UNIVERSITY *of* WASHINGTON | BOTHELL

ARM CPU on FPGA

Project By: Sam Ma, Angelica Quinto, and Forrest Zhang
Advisor: Joe Decuir
Sponsor: Arnie Berger

Table of Contents

Introduction

Abstract

The primary objective of this project is to build a new lab kit - a FPGA based ARM CPU platform for the Microprocessor System Design course at the University of Washington Bothell, so the Electrical Engineering students could learn the modern ARM architecture.

The project will incorporate FPGB, logic analyzer, custom PCB board and Quartus Prime.

Background and Motivation

Currently in the electrical engineering department there is no available streamlined method to effectively teach students about the inner workings of ARM architecture. This poses a potential lost opportunity for helping students getting familiar with the modern technological CPU architecture landscape of today. With the University of Washington Bothell lack of effective teaching tools in ARM architecture, it would greatly benefit to have the resources to prepare students for a potential future in a career with modern technology architecture.

By understanding the motivation behind the goal of modernizing the academic course, this project hopes to offer the tools to better prepare students with practical knowledge of ARM architecture. The primary goal of this project is to offer students an in-depth, hands-on understanding of an ARM CPU's operations and architecture, which is otherwise obscured in contemporary CPUs. In achieving this, the project aims to equip students with the ability to analyze and comprehend an ARM CPU's flow of instructions via the data and address buses, which is vital to understanding modern CPU design and operation.

Beyond providing a practical tool for learning, the project also serves to strengthen students' skills in several technical areas such as digital design, FPGA programming, computer architecture, and the practical use of logic analyzers. It will contribute to the development of their core and advanced technical competencies, better preparing them for their future careers in the industry.

Lastly, the project embodies the spirit of continuous improvement in educational methods by integrating practical, hands-on experience with theoretical knowledge. The motivation is not only to enhance the educational experience of current students but also to pave the way for future improvements and adaptations, fostering an environment of continuous learning and innovation.

Overview of Proposed Project

The goal of this project is to complete the implementation of the ARM processor on a Field Programmable Gate Array (FPGA) and using a LogicPort logic analyzer to measure and analyze signals to replace the outdated Z80 and 68008 experiments currently in use in B EE 425. The project involves implementing an ARM processor on a Field Programmable Gate Array (FPGA) and using a LogicPort logic analyzer to measure and analyze signals. This setup will enable students to write programs, observe, and analyze the behavior of a modern CPU, providing hands-on experience in ARM architecture. The success of this project will be achieved by completing the ARM core and external memory interfaces, as well as other necessary components and documentation for future students to effectively learn, analyze, and debug the ARM CPU using minimal resources.

The tasks of this project includes designing and programming the ARM CPU on an FPGA using Quartus Prime software and System Verilog. This includes creating the necessary state machines and control logic to ensure proper functionality. ARM assembly code will also be created to demonstrate the ARM processor's capabilities and interface it with external memories. Additionally, further implementations will be done on components such as memory I/O to interface the ARM CPU with the external switches, buttons, hex displays, and LogicPort logic analyzer for allowing real-time signal analysis. They will conduct rigorous testing to ensure the system's accuracy and reliability. Furthermore, comprehensive lab documentation will be created, including setup guides, example codes, and detailed procedures, to facilitate student learning and experimentation. This documentation will be crucial for future EE 425 students to understand and utilize the ARM-based system effectively.

Deliverables

The deliverables of this project is to implement, test and document the ARM emulation on FPGA, as well as creating the necessary lab materials and documentations for the EE 425 course at the University of Washington Bothell. The deliverables include extending state machines to display DMEM status by decoding specific OpCode bits, modifying the ARM core to extract instructions from EEPROM, demonstrating flash memory by writing the initialized IMEM array to external EEPROM, and creating a memory-mapped I/O system for hardware interfaces. Additionally, the project will develop example ARM assembly programs to demonstrate functionality, including simplified code to control LEDs and more complex code to utilize external interfaces like switches and buttons. Detailed lab documentation will also be provided to guide students in setting up and using the ARM lab kit.

