TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

Khoa Điện – Điện tử

Bộ môn Điện tử

_____***_____

BÁO CÁO CẤU TRÚC MÁY TÍNH

LAB 3: VENDING MACHINE

GVHD: Trần Hoàng Linh

Sinh viên thực hiện: Đinh Thế Bảo

MSSV: 1510152

Tp. Hồ Chí Minh, tháng 12 năm 2019

I. Yêu cầu

Thiết kế máy bán nước ngot tự động (vending machine) tuân thủ các nguyên tắc sau:

- Nước ngọt (Soda) giá 9000, nước suối (Water) giá 7000
- Máy nhận xu: 1000, 2000 và 5000 (N, D, Q)
- Số tiền trả lại sao cho số xu ít nhất (Give change in the smallest # coins possible)
- Nếu số tiền bỏ vào lớn hơn 9000 máy sẽ tự trả lại tiền vừa bỏ vào sau.
- Máy có nút Coin Return (CR) dùng để trả lại hết tiền vừa bỏ vào.
- Nếu không có nút nào được ấn thì máy trạng thái giữ nguyên trạng thái cũ.
- Các ngõ ra:
    - Coin Return out (trả hết tiền khi CR được bấm)
    - Water out (WO) (mua nước suối)
    - Soda out (SO) (mua nước ngọt)
    - Change (CO) (trả tiền thừa)

II. Mục tiêu

Thực hiện thiết kế một máy bán nước ngọt tự động thỏa mãn những yêu cầu trên.

Giả sử các các đồng xu đã được đếm qua một bộ đếm để xác định số lượng và mệnh giá.

Dựa vào những yêu cầu trên, ta có thể chia máy này thành 3 khối chủ yếu: khối thực hiện chức năng điều khiển, khối thực hiện chức năng so sánh, khối thực hiện chức năng trả lại tiền thừa.

Quá trình hoạt động:

Khi máy nhận được số tiền, khối so sánh sẽ kiểm tra giá trị đã nhận đó, sẽ có 3 khả năng có thể xảy ra:

- Nhỏ hơn 7000: led lỗi số 1 sẽ được bật, báo hiệu số tiền đưa vào nhỏ hơn số tiền cần thiết.
- Lớn hơn 7000 và nhỏ hơn 9000: Máy sẽ thực hiện chờ cho đến khi nhận được sự lựa chọn.
- Lớn hơn 9000: Máy sẽ thực hiện trả lại số tiền thừa, và chỉ giữ lại giá trị 9000 trong bộ nhớ, sau đó sẽ chờ đến khi nhận được sự lựa chọn.

III. Chương trình đã viết

3.1. Khối control

```
module control (clk, rst, soda, water, return_coin, emty, seven, eight, nine,
    soda_out, water_out, recoin, led_error);

input clk, rst, emty, seven, eight, nine, water, soda, return_coin;
```

```
output  soda_out, water_out, led_error;
output [3:0] recoin;

wire emty, seven, eight, nine, water, soda;
wire soda_out, water_out, led_error;
wire [5:0] ngo_vao;
wire [2:0] temp;

reg [6:0] ngo_ra;


assign ngo_vao [5] = water;
assign ngo_vao [4] = soda;
assign ngo_vao [3] = seven;
assign ngo_vao [2] = eight;
assign ngo_vao [1] = nine;
assign ngo_vao [0] = emty;

always @ (posedge clk)
begin
        if(rst)
                ngo_ra=7'b0000000;
        else if (return_coin) begin
                ngo_ra = 7'b0000000;
        end
        else
                case (ngo_vao)
                    6'b100001: ngo_ra=7'b1000000;
                    6'b101001: ngo_ra=7'b0100000;
                    6'b100101: ngo_ra=7'b0100001;
                    6'b100011: ngo_ra=7'b0100010;
                    6'b010001: ngo_ra=7'b1000000;
                    6'b011001: ngo_ra=7'b1000000;
                    6'b010101: ngo_ra=7'b1000000;
                    6'b010011: ngo_ra=7'b0010000;
                    default: $display ("Error in SEL");
                endcase
end
assign led_error = ngo_ra [6];
assign water_out = ngo_ra [5];
assign soda_out = ngo_ra [4];
assign recoin = ngo_ra [3:0];
endmodule
```

