ BẢN TRÊN FPGA

MÔN: KỸ THUẬT SỐ (TN) (EE1010)

GVHD: Nguyễn Trung Hiếu

Sinh viên thực hiện

Nhóm 6 – Lớp L21

1) Lâm Thành Phát MSSV: 2111974

2) Nguyễn Đăng Khoa MSSV: 2111529

3) Trần Thanh Tâm MSSV:2114720

TPHCM 12, 2022

A. HƯỚNG DẪN THÍ NGHIỆM

I. MỤC TIÊU

- Nắm được cách sử dụng kit thí nghiệm, phần mềm lập trình.
- Nắm được cách khảo sát và thiết kế hệ mạch đếm sử dụng các IC chức năng cơ bản.
- Nắm được quy trình mô tả phần cứng trên FPGA.

II. CHUẨN BỊ:

- Để chuẩn bị tốt cho bài thí nghiệm, sinh viên PHẢI đọc trước phần Phụ lục 1 và hoàn thành các bước của Sample lab trong Phụ lục 2.
- Sinh viên phải hoàn thành và nộp PRELAB 4 trước khi vào lớp.

III. HƯỚNG DẪN THÍ NGHIỆM

THÍ NGHIỆM 1

Mục tiêu: Nắm được các thức mô tả mạch tính giá trị tuyệt đối của một số 4 bit sử dụng ngôn ngữ systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên thực hiện mô tả mạch tính giá trị tuyệt đối của một số 4 bit (số ngõ vào lần lượt là A, ngõ ra là S).

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối và/hoặc diễn giải để giáo viên hiểu được ý tưởng của mình)

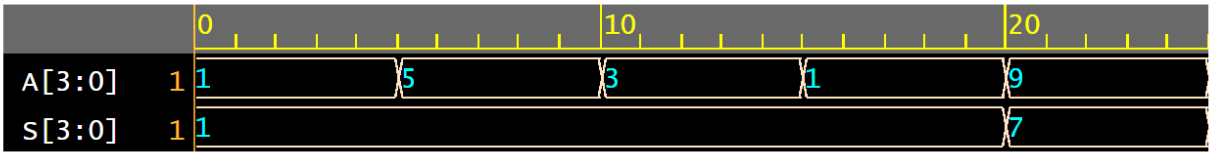
  Tạo biến sel = A[3].

- Nếu sel = 1 thì ngõ vào A là số âm 4 bit. Khi đó, $|A| = -A$.
- Nếu sel = 0 thì ngõ vào A là số không âm 4 bit. Khi đó, $|A| = A$.

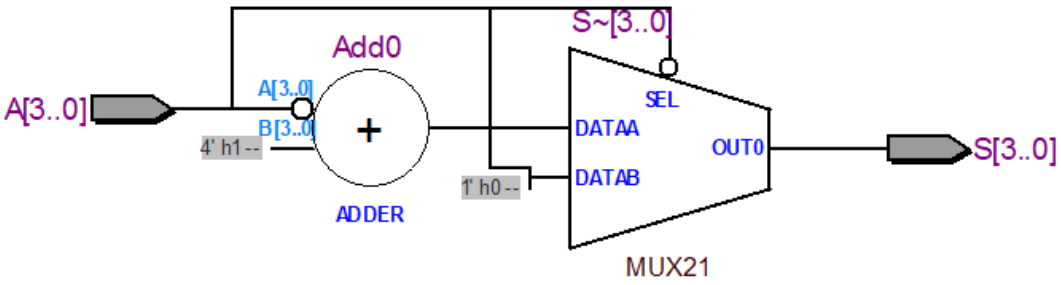
➢ Chương trình mô tả hoạt động của thiết kế.

```
// Code your design here
module LAB4_TN1(
  input logic [3:0] A,
  output logic [3:0] S
);
  assign sel = A[3];
  always@(sel) begin
    if (sel == 1'b0) begin
      S = A;
          end
  else begin
      S = 0 + ~A + 1;
           end
  end
  endmodule
```

➢ Kết quả mô phỏng dạng sóng.

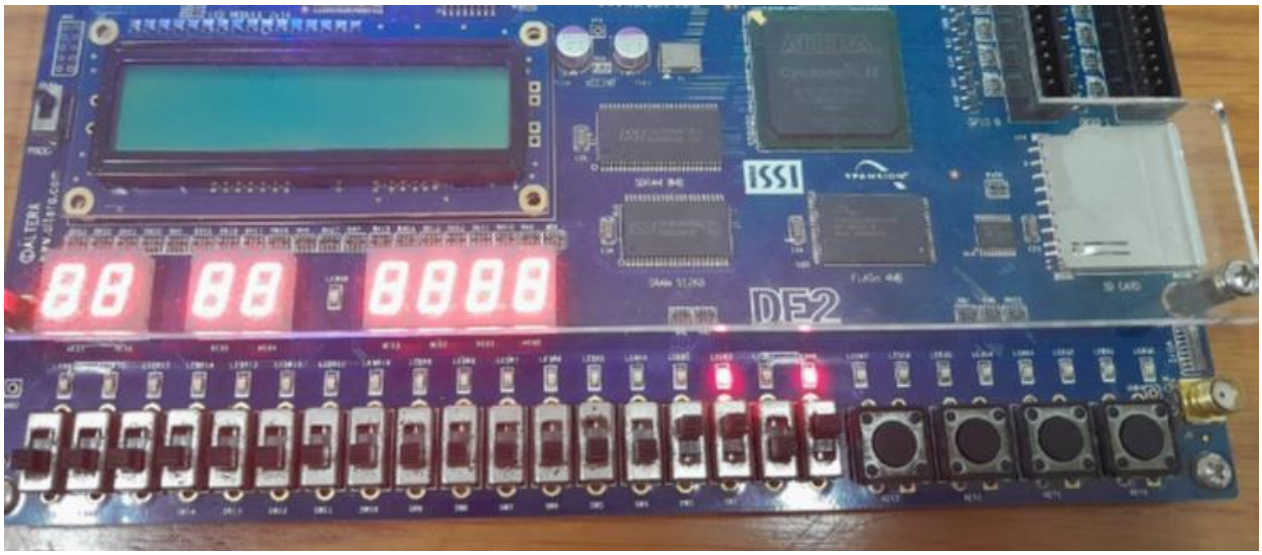


➢ Kết quả RTL viewer.



