```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 05/24/2022 05:06:34 PM
// Design Name:
// Module Name: SM
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module SM(
    input clk,
    input btnC,
    input btnD,
    input twoSec,
    input colision,
    input btnU,
    input frog232,
    input frog136,
    input frog328,

    output up,
    output down,
    output reset,
    output pause,
    output blink,
    //output score,
    output resetTime,
    output blinkscore
    );

    wire [9:0] d;
    wire [9:0] q;
```

```
   //hot logic
    assign up = q[4] | q[6] | q[8];
    assign down = q[5] | q[7] | q[9];
    assign reset = q[0] | q[1];
    assign pause = q[0] | q[1] | q[3] ;
    assign blink = q[1] | q[3];
    //assign score = q[0]| q[1] | q[2];
    assign resetTime = ~q[1];
    assign blinkscore = q[3];
    //transition states

    assign d[0] = (q[0] & ~btnC); //start
    assign d[1] = (q[0] & btnC) | (q[1] & ~twoSec) | (q[3] & btnC); //flick
    assign d[2] = (q[2] & ~colision & ~btnD & ~btnU) | (q[1] & twoSec) | (q[7] &
frog232 & ~colision) | (q[8] & frog232 & ~colision);//play
    assign d[3] = ((q[2] | q[9] | q[8] | q[7] | q[6] |q[5] | q[4]) & colision) |
(q[3] & ~btnC);//colision //NOT SURE IF I CAN DO Q[9:4] TO OR THEM ALL TOGETHER
    assign d[4] = (q[2] & btnU & ~colision) | (q[4] & frog232 & ~colision) ; // up
    assign d[5] = (q[2] & btnD & ~colision) | (q[5] & frog232 & ~colision) ; //down
    assign d[6] = (q[6] & ~frog136 & ~colision) | (q[4] & ~frog232 & ~colision) ;
//incUp
    assign d[7] = (q[6] & frog136 & ~colision) | (q[7] & ~frog232 & ~colision); //dec
    assign d[8] = (q[9] & frog328 & ~colision) | (q[8] & ~frog232 & ~colision);
//incDown
    assign d[9] = (q[5] & ~frog232 & ~colision) | (q[9] & ~frog328 & ~colision);
//decDown


    FDRE #(.INIT(1'b1) ) sm0 (.C(clk), .CE(1'b1), .D(d[0]), .Q(q[0]));
    FDRE #(.INIT(1'b0) ) sm1 (.C(clk), .CE(1'b1), .D(d[1]), .Q(q[1]));
    FDRE #(.INIT(1'b0) ) sm2 (.C(clk), .CE(1'b1), .D(d[2]), .Q(q[2]));
    FDRE #(.INIT(1'b0) ) sm3 (.C(clk), .CE(1'b1), .D(d[3]), .Q(q[3]));
    FDRE #(.INIT(1'b0) ) sm4 (.C(clk), .CE(1'b1), .D(d[4]), .Q(q[4]));
    FDRE #(.INIT(1'b0) ) sm5 (.C(clk), .CE(1'b1), .D(d[5]), .Q(q[5]));
    FDRE #(.INIT(1'b0) ) sm6 (.C(clk), .CE(1'b1), .D(d[6]), .Q(q[6]));
    FDRE #(.INIT(1'b0) ) sm7 (.C(clk), .CE(1'b1), .D(d[7]), .Q(q[7]));
    FDRE #(.INIT(1'b0) ) sm8 (.C(clk), .CE(1'b1), .D(d[8]), .Q(q[8]));
    FDRE #(.INIT(1'b0) ) sm9 (.C(clk), .CE(1'b1), .D(d[9]), .Q(q[9]));

    endmodule
```