Documentation Calculator Project

Pi-Sense

16.06.2023

1 Team members

1. Arsany Girgis
2. Emirkan Sali
3. Lochana Abhayawardana

2 Introduction

Digital system design encompasses a wide range of applications, from simple control circuits to complex computing systems. The ability to efficiently design and implement such systems is crucial in various domains. FPGAs are programmable integrated circuits that offer significant advantages in the design and development of digital systems. Unlike fixed-function ASICs (Application-Specific Integrated Circuits), FPGAs can be reconfigured to implement custom logic circuits. This flexibility allows for rapid prototyping, iterative refinement, and customization of designs based on specific application requirements. VHDL enables efficient design representation, simulation, synthesis, and verification. VHDL serves as an essential tool for capturing the intended functionality of digital systems, facilitating their implementation on FPGAs.

The 4-bit calculator project serves as a concrete example to demonstrate the significance of FPGAs and VHDL in digital system design. FPGAs offer a highly configurable platform that can accommodate the calculator's logic elements, arithmetic circuits, and interconnections. The reconfigurable nature of FPGAs allows for iterative development, testing, and refinement of the calculator design. Additionally, VHDL provides a robust and expressive language to describe the calculator's behavior and structure. By leveraging VHDL, we can efficiently capture the complex arithmetic operations, control logic, and interface requirements of the calculator, facilitating its implementation on the FPGA.

The objective of our project is to design and implement a 4-bit calculator using FPGAs and VHDL. The calculator supports basic arithmetic operations, including addition, subtraction, multiplication, and division. We aim to explore the capabilities of FPGAs and the power of VHDL in efficiently representing the calculator's functionality. Furthermore, we intend to demonstrate the practical application of FPGAs and VHDL in the digital system design domain, showcasing their importance in implementing complex arithmetic computations.

3 Concept description

The concept of a 4-bit calculator on an FPGA (Field-Programmable Gate Array) involves designing and implementing a digital circuit that performs basic arithmetic operations on 4-

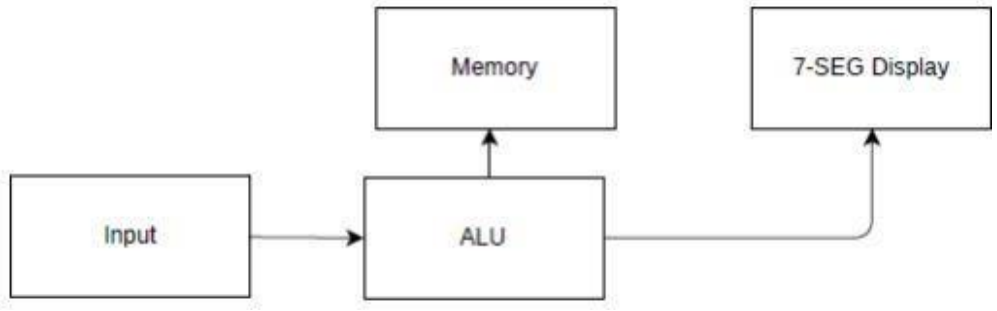


Figure 1: Concept block diagram of our calculator

bit binary numbers. The FPGA serves as a programmable hardware platform that allows us to create and configure custom digital circuits to perform specific functions.

- Designing the Calculator:

The calculator design starts with defining the input and output requirements. In this case, the inputs are two 4-bit binary numbers (a and b), an operation select signal (op), and the outputs are the result (result), carry-out (carry_out), and overflow (overflow) signals. The desired operations (addition, subtraction, multiplication, and division) are implemented using appropriate arithmetic and logical circuits within the FPGA design.

- VHDL Implementation:

The calculator's functionality is described using a hardware description language like VHDL or Verilog.
The operations are typically implemented using standard arithmetic and logical operators provided by the language, such as addition (+), subtraction (-), multiplication (*), division (/), and remainder (rem).
Additional logic is implemented to handle special cases, such as division by zero.

- Synthesis and Place & Route:

Once the VHDL/Verilog code is written, it is synthesized using a synthesis tool that converts the code into a gate-level representation. The synthesized design is then subjected to a place and route (P&R) process, where the FPGA resources (logic elements, flip-flops, interconnects, etc.) are allocated and connected to implement the desired functionality.

