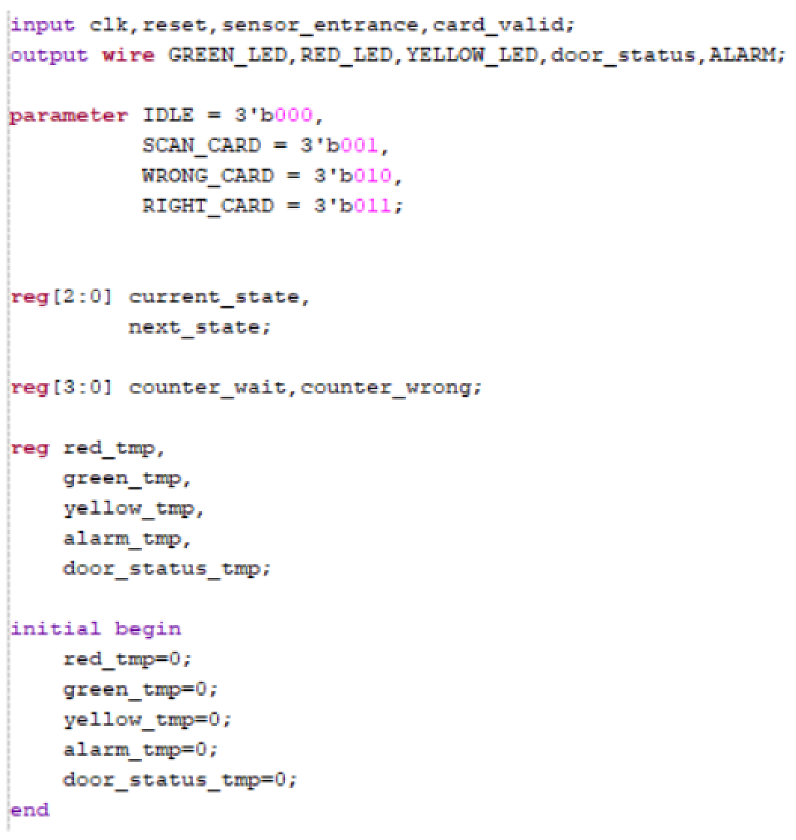
CARD SCANNER SYSTEM FOR DOOR

PURPOSE

The main purpose of this project is to scan the card that read into the scanner. Then perform the necessary applications at the door, which is open the door, lock the door, activate the alarm etc.