➢ Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

Gán chân theo mẫu sau:
Chân A[3:0] được nối với SW0-SW3
Chân S[3:0] được nối với LEDR[3:0]



THÍ NGHIỆM 2

Mục tiêu: Nắm được các thức mô tả khối ALU (bộ tính toán) có chức năng đơn giản sử dụng systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên thực hiện mô tả mạch cho mạch thực hiện bộ ALU tính toán 2 số 4 bit (hai số ngõ vào lần lượt là A và B, ngõ ra là S, cờ nhớ Ci và Co) thông qua ngõ vào điều khiển **2 bit** Sel.

- Nếu Sel=00: S=A+B

- Nếu Sel=01: S=A-B

- Nếu Sel=10: S=A AND B

- Nếu Sel=11: S=A OR B

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối và/hoặc diễn giải để giáo viên hiểu được ý tưởng của mình)

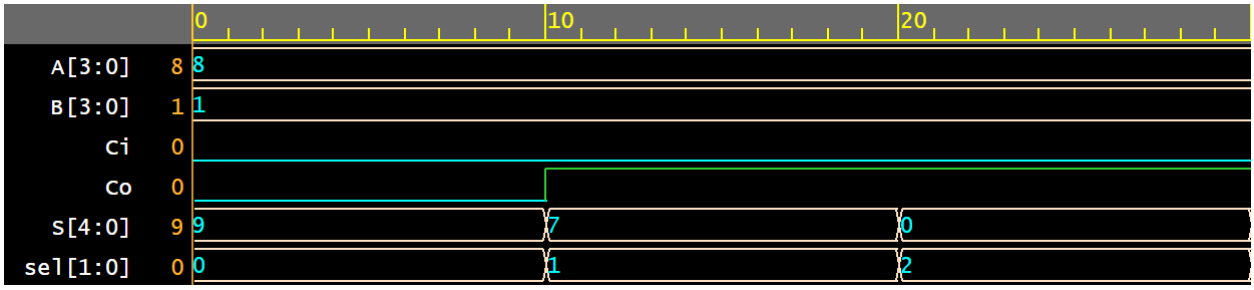
  Tạo biến logic ngõ vào lựa chọn Sel gồm 2 bit, căn cứ vào các trường hợp giá trị của biến Sel theo yêu cầu để thực hiện các phép tính sử dụng câu lệnh If-else.

- Chương trình mô tả hoạt động của thiết kế.

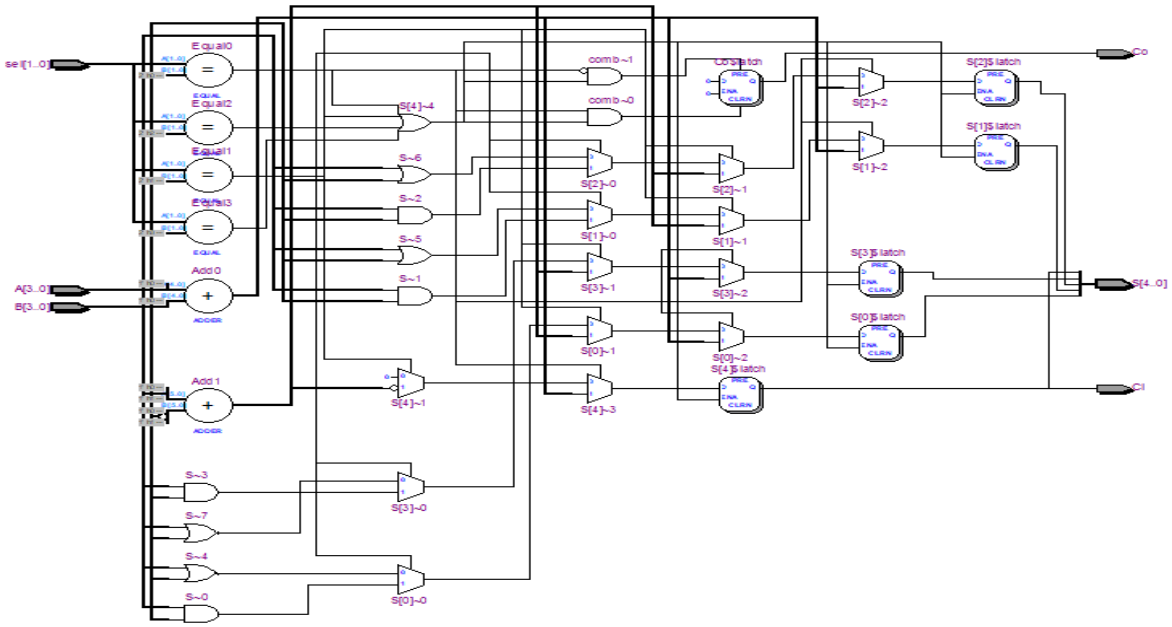
```
// Code your design here
module LAB4_TN2(
  input logic [3:0] A,
  input logic [3:0] B,
  input logic [1:0] sel,
  output logic [:0] S,
  output logic Ci,Co
);
  always@(sel) begin
    if (sel == 2'b00) begin
      S = A + B;
      Co = 0;
           end
    else if (sel == 2'b01) begin
      S = A - B;
      Co = 1;
           end
    else if (sel == 2'b10) begin
      S = A & B;
      Co = 0;
           end
    else if(sel == 2'b11) begin
```

```
    S = A | B;
    Co = 0;
        end
end
  assign Ci = S[4];
endmodule
```

- Kết quả mô phỏng dạng sóng.