4 Project/Team management

We used agile as a project method where we had weekly sprints. All members participated in all tasks, but for some tasks specific members focused on it more:

Lochana: Lochana enabled the seven segment display in the FPGA and implemented the clock divider. He also helped with designing the schematic, especially the schematic for the Keypad. Finally, he wrote the explanation of the VHDL files and the section about schematics in this documentation.
Pi-Sense

Arsany: Arsany mainly worked on the VHDL code for realizing the basic function of the calculator. Later he made the main VHDL file containing all the elements of the system to port it to the FPGA. He wrote the introduction and the concept of the documentation.

Emirkan: Emirkan helped with designing and assembling the schematic and has the main task of designing the final PCB. He wrote the technologies section and the VHDL and FPGA implementation section.

5 Technologies

- VHDL Hardware Description Language: A low level language used to develop and synthesize analog and digital circuits especially prevalent in Europe [1]. Using VHDL one can create simple to complex logic circuits by describing the hardware in bits and gates in combinational logic and if and case sentences in sequential logic. Furthermore, One can test the developed circuit using a testbench and synthesize it as well.
- FPGA: FPGA's (Field programmable gate array) are programmable digital building blocks used for testing prototype circuits without having to manufacture a PCB [2]. They can include several modules and sensors, but one can create almost every component on the FPGA itself.
- KiCAD: KiCAD is an open source PCB layout program used to create schematics of a circuit or a system and translate the schematics into netlists, which are then used to create an PCB design.
- Xilinx Vivado: Vivado is used to program the FPGA board and upload the VHDL code on it to enable the building blocks.

6 VHDL and FPGA Implementation

Our digital design of the calculator was implemented using VHDL in the following way:

- A calculator was coded and tested in VHDL, which takes A and B as inputs and the operation and outputs the result. A testbench can be seen in figure 2

Figure 2: Testbench of the calculator

- A decoder for the seven segment display is implemented for displaying the right number and ultimately converting the inputted binary value into a decimal digit.
- A decoder for the Keypad was created to determine button presses and receiving the corresponding code as an input
- The built in seven segment displays are used by defining the constraint file
- User Constraint File is used to enable the clock, LEDs, Pmod header(JA) and the seven-

segment display in the vivado IDE.

- Finally, every component is put into one main VHDL file which uses the decoders etc. as components to create the full circuit and functionality.

An updated block diagram is visualized in figure 3

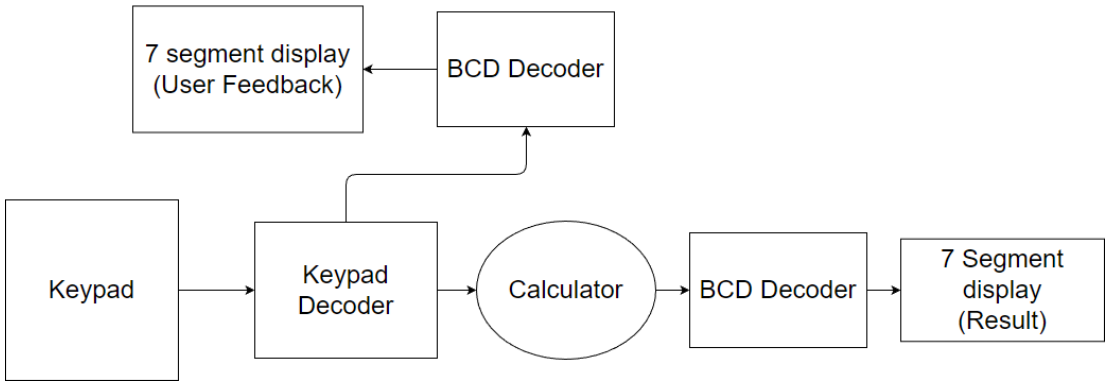


Figure 3: Updated block diagram

Here are the explanations of the individual VHDL files we have designed during our project.

- calculator.vhd
  This code is defining the entity of the calculator with the inputs, reset, a clock, two 4-bit operands which are a and b and the two bit operator (op). Output is 4-bit result named resultOut and the error flag is implemented (errorOut). In the behavioral section it contains the functions which handles the functionality of the calculator. The clock is checking the rising edge of the clock signal to trigger the reset function. Throughout the code switch case statements are used to implement the arithmetic operations and generate appropriate results. If there is an error or division by zero detected , the error-code will display.
- Decoder.vhd
  The decoder defines 4*4 matrix keypad inputs to 4-bit output. The rising edge of the clock signal is used to control the column select lines (col). Within each time interval the process check for the inputs based on rows and columns and sets the corresponding output value. . The code specifies a time-division multiplexing strategy for scanning the keypad and decoding the pushed key to 4-bit output.
- DisplayController.vhd
  This defines the display controller module to control the seven segment display by taking the 4-bit inputs (DispVal). Rightmost digit was selected to display the values. Use case statement is used to map the input values (DispVal) to display appropriate 7-segment display values.
- decoderFSM.vhd
  This code provides a finite state machine for allocating the three keys based on input in order to decode it. There are inputs for the clock(clk), reset and decodeOut. Furthermore it has outputs for key assignments such as Key1Assigned and assigned key values such as Key1Value. The processors inside the behavioral are triggering by the

