基于 2023 年和 2024 年的 FPGA 国赛能力测试题风格（注重实际应用逻辑、状态判断、计数器及简单的算术运算），我为你设计了一道模拟题。

这道题结合了**计数逻辑**（类似电梯题）和**费率计算**（类似游戏机题），难度适中，非常适合作为考前练手。

2025 FPGA 国赛能力测试模拟题

注：题目二选一即可，所有题目只验收仿真波形。

一、智能停车场管理系统

设计一个智能停车场管理控制模块，要求如下：

1. 停车场总共有 **16** 个车位，初始状态车位全空（剩余车位为 16）。
2. **入场逻辑**：当检测到入场请求 i_entry 上升沿时：
    - 若有剩余车位，剩余车位减 1，栏杆控制信号 o_barrier 拉高一个时钟周期（模拟抬杆）。
    - 若车位已满（剩余车位为 0），则 o_full_alarm 信号拉高，直到下一次有车离场导致车位空出为止。
3. **出场与计费逻辑**：当检测到出场请求 i_exit 上升沿时：
    - 剩余车位加 1，o_barrier 拉高一个时钟周期。
    - 同时根据输入的停车时长 i_duration（单位：小时）和会员状态 i_vip 计算停车费 o_fee。
    - **计费规则**：
        - 普通用户（i_vip=0）：首小时 5 元，之后每小时 3 元。
        - 会员用户（i_vip=1）：全程每小时 2 元。
        - 若 i_duration 为 0，费用为 0。
4. **优先级**：复位信号 i_rst_n 优先级最高。入场和出场信号不会同时发生（无需考虑冲突）。

程序头：

code Verilog

downloadcontent_copy

expand_less

```
module parking_lot(
input                   i_clk,
input                   i_rst_n,
input                   i_entry,      // 入场请求
input                   i_exit,       // 出场请求
input                   i_vip,        // 1=会员, 0=普通
input         [3:0] i_duration, // 停车时长(小时)
output reg [4:0] o_spaces,    // 剩余车位
output reg            o_barrier,    // 栏杆控制(脉冲)
output reg            o_full_alarm, // 车位满报警
output reg [7:0] o_fee            // 停车费用
);
endmodule
```

Benchmark：

code Verilog

downloadcontent_copy

expand_less

```
module tb();

reg           i_clk = 0;
reg           i_rst_n = 0;
reg           i_entry = 0;
reg           i_exit = 0;
reg           i_vip = 0;
reg [3:0] i_duration = 0;
```

```
wire [4:0] o_spaces;
wire       o_barrier;
wire       o_full_alarm;
wire [7:0] o_fee;

parking_lot test(
    .i_clk(i_clk),
    .i_rst_n(i_rst_n),
    .i_entry(i_entry),
    .i_exit(i_exit),
    .i_vip(i_vip),
    .i_duration(i_duration),
    .o_spaces(o_spaces),
    .o_barrier(o_barrier),
    .o_full_alarm(o_full_alarm),
    .o_fee(o_fee)
);

always #5 i_clk = ~i_clk;

initial begin
    #15 i_rst_n = 1;

    // 1. 正常入场测试
    #20 i_entry = 1; #10 i_entry = 0; //  第 1 辆车
    #20 i_entry = 1; #10 i_entry = 0; //  第 2 辆车
```

```
        // 2. 快速填满测试 (模拟剩余 14 辆车入场)
        repeat(14) begin
            #20 i_entry = 1; #10 i_entry = 0;
        end

        // 3. 满位报警测试
        #20 i_entry = 1; #10 i_entry = 0; // 第 17 辆车尝试进入，应报警

        // 4. 普通用户出场计费测试 (时长 3 小时: 5 + 3*2 = 11 元)
        #50;
        i_vip = 0;
        i_duration = 4'd3;
        i_exit = 1; #10 i_exit = 0;

        // 5. 会员用户出场计费测试 (时长 5 小时: 5 * 2 = 10 元)
        #50;
        i_vip = 1;
        i_duration = 4'd5;
        i_exit = 1; #10 i_exit = 0;

        #100 $stop;
    end

endmodule
```

