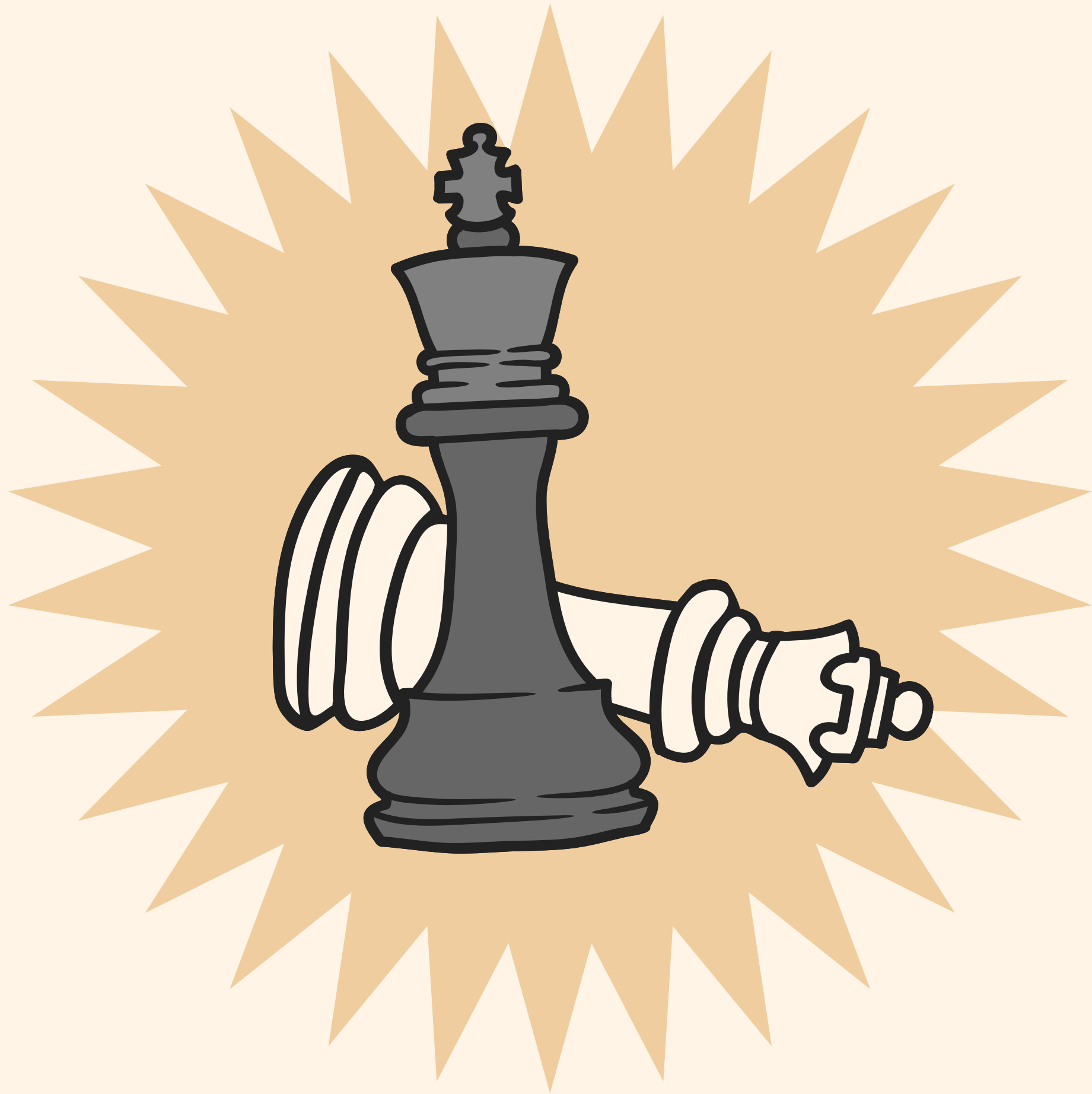


# CHESS ON THE FPGA



BY SOMETHING EASY TO  
REMEMBER

Dominic Murphy, Astrid Mihalopoulos, Rohan Alexander , Leonardo Mattos Martins



# GOALS & MOTIVATION

- **Create a pass n play game of chess on the FPGA using button inputs and the VGA display**
- **This design can be used for chess practice, fun among friends**
- **Chess is a notoriously difficult game to master, and we wanted get a different level of understanding about the game in order to improve**