Description of Project

The goal of this project is to implement an ARM processor on an FPGA and use a LogicPort logic analyzer to measure and analyze signals, replacing the outdated Z80 and 68K lab experiments currently in B EE 425. The project involves improving the ARM CPU using Quartus Prime and System Verilog, creating state machines and control logic, and interfacing with external memories. The team will develop ARM assembly code to demonstrate the processor's capabilities and conduct rigorous testing to ensure system accuracy and reliability. Comprehensive lab documentation will be created to guide future students in setting up and using the ARM-based system.

Key deliverables include extending state machines to display DMEM status by decoding specific OpCode bits, modifying the ARM core to extract instructions from EEPROM, and demonstrating flash memory by writing the initialized IMEM array to external EEPROM. The project will also create a memory-mapped I/O system for hardware interfaces, develop example ARM assembly programs to demonstrate functionality, and provide detailed lab documentation. These deliverables will equip students with the tools to effectively learn and experiment with ARM architecture in a lab setting.

In order to complete this project and the technical tasks, the team will start by improving and programming the ARM CPU on an FPGA using Quartus Prime and System Verilog. This includes creating state machines and control logic, and writing ARM assembly code to interface with external memories. They will also implement a memory-mapped I/O system to interface the ARM CPU with external switches, buttons, hex displays, and the LogicPort logic analyzer. Each component will undergo rigorous testing, and comprehensive lab documentation will be created to facilitate student learning and experimentation.

The integration and testing phase involves several steps to ensure the system's functionality and reliability. Each component will be tested individually before integrating them into the overall system. FPGA programming will be verified using Quartus Prime, System Verilog, and KEIL software. PCB functionality will be tested to ensure proper signal transfer, and the LogicPort logic analyzer will be used to capture and analyze ARM CPU signals. The team will also test ARM assembly programs to verify their functionality and compatibility with the FPGA emulation. Once all parts are confirmed to work individually, they will be integrated and tested as a complete system to achieve the anticipated final results.

In summary, the ARM CPU on FPGA Capstone Project will provide future electrical engineering students with a comprehensive learning environment to explore modern ARM CPU architectures. By completing these tasks and deliverables, the project will enhance the curriculum, giving students hands-on experience with ARM CPUs, FPGA programming, and system integration, thus preparing them for future careers.

Software Description

Identification of Software that will be Used

There's multiple software programs that will be used for this project such as Intel Quartus Prime, Keil, LogicPort, and ExpressPCB. All of these software are free to download and use. The main software utilized in this project is Intel Quartus Prime, a programmable logic software that facilitates communication between the FPGA board and the ARM board, specifically using SystemVerilog.

LogicPort is an additional piece of software that illustrates what's happening on the ARM board visually. This will probably be utilized for debugging and verifying our work initially, as well as confirming the work of the prior team. Since this is intended for upcoming B EE 425 students, logicPort will be implemented to visualize the ARM board's operations.

Furthermore, HEX ARM instructions will be generated by the software Kiel tool and placed into the instruction memory. By using switches from the FPGA board, students can input a number and then output a new value based on the command. This allows us to add certain ALU instructions, notably division and more.

Finally, we will utilize the ExpressPCB software to retrieve and confirm the ARM board schematic and PCB design. Once we confirmed that all of our tasks are almost complete we will then order approximately 15 ARM boards. This will likely take approximately 2 weeks to complete due to shipping, soldering, and testing all the boards.

Integration and Testing

Procedure for Testing and Debugging

To ensure quality control for this project, we must test and debug each part. The implementation of ARM CPU encompasses follows the current plan:

1. **Individual component testing:** Each component will be tested individually to ensure each part is in expected working order and shows the ability to be implemented together once each part is successfully working.

2. **FPGA programming verification:** Programming the FPGA is functionability able to work in conjunction with the PCB and ARM CPU architecture. The main method will be using Quartus prime, System Verilog and KEIL software to verify the program successfully compiling and is able to work in relation to the PCB.