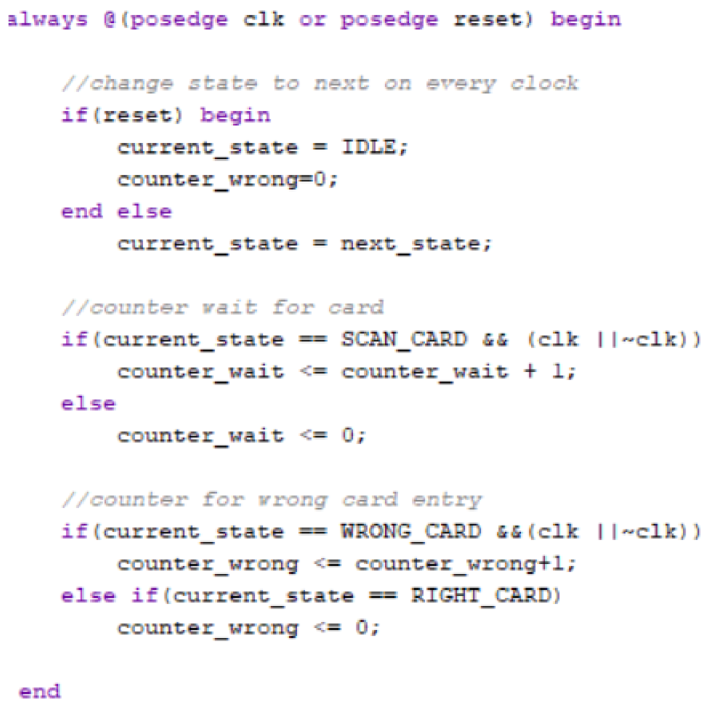
HOW IT WORKS

The system has 4 states (IDLE, SCAN_CARD, RIGHT_CARD, WRONG_CARD). First it starts with idle state. It does nothing in this state. When there is a card entrance, the system switches to SCAN_CARD state. In this state, the system scans the card for 3 clock cycle with help of counter (counter_wait) and also the yellow led turns on. After 3 cycles it looks whether the card is valid or not. First let's look at the wrong card case. If the card is not valid the state changes to WRONG_CARD. At this state, red led turns on and the wrong card counter increases by one (wrong_counter) if the counter reaches to 3 it triggers the alarm and alarm stays on until there is a valid card entry. Now let's look at the right card case. If the card is valid the state changes to RIGHT_CARD. At this state the door opens, the green led turns on and the wrong counter resets.

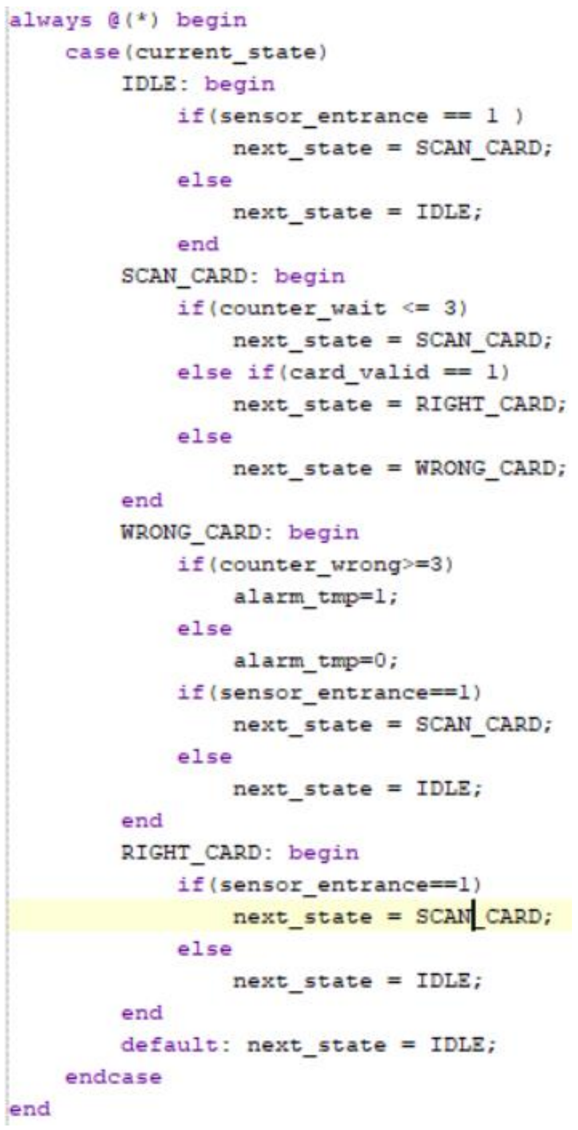


```
input clk,reset,sensor_entrance,card_valid;
output wire GREEN_LED,RED_LED,YELLOW_LED,door_status,ALARM;

parameter IDLE = 3'b000,
          SCAN_CARD = 3'b001,
          WRONG_CARD = 3'b010,
          RIGHT_CARD = 3'b011;


reg[2:0] current_state,
         next_state;

reg[3:0] counter_wait,counter_wrong;

reg red_tmp,
    green_tmp,
    yellow_tmp,
    alarm_tmp,
    door_status_tmp;

initial begin
    red_tmp=0;
    green_tmp=0;
    yellow_tmp=0;
    alarm_tmp=0;
    door_status_tmp=0;
end
```

