



طراحی سیستم های دیجیتال

امتیازی



امیر محمد محفوظی

401106469

بهار ۱۴۰۳

سوال ۸:

برای حل این سوال به روش سنکرون عمل کرده و همچنین تمام input,output های مورد نیاز به همراه clock,reset قرار داده شده است. منطق کد در زیر هر مرحله توضیح داده شده است.

```
module ParkingManagement (  
    input wire clk,           // Clock signal  
    input wire reset,        // Reset signal  
    input wire car_entered,   // Car entered signal  
    input wire is_uni_car_entered, // Alumni car entered signal  
    input wire car_exited,    // Car exited signal  
    input wire is_uni_car_exited, // Alumni car exited signal  
    input wire [4:0] hour,    // Hour of the day (5 bits, 0 to 23)  
  
    output reg [9:0] uni_parked_car, // Number of alumni cars parked  
    output reg [9:0] parked_car,    // Number of non-alumni cars parked  
    output reg [9:0] uni_vacated_space, // Number of available spaces for alumni cars  
    output reg [9:0] vacated_space,    // Number of available spaces for non-alumni cars  
    output reg uni_is_vacated_space,    // Flag if there is space for alumni cars  
    output reg is_vacated_space         // Flag if there is space for non-alumni cars  
);  
  
    // Parameters for maximum capacities  
    parameter MAX_ALUMNI_CAPACITY = 500;  
    parameter TOTAL_CAPACITY = 700;  
  
    // Registers to keep track of capacities  
    reg [9:0] allUniSpace;  
    reg [9:0] allNonUniSpace;
```

در این قسمت صرفاً تعریف ورودی ها و خروجی ها با توجه به نیاز های سوال آمده است و اینکه هر کدام چه کاری انجام میدهند در کامنت آورده شده است. رجیستر های all هم بیشینه ظرفیت هر ۲ گروه را در هر لحظه نشان میدهد.

```
// Initialization  
initial begin  
    uni_parked_car = 0;  
    parked_car = 0;  
    uni_vacated_space = 500;  
    vacated_space = 200;  
    uni_is_vacated_space = 1;  
    is_vacated_space = 1;  
end
```

آماده سازی اولیه برای اولین مرتبه اجرا شدن.

```

// Adjust capacities based on the hour
always @(hour) begin
    if (hour < 8) begin
        allUniSpace = 0;
        allNonUniSpace = 0;
    end else if (hour >= 8 && hour < 13) begin
        allUniSpace = 500;
        allNonUniSpace = 200;
    end else if (hour >= 13 && hour < 16) begin
        allUniSpace = 500 - (hour - 12) * 50;
        allNonUniSpace = 200 + (hour - 12) * 50;
    end else if (hour >= 16) begin
        allUniSpace = 200;
        allNonUniSpace = 500;
    end
end
end

```

در این قسمت با منطق توضیح داده شده بیشینه ظرفیت هر گروه را با توجه به ساعت در این لحظه (منطقاً لیست حساسیت همان ساعت است) را محاسبه میکنیم. توجه داشته باشید که قبل از ساعت ۸ بارکینگ بسته و ظرفیت ۰ فرض شده است. همچنین رعایت شده که مجموع در تمام لحظات برابر ۷۰۰ باشد.

```

// Adjust non-alumni capacity if necessary
always @(posedge clk or posedge reset) begin
    if (reset) begin
        uni_parked_car <= 0;
        parked_car <= 0;
    end else begin
        if (uni_parked_car > allUniSpace && hour >= 13) begin
            parked_car <= parked_car + (uni_parked_car - allUniSpace);
            uni_parked_car <= allUniSpace;
            vacated_space <= allNonUniSpace - parked_car;
            uni_vacated_space <= 0;
            uni_is_vacated_space <= 0;
        end
    end
end
end

```

تکه کد بالا مطمئن میشود که در هر لحظه تعداد ماشین های دانشگاه از ظرفیت مجاز بیشتر نشود وگرنه باید به عنوان ظرفیت آزاد وارد شوند.

```

// Handle car entry and exit
always @(posedge car_entered or posedge car_exited) begin
    if (car_entered && hour >= 8) begin
        if (is_uni_car_entered) begin
            if (uni_is_vacated_space) begin
                uni_parked_car <= uni_parked_car + 1;
                uni_vacated_space <= uni_vacated_space - 1;
            end else if (is_vacated_space) begin
                parked_car <= parked_car + 1;
                vacated_space <= vacated_space - 1;
            end
        end
    end else begin
        if (is_vacated_space) begin
            parked_car <= parked_car + 1;
            vacated_space <= vacated_space - 1;
        end
    end
end

if (car_exited && hour >= 8) begin
    if (is_uni_car_exited && uni_parked_car > 0) begin
        uni_parked_car <= uni_parked_car - 1;
        uni_vacated_space <= uni_vacated_space + 1;
        uni_is_vacated_space <= 1;
    end else if (parked_car > 0) begin
        parked_car <= parked_car - 1;
        vacated_space <= vacated_space + 1;
        is_vacated_space <= 1;
    end
end
end

