ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN ĐIỆN TỬ

----------o0o----------

COMPUTER ARCHITECTURE

Milestone 1

Design of a Vending Machine

Sinh viên thực hiện	Mã số sinh viên
Lê Đức Quý	2114597
Nguyễn Hữu Nhân	2111906

Giáo viên giảng dạy: TS. Trần Hoàng Linh

Người hướng dẫn: anh Cao Xuân Hải

TP. HỒ CHÍ MINH, THÁNG 3 NĂM 2024

I. VẤN ĐỀ

Vending Machine is a dispenser machine that receives coins or bills and dispenses soft drinks or snacks.

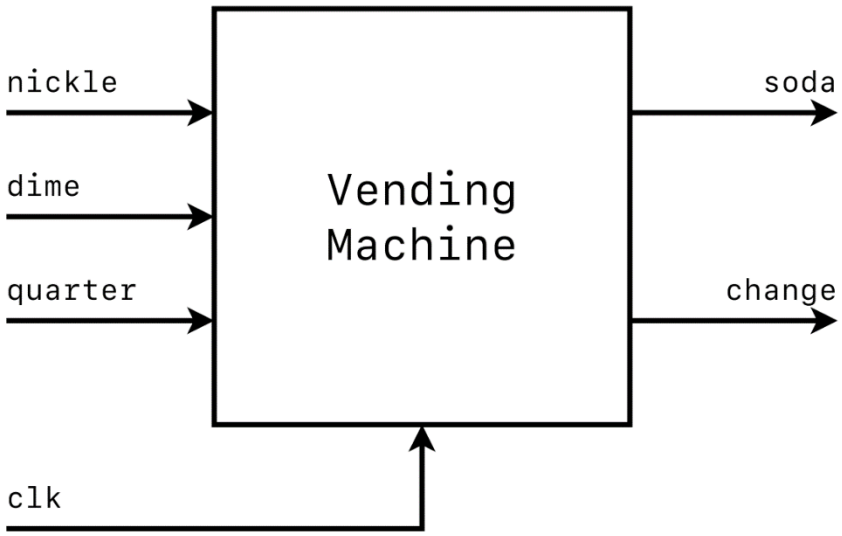


Figure 1: Vending machine's ports

In this problem, you will design a vending machine that satisfies the requirements below:

1. It accepts coins: ¢5 (Nickel), ¢10 (Dime), ¢25 (Quarter), but only one coin at a time (or clock).
2. When the deposit exceeds ¢20, it dispenses a soda and a change.
3. Change is a 3-bit data

000 ¢0

001 ¢5

010 ¢10

011 ¢15

100 ¢20

In this example, if a customer put a dime and then a quarter, in the next cycle, he received a soda and a change of ¢15.

I. GIẢI QUYẾT

1. Ý tưởng

Máy bán hàng tự động của nhóm hoạt động dựa trên một hệ thống trạng thái, với mỗi trạng thái tương ứng với một số tiền đã được nạp vào máy. Máy sẽ chuyển trạng thái dựa trên loại tiền được nạp vào, và sẽ phát ra soda và tiền thừa tương ứng khi đủ 20 cent.

Ví dụ, nếu máy đang ở trạng thái State_0 (chưa nhận tiền nào), và người dùng nạp vào một nickel, máy sẽ chuyển sang trạng thái State_1 (đã nhận 5 cent). Nếu tiếp tục nạp vào một dime, máy sẽ chuyển sang trạng thái State_3 (đã nhận 15 cent). Cuối cùng, nếu nạp thêm một nickel, máy sẽ chuyển sang trạng thái State_4 (đã nhận 20 cent), phát ra một lon soda, và quay trở lại trạng thái State_0.

