问题一

● **问题要求**

设计一个状态机，状态机有三个状态，IDLE，S1 和 S0，有两个输入 key1 和 key2，以及两个输出 cnt_en 和 load。注：初始状态为 IDLE，所有信号与时钟同步。
在 IDLE 状态时，cnt_en=0，load=1。当 key1=1，则跳转到 S1 状态，同时 cnt_en=1，load=0，否则保持当前状态不变；
在 S1 状态下，当 key1=1，则跳转到 S2 状态，同时 cnt_en=0，否则保持当前状态不变；
在 S2 状态下，当 key1=1，则跳转到 S1 状态，同时 cnt_en=1；当 key2=1，则跳转到 IDLE 状态，同时 load=1。
根据以上描述练习画出状态转移图，下图是参考状态图，（也可以练习写出状态转移表）使用三段式状态机描述方法编写代码，并仿真验证。提交源程序，仿真程序和仿真波形（最好包含状态寄存器）。

● **源程序：**

fsm.v

```
module fsm(
input clk,
input key1,
input key2,
output cnt_en,load
    );
    parameter IDLE = 2'b00,
    S1 = 2'b01,
    S2 = 2'b10;
    reg [1:0] cstate = IDLE, nstate = IDLE;

    always @(posedge clk)
    cstate = nstate;

    always @*
    if((cstate == IDLE) && (key1 == 1)) nstate <= S1;
    else if((cstate == S1) && (key1 == 1)) nstate <= S2;
    else if((cstate == S2) && (key1 == 1)) nstate <= S1;
    else if((cstate == S2) && (key2 == 1)) nstate <= IDLE;

    assign cnt_en = (cstate == S1)?1:0;
    assign load = (cstate == IDLE)?1:0;
endmodule
```

● **仿真程序：**

sim_fsm.v

```
module sim_fsm(
output reg clk = 0,
output reg [1:0] key = 0
    );
```

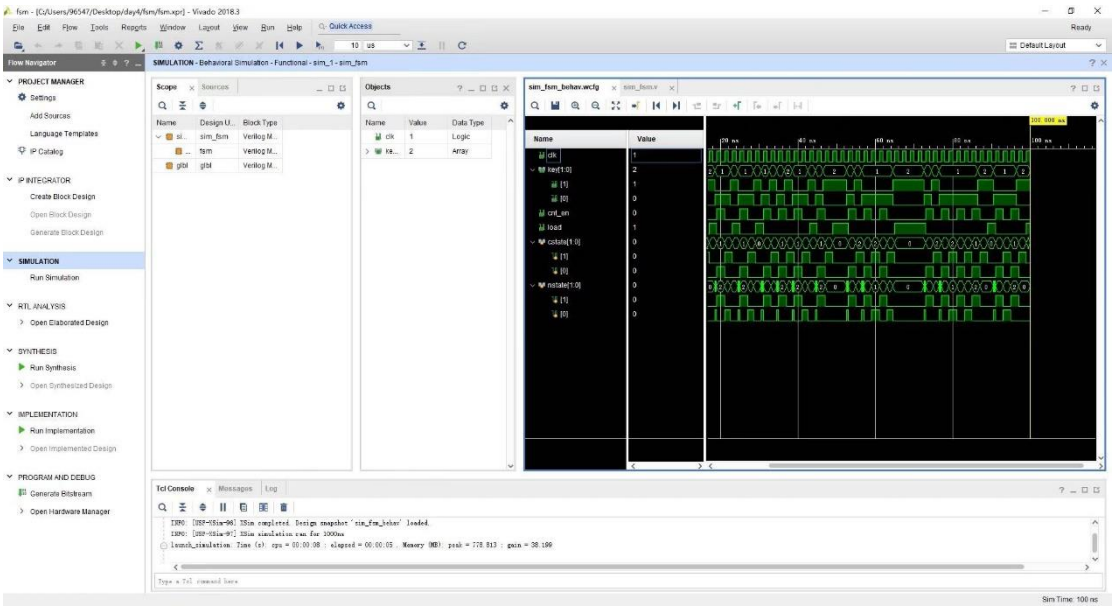
```
    fsm u1(
    .clk(clk),
    .key1(key[0]),
    .key2(key[1]));

    always
    #1 clk = !clk;

    initial
    #0.5
    while(1)
    begin
    #2 key[0] = $random;
    key[1] = !key[0];
    end

    initial
    #100 $stop;
endmodule
```

● **仿真波形：**



问题二

● **问题要求**

按键去抖动设计，按键去抖动模块至少有一个输出信号为 1 个 clk（系统时钟）周期的按键有效信号。提交设计程序。按键去抖动有多种方法，也可以试着采用状态机完成。

● **源程序：**

top.v

```
module top#(parameter DEBUG = 0) (
input clk,
input keyIn,
input keyOut
    );
    wire clk100ms, keyDejitter;

     sec
     #(.MAX((DEBUG == 1)?50:5_000_000)) clklow
     (
     .clk(clk),
     .rst(1),
     .sec(clk100ms));

    fsm trigger(
    .clk(clk),
    .clk100ms(clk100ms),
    .keyIn(keyIn),
    .keyOut(keyDejitter));

    differenciator diff(
    .clk(clk),
    .NumIn(keyDejitter),
    .NumOut(keyOut));

Endmodule
Fsm.v
module fsm#(parameter DEBUG = 0)
(
input clk,
input clk100ms,
input keyIn,
output keyOut
    );

    parameter [1:0] IDLE = 0, RESPOND = 1,PRESSED = 2;
    reg [1:0] cstate = IDLE, nstate = IDLE;
    wire keyHold;

    always @(posedge clk)
    cstate = nstate;

    always @(posedge clk100ms)
    if(keyIn == 1)
```

```
    if(cstate == IDLE) nstate = RESPOND;
     else nstate = PRESSED;
    else nstate = IDLE;

    assign keyOut = (cstate == RESPOND)?1:0;
endmodule
```

differenciator.v

```
module differenciator(
input clk,
input NumIn,
input NumOut
    );

    reg cstate = 0, nstate = 0;
    always @(posedge clk)
    cstate = nstate;

    always @(posedge clk)
    nstate = NumIn;

    assign NumOut = (cstate == 0) && (nstate == 1)?1:0;

endmodule
```

- **仿真程序：**

sim_fsm.v

```
module sim_fsm(
output reg clk = 0,
output reg keyIn = 0
    );

    top #(1)
    u1(
    .clk(clk),
    .keyIn(keyIn));

    always
    #1 clk = !clk;

    always begin
    #(200 + $random %200)
    repeat(20) #1 keyIn = $random;
    keyIn = 1;
    #(200 + $random %100)
    repeat(20) #1 keyIn = $random;
```

```
    keyIn = 0;
    end

    initial
    #10000 $stop;
Endmodule
```

- **仿真波形：**