```

در این مرحله از کد هم به راحتی ورود و خروج ماشین ها با توجه به نوع سیگنال ورودی و خروجی و ظرفیت های مربوطه یکی یکی کنترل شده است.

```
// Update vacated spaces and flags
always @(*) begin
    uni_vacated_space = allUniSpace - uni_parked_car;
    vacated_space = allNonUniSpace - parked_car;
    uni_is_vacated_space = (uni_vacated_space > 0);
    is_vacated_space = (vacated_space > 0);
end
```

تکه کد بالا هم برای آبدیت کردن بولین های مربوط به جای خالی و همچنین تعداد فضای خالی در هر لحظه (با هر تغییر) است. برای اطمینان از صحت عملکرد به یک testbench احتیاج داریم :

```
module ParkingManagement_tb;

    // Inputs
    reg clk;
    reg reset;
    reg car_entered;
    reg is_uni_car_entered;
    reg car_exited;
    reg is_uni_car_exited;
    reg [4:0] hour;

    // Outputs
    wire [9:0] uni_parked_car;
    wire [9:0] parked_car;
    wire [9:0] uni_vacated_space;
    wire [9:0] vacated_space;
    wire uni_is_vacated_space;
    wire is_vacated_space;

    // Instantiate the ParkingManagement module
    ParkingManagement uut (
        .clk(clk),
        .reset(reset),
        .car_entered(car_entered),
        .is_uni_car_entered(is_uni_car_entered),
        .car_exited(car_exited),
        .is_uni_car_exited(is_uni_car_exited),
        .hour(hour),
        .uni_parked_car(uni_parked_car),
        .parked_car(parked_car),
```

```

// Clock generation
always #5 clk = ~clk;

// Monitor task for displaying the state
initial begin
    $monitor("Time: %0t | Hour: %2d | Car Entered: %b | Is Uni Car Entered: %b | Car Exited: %b | Is Uni Car Exited: %b | Uni Parked Car: %3d | Non-Uni Parked Car: %3d | Uni Vacated Space: %3d | Non-Uni Vacated Space: %3d | Uni Vacated Flag: %b | Non-Uni Vacated Flag: %b",
        $time, hour, car_entered, is_uni_car_entered, car_exited, is_uni_car_exited, uni_parked_car, parked_car, uni_vacated_space, vacated_space, uni_is_vacated_space, is_vacated_space);
end

// Test cases
initial begin
    // Initialize inputs
    clk = 0;
    reset = 1;
    car_entered = 0;
    is_uni_car_entered = 0;
    car_exited = 0;
    is_uni_car_exited = 0;
    hour = 0;

    // Apply reset
    #10 reset = 0;

    // Test case: Before 8 AM, parking is closed
    hour = 7;
    #10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;

    // Test case: Parking opens at 8 AM, alumni car enters
    hour = 8;
    #10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;

    // Test case: Non-alumni car enters
    #10 car_entered = 1; is_uni_car_entered = 0; #10 car_entered = 0;

    // Test case: Alumni car exits
    #10 car_exited = 1; is_uni_car_exited = 1; #10 car_exited = 0;

    // Test case: Non-alumni car exits
    #10 car_exited = 1; is_uni_car_exited = 0; #10 car_exited = 0;
end

```

```

// Test case: Parking capacity changes at 1 PM
hour = 13;
#10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;

// Test case: Parking capacity further changes at 3 PM
hour = 15;
#10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;

// Test case: Parking capacity reaches max at 4 PM
hour = 16;
#10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;

// Test case: Overflow alumni cars after 1 PM
hour = 13;
repeat (510) begin
    #10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;
end

// Test case: Overflow non-alumni cars
hour = 8;
repeat (200) begin
    #10 car_entered = 1; is_uni_car_entered = 0; #10 car_entered = 0;
end

// Test case: Mixed entries and exits
hour = 15;
#10 car_entered = 1; is_uni_car_entered = 1; #10 car_entered = 0;
#10 car_exited = 1; is_uni_car_exited = 1; #10 car_exited = 0;
#10 car_entered = 1; is_uni_car_entered = 0; #10 car_entered = 0;
#10 car_exited = 1; is_uni_car_exited = 0; #10 car_exited = 0;

// Test case: Reset condition
#10 reset = 1; #10 reset = 0;

$stop;
end

endmodule

```

در کد های بالا تمام case ها در نظر گرفته شده اند. اعم از ورود و خروج و اورفلو شدن ظرفیت و حتی ورود غیر مجاز. نتایج حاصل از simulate کردن هم در زیر آمده است.