参考答案与解析

1. 设计思路

- **边沿检测**：题目要求在 i_entry 和 i_exit 的上升沿触发动作，因此需要设计打两拍的边沿检测逻辑（posedge detection）。
- **计数器逻辑**：维护一个 o_spaces 寄存器。入场减 1，出场加 1。需要注意边界条件（0 和 16）。
- **报警逻辑**：当 o_spaces == 0 且尝试入场时触发报警；当有车出场（o_spaces > 0）时解除报警。
- **算术逻辑**：
    - VIP：fee = duration * 2
    - Normal：fee = 5 + (duration - 1) * 3 (注意：如果 duration=0，费用为 0；如果 duration=1，费用为 5)。

2. 参考代码 (Verilog)

code Verilog

downloadcontent_copy

expand_less

```
module parking_lot(
input           i_clk,
input           i_rst_n,
input           i_entry,
input           i_exit,
input           i_vip,
input     [3:0] i_duration,
output reg [4:0] o_spaces,
output reg      o_barrier,
output reg      o_full_alarm,
output reg [7:0] o_fee
);
```

```
// ---  边沿检测逻辑  ---
reg r_entry_d1, r_entry_d2;
reg r_exit_d1, r_exit_d2;

always @(posedge i_clk or negedge i_rst_n) begin
    if (!i_rst_n) begin
        r_entry_d1 <= 1'b0; r_entry_d2 <= 1'b0;
        r_exit_d1  <= 1'b0; r_exit_d2  <= 1'b0;
    end else begin
        r_entry_d1 <= i_entry; r_entry_d2 <= r_entry_d1;
        r_exit_d1  <= i_exit;  r_exit_d2  <= r_exit_d1;
    end
end

wire w_entry_rise = r_entry_d1 & (~r_entry_d2);
wire w_exit_rise  = r_exit_d1 & (~r_exit_d2);

// ---  主逻辑  ---
always @(posedge i_clk or negedge i_rst_n) begin
    if (!i_rst_n) begin
        o_spaces     <= 5'd16; //  初始 16 个车位
        o_barrier    <= 1'b0;
        o_full_alarm <= 1'b0;
        o_fee        <= 8'd0;
    end else begin
        //  默认脉冲信号拉低
```

```
o_barrier <= 1'b0;

// 处理入场
if (w_entry_rise) begin
    if (o_spaces > 0) begin
        o_spaces    <= o_spaces - 1'b1;
        o_barrier <= 1'b1; // 抬杆
    end else begin
        o_full_alarm <= 1'b1; // 车位已满，报警
    end
end

// 处理出场
else if (w_exit_rise) begin
    // 只要有车出场，车位+1，且报警必然解除
    if (o_spaces < 5'd16) begin
        o_spaces         <= o_spaces + 1'b1;
        o_barrier        <= 1'b1; // 抬杆
        o_full_alarm <= 1'b0; // 解除报警

        // 计费逻辑
        if (i_duration == 0) begin
            o_fee <= 8'd0;
        end else if (i_vip) begin
            // 会员：2 元/小时
            o_fee <= {4'd0, i_duration} * 8'd2;
        end else begin
```

```
                    // 普通：首小时 5 元，之后 3 元
                    if (i_duration == 1)
                        o_fee <= 8'd5;
                    else
                        o_fee <= 8'd5 + ({4'd0, i_duration} - 1'b1) * 8'd3;
                end
            end
        end
    end
end

endmodule
```

3. 仿真波形预期分析

1. **0-15ns**: 复位，o_spaces 置为 16。
2. **入场阶段**: 随着 i_entry 的脉冲，o_spaces 递减（16 -> 15 -> ... -> 0）。每次成功入场 o_barrier 都会产生一个高电平脉冲。
3. **满位报警**: 当 o_spaces 降为 0 后，再次给 i_entry 脉冲，o_spaces 保持 0，o_barrier 不动作，o_full_alarm 变高。
4. **普通用户出场**: i_exit 上升沿，i_vip=0, i_duration=3。
    - o_spaces 变为 1。
    - o_full_alarm 变低。
    - o_fee 更新为 11 (5 + 2*3)。
5. **会员用户出场**: i_exit 上升沿，i_vip=1, i_duration=5。
    - o_spaces 变为 2。
    - o_fee 更新为 10 (5 * 2)。