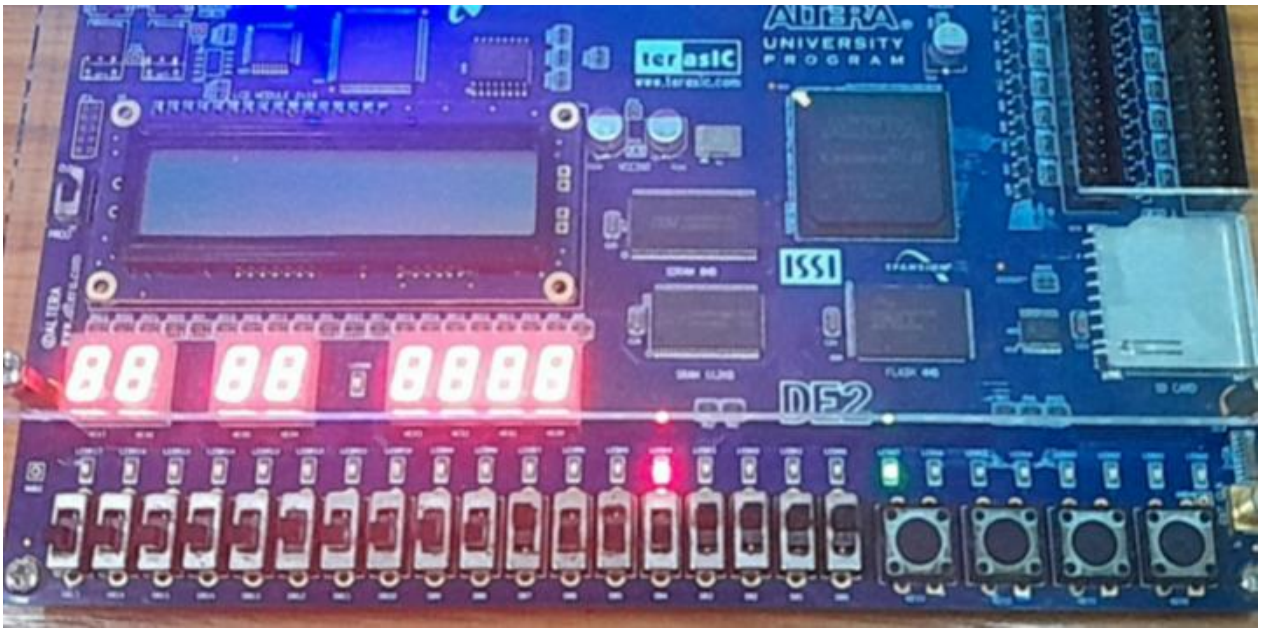
- Kết quả RTL viewer.



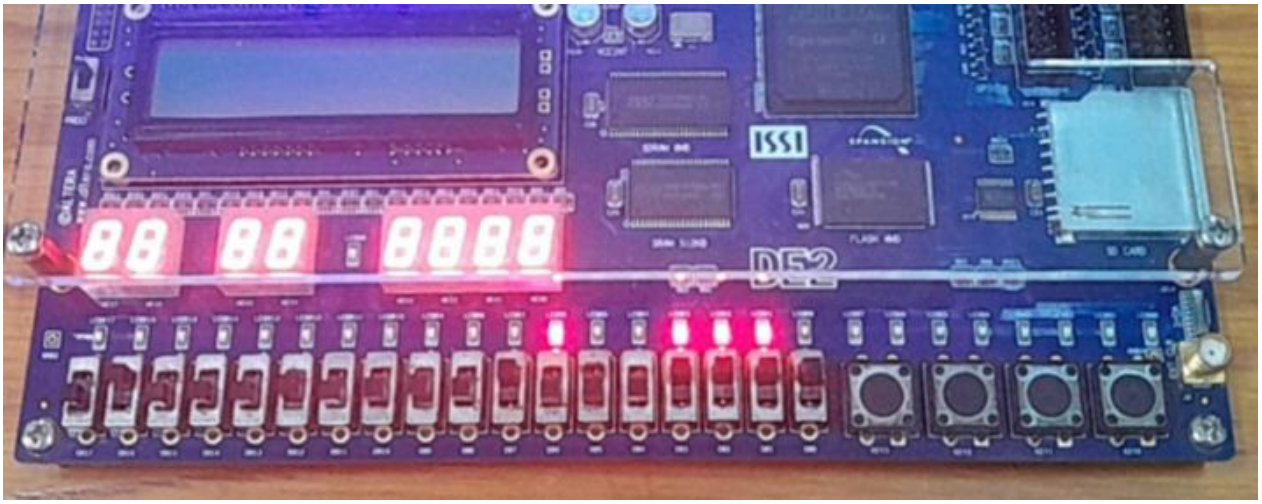
- Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

Gán chân theo mẫu sau:
Chân A[3:0] được nối với SW0-SW3
Chân B[3:0] được nối với SW4-SW7
Chân Ci được nối với SW8
Chân S[3:0] được nối với LEDR[3:0]
Chân Co được nối với LEDR[4]

Sel = 00

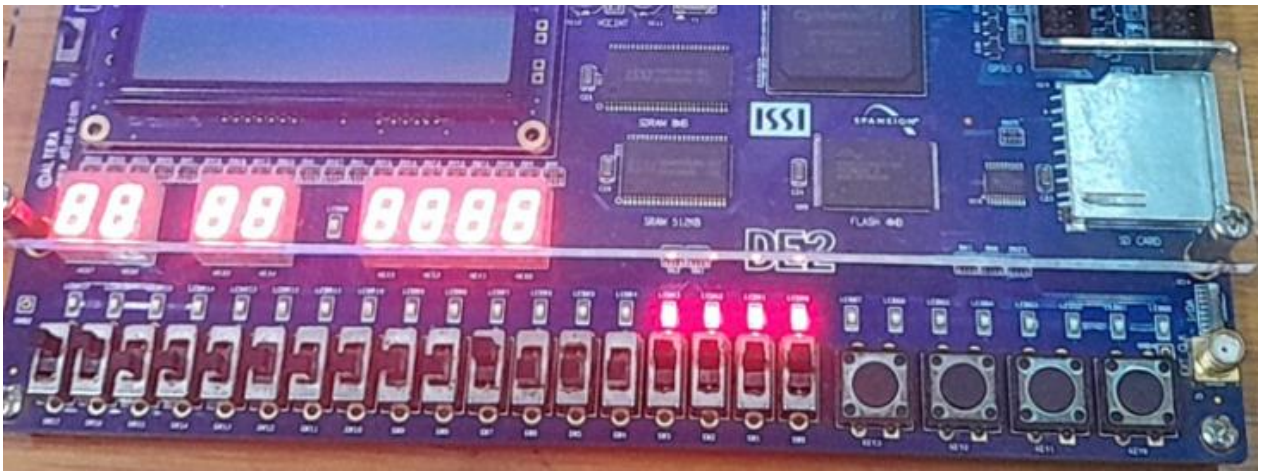


Sel = 01



Sel = 10

Sel = 11



THÍ NGHIỆM 3

Mục tiêu: Nắm được các thức mô tả mạch đếm đầy đủ sử dụng ngôn ngữ systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên thực hiện thiết kế mô tả mạch đếm xuống 3 bit đầy đủ sử dụng ngôn ngữ system verilog. Giá trị đếm thay đổi sau mỗi 1s. Ngõ ra được kết nối với LED 7 đoạn loại anode chung. Ngoài ra, bộ đếm còn có chân RST (tích cực cao) dùng để reset trạng thái bộ đếm về 0.