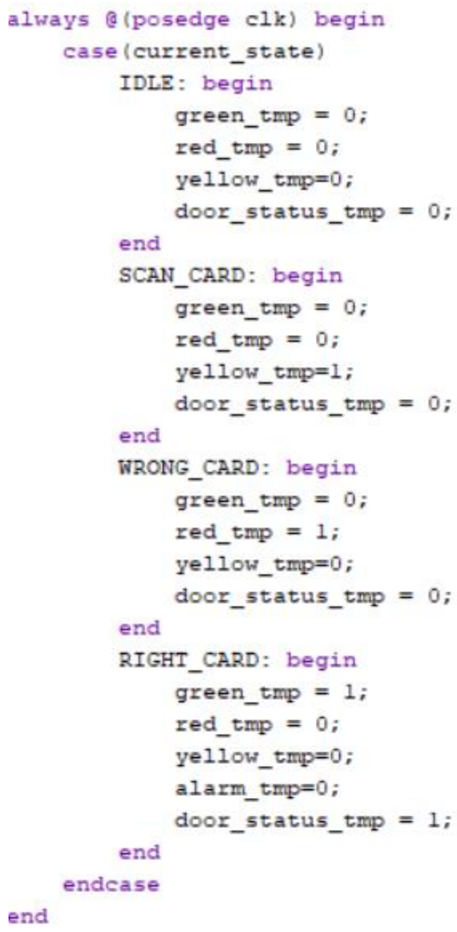
ALWAYS BLOCKS:



```
always @(posedge clk or posedge reset) begin

    //change state to next on every clock
    if(reset) begin
        current_state = IDLE;
        counter_wrong=0;
    end else
        current_state = next_state;

    //counter wait for card
    if(current_state == SCAN_CARD && (clk ||~clk))
        counter_wait <= counter_wait + 1;
    else
        counter_wait <= 0;

    //counter for wrong card entry
    if(current_state == WRONG_CARD &&(clk ||~clk))
        counter_wrong <= counter_wrong+1;
    else if(current_state == RIGHT_CARD)
        counter_wrong <= 0;

end
```

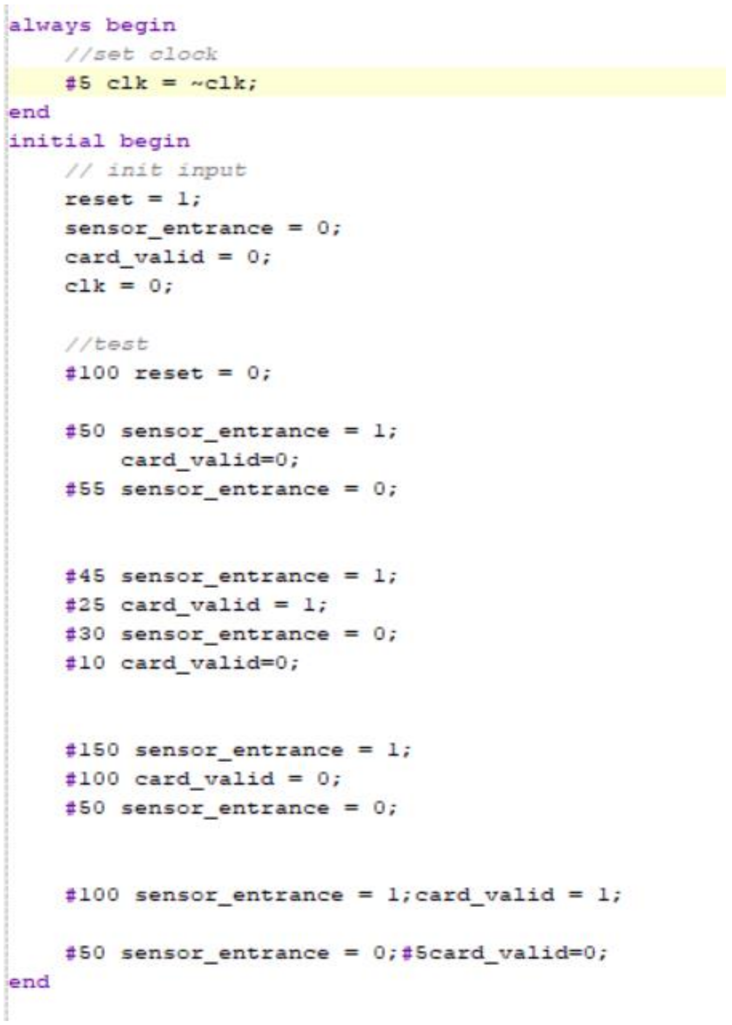


```
always @(*) begin
    case(current_state)
        IDLE: begin
            if(sensor_entrance == 1 )
                next_state = SCAN_CARD;
            else
                next_state = IDLE;
            end
        SCAN_CARD: begin
            if(counter_wait <= 3)
                next_state = SCAN_CARD;
            else if(card_valid == 1)
                next_state = RIGHT_CARD;
            else
                next_state = WRONG_CARD;
        end
        WRONG_CARD: begin
            if(counter_wrong>=3)
                alarm_tmp=1;
            else
                alarm_tmp=0;
            if(sensor_entrance==1)
                next_state = SCAN_CARD;
            else
                next_state = IDLE;
        end
        RIGHT_CARD: begin
            if(sensor_entrance==1)
                next_state = SCAN_CARD;
            else
                next_state = IDLE;
        end
        default: next_state = IDLE;
    endcase
end
```