2. Quy ước

Quy ước bit biểu diễn và ý nghĩa của trạng thái

State_0=4'b0000; chưa nhận tiền vào

State_1=4'b0001; đã nhận 5 cent

State_2=4'b0010; đã nhận 10 cent

State_3=4'b0011; đã nhận 15 cent

State_4=4'b0100; đã nhận 20 cent

State_5=4'b0101; đã nhận 25 cent

State_6=4'b0110; đã nhận 30 cent

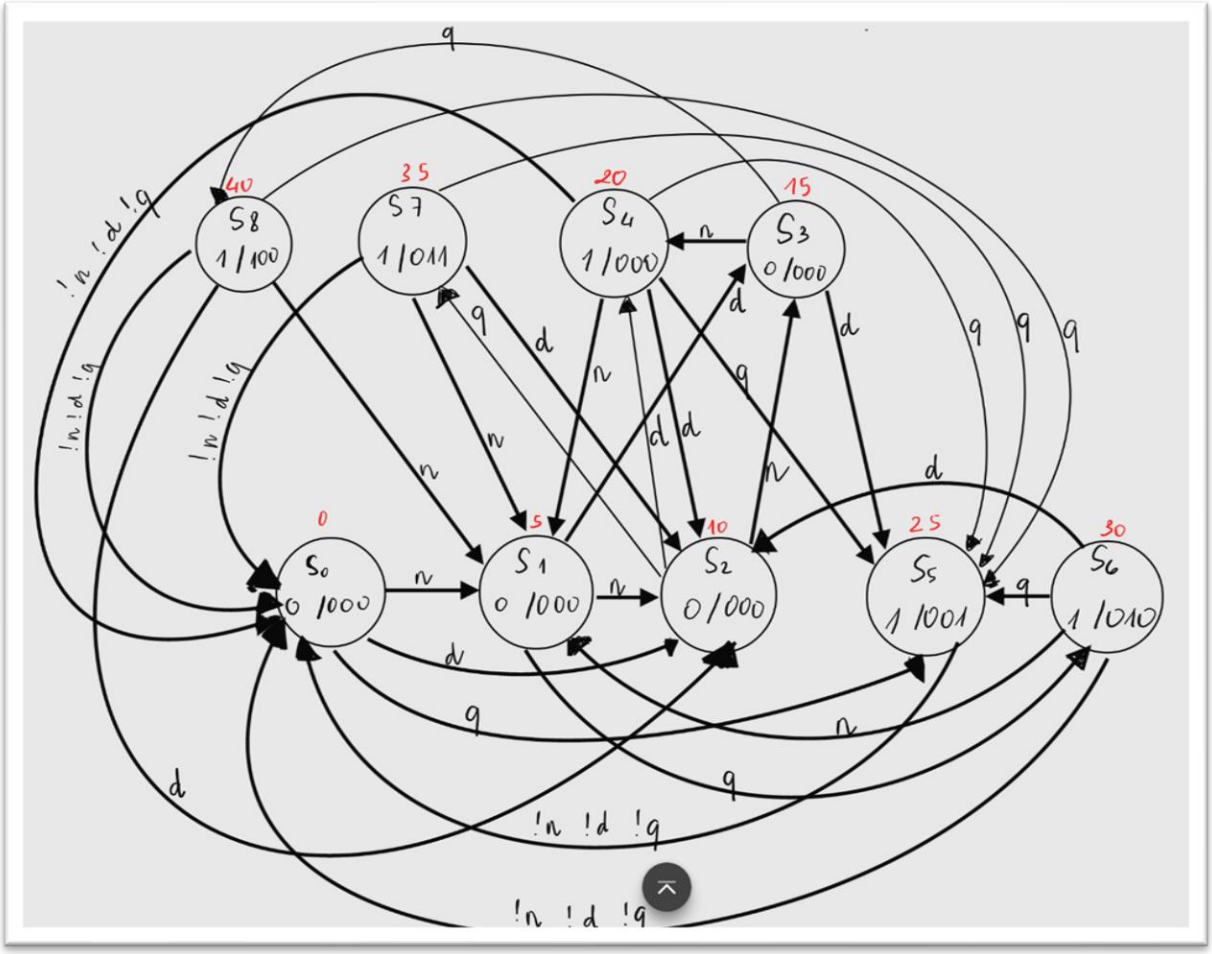
State_7=4'b0111; đã nhận 35 cent

State_8=4'b1000; đã nhận 40 cent

3. Hiện thực máy trạng thái (FSM)

TT hiện tại	TT kế tiếp						Ngõ ra	
	n=0	n=1	d=0	d=1	q=0	q=1	Soda	Change
S0	S0	S1	S0	S2	S0	S5	0	/000
S1 (5¢)	S1	S2	S1	S3	S1	S6	0	/000
S2 (10¢)	S2	S3	S2	S4	S2	S7	0	/000
S3 (có 15¢)	S3	S4	S3	S5	S3	S8	0	/000
S4 (có 20¢)	S0	S1	S0	S2	S0	S5	1	/000
S5 (có 25¢)	S0	S1	S0	S2	S0	S5	1	/001
S6 (có 30¢)	S0	S1	S0	S2	S0	S5	1	/010
S7 (có 35¢)	S0	S1	S0	S2	S0	S5	1	/011
S8 (có 40¢)	S0	S1	S0	S2	S0	S5	1	/100

Hình: Bảng chuyển trạng thái (State_transistion table).



Hình: Vẽ tay giản đồ trạng thái.

4. System Verilog code

```
module vendingmachine(
 input clk,
 input rst,
 input logic nick_i,
 input logic dime_i,
 input logic quar_i,
 output logic soda_o,
 output logic [2:0] chan_o
 );
 parameter State_0=4'b0000; // 0$
 parameter State_1=4'b0001; // 5$
 parameter State_2=4'b0010; // 10$
 parameter State_3=4'b0011; // 15$
 parameter State_4=4'b0100; // 20$
 parameter State_5=4'b0101; // 25$
 parameter State_6=4'b0110; // 30$
 parameter State_7=4'b0111; // 35$
 parameter State_8=4'b1000; // 40$

 logic [3:0] curr_state;
 logic [3:0] next_state;

 always@(posedge clk)
 begin
  if (rst)
    begin
       curr_state=4'b0000;
       next_state=4'b0000;
    end
  else
```

```
begin
  curr_state = next_state;

  case (curr_state)
    State_0: //0$
      begin
        if(nick_i)      begin next_state = State_1; end
        else if(dime_i) begin next_state = State_2; end
        else if(quar_i) begin next_state = State_5; end
        else            begin next_state = State_0; end
      soda_o <= 1'b0;
      chan_o <= 3'b000;
      end
    State_1: // 5$
      begin
        if(nick_i)      begin next_state = State_2; end
        else if(dime_i) begin next_state = State_3; end
        else if(quar_i) begin next_state = State_6; end
        else            begin next_state = State_1; end
      soda_o <= 1'b0;
