Documentation
VendTech
28/06/2024

1 Team members

Masrur Jamil Prochchhod
Mehedi Hasan
Pisula Guruge

2 Introduction

The main focus of our project was to design and implement a vending machine using FPGA board. By integrating an FPGA for the vending machine mechanism, our project leverages the high-speed processing and flexibility of FPGAs to manage real-time data from various sensors and control the vending operations efficiently. Using the FPGA we integrate it to handle tasks such as user input processing, product dispensing, inventory management, transaction and communication with IoT and WSN components, making the vending machine smarter and more reliable.

3 Concept description

User Interface **(Switches)**
FPGA **Circuits**
Transaction

Main Application: The primary application of our prototype is to create a vending machine that efficiently dispenses cold drinks, coffee and candy. This vending machine leverages FPGA technology for real-time processing and control, ensuring reliable operation and enhanced performance.

Devices and Components:

1. **FPGA (Field-Programmable Gate Array)**:
   - Central processing unit for managing the vending machine operations.
2. **User Interface**:
   - **Push Buttons**: For user input to select products.
   - **7-Segment Display**: Displays information such as product selection and status.
3. **Inventory Management:** The inventory management system in our vending machine uses sensors to track the stock levels of each product and updates this information in real-time via the FPGA.
4. **Transaction:** The transaction process in our vending machine involves the user selecting a product via the push buttons, which the FPGA processes to dispense the chosen item. The transaction system calculates the amount due, deducts the payment and provide appropriate changes.
5. **Power Supply**:
   - **Voltage Regulator (LM7805)**: Provides a stable 5V supply to the FPGA and other components.
6. **Decoupling Capacitors**: Ensure stable voltage supply to the FPGA and minimize noise.

4 Project/Team management

Group Member	Task 01	Task 02	Task 03	Task 04	Task 05
Masrur Jamil Prochchhod	Project Idea	Implementation	VHDL Coding	FPGA Programming	PCB Design
Mehedi Hasan	Project Idea	Implementation	VHDL Coding	FPGA Programming	PCB Design
Pisula Guruge	Project Idea	Implementation	VHDL Coding	FPGA Programming	PCB Design

5 Technologies

- Kicad
- VHDL
- FPGA

6 Implementation

VHDL and FPGA

- **Entity Declaration**: We defined the main entity for the vending machine, specifying inputs (clock, reset, product selection buttons) and outputs (LED indicators, seven-segment display, dispensing actuators). This step establishes the interface of our design.
- **Architecture Definition**: Within the architecture, we described the internal workings of the vending machine. We used behavioral VHDL to define how the vending machine transitions between states such as idle, product selection, waiting for coins, and dispensing.
- **State Machine Implementation**: We developed finite state machines (FSMs) to manage the various operational states. Each state handles specific tasks like displaying options, accepting coins, and activating the dispensing mechanism.
- **Component Instantiation**: We integrated sub-modules within the main architecture for specific tasks such as controlling the display and driving the motors. This modular approach simplifies design and testing.
- **RTL Simulation**: We performed Register Transfer Level (RTL) simulation to verify the functionality of the VHDL design at the register level. This involved creating testbenches to simulate the operation of the design and check if it behaves as expected.
- **Synthesis Constraints**: We defined synthesis constraints to ensure that the design meets the required performance metrics, such as timing constraints. These constraints guide the synthesis tool to optimize the design appropriately.

- **Synthesis**: We used synthesis tools (such as Xilinx Vivado) to convert the VHDL code into a netlist that maps the design onto the FPGA's hardware resources. Synthesis translates high-level VHDL descriptions into gate-level logic. This step transforms our abstract VHDL code into a concrete implementation that the FPGA can use.
- **Place and Route**: After synthesis, we performed place and route to fit the netlist onto the FPGA. This step involves arranging the synthesized logic elements in the physical FPGA fabric and routing the interconnections between them. The goal is to optimize the placement and routing to meet timing requirements, ensuring that signals propagate through the FPGA within the necessary time constraints.
- **Bitstream Generation**: Once place and route were completed, we generated a bitstream file from the placed and routed design. This file contains the configuration data needed to program the FPGA. The bitstream encodes the entire configuration of the FPGA, including the arrangement of logic elements and routing of signals.
- **Programming the FPGA**: Finally, we used an FPGA programmer to load the bitstream onto the FPGA (e.g., Nexys XC7A100TCSG324A). This step configures the FPGA to operate as per the VHDL design. We connected the FPGA board to our computer via a USB cable and used the programming software to transfer the bitstream file to the FPGA. Once programmed, the FPGA was ready to control the vending machine as designed.