3. **PCB functionality verification:** We will test if all programs are able to work with PCB hardware by checking all connections and signal information is properly transferred which can be analyzed with the help of the logic analyzer. The PCB fabrication and full implementation must work as desired.

4. **Logic analyzer signal testing:** For signal capture, we ensure that our ARM CPU’s inputted program will output the expected electronic signal. The method to analyze these digital signals is by using the Intronix Logic Analyzer - Logic Port.

5. **ARM programming verification:** We will have to test if all the inputted ARM program is able to verify all instructions given through the FPGA emulation and its results capable of being read by the logic analyzer.

6. **Connected implementation:** Once we have ensured that all individual parts of the systems are working, we will work with full incorporation of smaller parts for our anticipated final results.

Procedure for Data Reduction and Analysis

To successfully analyze data we will use the logic analyzer to view the digital signals of the CPU. Besides these digital signals we must implement methods to view and debug other behavior of our program. Some methods of debugging might include:

1. **Verification of memory mapping:** By adding functionality to our given I/O ports and switches, we can tie program functions. Creating memory mapped I/O that has a visual display will allow for greater understanding of ARM architecture.
2. **Memory access visibility:** Ensure that we are able to read DMEM and IMEM access as well as tracing memory movement will help expose program behavior.
3. **Signal Analysis**: Detailed signal analysis will be performed using the logic analyzer. This involves capturing waveforms of the control signals, address lines, and data lines to verify the timing and sequence of operations. By analyzing these signals, we can ensure that the ARM CPU is functioning as expected and identify any timing-related issues or logic errors.

Anticipated Final Results

The expected conclusion of this project is being able to load a self programmed ARM programming instruction file that is able to be uploaded or emulated though the FPGA system to successfully program an ARM's functionality. This functionality will be able to be tested by displaying the expected output that will be viewable by the user. Such visualization will be viewable data movement such as IMEM or DMEM access though logicport software and physical display outputs.

- **Functionality Demonstration**: The ARM CPU should be able to perform all specified operations accurately, and this will be verified through the use of test programs. These test programs will be designed to exercise various aspects of the CPU, such as instruction fetch, decode, execution, and memory access.
- **Visual Output**: The functionality will be tested by displaying the expected output through visual indicators. For instance, specific memory-mapped I/O operations will light up LEDs or display

values on hex displays, providing immediate visual feedback on the CPU's operations. This will show that the system is working correctly and that instructions are being processed as intended.

- **Data Movement Visualization**: The movement of data between IMEM and DMEM will be made visible through the logic analyzer and physical display outputs. The logic analyzer will capture and display waveforms that show the address and data lines' activity, allowing us to trace the flow of instructions and data throughout the system. This will help confirm that the ARM CPU is correctly interfacing with external memory and peripherals.
- **Comprehensive Testing**: Extensive testing will be conducted to ensure the system's reliability and accuracy. This will include testing under various conditions and scenarios to ensure that the ARM CPU can handle different types of instructions and workloads. The final results will be documented, including test cases, expected outcomes, and actual outcomes, to provide a clear record of the system's performance.

Bill of Materials (BOM) Proposed

Table 1 shows the materials required for this project, including the cost and quantity of each component.

Table 1: Bill of Materials

Materials	Quantity	Cost
FPGA Board: DE10 - Nano	3	Already Provided
Intronix Logic Analyzer - Logic Port	3	Already Provided
EEPROM Burner	1	Already Provided
PCB	3	Already Provided
Miscellaneous (wires, pin headers, etc.)	As needed	$40

Note that the goal is to manufacture more PCB boards for future B EE 425 students; the quantity and cost will vary according to the number of boards requested by the sponsor.