rising edge of the clock. When the inputs form DecodeOut is changing, the process check if any of the keys are not yet assigned and assigns the values accordingly. Assigned key values, assignment flags are update accordingly. Then the updated values assigned to output ports accordingly.

- PmodKYPD.vhd
  This code includes the components declarations including the decoderFSM module, Decoder module, DisplayController module and the calculator module. The decoder module decodes the keypad input, decoderFSm assigns specific meanings to the decoded keys. The calculator module performs the arithmetic operations based on assigned keys. The displayController module displays result on seven segment display.

  This code connects the input and output signals of all the above-mentioned segments, allowing them to interact and coordinate to form a complete system.

- Testbenches
  During our project we wrote some techbenches such as calculator_tb, PmodKYPD_tb,decoder_tb,decoderFSM. Each testbench instantiates the associated module and applies stimulus signals to replicate the functioning of the respective modules and monitor the output responses.

7 PCB Design

Kicad 7.0 Schematic editor is used For designing the schematic of the calculator. The design mainly has five categories.

7.1 Power Schematic

- Voltage regulator which is able to give a voltage input from 4.5 volts to 18 volts and the current input varies from 1.5 amperes to 2.5 amperes. It is a triple synchronous and a step-down converter. The schematic is visualized in figure 4

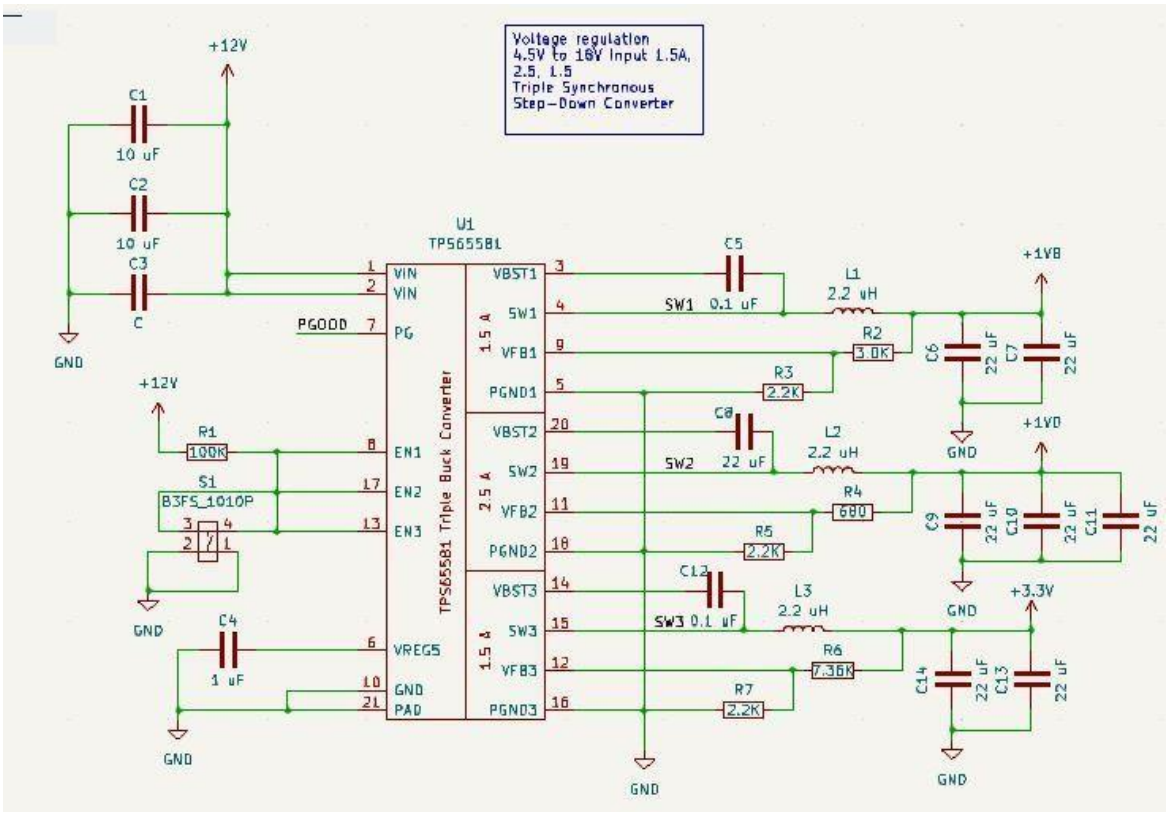


Figure 4: Schematic of the voltage regulator

- A simple switch to turn on and off the power supply “J1 Barrel_Jack_Switch” is used visualized in figure 6.

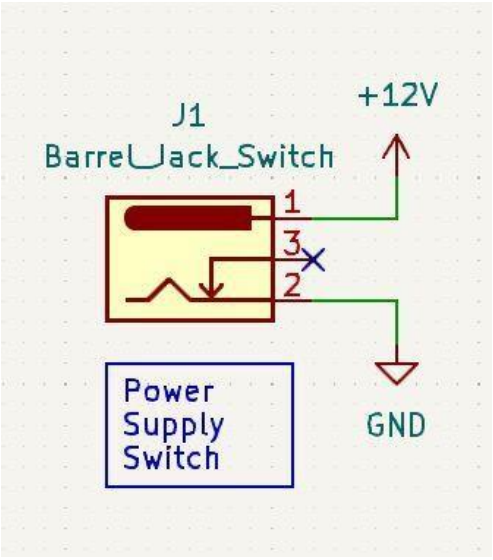
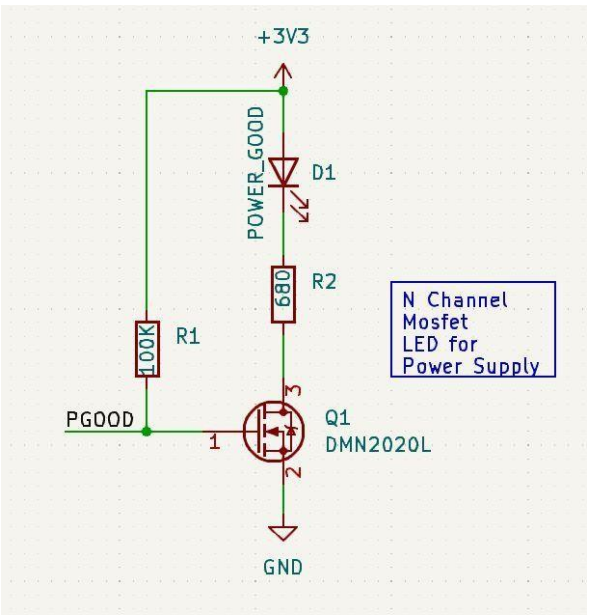


Figure 6: Barrel switch to turn on and off the power supply



- LED: N channel MOSFET LED for Power supply is used. It is included with a Q1 DMN2020L N channel MOSFET, an LED and 100k and 680 ohm resistors visualized in figure 5.

7.2 Keypad Schematic

This is an external input device which has been used in the project. It is a 4x4 matrix keypad with 4 columns and rows and a total of 16 push buttons. It is included with the global labels such as ROW 1 to 4 and Col 1 to 4 in order to connect the schematic with the other hierarchical schematics. To create this keypad on an PCB, we simulated it by creating a 4x4 matrix of pushbuttons instead of using the entire provided keypad. This is visualized in figure 7.

