**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**



**BÁO CÁO LAB02 CE213**

**CONTINUOUS ASSIGNMENT**

|  |  |
| --- | --- |
| **HỌ VÀ TÊN:** | **VÕ ĐÔNG THÀNH** |
| **MSSV:** | **21520457** |
| **LỚP:** | **CE213.O11.1** |

**GIẢNG VIÊN HƯỚNG DẪN:**

**HỒ NGỌC DIỄM**

**I. Mục tiêu**

Trong bài thực hành này, sinh viên sẽ thực hành lệnh assign (continuous assignment) và cách gọi module theo thứ bậc để thiết kế các mạch tổ hợp.

**II. Chuẩn bị thực hành**

- Sinh viên phải chuẩn bị code Verilog cho tất cả các câu trong phần nội dung thực hành và nộp cho GVHD vào đầu buổi học.

- Sinh viên nào không có bài chuẩn bị được xem là vắng buổi học hôm đó.

- Bài chuẩn bị được tính vào điểm bài báo cáo của Lab.

**III. Nội dung thực hành**

**Câu 2.**

**a) Thực hiện bộ Mux\_5to1\_1-bit từ các Mux\_2to1\_bitA diagram of a circuit

Description automatically generated**

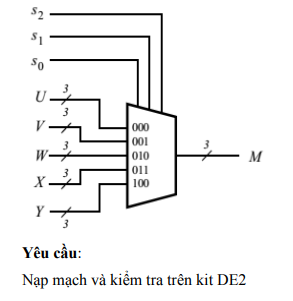
|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module mux2to1\_1bit(  input wire x,  input wire y,  input wire select,  output m  );  assign m = select ? y : x;  endmodule |  |
| module mux5to1\_1bit(  input u,v,w,x,y,  input s0,s1,s2,  output m  );  wire uv,wx,uvmx;  mux2to1\_1bit u1 (u,v,s0,uv);  mux2to1\_1bit u2 (w,x,s0,wx);  mux2to1\_1bit u3 (uv,wx,s1,uvmx);  mux2to1\_1bit u4 (uvmx,y,s2,m);  endmodule |  |

**Kết quả mô phỏng:**

**A screenshot of a graph

Description automatically generated**

**b) Thiết kế bộ Mux\_5to1\_3-bit**

****

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| (Behavior)  module mux5to1\_3bit(  input [2:0] U,V,W,X,Y,  input s0,s1,s2,  output [2:0] M  );  assign M = (s2 == 1'b0 && s1 == 1'b0 && s0 == 1'b0) ? U :  (s2 == 1'b0 && s1 == 1'b0 && s0 == 1'b1) ? V :  (s2 == 1'b0 && s1 == 1'b1 && s0 == 1'b0) ? W :  (s2 == 1'b0 && s1 == 1'b1 && s0 == 1'b1) ? X :  Y ;  endmodule |  |
| (Structural)  module mux5to1\_3bit\_2(  input [2:0] U,V,W,X,Y,  input S0,S1,S2,  output [2:0] M  );  mux5to1\_1bit u0(  .u(U[0]),.v(V[0]),.w(W[0]),  .x(X[0]),.y(Y[0]),  .s2(S2),.s1(S1),.s0(S0),  .m(M[0])  );  mux5to1\_1bit u1(  .u(U[1]),.v(V[1]),.w(W[1]),  .x(X[1]),.y(Y[1]),  .s2(S2),.s1(S1),.s0(S0),  .m(M[1])  );  mux5to1\_1bit u2(  .u(U[2]),.v(V[2]),.w(W[2]),  .x(X[2]),.y(Y[2]),  .s2(S2),.s1(S1),.s0(S0),  .m(M[2])  );  endmodule |  |

**Kết quả mô phỏng**

**A screenshot of a computer

Description automatically generated**

**Câu 3.**

**a) Hiện thực bộ Decoder cho Led 7 đoạn (7-segment) có chức năng giải mã giá trị nhị phân 3 bit như trong bảng sự thật bên dưới.**

**A screen shot of a number

Description automatically generated**

**Bảng chuyển đổi**

|  |  |  |
| --- | --- | --- |
| **c2c1c0** | **Character** | **Led** |
| 000 | H | 1001000 |
| 001 | E | 0110000 |
| 010 | L | 1110001 |
| 011 | L | 1110001 |
| 100 | O | 0000001 |

