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

|  |  |
| --- | --- |
| **Họ và tên:** Lâm Thành Phát | **Lớp TN:** L21 |
| **MSSV:** 2111974 | **Ngày:** 12/11/2022 |

1. **PRELAB**

**BÀI CHUẨN BỊ 1**

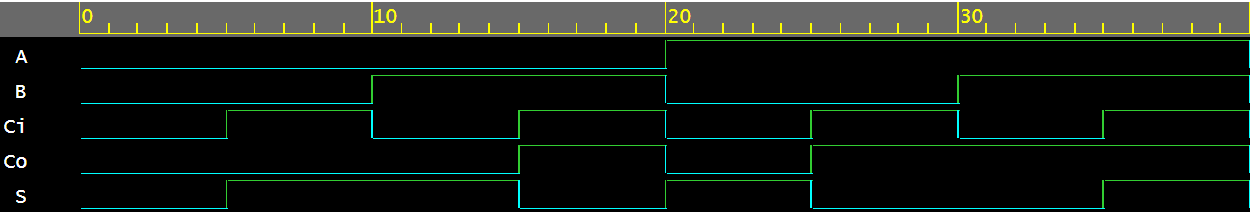
***Mục tiêu:***

Nắm được các thức mô tả mạch cộng toàn phần (FA) sử dụng ngôn ngữ SystemVerilog.

***Yêu cầu:*** Sinh viên đọc đoạn chương trình sau đây dùng để mô tả bộ FA.

|  |  |
| --- | --- |
| module FA\_ex1(  input A,  input B,  input Ci,  output S,  output Co);  assign S = A ^ B ^ Ci;  assign Co=(A&B)|(A&Ci)|(B&Ci);  endmodule | ***Hình 4.1:*** *Bộ cộng toàn phần Full Adder* |

* Sinh viên tiến hành biên dịch đoạn chương trình trên. Sau đó, sinh viên tiến mô phỏng dạng sóng ngõ ra trong tất cả các trường hợp của ngõ vào. Chụp hình dạng sóng ngõ ra.



**BÀI CHUẨN BỊ 2**

***Mục tiêu:*** Nắm được các thức mô tả mạch cộng trừ 2 số 4 bit sử dụng ngôn ngữ systemverilog.

***Yêu cầu:*** Sinh viên thực hiện mô tả mạch cộng trừ 2 số 4 bit (2 số ngõ vào lần lượt là A và B, ngõ ra là S, cờ nhớ Ci và Co) bằng thông qua ngõ vào điều khiển sel:

- sel=0: S = A + B

- sel=1: S = A - B

Sinh viên thực hiện mô tả bằng 2 cách:

- Cách 1: sử dụng các Full Adder ghép nối lại với nhau (sử dụng bài chuẩn bị 1)*.*

- Cách 2: mô tả theo phương pháp luồng dữ liệu (data flow model).

* Sinh viên thực hiện mô tả theo yêu cầu của bài toán sử dụng ngôn ngữ systemverilog.

*Cách 1: Gợi ý:* sử dụng dạng cấu trúc, ghép nối các bộ FA.

|  |
| --- |
| wire wire0,wire1,wire2  FA\_ex1 u0(.Ci(0),.A(A[0]),.B(B[0]),.S(S[0]),.Co(wire0));  FA\_ex1 u1(.Ci(wire1),.A([1]),.B([1]),.S(S[1]),.Co(wire1));  … |

*Cách 2: Gợi ý:* sử dụng câu điều kiện *if … else ...*

|  |  |
| --- | --- |
| always@(\*)  if (sel)  …  else  … | assign S=(sel)?A+B:A-B; |

* Chương trình System Verilog theo cách 1:

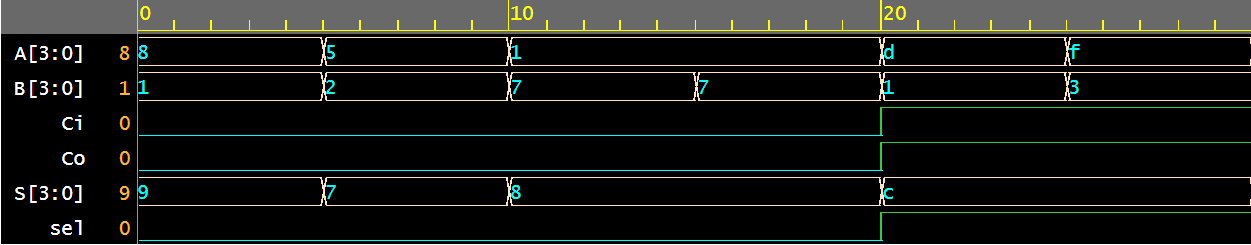
**File design.sv:**

|  |
| --- |
| // Code your design here  // Code your design here  module FA\_ex2(  input A,  input B,  input C,  input sel, //C0  output S,  output Ci);  logic B\_res;  assign B\_res = B ^ sel;  assign S = A ^ B\_res ^ C;  assign Ci=(A&B\_res)|(A&C)|(B\_res&C);  endmodule  module LAB4\_TN2(  input logic [3:0] A,  input logic [3:0] B,  input logic sel,  output logic [3:0] S,  output Ci,Co  );  assign Co = sel;  logic C1,C2,C3;  FA\_ex2 FA1(.A(A[0]),.B(B[0]),.C(sel),.sel(sel),.S(S[0]),.Ci(C1));  FA\_ex2 FA2(.A(A[1]),.B(B[1]),.C(C1),.sel(sel),.S(S[1]),.Ci(C2));  FA\_ex2 FA3(.A(A[2]),.B(B[2]),.C(C2),.sel(sel),.S(S[2]),.Ci(C3));  FA\_ex2 FA4(.A(A[3]),.B(B[3]),.C(C3),.sel(sel),.S(S[3]),.Ci(Ci));  endmodule |

**File testbench.sv:**

|  |
| --- |
| // Code your testbench here  // or browse Examples  `timescale 1ns/1ns  module testbench;  reg [3:0] A;  reg [3:0] B;  reg sel;  wire [3:0] S;  wire Ci,Co;    LAB4\_TN2 dut(.A(A),.B(B),.sel(sel),.S(S),.Ci(Ci),.Co(Co));  initial begin  $dumpfile("dump.vcd");  $dumpvars(1);  A = 4'b1000;  B = 4'b0001;  sel = 1'b0;  #5  A = 4'b0101;  B = 4'b0010;  sel = 1'b0;  #5  A = 4'b0001;  B = 4'b0111;  sel = 1'b0;  #5  A = 4'b0001;  B = 4'b0101;  sel = 1'b0;  A = 4'b0001;  B = 4'b0111;  sel = 1'b0;  #5  A = 4'b1101;  B = 4'b0001;  sel = 1'b1;  #5  A = 4'b1111;  B = 4'b0011;  sel = 1'b1;  #5  A = 4'b1001;  B = 4'b0011;  sel = 1'b1;  #5  $finish;  end  endmodule |

* Sinh viên tiến mô phỏng dạng sóng ngõ ra trong một vài trường hợp của ngõ vào. Chụp hình dạng sóng ngõ ra. (Cách 1).