Gợi ý:

- Tín hiệu clock được tạo từ bộ chia tần số từ 50MHz sang 1s.

- Sinh viên sử dụng chương trình chuyển từ mã BCD sang LED 7 đoạn trong PRELAB, kết nối ngõ ra của bộ đếm với ngõ vào của bộ chuyển đổi.

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối và/hoặc diễn giải để giáo viên hiểu được ý tưởng của mình)

- Thực hiện chia tần số từ 50Mhz sang 1Hz, ngõ ra lưu tại tín hiệu clock trong chương trình.
- Tạo biến bcd gồm 3 bit để lưu các giá trị nhị phân từ 7-0 và biến seg 7 bit biểu diễn các đèn của led 7 đoạn. Với mỗi giá trị của bcd thì tương ứng có được các giá trị logic của 7 đoạn để led hiển thị giá trị thập phân của bcd.

- Chương trình mô tả hoạt động của thiết kế.

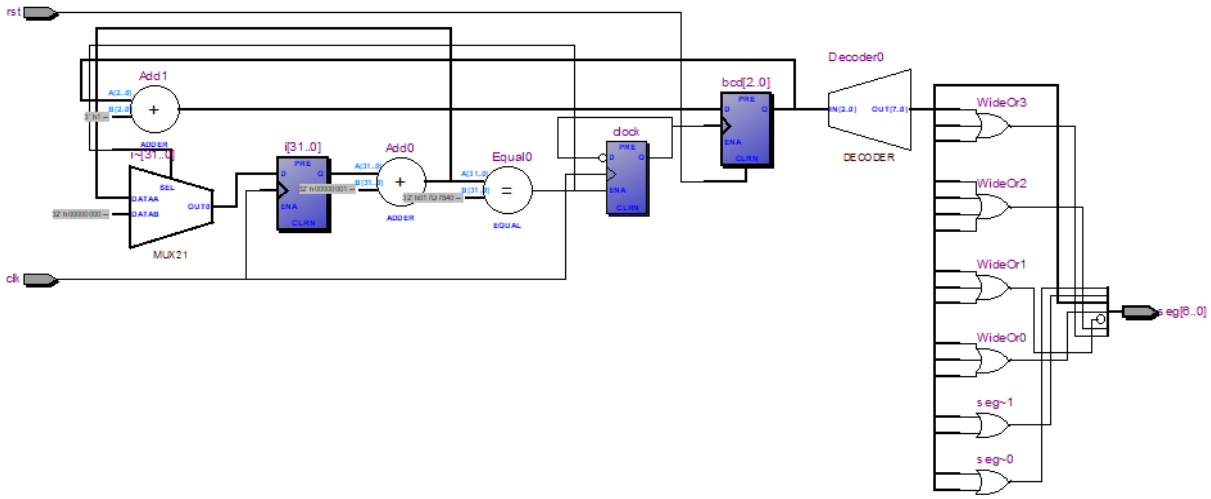
```
module LAB4_TN3(
 input clk,
 input rst,
 output reg [6:0] seg);
 integer i = 0;
 logic out;
 logic[2:0] bcd;
 logic clock;
 //integer seg_;
 always_ff@(posedge clk)
        begin
   i = i + 1;
   if (i == 25000000) begin
       clock = ~ clock;
```

```
        i = 0;
   end
        end

always@(posedge clock or posedge rst)
begin
if (rst == 1)
                begin
                        bcd <= 1'b111 ;
    end
        else
                begin
                        bcd <= bcd - 1'b1;
                end
        end

 always @(bcd)
  begin
     case (bcd)
        3'b000: seg = 7'b0000001;
        3'b001: seg = 7'b1001111;
        3'b010: seg = 7'b0010010;
        3'b011: seg = 7'b0000110;
        3'b100: seg = 7'b1001100;
        3'b101: seg = 7'b0100100;
        3'b110: seg = 7'b0100000;
```

```
            3'b111: seg = 7'b0001111;
            default : seg = 7'b1111111;
        endcase
                    end
endmodule
```

- Kết quả mô phỏng dạng sóng.

- Kết quả RTL viewer.



- Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

Gán chân theo mẫu sau:
LED 7 đoạn là HEX0.
Chân RST là SW0.

THÍ NGHIỆM 4

Mục tiêu: Nắm được các thức mô tả mạch đếm đầy đủ sử dụng ngôn ngữ systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên thực hiện thiết kế mô tả mạch **đếm lên 4 bit từ 5 đến 14** sử dụng ngôn ngữ system verilog. Giá trị đếm thay đổi sau mỗi 1s. Ngõ ra 4 bit được kết nối với 2 LED 7 đoạn loại anode chung. Ngoài ra, bộ đếm còn có chân RST (tích cực cao) dùng để reset trạng thái bộ đếm về 0.

Gợi ý:

- Tín hiệu clock được tạo từ bộ chia tần số từ 50MHz sang 1s.

- Sinh viên sử dụng chương trình chuyển từ mã BCD sang LED 7 đoạn trong PRELAB, kết nối ngõ ra của bộ đếm với ngõ vào của bộ chuyển đổi.

- Sinh viên cần viết thêm bộ chuyển đổi từ số 4 bit sang số BCD.

