```
`timescale 1ns / 1ps
/**********************************************
**********************************
*
* Module: Codebreaker
*
* Author: Eric Christie
* Class: ECEN 220, Section 1, Winter 2021
* Date: 3/30/21
*
* Description: Instances into a larger RC4
codebreaker module
*
*
**********************************************
**********************************/

`default_nettype none
module Codebreaker(
    input wire logic clk, reset, start,
    output logic [15:0] key_display,
    output logic stopwatch_run,
draw_plaintext,
    input wire logic
```

```
done_drawing_plaintext,
    output logic [127:0]
plaintext_to_draw);

    assign key_display = 0;   //Ex 1
    /*assign stopwatch_run = 1;
    assign plaintext_to_draw =
{"HELLO123"};
    assign draw_plaintext = start;*/

    //Logic wires
    logic [23:0] key;
    logic [127:0] cyphertext;
    logic decrypted, begin_decryption;

    //RC4 module    clk, reset, enable,
[24] key, [128] bytes_in, [128] bytes_out,
done
    decrypt_rc4 RC4(.clk(clk),
.reset(reset), .enable(begin_decryption),
.key(key), .bytes_in(cyphertext),
.bytes_out(plaintext_to_draw),
.done(decrypted));
```

```
    //assign key and cyphertext
    assign key = 24'h79726a;
    assign cyphertext =
128'h93a931affae622e10a029bd3d4bd6ced;

    //FSM
    typedef enum logic [2:0] {Wait,
Decrypting, Drawing, Terminate, ERR='X}
StateType;
    StateType ns, cs;

    always_comb begin
        ns = ERR;
        begin_decryption = 0;
        stopwatch_run = 0;
        draw_plaintext = 0;

        if(reset) ns = Wait;
        else
            case(cs)
                Wait:
                    if(start) ns =
Decrypting;
                    else ns = Wait;
```

```
                Decrypting: begin
                    begin_decryption = 1;
                    stopwatch_run = 1;
                    if(decrypted) ns =
Drawing;
                    else ns = Decrypting;
                end

                Drawing: begin
                    draw_plaintext = 1;

if(done_drawing_plaintext) ns = Terminate;
                    else ns = Drawing;
                end

                Terminate:
                    if(!reset) ns =
Terminate;
            endcase
    end

    always_ff @(posedge clk) cs <= ns;
```

```
endmodule
```