**Cách 2:**  Sử dụng sử dụng câu điều kiện *if … else ...*

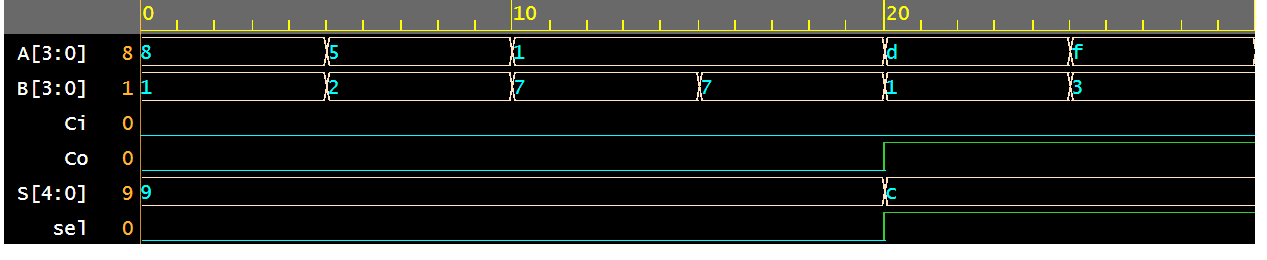
**File design.sv:**

|  |
| --- |
| module LAB4\_TN2(  input logic [3:0] A,  input logic [3:0] B,  input logic sel,  output logic [4:0] S,  output Ci,Co  );  assign Co = sel;  always@(sel) begin  if (sel == 1'b0) begin  S = A + B;  end  else begin  if (sel == 1'b1) begin  S = A - B;  end  end  end  assign Ci = S[4];  endmodule |

**File testbench.sv:**

|  |
| --- |
| // Code your testbench here  // or browse Examples  // Code your testbench here  // or browse Examples  `timescale 1ns/1ns  module testbench;  reg [3:0] A;  reg [3:0] B;  reg sel;  reg [4:0] S;  wire Ci,Co;    LAB4\_TN2 dut(.A(A),.B(B),.sel(sel),.S(S),.Ci(Ci),.Co(Co));  initial begin  $dumpfile("dump.vcd");  $dumpvars(1);  A = 4'b1000;  B = 4'b0001;  sel = 1'b0;  #5  A = 4'b0101;  B = 4'b0010;  sel = 1'b0;  #5  A = 4'b0001;  B = 4'b0111;  sel = 1'b0;  #5  A = 4'b0001;  B = 4'b0101;  sel = 1'b0;  A = 4'b0001;  B = 4'b0111;  sel = 1'b0;  #5  A = 4'b1101;  B = 4'b0001;  sel = 1'b1;  #5  A = 4'b1111;  B = 4'b0011;  sel = 1'b1;  #5  A = 4'b1001;  B = 4'b0011;  sel = 1'b1;  #5  $finish;  end  endmodule |

* Sinh viên tiến mô phỏng dạng sóng ngõ ra trong một vài trường hợp của ngõ vào. Chụp hình dạng sóng ngõ ra. (Cách 2).



**BÀI CHUẨN BỊ 3**

***Mục tiêu:*** Nắm được các thức mô tả mạch giải mã số BCD sang mã LED 7 đoạn loại anode chung sử dụng ngôn ngữ systemverilog.

***Yêu cầu:*** Sinh viên thực hiện mô tả mạch giải mã số BCD sang mã LED 7 đoạn loại anode chung.

* Sinh viên hoàn thành bảng chân trị của mạch giải mã.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | I3 | I2 | I1 | I0 | g | f | e | d | c | b | a | | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | | 1 | 0 | 1 | 0 | - | - | - | - | - | - | - | | 1 | 0 | 1 | 1 | - | - | - | - | - | - | - | | 1 | 1 | 0 | 0 | - | - | - | - | - | - | - | | 1 | 1 | 0 | 1 | - | - | - | - | - | - | - | | 1 | 1 | 1 | 0 | - | - | - | - | - | - | - | | 1 | 1 | 1 | 1 | - | - | - | - | - | - | - |   ***Bảng 4.1:*** *Bảng chân trị mạch giải mã LED 7 đoạn* | Buy 7 Segment Common Anode Display Online in India | Robocraze  ***Hình 4.2*** *LED 7 đoạn loại anod chung*  *(source: internet)* |

* Sinh viên thực hiện mô tả theo yêu cầu của bài toán sử dụng ngôn ngữ systemverilog.

*Gợi ý:* sử dụng câu lệnh *case …. .*

|  |
| --- |
| input [3:0]I;  always@(\*) begin  case(I)  4’b0000: out <= 7’b1000000;  4’b0001: out <= …  …  endcase;  end |