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối và/hoặc diễn giải để giáo viên hiểu được ý tưởng của mình)

- Tạo bộ chuyển tần số sang 1Hz lưu vào biến Clock.
- Tạo biến bcd nhị phân gồm 4 bit để biểu diễn các giá trị thập phân của bộ đếm.
- Tạo 2 biến o1, o2, mỗi biến gồm 7 bit để biểu diễn các led của hai led 7 đoạn biểu diễn các số từ 5-14.
- Thiết đặt các điều kiện biên cho bộ đếm và chức năng của biến RST.
- Với mỗi giá trị của biến bcd thì mỗi led 7 đoạn sẽ có tương ứng các tổ hợp logic của các led con sao cho số đếm được hiển thị lên 2 led 7 đoạn.

- Chương trình mô tả hoạt động của thiết kế.

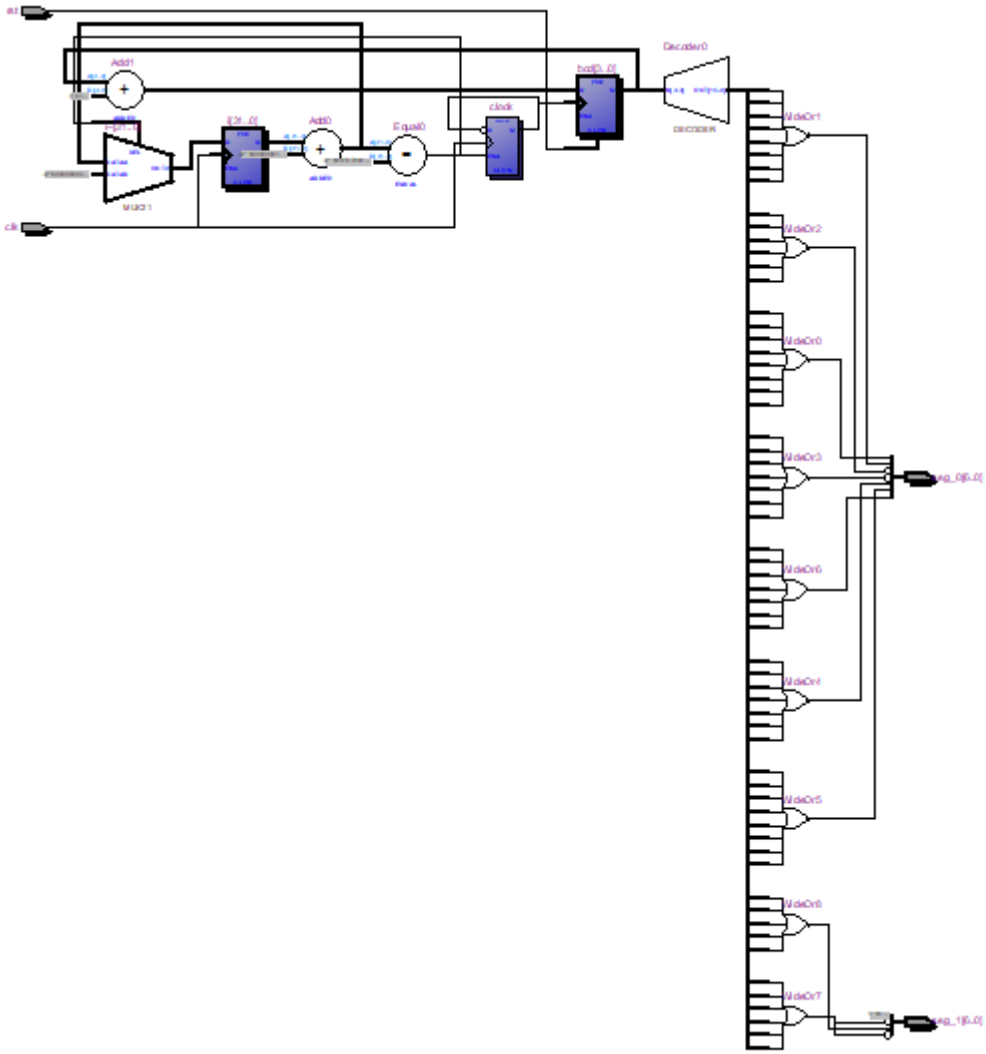
```
// Code your design here
module LAB4_TN4 (
        input logic re,
        input logic clk,
        output logic [6:0] o1,o2
);
logic out;
integer k = 0;
always_ff@(posedge clk) begin
        k = k+1;
        if ( k == 25000000) begin
                out = ~out;
                k = 0;
        end
end
logic [3:0] i;
always @(posedge out or posedge re) begin
if (re == '1)
    i <= 4'b0000;
else
if (i == 4'b1110 || i == 4'b0000)
    i <= 4'b0101;
else
```

```
    i <= i + 4'b0001;
end
seg sus (
        .i(i),
        .o1(o1),
        .o2(o2)
);
endmodule

module seg (
        input  logic [3:0] i,
        output logic [6:0] o1,o2
);
always_comb begin
case(i)
4'b0101: begin
 o1 = 7'b1000000;
 o2 = 7'b0010010;
  end
4'b0110: begin
 o1 = 7'b1000000;
 o2 = 7'b0000010;
  end
4'b0111: begin
 o1 = 7'b1000000;
 o2 = 7'b1111000;
```

```
  end
4'b1000: begin
 o1 = 7'b1000000;
 o2 = 7'b0000000;
  end
4'b1001: begin
 o1 = 7'b1000000;
 o2 = 7'b0010000;
  end
4'b1010: begin
 o1 = 7'b1111001;
 o2 = 7'b1000000;
  end
4'b1011: begin
 o1 = 7'b1111001;
 o2 = 7'b1111011;
  end
4'b1100: begin
 o1 = 7'b1111001;
 o2 = 7'b0100100;
  end
4'b1101: begin
 o1 = 7'b1111001;
 o2 = 7'b0110000;
  end
4'b1110: begin
```

```
 o1 = 7'b1111001;
 o2 = 7'b1111000;
  end
4'b0000: begin
 o1 = 7'b1000000;
 o2 = 7'b1000000;
  end
  default: begin o1 = 7'b1111111; o2 = 7'b1111111; end
endcase
end
endmodule
```

➢ Kết quả RTL viewer.