- **Transaction Process:** The transaction process begins when the user selects a product using the push buttons. The FPGA processes this input and verifies the selection, displaying the selected product on the seven-segment display. The system then waits for the user to insert the required amount of money, tracked by input sensors. Once the payment is confirmed, the FPGA activates the appropriate actuators to dispense the selected product, completing the transaction and updating the display to reflect the completed purchase.

- **FPGA Logic Circuits:** The FPGA logic circuits include a combination of control logic, data paths, and timing control elements. Control logic is implemented using finite state machines (FSMs) that manage the different operational states of the vending machine, such as idle, selection, payment, and dispensing. Data paths handle the routing of signals between various modules, ensuring correct processing and output of data. Timing control ensures all operations are synchronized with the system clock, meeting the necessary timing requirements for reliable performance.

- **FPGA Platform:** We utilized the Nexys XC7A100TCSG324A FPGA board for this project, which provides the necessary input/output (I/O) capabilities, processing power, and flexibility to implement the vending machine control system. Using Xilinx Vivado, we performed synthesis, place and route, and bitstream generation. This process allowed us to convert our VHDL design into a configuration that could be loaded onto the FPGA, enabling it to perform the specified control tasks.

- **VHDL Implementation of User Interface:** The VHDL implementation of the user interface involves creating modules for interfacing with buttons, LEDs, and the seven-segment display. The VHDL code handles signal debouncing to ensure clean input from buttons, drives the LEDs to indicate system status, and generates control signals for the seven-segment display to show product information and transaction status. These modules are integrated within the main architecture, allowing the FPGA to manage user interactions effectively and provide clear feedback through the display and LEDs.

- **User Interface:** The user interface consists of push buttons for product selection, LEDs for status indicators, and a seven-segment display to show product information and transaction status. The buttons allow users to choose products, and the FPGA processes these inputs to display the selection and await payment. LEDs indicate various statuses, such as product availability and transaction progress. The seven-segment display provides real-time feedback on the selected product and transaction status, ensuring a seamless and user-friendly experience.

Kicad (PCB)

- **Component Selection**: We carefully selected all necessary components for the vending machine, including the FPGA, voltage regulator, capacitors, LEDs, switches, and connectors. Each component was chosen based on its electrical characteristics, compatibility, and ability to meet the design requirements.
- **Schematic Entry**: Using KiCad's schematic editor, we created the schematic by placing and connecting all components according to the design. This involved drawing connections between components using wires to create a logical representation of the circuit. We ensured that each connection was correctly made to reflect the actual circuit design, allowing for accurate simulation and layout in subsequent steps.
- **Annotation and ERC**: We annotated the schematic to assign unique identifiers to each component. This step ensures that each component can be individually identified and referenced throughout the design process. We then ran an Electrical Rules Check (ERC) to verify that there were no errors or missing connections in the schematic. The ERC checks for issues such as unconnected pins, missing connections, and incorrect electrical properties, ensuring that the schematic is error-free before proceeding to the layout stage.

Footprint Assignment

- **Assign Footprints**: We assigned appropriate physical footprints to all components in the schematic. This step ensures that each component will have the correct size and shape on the PCB layout. Footprints represent the physical layout of the component pins and pads on the PCB, and assigning the correct footprints is crucial for accurate component placement and soldering. We used KiCad's extensive library of component footprints, selecting those that matched our components' specifications.