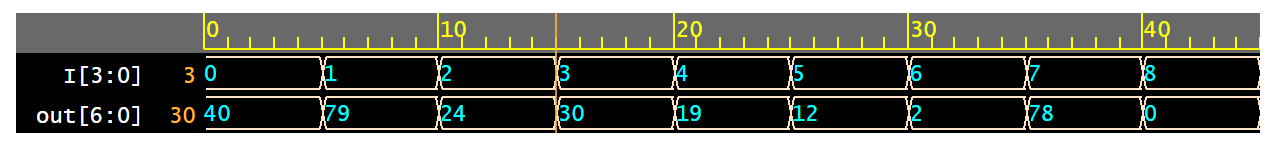
**File design.sv:**

|  |
| --- |
| // BCD to led 7 doan  module LAB4\_TN3(  input logic [3:0] I,  output logic [6:0] out  );  always@(I) begin  case (I)  4'b0000: out = 7'b1000000;  4'b0001: out = 7'b1111001;  4'b0010: out = 7'b0100100;  4'b0011: out = 7'b0110000;  4'b0100: out = 7'b0011001;  4'b0101: out = 7'b0010010;  4'b0110: out = 7'b0000010;  4'b0111: out = 7'b1111000;  4'b1000: out = 7'b0000000;  4'b1001: out = 7'b0010000;  endcase  end  endmodule |

**File testbench.sv:**

|  |
| --- |
| // Code your testbench here  // or browse Examples  // Code your testbench here  // or browse Examples  `timescale 1ns/1ns  module testbench;  reg[3:0] I;  wire logic [6:0] out;    LAB4\_TN3 dut(.I(I),.out(out));  initial begin  $dumpfile("dump.vcd");  $dumpvars(1);  I = 4'b0000;  #5  I = 4'b0001;  #5  I = 4'b0010;  #5  I = 4'b0011;  #5  I = 4'b0100;  #5  I = 4'b0101;  #5  I = 4'b0110;  #5  I = 4'b0111;  #5  I = 4'b1000;  #5  I = 4'b1001;  #5  $finish;  end  endmodule |

* Sinh viên tiến mô phỏng dạng sóng ngõ ra trong một vài trường hợp của ngõ vào. Chụp hình dạng sóng ngõ ra.



**BÀI CHUẨN BỊ 4**

***Mục tiêu:*** Nắm được các thức mô tả mạch chuyển đổi 1 xung có tần số 50MHz sang 1 xung có tần số 1Hz sử dụng ngôn ngữ systemverilog.

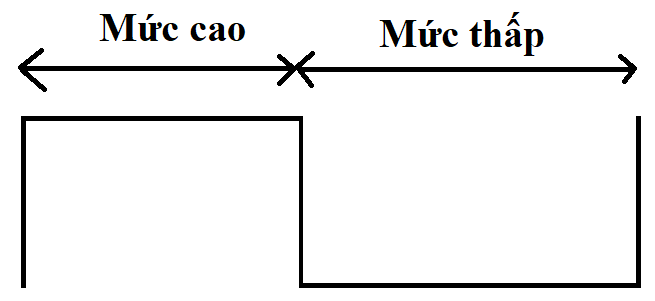
***Yêu cầu:*** Sinh viên thực hiện mô tả mạch chuyển đổi 1 xung có tần số 50MHz sang 1 xung có tần số 1Hz.

*Gợi ý:*

- Xung có tần số 50MHz sẽ thực hiện 50 000 000 dao động trong 1s. (1 dao động bao gồm 1 lần mức cao và 1 lần mức thấp).

**-** Xung có tần số 1Hz sẽ thực hiện 1 dao động trong 1s.

- Vận dụng nguyên lí của mạch đếm, mạch nhận ngõ vào là xung 50MHz, khi xung ngõ vào đếm 25 000 000 sẽ tiến hành đảo trạng thái của xung ngõ ra.

****

Đoạn chương trình minh họa:

|  |
| --- |
| integer i=0;  reg temp=0;  always\_ff@(posedge clk)  begin  i = i + 1;  if (i == 25 000 000) begin  out = ~ out;  i = 0;  end  end |

* Sinh viên sử dụng gợi ý trên và thực hiện mô tả theo yêu cầu của bài toán sử dụng ngôn ngữ systemverilog.