➢ Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

Gán chân theo mẫu sau:
Hai LED 7 đoạn là HEX1 (trọng số cao) và HEX0 (trọng số thấp).
Chân RST là SW0.

THÍ NGHIỆM 5

Mục tiêu: Nắm được các thức mô tả máy trạng thái sử dụng ngôn ngữ systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên tiến hành mô tả hệ tuần tự sau bằng VHDL sau đó nạp chương trình xuống kit DE2 để kiểm tra hoạt động:

Hệ tuần tự có 1 ngõ vào (X) và 1 ngõ ra (Z). Ngõ ra Z = 1 nếu tổng số bit 1 nhận được chia hết cho 3 (quy ước 0, 3, 6, 9, … là các số chia hết cho 3) và tổng số bit 0 nhận được là 1 số chẵn (lớn hơn 0).

Ghi chú: Sinh viên có thể lựa chọn thiết kế theo máy trạng thái kiểu Mealy hoặc Moore.

Gán chân theo mẫu sau:
Ngõ vào X được nối với SW0.
Tín hiệu CLK được nối với xung clock 1Hz (Trong bài PRELAB).
Ngõ ra Z được nối với LED0.

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối (máy trạng thái) và/hoặc diễn giải để giáo viên hiểu được ý tưởng của mình)

Ta đặt các trạng thái như sau:

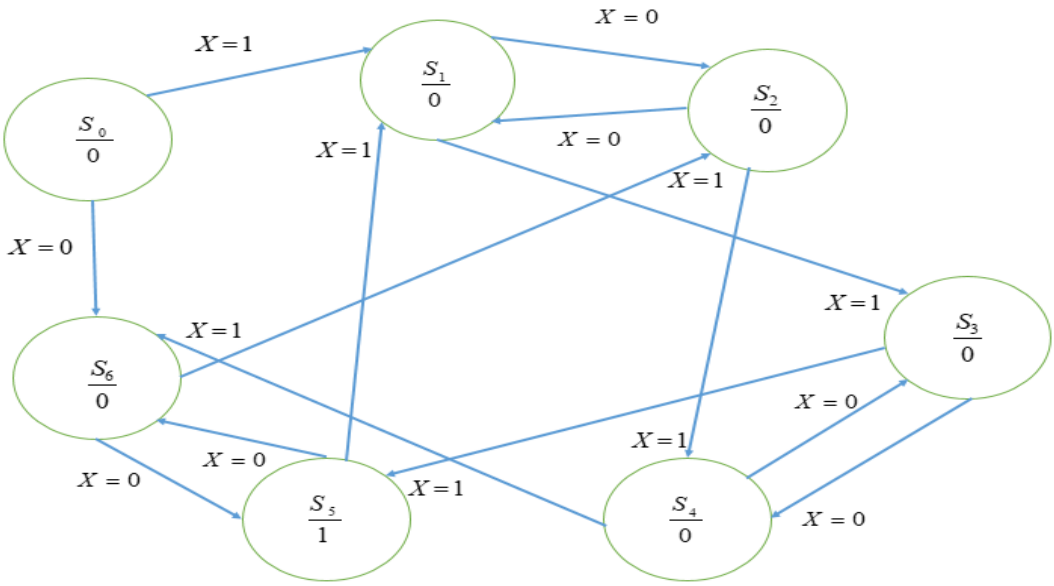
- S_0 : Trạng thái ban đầu, chưa nhận được bit nào.
- S_1 : Số bit 1 nhận được có dạng $3k+1$, số bit 0 là số chẵn.
- S_2 : Số bit 1 nhận được có dạng $3k+1$, số bit 0 là số lẻ.
- S_3 : Số bit 1 nhận được có dạng $3k+2$, số bit 0 là số chẵn.
- S_4 : Số bit 1 nhận được có dạng $3k+2$, số bit 0 là số lẻ.
- S_5 : Số bit 1 nhận được có dạng $3k$, số bit 0 là chẵn
- S_6 : Số bit 1 nhận được có dạng $3k$, số bit 0 là số lẻ.

Mã hóa dưới dạng chuỗi bit:

X: 0 1 1 1 0 0 0 1 0 0 1 1

Z: 0 0 0 0 1 0 1 0 0 0 0 1

Giản đồ trạng thái như hình dưới;



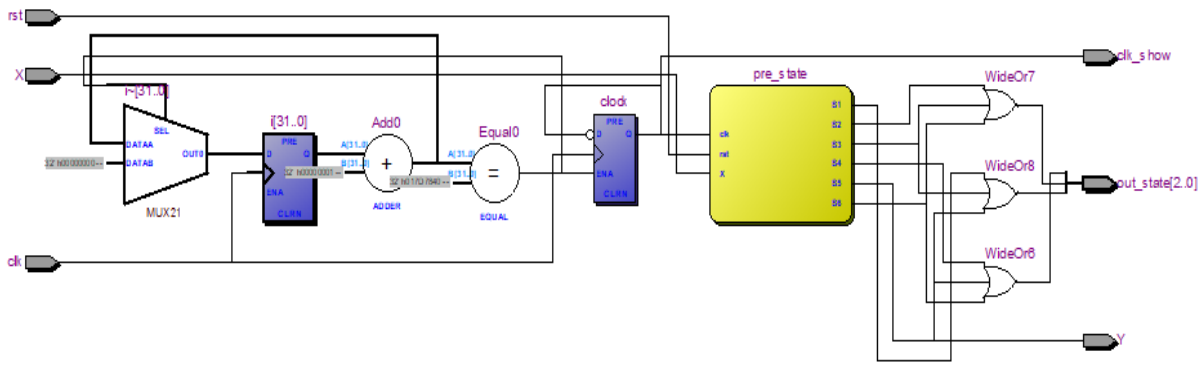
- Chương trình mô tả hoạt động của thiết kế.

```
// Code your design here

module LAB4_TN5(input X,input clk, input rst, output Y ,output clk_show, output [2:0]out_state);

   parameter S0=3'b000, S1=3'b001, S2=3'b010, S3=3'b011, S4 =3'b100, S5 = 3'b101, S6= 3'b110;

   reg [2:0]  pre_state, next_state;

         logic clock;

         integer i = 0;

         assign clk_show = clock;

//Các thanh ghi chưa trạng thái

         //Khối chuyển trạng thái

         always_ff@(posedge clk)

         begin

    i = i + 1;
```

```
    if (i == 25000000) begin
        clock = ~ clock;
        i = 0;
    end
        end
always@(posedge clock) begin
            if (rst)  begin
                pre_state <= S0;
                end
            else
                pre_state <= next_state;
        end
        //Khối chuyển trạng thái
        always@(pre_state or X) begin
            case(pre_state)
                S0: if (X) next_state <= S1;
                  else next_state <= S6;
                S1: if (!X) next_state <= S2;
                   else next_state <= S3;
        S2:  if (!X) next_state <= S1;
                  else next_state <= S4;
        S3: if (!X) next_state <= S4;
                   else next_state <= S5;
        S4: if (!X) next_state <= S3;
                   else next_state <= S6;
        S5: if (!X) next_state <= S6;
```