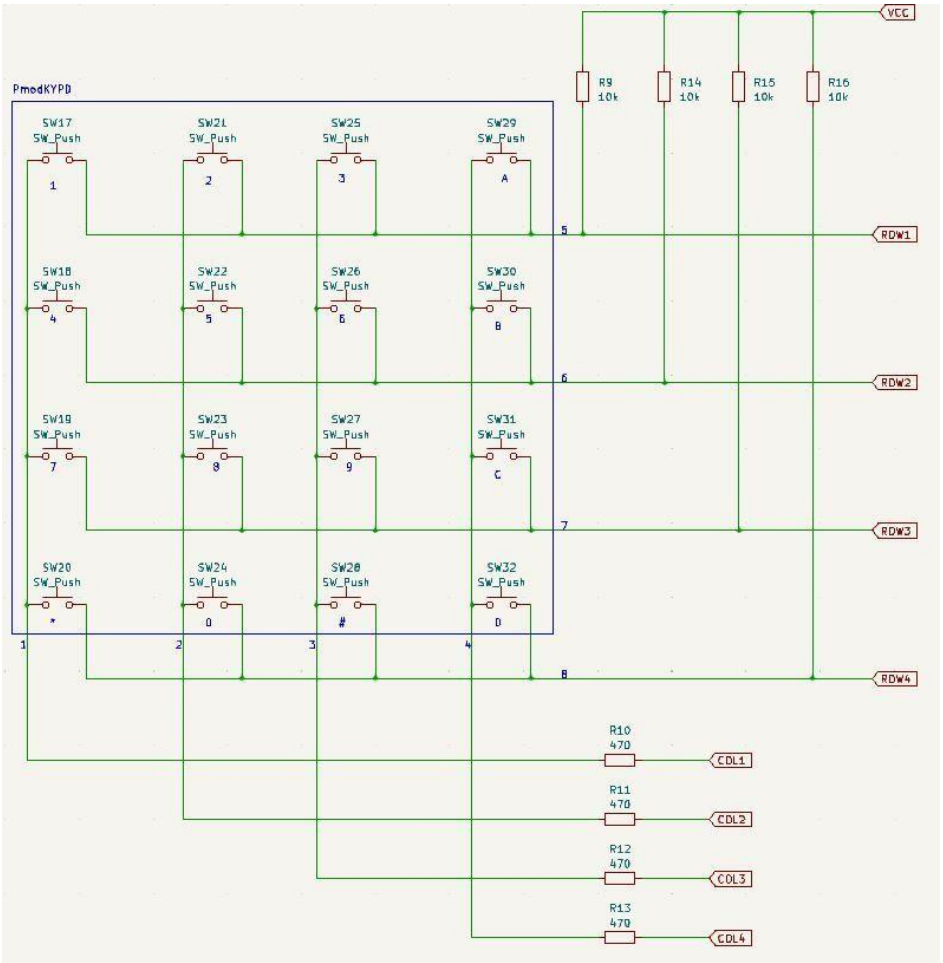


Figure 7: Schematic of a 4x4 keypad used as input for our calculator

7.3 FPGA Schematic

- Ground: The schematic to represent grounded pins seen in figure 9.
- Power Input: The schematic to represent Power inputs seen in figure 10.
- Input Outputs: The schematic to represent Input and Outputs seen in figure 8.