|  |
| --- |
| // Code your design here  // Code your design here  module LAB4\_TN4(  input logic clk,  output logic out  );  longint i;  assign i = 0;  reg temp=0;  always\_ff@(posedge clk) begin  i = i + 1;  if (i == 25 000 000) begin  out = ~ out;  i = 0;  end    endmodule |

|  |
| --- |
| // Code your testbench here  // or browse Examples  // Code your testbench here  // or browse Examples  `timescale 1ns/1ns  module testbench;  reg cl;  wire ou;    LAB4\_TN4 dut(.clk(cl),.out(ou));  initial begin  $dumpfile("dump.vcd");  $dumpvars(1);  #10  cl = 1'b1;  #10  cl = 1'b0;  #10  cl = 1'b1;  $finish;  end  endmodule |

**BÀI CHUẨN BỊ 5**

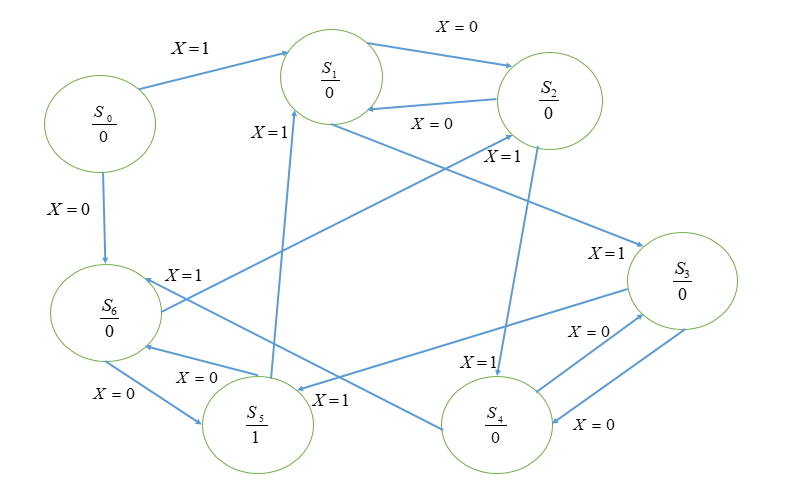
***Mục tiêu:*** Nắm được các thức mô tả máy trạng thái sử dụng ngôn ngữ systemverilog.

***Yêu cầu:***

* Sinh viên vẽ máy trạng thái cho yêu cầu ở thí nghiệm 5 của bài Lab 4. Sinh viên giải thích ý nghĩa của các trạng thái và tiến hành mã hóa trạng thái thành chuỗi bit.

**Thí nghiệm 5, Lab 4:** 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).

Ta đặt các trạng thái như sau:

* : Trạng thái ban đầu, chưa nhận được bit nào.
* : Số bit 1 nhận được có dạng , số bit 0 là số chẵn.
* : Số bit 1 nhận được có dạng , số bit 0 là số lẻ.
* : Số bit 1 nhận được có dạng , số bit 0 là số chẵn.
* : Số bit 1 nhận được có dạng , số bit 0 là số lẻ.
* : Số bit 1 nhận được có dạng , số bit 0 là chẵn
* : Số bit 1 nhận được có dạng , số bit 0 là số lẻ.