3.2. Khối compare

```
module compare (clk, rst, coin, seven, eight, nine, return_coin,
    recoin, error_led);

    input clk, rst, return_coin;
    input [3:0] coin;    //Ngõ vào nơi tổng số tiền xu được bỏ vào được đưa v
ào


    output seven; //Ngõ ra được setb khi tổng số tiền đó x<7000
    output eight; //Ngõ ra được setb khi tổng số tiền đó 7000<= x =<9000

    output nine; //Ngõ ra khi tổng số tiền đó x>9000
    output wire [3:0] recoin;
    output reg error_led;

    //reg [3:0] coin;
    reg [3:0] recoin_0;
    reg [3:0] temp;
    reg [2:0] number;

    always @(temp)
    temp = coin;

  // always @(sel or a or b)
    always @(posedge clk)
        if(rst) begin
                error_led=1'b0;
            end
        else if (return_coin) begin
            number = 3'b000;
            recoin_0 = coin;
            error_led=1'b0;
        end
        else begin
            if  (coin < 4'b0111) begin
                number = 3'b000;
                recoin_0 = coin;
                error_led=1'b1;
            end

            if (4'b0111 <= coin <= 4'b1001) begin
                recoin_0 = 4'b0000;
                error_led=1'b0;
                case(coin)
```

```
                    4'b0111: number=3'b001;
                    4'b1000: number=3'b010;
                    4'b1001: number=3'b100;
                endcase
            end

            if (coin > 4'b1001) begin
                number = 3'b100;
                recoin_0 = coin-9;
                error_led=1'b0;
            end

        end
    assign recoin = recoin_0;
    assign seven = number [0];
    assign eight = number [1];
    assign nine = number [2];
endmodule
```

3.3. Khối return coin

```
module charge (money, coin_1000,coin_2000,coin_5000);
input [3:0] money;
output [3:0] coin_1000, coin_2000, coin_5000;
wire [3:0] coin;
wire [3:0] nam, hai, mot;
wire [3:0] coin_5, coin_2;
assign coin= money;
assign nam=coin/5;
assign coin_5= coin%5;
assign hai=coin_5/2;
assign coin_2=coin_5%2;
assign mot=coin_2;
assign coin_1000=mot;
assign coin_2000=hai;
assign coin_5000=nam;
endmodule
```

3.4. Vending machine

```
module top (clk, rst, count_1000_vnd, count_2000_vnd, count_5000_vnd,
soda, nuoc, return_coin, coin_1000, coin_2000, coin_5000, error_led,
soda_out, water_out, recoin_1, recoin_2, money);
input clk, rst;
input [3:0] count_1000_vnd, count_2000_vnd, count_5000_vnd;
input soda, nuoc, return_coin;

output wire soda_out, water_out;
```

```
output wire [3:0] coin_1000,coin_2000,coin_5000, money;
output [1:0] error_led;

wire [3:0] total;//, money;
wire seven, eight, nine;
output wire [3:0] recoin_1, recoin_2;

wire error_led_1, error_led_2;
wire [3:0] temp_recoin;

assign total = count_1000_vnd + 2*count_2000_vnd + 5*count_5000_vnd;

/*
countMoneyInput countMoneyInput_2(
     .rst(rst),
     .clk(clk),
     .count_1000_vnd(count_1000_vnd),
     .count_2000_vnd(count_2000_vnd),
     .count_5000_vnd(count_5000_vnd),
     .total(total)
);
*/

    compare compare_2(  .clk(clk),
                        .rst(rst),
                        .return_coin(return_coin),
                        .coin(total),
                        .seven(seven),
                        .eight(eight),
                        .nine(nine),
                        .recoin(recoin_1),
                        .error_led(error_led_1)
                        );

    control control_2(  .clk(clk),
                        .rst(rst),
                        .soda(soda),
                        .water(nuoc),
                        .return_coin(return_coin),
                        .emty(1),
                        .seven(seven),
                        .eight(eight),
                        .nine(nine),
                        .soda_out(soda_out),
                        .water_out(water_out),
```

```
                        .recoin(recoin_2),
                        .led_error(error_led_2)
                        );

    fulladder_4b fulladder_4b_recoin(   .a(recoin_1),
                                        .b(recoin_2),
                                        .ci(0),
                                        .co(),
                                        .sum(money)
                                        );

    charge charge_1(.money(money),
                    .coin_1000(coin_1000),
                    .coin_2000(coin_2000),
                    .coin_5000(coin_5000)
                    );

    assign error_led [1] = error_led_2;
    assign error_led [0] = error_led_1;

endmodule
```