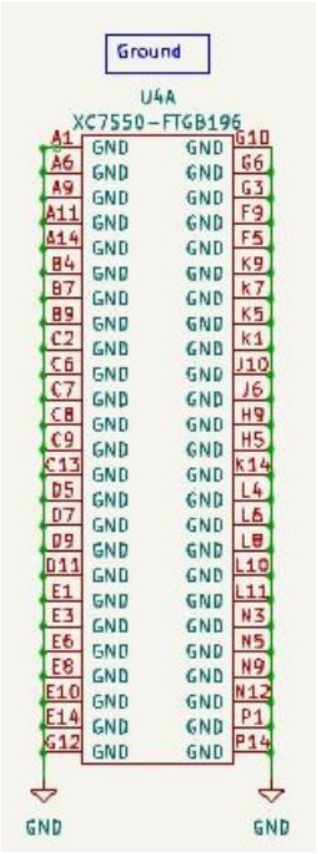


Figure 9: Ground pins components from the FPGA

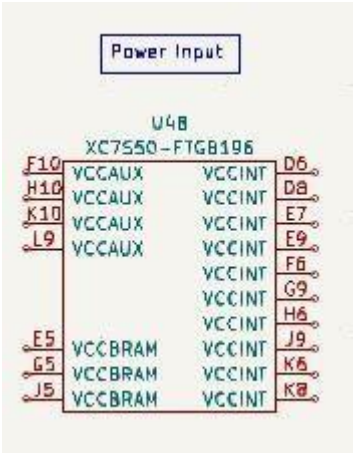


Figure 10: Input Voltage of the FPGA part of the board

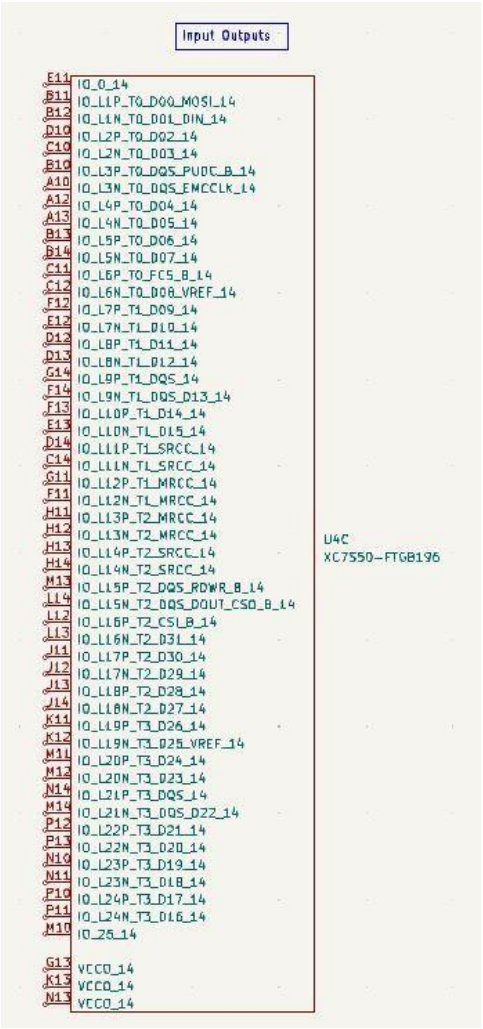


Figure 8: Input output ports of the FPGA board

7.4 Seven Segment display Schematic:

- Two seven segment display “U2 CC 56-12CGKWA” : One for feedback and one for Displaying the result. Shown in figure 11.

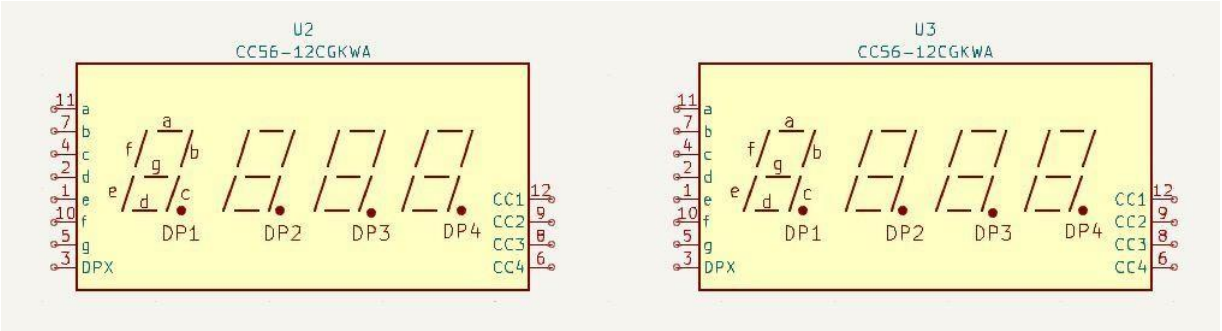
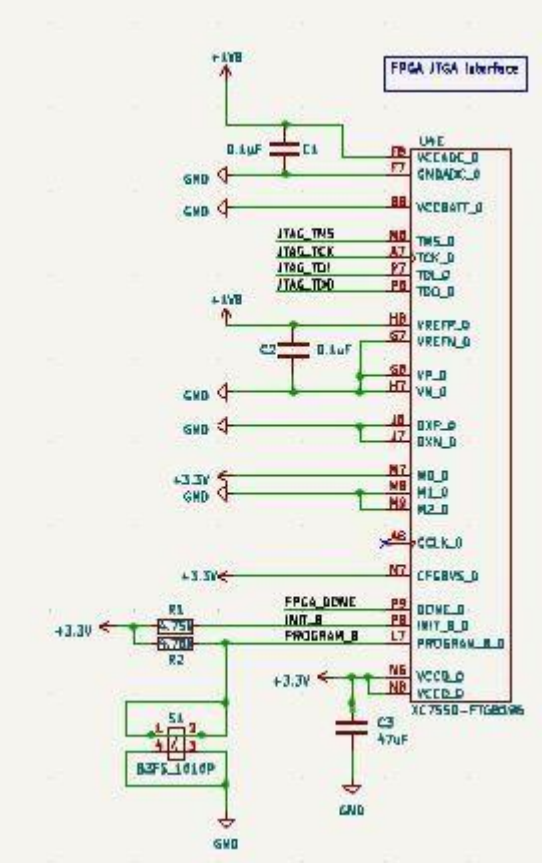