# FUNCTIONALITY

**What was the design supposed to do:**

- **store the board as an array with 2 dimensions**
- **use an FSM to determine whether it is a pawn's first move to check if it can move 2 pieces**

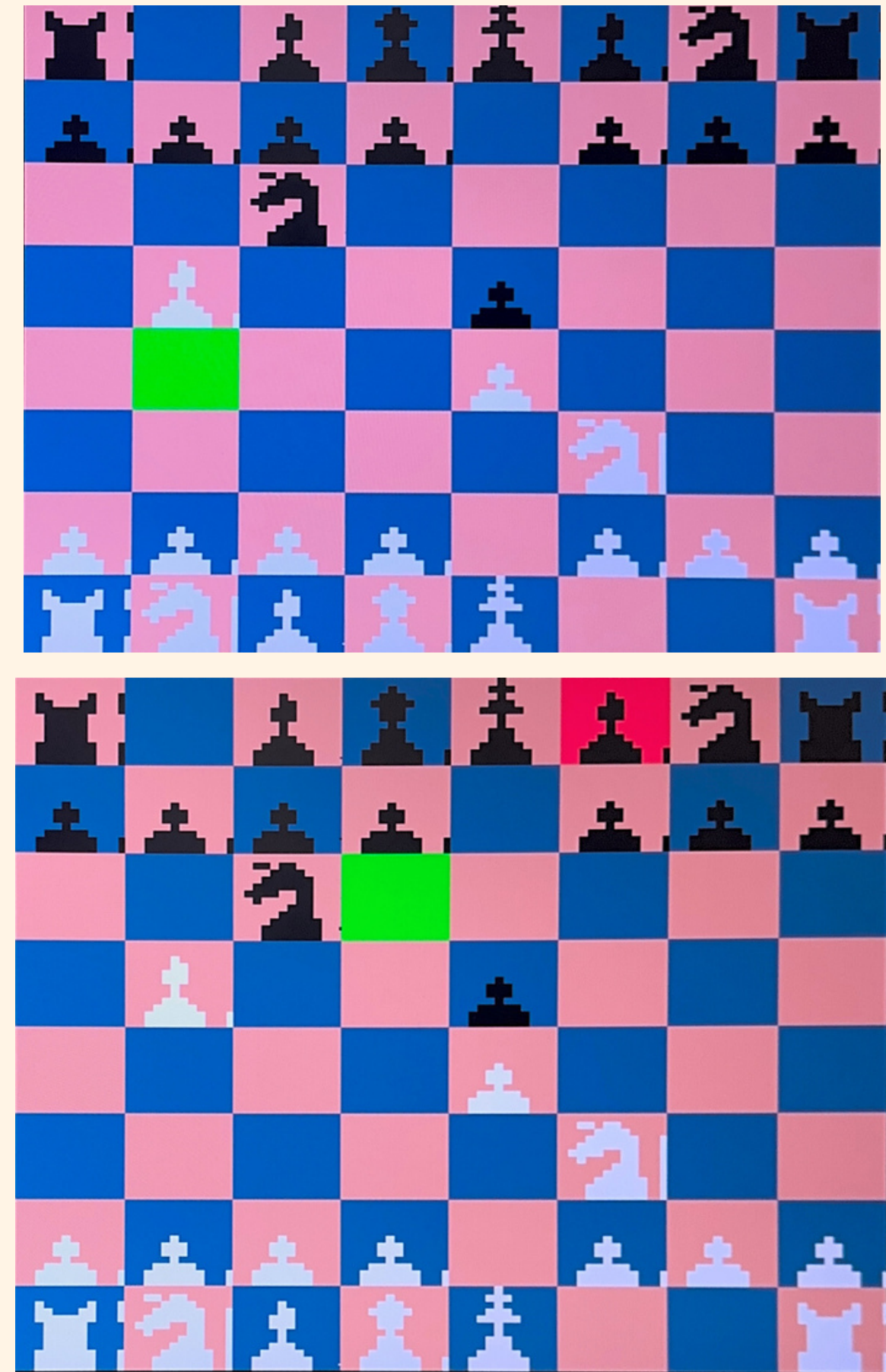
**What we are doing:**

- **store the board as a 256 bit register, and then convert to a board with 2 dimensions using genvar when necessary**
- **Using the pawn's coordinate to determine if movement is allowed;**