      chan_o <= 3'b000;
      end
    State_2: // 10$
      begin
        if(nick_i)       begin next_state = State_3; end
        else if(dime_i)   begin next_state = State_4; end
        else if(quar_i)   begin next_state = State_7; end
        else             begin next_state = State_2; end
      soda_o <= 1'b0;
      chan_o <= 3'b000;
      end
```

```
State_3: // 15$
  begin
    if(nick_i)       begin next_state = State_4; end
    else if(dime_i)  begin next_state = State_5; end
    else if(quar_i)  begin next_state = State_8; end
    else             begin next_state = State_3; end
  soda_o <= 1'b0;
  chan_o <= 3'b000;
  end
State_4: //20$
  begin
    if(nick_i)       begin next_state = State_1; end
    else if(dime_i)  begin next_state = State_2; end
    else if(quar_i)  begin next_state = State_5; end
    else             begin next_state = State_0; end
  soda_o <= 1'b1;
  chan_o <= 3'b000;
  end
State_5://25$
  begin
    if(nick_i)      begin next_state = State_1; end
    else if(dime_i) begin next_state = State_2; end
    else if(quar_i) begin next_state = State_5; end
    else            begin next_state = State_0; end
  soda_o <= 1'b1;
  chan_o <= 3'b001;
  end
State_6://30$
  begin
    if(nick_i)      begin next_state = State_1; end
    else if(dime_i) begin next_state = State_2; end
```

```
            else if(quar_i) begin next_state = State_5; end
            else            begin next_state = State_0; end
          soda_o <= 1'b1;
          chan_o <= 3'b010;
          end
        State_7://35$
          begin
            if(nick_i)      begin next_state = State_1; end
            else if(dime_i) begin next_state = State_2; end
            else if(quar_i) begin next_state = State_5; end
            else            begin next_state = State_0; end
          soda_o <= 1'b1;
          chan_o <= 3'b011;
          end
        State_8://40$
          begin
            if(nick_i)      begin next_state = State_1;end
            else if(dime_i) begin next_state = State_2;end
            else if(quar_i) begin next_state = State_5;end
            else            begin next_state = State_0;end
          soda_o <= 1'b1;
          chan_o <= 3'b100;
          end
      endcase
    end
  end
endmodule : vendingmachine
```