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module segment7\_decoder(  input [2:0] C,  output[6:0] seg  );  assign seg = (C == 3'b000 ) ? 7'b1001000:  (C == 3'b001 ) ? 7'b0110000:  (C == 3'b010 ) ? 7'b1110001:  (C == 3'b011 ) ? 7'b1110001:  (C == 3'b100 ) ? 7'b0000001: 7'b0000000;  endmodule |  |

**Kết quả mô phỏng**

**A screenshot of a computer

Description automatically generated**

**b) Kết hợp với mạch trong câu a và mạch trong câu 3 để thiết kế mạch sau:A diagram of a device

Description automatically generated**

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module segment7\_controller(  input [2:0] a,b,c,d,e,  input S2,S1,S0,  output [6:0] out\_seg  );  wire [2:0] out\_mux;  mux5to1\_3bit mux\_inst (  .U(a),.V(b),.W(c),.X(d),.Y(e),  .s2(S2),.s1(S1),.s0(S0),  .M(out\_mux)  );  segment7\_decoder decoder\_inst(  .C(out\_mux),  .seg(out\_seg)  );  endmodule |  |

**Kết quả mô phỏng**

**A screenshot of a computer

Description automatically generated**

**Câu 4.**

**Thiết kế mạch chuyển đổi số Binary sang số Decimal như bảng sau:**A screenshot of a table

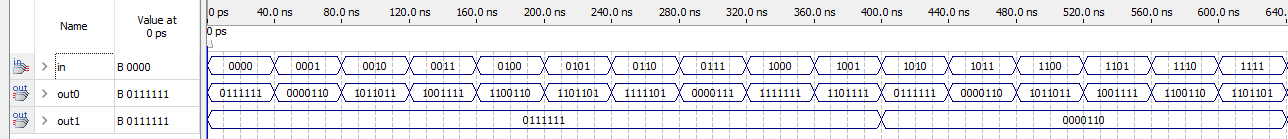
Description automatically generated

A diagram of a circuit

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Name | Code | RTL Viewer |
| comparator | module comparator(in,out);  input[3:0] in;  output out;  reg out;    always@(in)  begin  if(in > 4'b1001)  begin out = 1'b1; end  else  begin out = 1'b0; end  end  endmodule |  |
| CircuitA | module CircuitA(in,out);  input[2:0] in;  output reg[2:0] out;  always@(\*)  begin  case(in)  3'b000: out=3'b000;  3'b001: out=3'b001;  3'b010: out=3'b000;  3'b011: out=3'b001;  3'b100: out=3'b010;  3'b101:out=3'b011;  3'b110: out=3'b100;  3'b111: out=3'b101;  endcase  end  endmodule |  |
| CircuitB | module CircuitB(in,out);    input in;  output reg [6:0] out;  always@(\*)  begin  if(in==1'b1)  begin out=7'b0000110; end  else  begin out=7'b0111111; end  end  endmodule |  |
| DECODER | module DECODER(in,out);  input[3:0]in;  output reg [6:0]out;    always@(\*)  begin  case(in)  4'b0000: out=7'b0111111;  4'b0001: out=7'b0000110;  4'b0010: out=7'b1011011;  4'b0011: out=7'b1001111;  4'b0100: out=7'b1100110;  4'b0101: out=7'b1101101;  4'b0110: out=7'b1111101;  4'b0111: out=7'b0000111;  4'b1000: out=7'b1111111;  4'b1001: out=7'b1101111;  4'b1010: out=7'b0111111;  4'b1011: out=7'b0000110;  4'b1100: out=7'b1011011;  4'b1101: out=7'b1001111;  4'b1110: out=7'b1100110;  4'b1111: out=7'b1101101;  endcase  end  endmodule |  |
| Bai4 | module Bai4(in,out1,out0);  input[3:0]in;  output[6:0]out1;  output[6:0]out0;  wire COMOut;  wire[3:0]MUX;  wire[2:0]AOut;  comparator Compare(in,COMOut);  CircuitB B(COMOut,out1);  CircuitA A(in[2:0],AOut);    mux2to1 Mux3(in[3],0,COMOut,MUX[3]);  mux2to1 Mux2(in[2],AOut[2],COMOut,MUX[2]);  mux2to1 Mux1(in[1],AOut[1],COMOut,MUX[1]);  mux2to1 Mux0(in[0],AOut[0],COMOut,MUX[0]);      DECODER de(MUX,out0);    endmodule |  |

