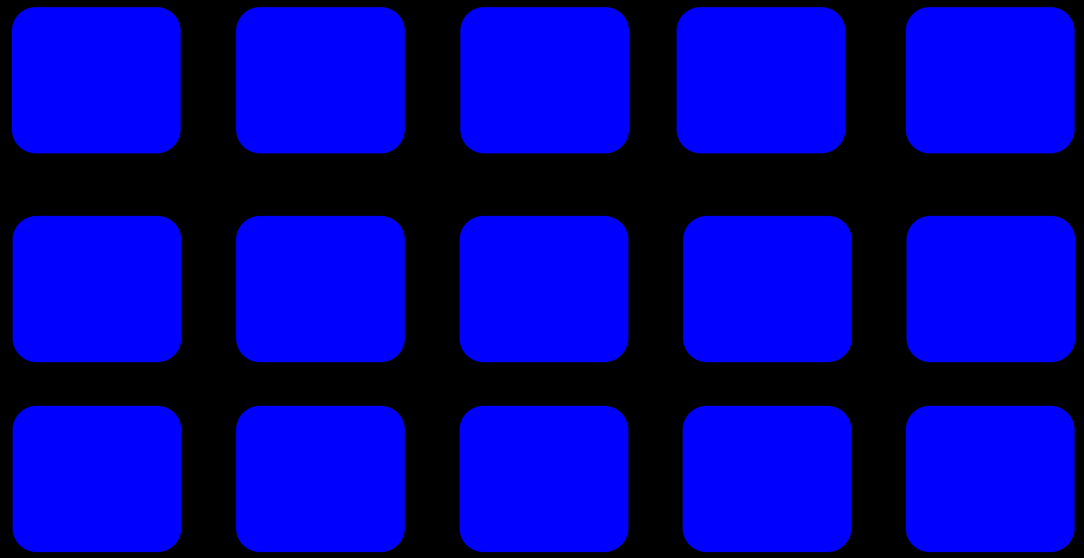
Memory Tile Game

By Anthony Carparu, Sam Kraft, Cole Knutsen, Josh A

Goals/Motivation

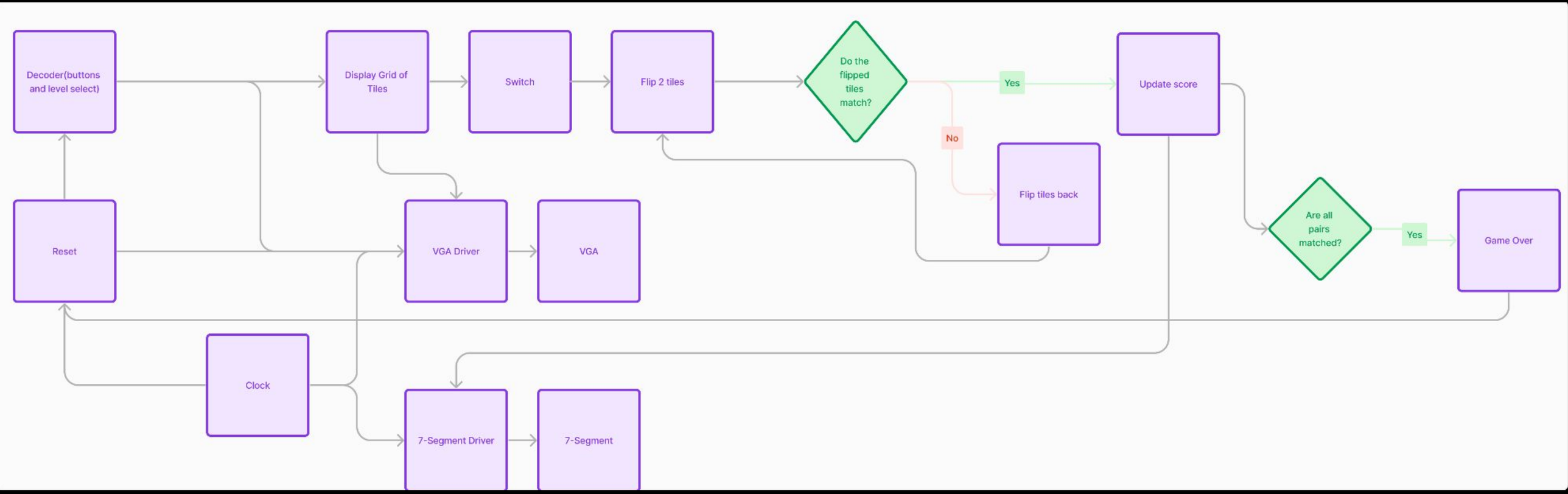
- ★ To recreate a memory game
- ★ Make a game where you match tiles.
- ★ We play this game when we have some free time and decided to recreate it.



Functionality

- We are creating the memory tile game in which you flip tiles over and try to match them.
- We display the game on the monitor through the VGA(it's a 4x4 tile) and use the switches to be able to flip tiles.
- Four levels of memory matching tiles
- Score displays on the 7 segment display on the FPGA.

Block Diagram



Specification

Requirements:

★ Need to use the VGA display as a screen to view the game.