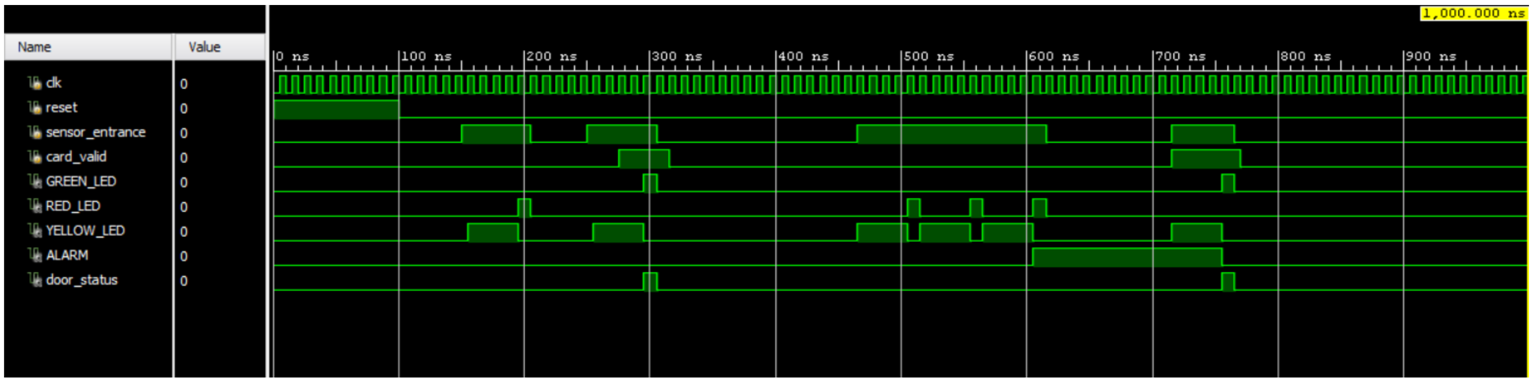


```
always @(posedge clk) begin
    case(current_state)
        IDLE: begin
            green_tmp = 0;
            red_tmp = 0;
            yellow_tmp=0;
            door_status_tmp = 0;
        end
        SCAN_CARD: begin
            green_tmp = 0;
            red_tmp = 0;
            yellow_tmp=1;
            door_status_tmp = 0;
        end
        WRONG_CARD: begin
            green_tmp = 0;
            red_tmp = 1;
            yellow_tmp=0;
            door_status_tmp = 0;
        end
        RIGHT_CARD: begin
            green_tmp = 1;
            red_tmp = 0;
            yellow_tmp=0;
            alarm_tmp=0;
            door_status_tmp = 1;
        end
    endcase
end
```

TEST BENCH AND SIMULATION



```
always begin
    //set clock
    #5 clk = ~clk;
end
initial begin
    // init input
    reset = 1;
    sensor_entrance = 0;
    card_valid = 0;
    clk = 0;

    //test
    #100 reset = 0;

    #50 sensor_entrance = 1;
        card_valid=0;
    #55 sensor_entrance = 0;


    #45 sensor_entrance = 1;
    #25 card_valid = 1;
    #30 sensor_entrance = 0;
    #10 card_valid=0;


    #150 sensor_entrance = 1;
    #100 card_valid = 0;
    #50 sensor_entrance = 0;


    #100 sensor_entrance = 1;card_valid = 1;

    #50 sensor_entrance = 0;#5card_valid=0;
end
```



ENES AKTURAN
150720050