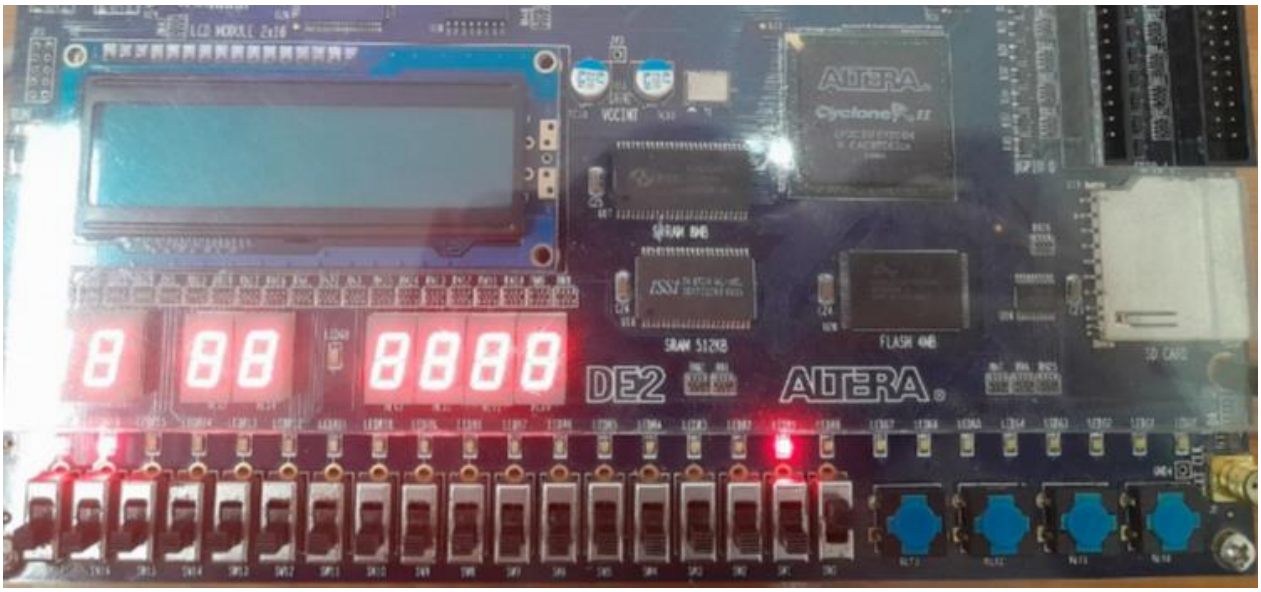
```
                    else next_state <= S1;
        S6: if (!X) next_state <= S5;
                    else next_state <= S2;
            endcase;
        end
        //Khối tạo ngõ ra
        always@(*) begin
            case (pre_state)
                S0: Y <= 1'b0;
                S1: Y <= 1'b0;
                S2: Y <= 1'b0;
                S3: Y <= 1'b0;
                S4: Y <= 1'b0;
                S5: Y <= 1'b1;
                S6: Y <= 1'b0;
            endcase;
        end
        assign out_state = pre_state;
endmodule
```

➢ Kết quả RTL viewer.



➢ Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

Lưu ý: Sinh viên nên nối clock 1Hz ra 1 LED để dễ quan sát tín hiệu clock.




SRAM 512KB
FLASH 4MB
DE2
ALTERA

THÍ NGHIỆM 6

Mục tiêu: Nắm được các thức mô tả máy trạng thái sử dụng ngôn ngữ systemverilog và thực hiện kiểm tra hoạt động trên kit FPGA.

Yêu cầu: Sinh viên tiến hành mô tả hệ tuần tự sau bằng VHDL sau đó nạp chương trình xuống kit DE2 để kiểm tra hoạt động:

Một hệ thống cung cấp thức ăn và nước uống tự động cho thú cưng gồm 2 ngõ vào là 2 nút nhấn
RED, BLUE; và 2 ngõ ra là tín hiệu FOOD, WATER để kích hoạt máy cung cấp thức ăn và nước uống

- Nút RED (tín hiệu R; khi nhấn nút R=1, ngược lại R=0): khi con vật đói muốn ăn thì cần nhấn nút RED 3 lần liến tiếp. Khi đó tín hiệu F (FOOD) = 1 để kích hoạt máy cung cấp thức ăn.
- Nút BLUE (tín hiệu B; khí nhấn nút B=1, ngược lại B=0): khi con vật khát muốn uống thì cần nhấn nút BLUE 2 lần liên tiếp. Khi đó tín hiệu W (WATER) = 1 để kích hoạt máy cung cấp nước uống.

Chú ý:

- Khi tín hiệu kích hoạt F hay W bằng 1, nếu nhấn 1 nút bất kỳ hệ thống sẽ trở lại trạng thái reset.
- Ở mỗi thời điểm, chỉ có 1 nút nhấn.

- Các nút nhấn cần phải được tác động liên tiếp, nếu có 1 nút sai trình tự, máy trạng thái sẽ quay trở về trạng thái ban đầu. (ví dụ: khi các nút nhấn được tác động theo trình tự (RED, RED, BLUE), máy trạng thái quay về trạng thái đầu reset).

Ghi chú: Sinh viên có thể lựa chọn thiết kế theo máy trạng thái kiểu Mealy hoặc Moore.