# SPECIFICATIONS

- **Display chess board on a VGA display that will update given player inputs**
- **Use button inputs to grab and move pieces**
- **Permit only allowed moves within the rules of chess**
- **Disallow moves which would cause a check**
- **Custom sprites for each piece**
- **Determine winner based on conditions such as checkmate, stalemate...**



# SUCCESSSES

- **VGA Display**
- **Button Inputs**
- **Grabbing Pieces**
- **Capturing**
- **Checking allowed moves**
- **Cursor**
- **Reset Functionality**



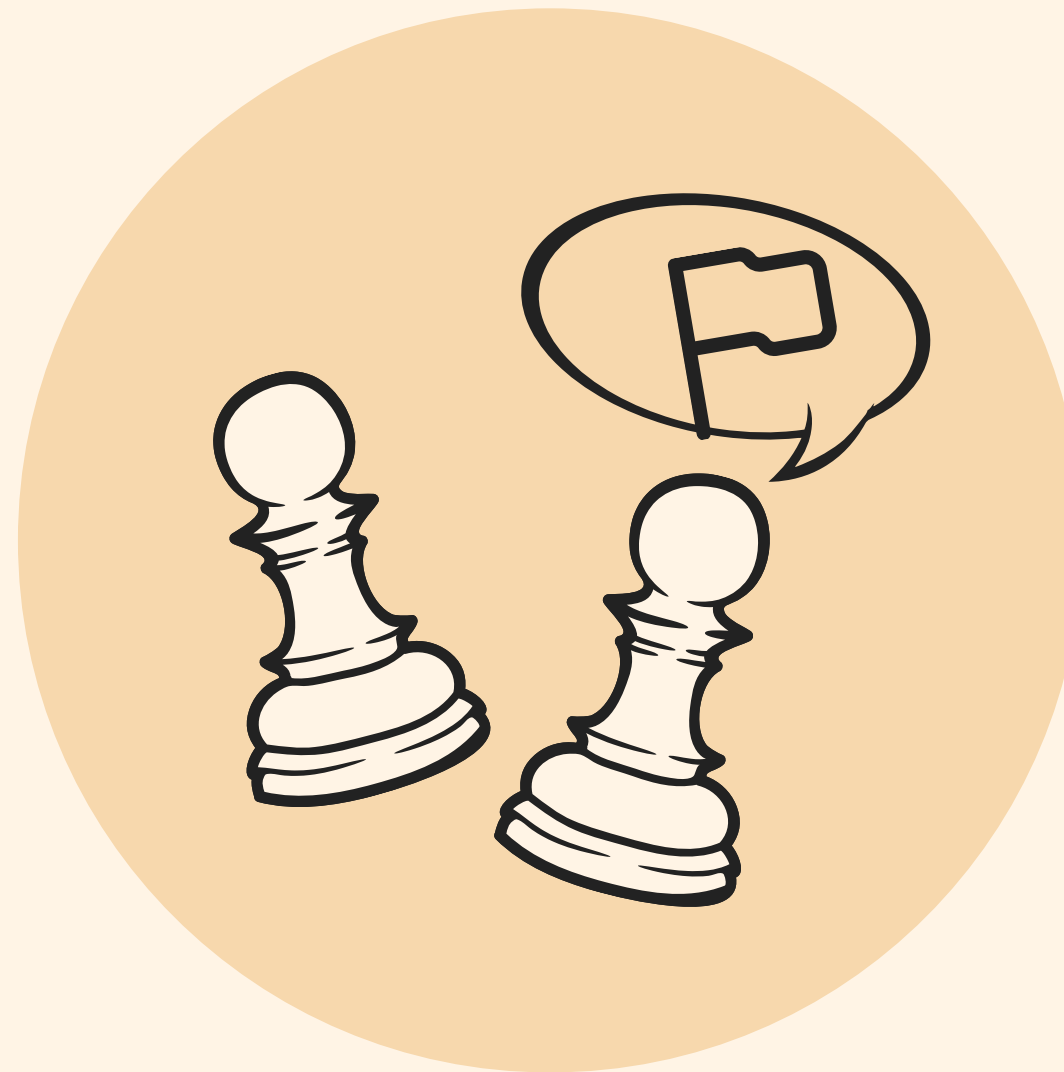


# FUTURE IMPROVEMENTS

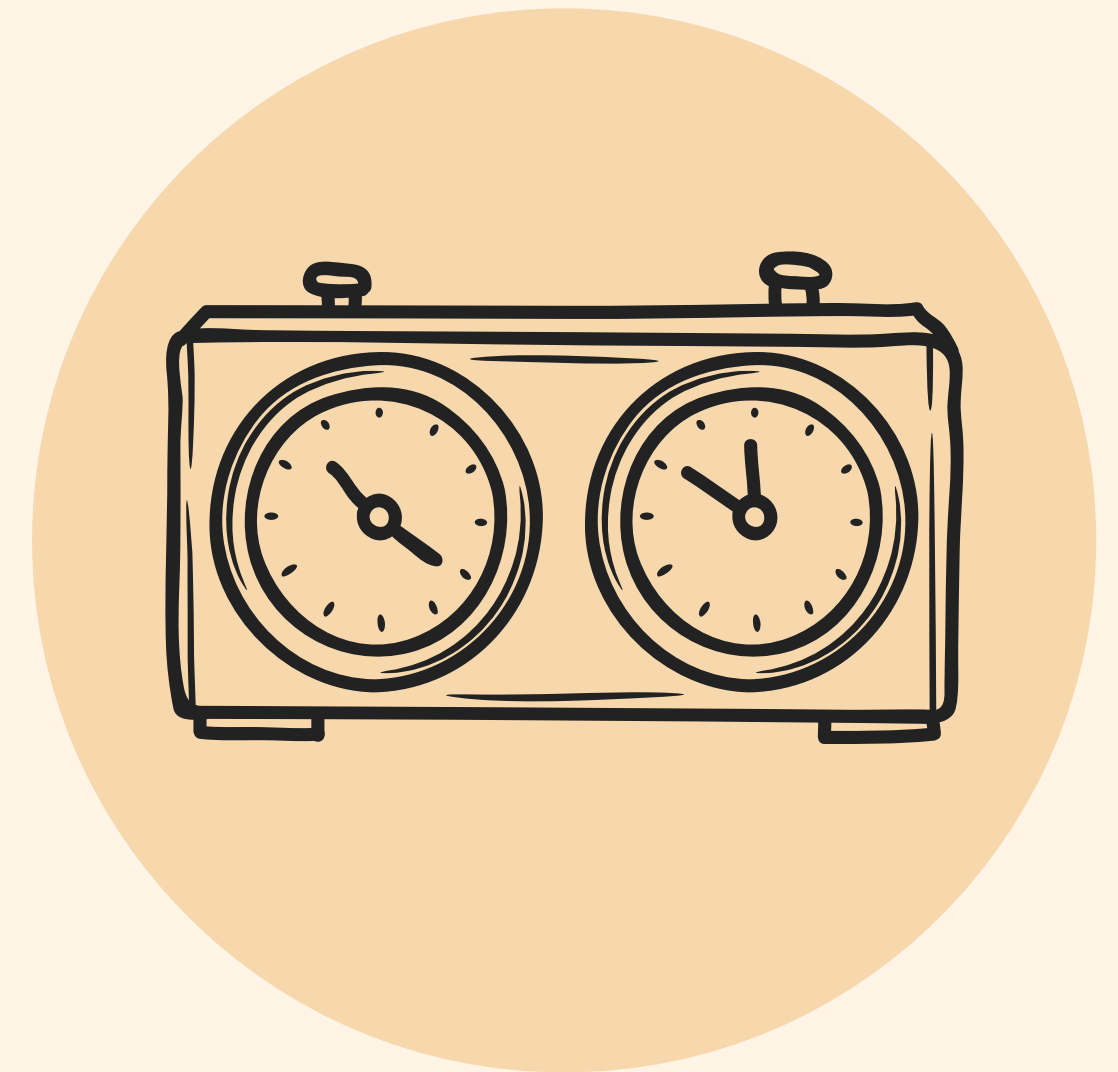
**WIN CONDITIONS**



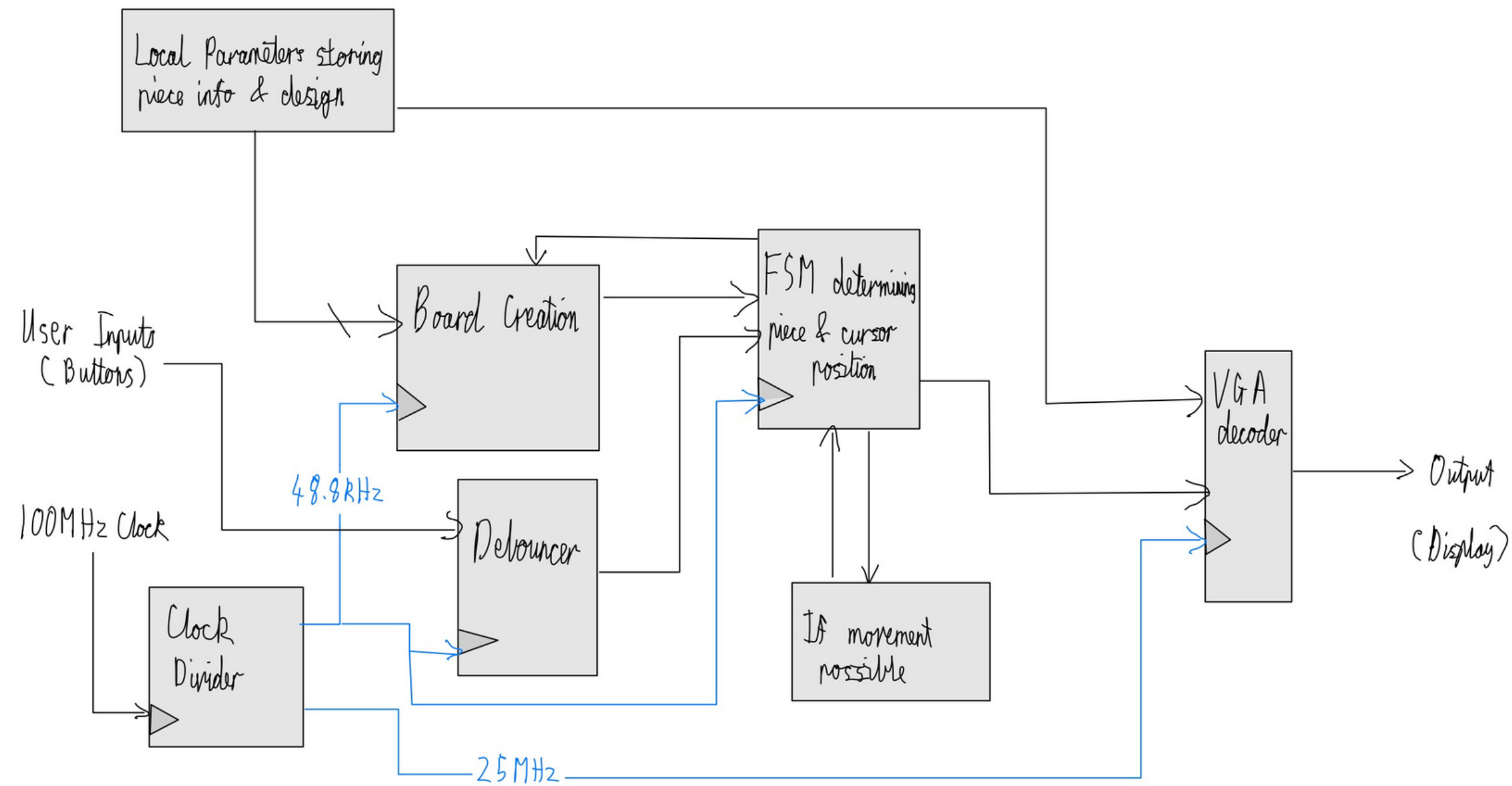
**EN PASSANT**



**TIMING OUT**



# BLOCK DIAGRAM



```

module top(
    input boardCLK,
    output [3:0] redVGA,
    output [3:0] greenVGA,
    output [3:0] blueVGA,