PCB Layout

- **Import Netlist**: We imported the netlist generated from the schematic into the PCB layout tool. The netlist contains information about all the connections that need to be made between components, serving as a blueprint for routing the PCB. Importing the netlist ensures that all connections defined in the schematic are transferred accurately to the PCB layout, providing a clear guide for placing and connecting components.
- **Component Placement**: We arranged components on the PCB, optimizing for trace lengths and ease of routing. Critical components like the FPGA were placed centrally to minimize the length of critical signal paths. We grouped related components

together to simplify routing and reduce signal interference. Proper placement of components is crucial for achieving a compact and efficient layout, reducing the likelihood of errors during manufacturing and assembly.

- **Routing**: We routed the connections between components using the PCB layout editor. Starting with critical traces such as power and ground, we used wide traces to handle higher currents and ensure stable power distribution. We then routed signal traces, ensuring that they were as short and direct as possible to minimize signal delay and interference. Routing involves drawing physical paths for electrical connections on the PCB, which must be done carefully to avoid issues such as crosstalk and electromagnetic interference.
- **Design Rule Check (DRC)**: We performed a DRC to ensure the layout met all manufacturing requirements. The DRC checks for issues such as trace width, spacing, and overlap, verifying that the layout adheres to the specified design rules. We fixed any issues identified by the DRC, ensuring that the PCB layout was compliant with manufacturing standards and ready for fabrication. This step is critical for preventing errors that could lead to defects in the final product.