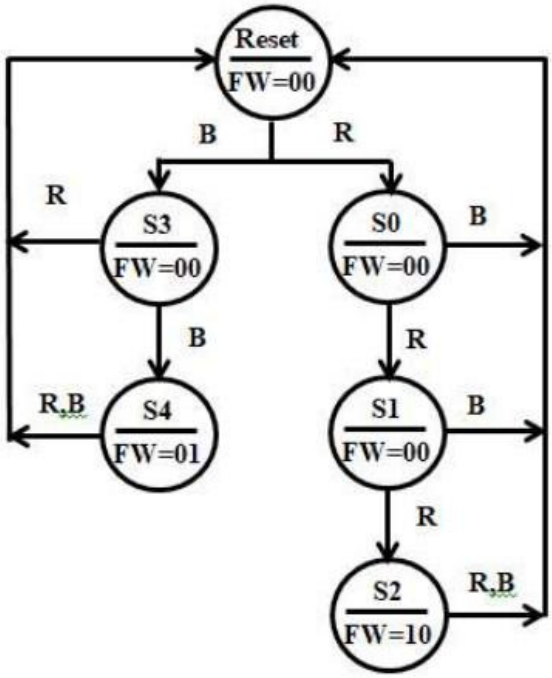
Gán chân theo mẫu sau:
Ngõ vào RED và BLUE lần lượt được nối với SW0 và SW1.
Tín hiệu CLK được nối với xung clok 1Hz (Trong bài PRELAB).
Ngõ ra FOOD và WATER lần lượt được nối với LED0 và LED1.

Kiểm tra:

- Sinh viên trình bày ý tưởng của thiết kế. (Sinh viên có thể vẽ sơ đồ khối (máy trạng thái) và/hoặc diễn dãi để giáo viên hiểu được ý tưởng của mình)

TTHT	TTKT		NGÕ RA	
	R	B	F	W
Reset	S0	S3	0	0
S0	S1	Reset	0	0
S1	S2	Reset	0	0
S2	Reset	Reset	1	0
S3	Reset	S4	0	0
S4	Reset	Reset	0	1

Định nghĩa trạng thái
S0: nhấn nút Red 1 lần
S1: nhấn nút Red 2 lần
S2: nhấn nút Red 3 lần
S3: nhấn nút Blue 1 lần
S4: nhấn nút Blue 2 lần



- Chương trình mô tả hoạt động của thiết kế.

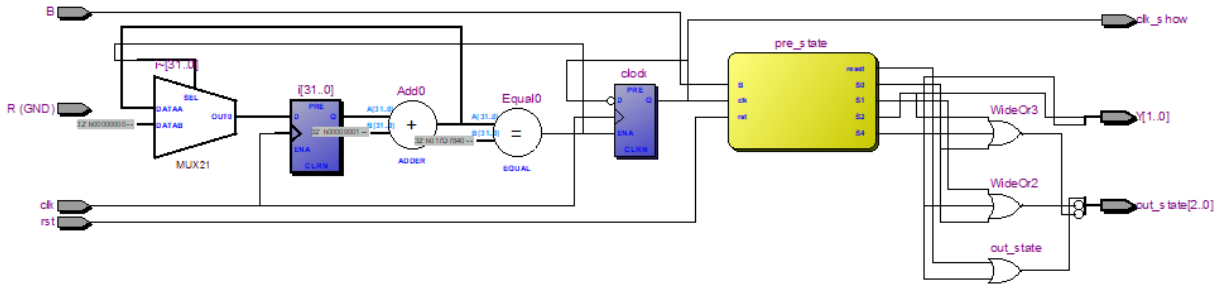
```
module LAB4_TN6(input clk, input rst, input R, input B,  output logic [1:0]Y ,output clk_show, output [2:0] out_state);
   parameter reset = 3'b111, S0=3'b000, S1=3'b001, S2=3'b010, S3=3'b011, S4 =3'b100;
   reg [2:0]  pre_state, next_state;
         logic clock;
         integer i = 0;
         assign clk_show = clock;
//Các thanh ghi chưa trạng thái
         //Khối chuyển trạng thái
         always_ff@(posedge clk)
         begin
```

```
    i = i + 1;
    if (i == 25000000) begin
        clock = ~ clock;
        i = 0;
    end
        end
always@(posedge clock) begin
            if (rst)  begin
                pre_state <= reset;
                end
            else
                pre_state <= next_state;
        end
        //Khối chuyển trạng thái
        logic [1:0]RB;
        assign RB = {R,B};
        always@(pre_state or RB) begin
            case(pre_state)
                reset: if (RB == 2'b10)
                                    next_state <= S0;
                                     else if (RB == 2'b01)
                                    next_state <= S3;
                                     else next_state <= reset;
                  S0: if (RB == 2'b10)
                                    next_state <= S1;
                                else if (RB == 2'b01)
```

```
                    next_state  <= reset;
                else
                    next_state <= S0;

        S1: if (RB == 2'b10)
                    next_state <= S2;
          else if (RB == 2'b01)
                    next_state <= reset;
                else
                    next_state <= S1;

S2:  if ((RB == 2'b10 || RB == 2'b01))
                    next_state <= reset;
                    else
                        next_state <= S2;
S3: if (RB == 2'b01)
                        next_state <= S4;
            else if (RB == 2'b10)
                        next_state <= reset;
                else
                        next_state <= S3;
S4: if (RB == 2'b10 || RB == 2'b01)
                    next_state <= reset;
                else
                    next_state <= S4;
      endcase;
```

```
        end
        //Khối tạo ngõ ra
        always@(*) begin
                case (pre_state)
                        reset: Y <= 2'b00;
                        S0: Y <= 2'b00;
                        S1: Y <= 2'b00;
                        S2: Y <= 2'b10;
                        S3: Y <= 2'b00;
      S4: Y <= 2'b01;
                endcase;
        end
        assign out_state = pre_state;
endmodule
```

- Kết quả RTL viewer.



- Sinh viên thực hiện gán chân theo yêu cầu và sau đó đổ lên kit FPGA DE2. Sau đó ghi nhận kết quả.

  Lưu ý: Sinh viên nên nối clock 1Hz ra 1 LED để dễ quan sát tín hiệu clock.