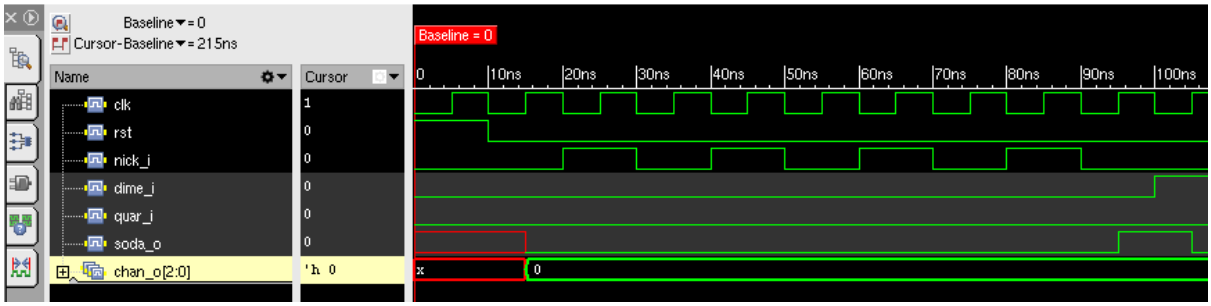
5. Code Testbench

```
module vendingmachine_tb();
    reg clk;
    reg rst;
    reg nick_i;
    reg dime_i;
    reg quar_i;
    wire soda_o;
    wire [2:0] chan_o;

    // Instantiate the vendingmachine
module
    vendingmachine dut(
        .clk(clk),
        .rst(rst),
        .nick_i(nick_i),
        .dime_i(dime_i),
        .quar_i(quar_i),
        .soda_o(soda_o),
        .chan_o(chan_o)
    );
    // Clock generation
    always begin
        #5 clk = ~clk;
    end
    // Testbench
    initial begin
        // Initialize inputs
        clk = 0;
        rst = 0;
        nick_i = 0;
        dime_i = 0;
        quar_i = 0;

        // Reset
        rst = 1;
        #10 rst = 0;
        // Test case: Insert 4 nickel
        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 nick_i = 1;
        #10 nick_i = 0;

      //Test case: Insert 2 dime
        #10 dime_i = 1;
        #10 dime_i = 0;

        #10 dime_i = 1;
        #10 dime_i = 0;

      //Test case: Insert 2 quarter
        #10 quar_i = 1;
        #10 quar_i = 0;

        #10 quar_i = 1;
        #10 quar_i = 0;

      // Test case: Insert 2 nickle + 1 dime
        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 dime_i = 1;
        #10 dime_i = 0;
      // test case : insert 1 nickle +1 dime+
1 quarter
        #10 nick_i = 1;
        #10 nick_i = 0;

        #10 dime_i = 1;
        #10 dime_i = 0;

        #10 quar_i = 1;
        #10 quar_i = 0;

        #10 $finish;
    end
endmodule
```