3.5. Test bench

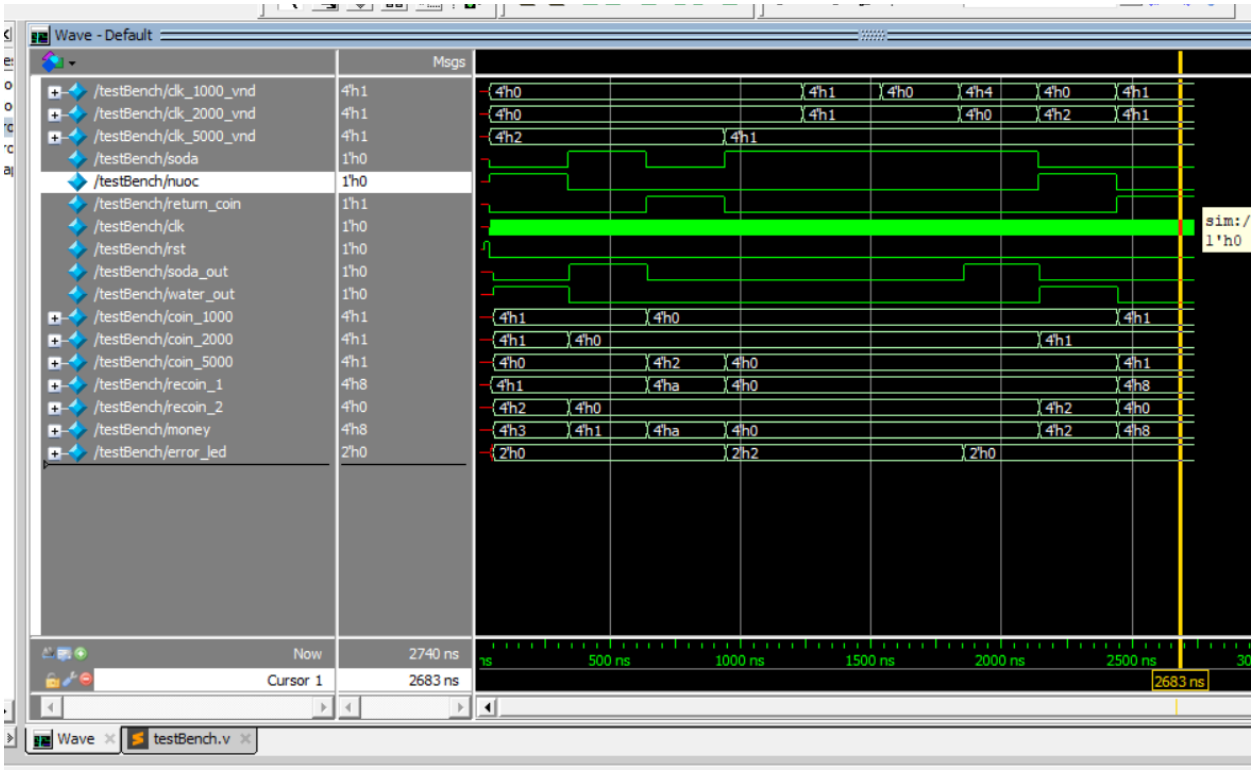
```
module testBench;
    reg [3:0] clk_1000_vnd; // Ngõ vào chứa số xu 1000 VNĐ được bỏ vào
    reg [3:0] clk_2000_vnd;// Ngõ vào chứa số xu 2000 VNĐ được bỏ vào
    reg [3:0] clk_5000_vnd; // Ngõ vào chứa số xu 5000 VNĐ được bỏ vào
    reg soda, nuoc, return_coin;
    reg clk, rst;

    wire soda_out,water_out;
    wire [3:0] coin_1000,coin_2000,coin_5000, recoin_1, recoin_2, money;
    wire [1:0] error_led;

    top top2 (
        .rst(rst),
        .clk(clk),
        .count_1000_vnd(clk_1000_vnd),
        .count_2000_vnd(clk_2000_vnd),
        .count_5000_vnd(clk_5000_vnd),
        .soda(soda),
        .nuoc(nuoc),
        .return_coin(return_coin),
        .money         (money),
        .recoin_1   (recoin_1),
```

```
        .recoin_2(recoin_2),
        .coin_1000(coin_1000),
        .coin_2000(coin_2000),
        .coin_5000(coin_5000),
        .error_led(error_led),
        .soda_out(soda_out),
        .water_out(water_out)
        );
    initial
    begin
        #20
        rst = 1'b1;
        #20
        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0000;
        clk_5000_vnd = 4'b0010;
        return_coin = 1'b0;
        soda = 1'b0;
        nuoc = 1'b1;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0000;
        clk_5000_vnd = 4'b0010;
        return_coin = 1'b0;
        soda = 1'b1;
        nuoc = 1'b0;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0000;
        clk_5000_vnd = 4'b0010;
        return_coin = 1'b1;
        nuoc = 1'b0;
        soda = 1'b0;
        #300

        rst = 1'b0;
        clk = 1'b1;
```

```
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0000;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b0;
        nuoc = 1'b0;
        soda = 1'b1;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0001;
        clk_2000_vnd = 4'b0001;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b0;
        nuoc = 1'b0;
        soda = 1'b1;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0001;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b0;
        nuoc = 1'b0;
        soda = 1'b1;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0100;
        clk_2000_vnd = 4'b0000;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b0;
        nuoc = 1'b0;
        soda = 1'b1;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0000;
        clk_2000_vnd = 4'b0010;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b0;
        nuoc = 1'b1;
```

```
        soda = 1'b0;
        #300

        rst = 1'b0;
        clk = 1'b1;
        clk_1000_vnd = 4'b0001;
        clk_2000_vnd = 4'b0001;
        clk_5000_vnd = 4'b0001;
        return_coin = 1'b1;
        nuoc = 1'b0;
        soda = 1'b0;
        #300

    $finish;
    end

    always
    #5
    clk = ~clk;


endmodule
```

3.6. Kết quả mô phỏng và nhận xét



Dựa vào biểu đồ dạng sóng, ta có thể thấy kết quả mô phỏng giống với yêu cầu.