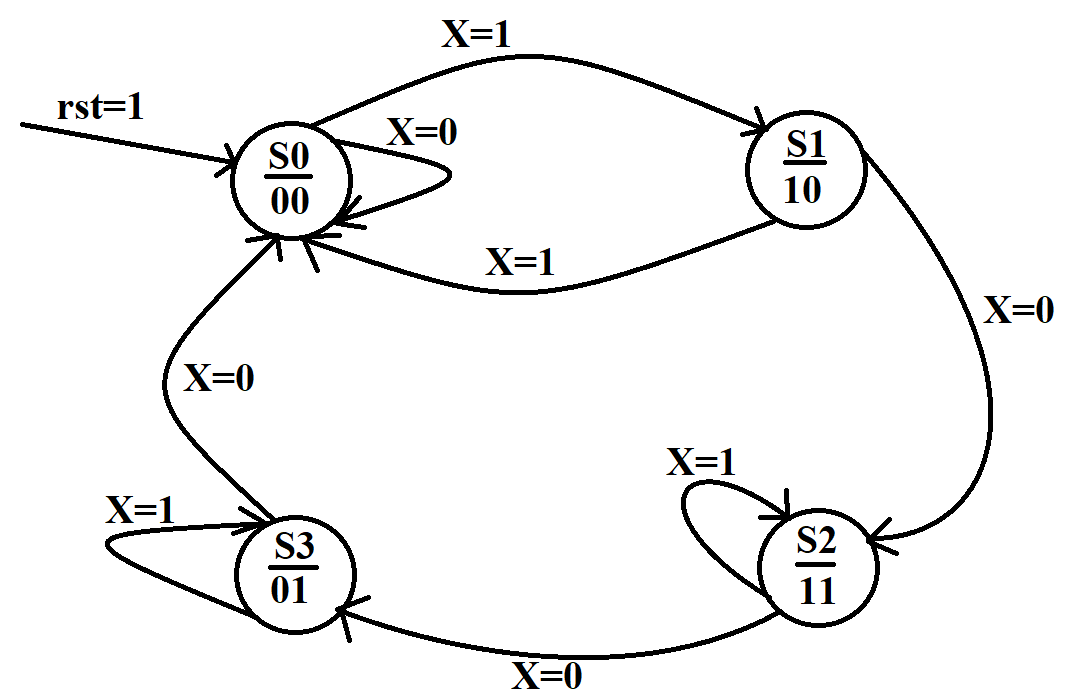
Mã hóa dưới dạng chuỗi bit:

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

* Sinh viên thực hiện mô tả máy trạng thái ở phần trên sử dụng ngôn ngữ systemverilog. (Tham khảo đoạn chương trình ở phần dưới về cách viết máy trạng thái).

Giả sử có máy trạng thái: Ngõ vào X, ngõ ra Y (2 bit)



Các trạng thái lần lượt là: S0 (00), S1 (01), S2 (10), S3(11)

Đoạn chương trình cho máy trạng thái trên:

|  |
| --- |
| input X; //Khai báo ngõ vào 1 bit  input clk; //Khai báo clock  output reg [1:0]Y; //Khai báo ngõ ra 2 bit  parameter S0=2'b00, S1=2'b01, S2=2'b10, S3=2'b11; //Định nghĩa các trạng thái  reg [1:0]pre\_state, next\_state; //Các thanh ghi chưa trạng thái  **//Khối chuyển trạng thái**  always@(posedge clk) 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 <= S0;  S1: if (!X) next\_state <= S2;  else next\_state <= S0;  S2: if (!X) next\_state <= S3;  else next\_state <= S2;  S3: if (!X) next\_state <= S0;  else next\_state <= S3;  endcase;  end  **//Khối tạo ngõ ra**  always@(\*) begin  case (pre\_state)  S0: Y <= 2'b00;  S1: Y <= 2'b10;  S2: Y <= 2'b11;  S3: Y <= 2'b01;  endcase;  end |

Thực hiện mô tả máy trạng thái ở **TN 5** sử dụng ngôn ngữ systemverilog:

|  |
| --- |
| // Code your design here  input X; //Khai báo ngõ vào 1 bit  input clk; //Khai báo clock  output reg Y; //Khai báo ngõ ra 1 bit  parameter S0=3'b000, S1=3'b001, S2=3'b010, S3=3'b011, S4=3'b100, S5=3'b101, S6=3'b110; //Định nghĩa các trạng thái  reg [1:0]pre\_state, next\_state; //Các thanh ghi chứa  //Khối chuyển trạng thái  always@(posedge clk) 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 <= S3;  else next\_state <= S2;  S2: if (X) next\_state <= S4;  else next\_state <= S1;  S3: if (X) next\_state <= S5;  else next\_state <= S4;  S4: if (X) next\_state <= S6;  else next\_state <= S3;  S5: if (X) next\_state <= S1;  else next\_state <= S6;  S6: if (X) next\_state <= S2;  else next\_state <= S5;  endcase;  end  //Khối tạo ngõ ra  always@(\*) begin  case (pre\_state)  S0: Y <= 3'b000;  S1: Y <= 3'b001;  S2: Y <= 3'b010;  S3: Y <= 3'b011;  S4: Y <= 3'b100;  S5: Y <= 3'b101;  S6: Y <= 3'b110;  endcase;  end |