**Kết quả mô phỏng**



**Câu 5.**

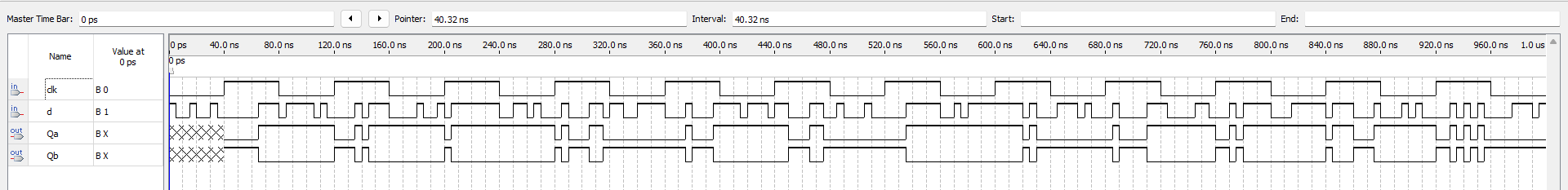
1. **Thực hiện mạch D-latch có sơ đồ như sau:**

**A diagram of a circuit

Description automatically generated**

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module D\_latch(  input d,clk,  output Qa,Qb  );  wire R,S\_g,R\_g;  not(R, d);  nand(S\_g, clk, d);  nand(R\_g, clk, R);  nand(Qb, R\_g, Qa);  nand(Qa, S\_g, Qb);  endmodule |  |

**Kết quả mô phỏng:**

****

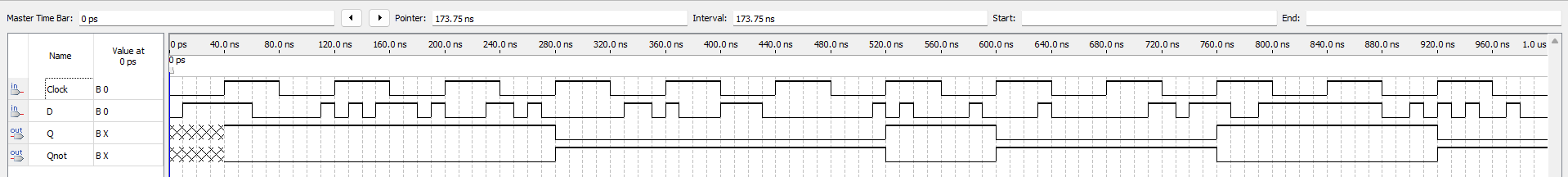
**b) Sử dụng D-latch trong câu a để thiết kế D-Flipflop kích cạnh lên theo dạng Master-Slave có sơ đồ như sau:**

**A diagram of a circuit

Description automatically generated**

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module D\_flipflop(  input D,Clock,  output Q,Qnot  );  wire Qm,nClk;  not(nClk,Clock);  D\_latch Master(  .d(D),.clk(nClk),  .Qa(Qm)  );  D\_latch Slave(  .d(Qm),.clk(Clock),  .Qa(Q),.Qb(Qnot)  );  endmodule |  |

**Kết quả mô phỏng:**

****

1. **Thực hiện mạch sau:**A diagram of a circuit

   Description automatically generated

|  |  |
| --- | --- |
| **Code** | **RTL Viewer** |
| module D\_flipflop\_3bit(  input D\_3b, Clock\_3bit,  output Qa\_3b, Qa\_not\_3b , Qb\_3b, Qb\_not\_3b, Qc\_3b, Qc\_not\_3b  );  not(nClk\_3bit,Clock\_3bit);  D\_latch Latch(  .d(D\_3b), .clk(Clock\_3bit),  .Qa(Qa\_3b), .Qb(Qa\_not\_3b)  );  D\_flipflop FF1(  .D(D\_3b),.Clock(Clock\_3bit),  .Q(Qb\_3b),.Qnot(Qb\_not\_3b)  );  D\_flipflop FF2(  .D(D\_3b),.Clock(nClk\_3bit),  .Q(Qc\_3b),.Qnot(Qc\_not\_3b)  );  endmodule |  |

**Kết quả mô phỏng:**

**A screenshot of a computer

Description automatically generated**