    input BTNC, BTNU, BTND, BTNR, BTNL,
    input resetSwitch,
    output horizontalVGA,
    output verticalVGA
);

//Reset
wire reset;
assign reset = resetSwitch;

//Clock Management
wire vgaCLK, gameCLK;
clock_divider divide(.boardCLK(boardCLK), .vgaCLK(vgaCLK), .gameCLK(gameCLK), .reset(reset));

//Button Management
wire cleanBTNC, cleanBTNU, cleanBTND, cleanBTNR, cleanBTNL;
button_debouncer debounceUp(.clk(gameCLK), .reset(reset), .BTN(BTNU), .clean(cleanBTNU));
button_debouncer debounceDown(.clk(gameCLK), .reset(reset), .BTN(BTND), .clean(cleanBTND));
button_debouncer debounceRight(.clk(gameCLK), .reset(reset), .BTN(BTNR), .clean(cleanBTNR));
button_debouncer debounceLeft(.clk(gameCLK), .reset(reset), .BTN(BTNL), .clean(cleanBTNL));
button_debouncer debounceCenter(.clk(gameCLK), .reset(reset), .BTN(BTNC), .clean(cleanBTNC));

//Game Logic
wire [255:0] board;
wire [12:0] moveData;

board create(
    .reset(reset),
    .clk(gameCLK),
    .BTNC(cleanBTNC), .BTNU(cleanBTNU), .BTND(cleanBTND), .BTNR(cleanBTNR), .BTNL(cleanBTNL),
    .board(board), .moveData(moveData)
);

//Display on the VGA
vga_paint paint(
    .board(board),
    .moveData(moveData),
    .clk(boardCLK),
    .reset(reset),
    .greenVGA(greenVGA), .blueVGA(blueVGA), .redVGA(redVGA), .horizontalVGA(horizontalVGA), .verticalVGA(verticalVGA)
);

endmodule

```

[illegible]

# ENCODINGS



```
module board(  
    input reset,  
    input clk, //25 MHz  
    input BTNC, BTNU, BTND, BTNR, BTNL,  
    output wire [255:0] board,  
    output wire [12:0] moveData  
);  
  
//Define local parameters  
//wire allowMove = 1; //TESTING wire allowMove;  
wire [10:0] changePiece;  
wire [2:0] currentState;  
  
//Run the State Machine  
user_state play(.clk(clk), .reset(reset), .allowMove(allowMove), .entireBoard(board),  
    .BTNC(BTNC), .BTNU(BTNU), .BTND(BTND), .BTNR(BTNR), .BTNL(BTNL),  
    .changePiece(changePiece), .moveData(moveData), .currentState(currentState));  
  
//Check if move is legal  
allow_move check(.allowMove(allowMove), .moveData(moveData)); //TESTING  
  
//Generate the board/Move the pieces  
build_board build(.clk(clk), .boardPass(board), .changePiece(changePiece), .currentState(currentState)); //TESTING try no boardPass  
  
endmodule
```

# GAMELOGIC

Lines of Code per File  
◀1560 total▶

top.v	clock_divider.v	button_debouncer.v	checkAllow.v	checkDistance.v	vga_paint.v
70	46	91	55	413	392
board.v	build_board.v	user_state.v	checkPath.v	checkSquare.v	vga_generator.v
47	139	141	55	41	70

THANK YOU FOR LISTENING

**LET'S PLAY CHESS!**

**[HTTPS://GITHUB.COM/LEOMATTOSMARTINS/EC311\\_PROJECT.GIT](https://github.com/leomattosmartins/ec311_project.git)**