Figure 11: The two seven segment displays used in our schematic

7.5 FPGA JTAG Interface Schematic

- The FPGA JTAG interface is visualized in figure 15
- Pin Header for Xilinx JTAG programmer visualized in figure 14



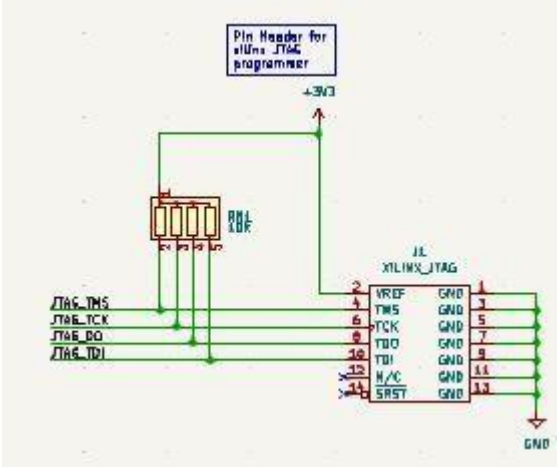


Figure 14: Pin header for Xilinx JTAG Programmer

- Clock: Oscillator 2KHz to 100MHz seen in figure 13

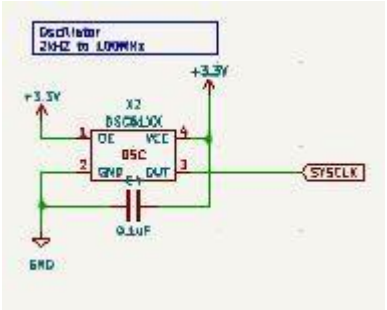


Figure 13: Oscilator clock of our system

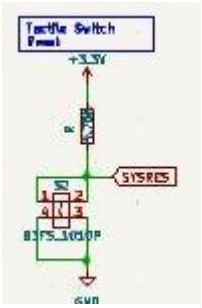


Figure 12: Reset Switch

- Tactile Switch: Used for the reset switch seen in figure 12

7.6 PCB Design

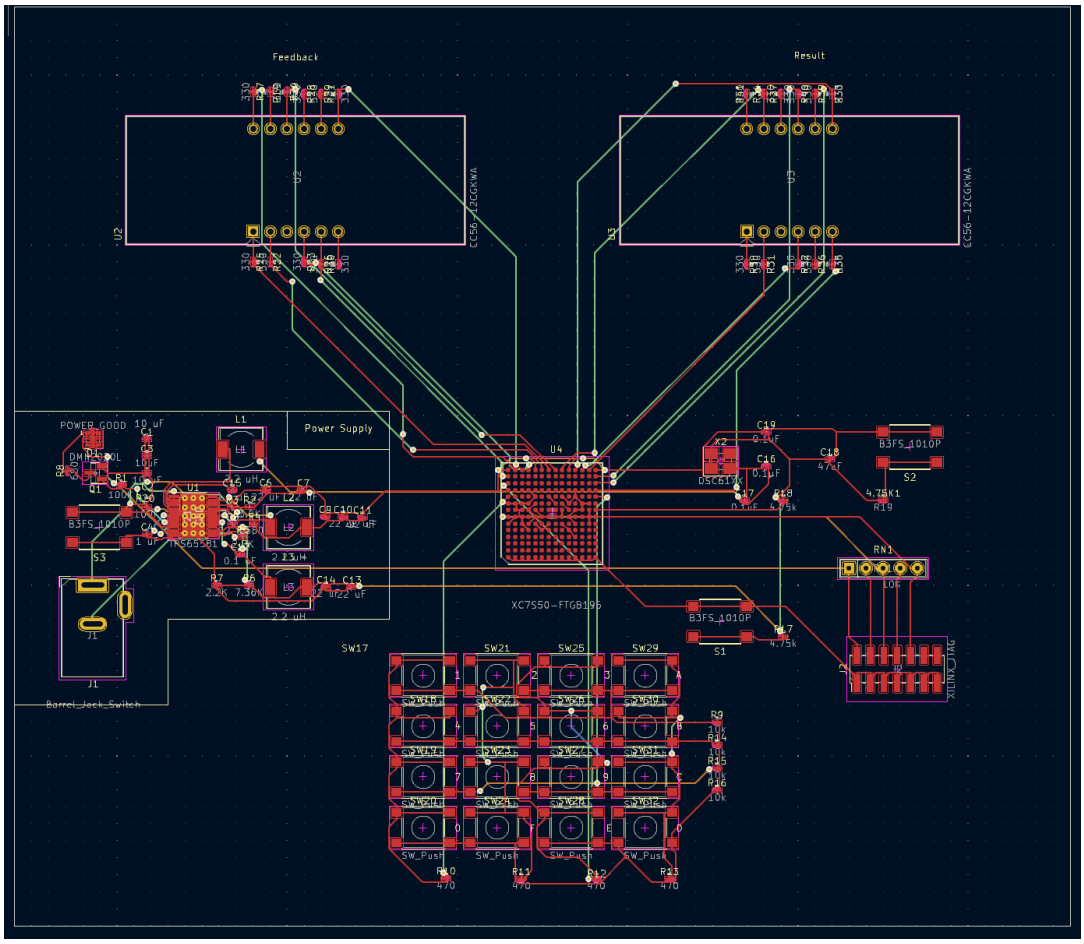


Figure 15: PCB Design in KiCad with routed tracks

For the PCB design, the first step was to finish the net assignment of the schematics and assign footprints to each component. To save space, we used SMD components when it was possible. Every component's footprint was transferred to the PCB design tool. From there on, the first step was to make the edge cut of the PCB. We settled for a rectangular form of the PCB. For the placement