II. KẾT QUẢ

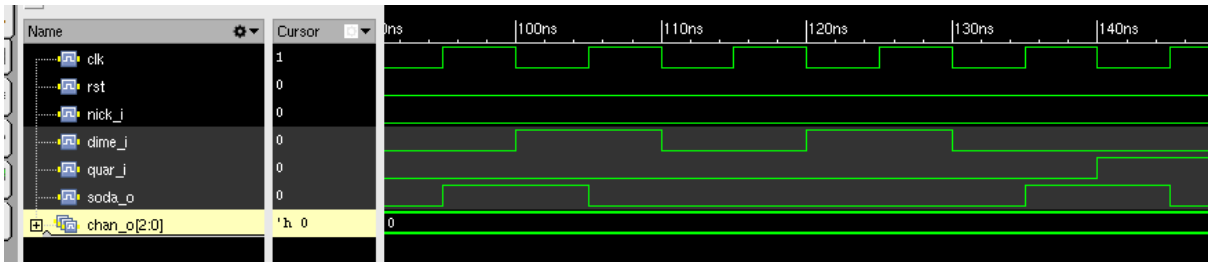
1. Mô phỏng dạng sóng

- Trường hợp 1



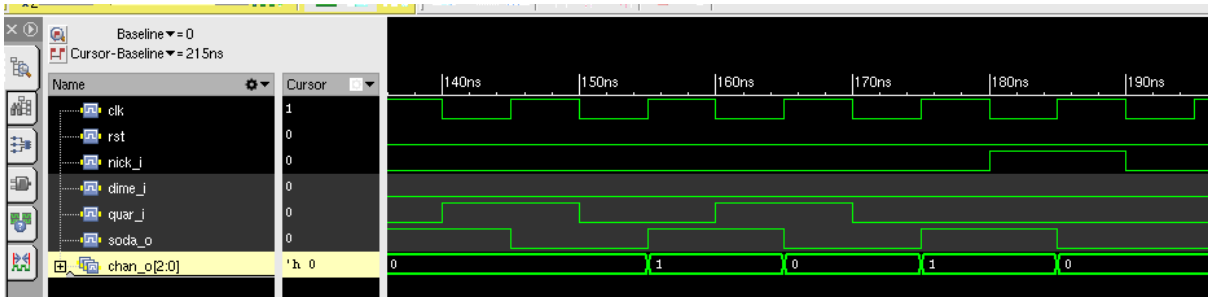
Hình 1: Trường hợp 4 nickle lên tiếp (cách nhau 1 xung clock).

- Trường hợp 2



Hình 2: Dạng sóng trường hợp 2 dime liên tiếp (cách nhau 1 xung clock).

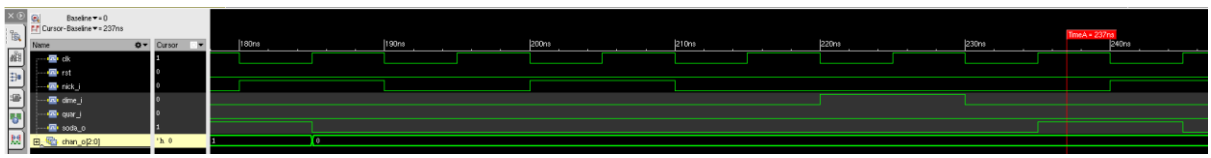
- Trường hợp 3



Hình 3: Dạng sóng trường hợp 2 quarter liên tiếp (cách nhau 1 xung clock).

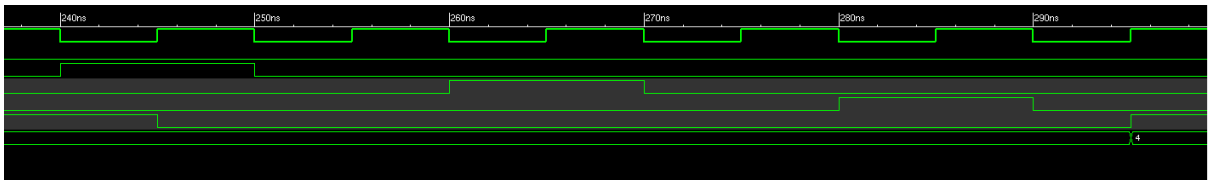
(Ngõ ra chan_o[2:0] có giá trị 1 decimal tức là 01 binary.