★ Making sure that the tiles are different colors

★ Keep track of the score on 7 segment

Constraints:

★ Lab issues

★ Making the grid of tiles

★ Deadline

Code snippets

```
`timescale 1ns/1ps

module level_select(
    input clk,
    input reset,
    input [1:0] level,
    output reg [47:0] tile_setup  // 16 tiles x 3 bits each
);

    always @(posedge clk or posedge reset) begin
        if (reset) begin
            tile_setup <= 48'b0;
        end else begin
            case (level)
                2'b00: begin
                    tile_setup <= {3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7};
                end
                2'b01: begin
                    tile_setup <= {3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5};
                end
                2'b10: begin
                    tile_setup <= {3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1};
                end
                2'b11: begin
                    tile_setup <= {3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3, 3'd4, 3'd5, 3'd6, 3'd7, 3'd0, 3'd1, 3'd2, 3'd3};
                end
                default: begin
                    tile_setup <= 48'b0;
                end
            endcase
        end
    end
endmodule
```

Code snippets

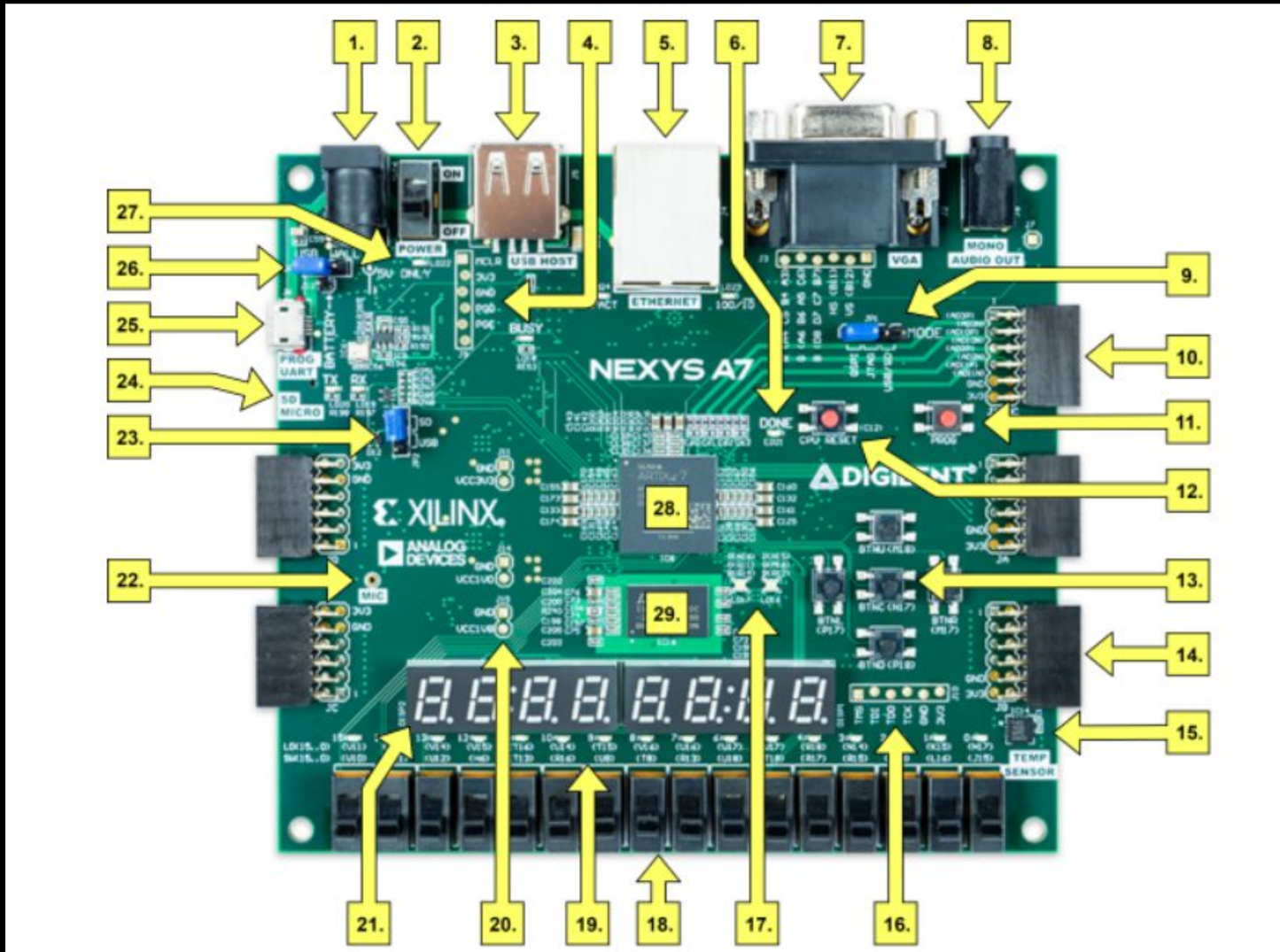
```
            first <= 4'b1111;
            second <= 4'b1111;
            select2 <= 0;
        end else begin
            for(i = 0; i < 16; i = i + 1) begin
                if(switches[i]) begin
                    if(!select2 && state[i] == 0) begin
                        first <= i;
                        select2 <= 1;
                        state[i] <= 1;
                    end else if(select2 && state[i] == 0) begin
                        second <= i;
                        select2 <= 0;
                        state[i] <= 1;

                        if(tiles[first[i]*3 +: 3] == tiles[second[i]*3 +: 3]) begin
                            match[first] <= 1;
                            match[second] <= 1;
                            leds[first] <= 1;
                            leds[second] <= 1;
                        end else begin
                            mismatch[first] <= 1;
                            mismatch[second] <= 1;
                        end
                            first <= 4'b1111;
                            second <= 4'b1111;
                    end
                end else if(!switches[i] && state[i] == 1) begin
                    state[i] <= 0;
                    if(i == first) first <= 4'b1111;
                    if(i == second) second <= 4'b1111;
                end
            end
        end
    end
endmodule
```

Failures

- Being able to flip the tiles
- Show animations when flipping(resorted to no animation)
- Getting the VGA to work
- switching our project idea

Successes

- ➔ Showing the tiles on the Monitor
- ➔ Game Logic
- ➔ Debouncer
- ➔ Inputs
- ➔ Four levels
- ➔ score



THANK YOU