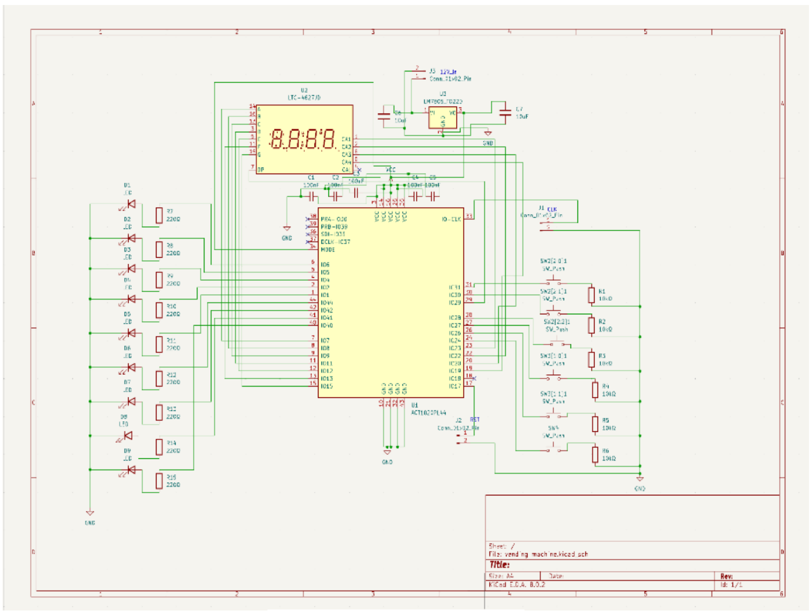


Fig: Schematic

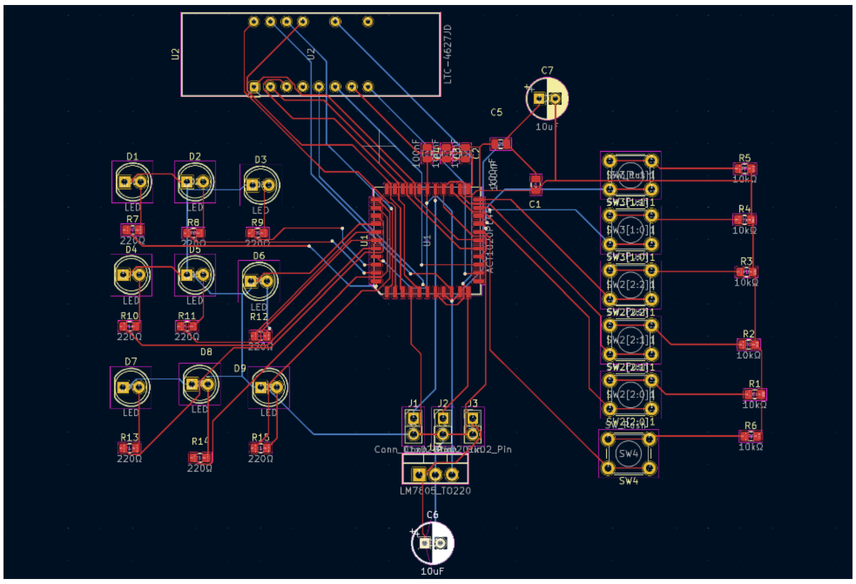
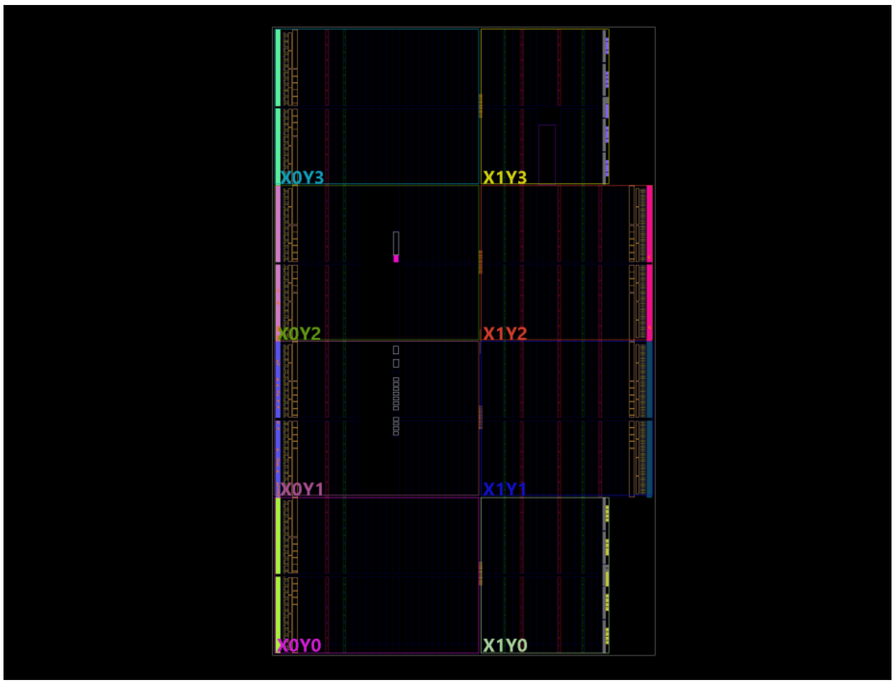


Fig: Board

7. Synthesis Design and Implementation Design

In the synthesis design and implementation phase, our high-level VHDL code is then translated into a gate-level netlist suitable for the FPGA; that was done within the Xilinx Vivado design environment synthesizing a design for optimal logic performance and area. In the image below, the place and route process is shown, where the synthesized netlist is mapped onto the FPGA's layout physically. This layout shows how the circuit logic resources are spread over the FPGA, with a variety of placements in regions such as X0Y3 and X1Y2.

Timing analysis ensured that all the signal paths had met the required timing constraints so that the FPGA could reliably work at a desired clock speed. After checking its timing, a bitstream file representing the configuration data for programming an FPGA was generated. The final step was to configure our FPGA for dispensing, as specified by our VHDL design in the vending machine, using an FPGA programmer and loading this bitstream. Artistically, it expresses a careful attention to resource usage with a view to optimizing performance and minimizing delays on critical paths.



8. Sources/References

- KiCad EDA. (n.d.). Learning resources. KiCad EDA. https://www.kicad.org/help/learning-resources/
- Surf-VHDL. (2021, December 27). Vivado project tutorial - surf-VHDL. https://surf-vhdl.com/vivado-project-tutorial/
- FPGA Learning https://www.xilinx.com/products/silicon-devices/resources/programming-an-fpga-an-introduction-to-how-it-works.html
- Kicad sources https://learn.sparkfun.com/tutorials/beginners-guide-to-kicad