- Trường hợp 4



Hình 4: Dạng sóng trường hợp 2 nickle và 1 dime.

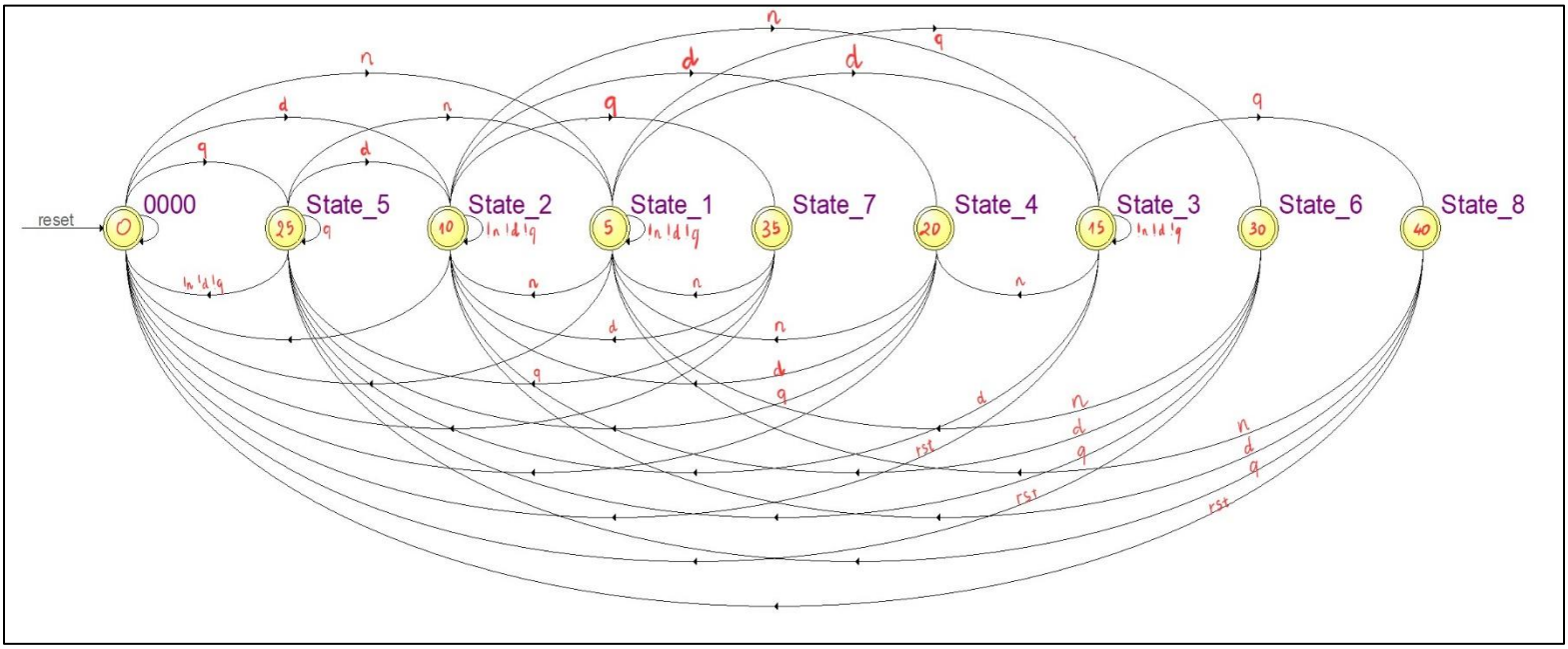
- Trường hợp 5



Hình 5: Dạng sóng trường hợp 1 nickle, 1 dime, 1 quarter.

(Ngõ ra chan_o [2:0] có giá trị 4 decimal, tức là 100 binary)

2. Giản đồ máy trạng thái tạo được.



3. RTL Viewer