of the components, we separated them based on their function and the connections between the components. It was also important to make it understandable for a user when he looks at the PCB instead of making it an incomprehensible mess. The seven-segment displays were placed on the upper side of the PCB with enough space between them and the edges to allow for the needed tracks. The keypad was placed in the middle of the bottom of the PCB. The FPGA component is in the middle of the PCB. All the components concerning the power supply of the device were placed on the left side of the PCB, and the JTAG environment was on the right side of the PCB. The placement of the components is visualized in Figures 15 and 16.

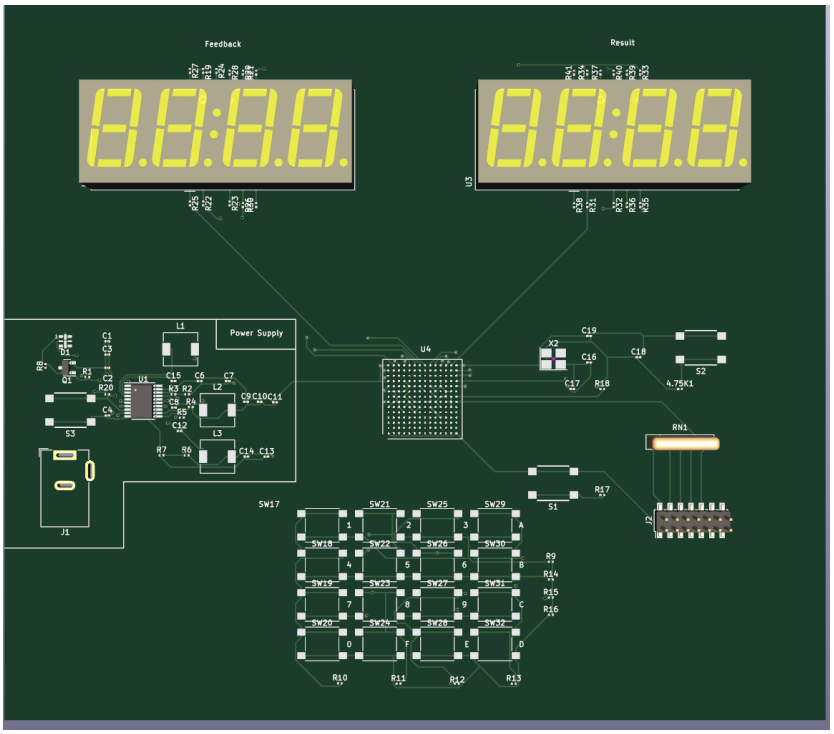


Figure 16: PCB Design in the 3D viewer

8 Sources/References

[https://www.mikrocontroller.net/articles/VHDL [1]](https://www.mikrocontroller.net/articles/VHDL)

[https://www.mikrocontroller.net/articles/FPGA [2]](https://www.mikrocontroller.net/articles/FPGA)

https://www.mikrocontroller.net/articles/KiCad *[3]*