Capstone II Schedule

Here is a projected timeframe for this project:

1. **June 20**:
    a. Retrieve the latest version of the code from the former ARM CPU team. Confirm the functionality of the updated code by reviewing the quality control processes established by the previous team.
    b. Brainstorm two lab experiments that students will do for B EE 425.
    c. Develop approaches on extending the state machines, key tasks include research on OpCode decoding and dmem displaying
2. **June 27**:
    a. Completed the development of the sample program intended for instructional purposes in ARM Assembly.
    b. Complete the extension of ARM Core state machines.
3. **July 3**:
    a. Successfully finalized the simplified ARM code, enabling it to execute and showcase LED lights, thereby indicating the proper functioning of the system.
    b. Develop approaches on building and initializing the EEPROM Interface, modifying the EEPROM module with System Verilog, design the instruction fetch mechanism
4. **July 10**:
    a. Complete setting up the EEPROM Interface, modify the EEPROM Module, and update the state machines for EEPROM module with updated instruction fetch mechanism.
5. **July 17**:
    a. Test and verify the state machine in the EEPROM module, Ensure the state machines responsible for fetching and decoding instructions can handle the timing and control signals of the EEPROM. Use simulation tools such as LA and DE -10 to verify that the modified design correctly fetches instructions from the EEPROM by single stepping. Check for correct address decoding, timing, and data integrity.
6. **July 24**:
    a. Successfully demonstrated the flash memory by writing a program where it initialized the IMEM array to external EEPROM.
    b. Ensure the ARM Core state machines, interfacing with EEPROM, and instructions fetching is working, by using single stepping and state/timing analysis.
7. **July 31**:
    a. Design Review with the sponsor and advisor. Confirming previous team's pcb schematic and design.

b. Order the ARM CPU board from ExpressPCB which will likely take a week.

8. **August 11**:
    a. Once the board arrives, test each one by inspecting the board, continuity test using an ohm meter, and signal testing by using an oscilloscope.
    b. Solder components for the boards.
    c. Once all components are soldered, we can do a continuity test with all of the pins. Then we can test the voltage and ground pins for the board. We can also analyze the logic port.
    d. Once everything has been checked, we connect the PCB board with the FPGA and see our outcomes.
9. **August 14**:
    a. Finalize all of the documentation such as the mega-document, poster, and presentation, and two LAB experiments.
10. **August 16**: Demonstrate the Final Product

Internal Specification

Team Collaboration and Communication

- **Regular Meetings**: The team will meet twice a week to discuss progress, address any issues, and plan upcoming tasks. Additional meetings will be scheduled as needed to resolve urgent matters or prepare for key milestones.
- **Task Allocation**: Responsibilities will be clearly defined and allocated based on each member's strengths and expertise. Task assignments will be documented and tracked to ensure accountability and timely completion.
- **Progress Updates**: Team members will provide weekly progress updates, highlighting completed tasks, current challenges, and next steps. This ensures transparency and allows for early identification of potential delays or issues.

Quality Assurance and Testing

- **Code Reviews**: All code will undergo peer review to ensure quality, correctness, and adherence to project standards. Reviews will focus on functionality, readability, and efficiency.
- **Debugging and Verification**: The LogicPort logic analyzer will be used to capture and analyze signals, helping to identify and fix any issues with signal integrity, timing, or functionality.

Documentation and Reporting

- **Project Documentation**: Comprehensive documentation will be maintained throughout the project. This includes design documents, implementation details, testing procedures, and user guides. All documentation will be reviewed and updated regularly.
- **Lab Manuals**: Detailed lab manuals will be created to guide students in setting up and using the ARM-based system. These manuals will include step-by-step instructions, example programs, and troubleshooting tips.

Risk Management

- **Identifying Risks**: Potential risks will be identified at the start of the project and reviewed regularly. This includes technical challenges, time constraints, and resource availability.
- **Mitigation Strategies**: For each identified risk, mitigation strategies such as make up meetings or adjusting the project timeline will be used to minimize its impacts.

Professional Conduct

- We promise to maintain a high standard of professionalism by showing respect to one another, open and clear communications by attending all of the scheduled meetings on time, sharing progress updates, and notifying team members about any potential delays or issues as soon as possible. In addition, we promise to treat each other with respect by having team member's ideas and opinions being valued and acknowledged. We are committed to addressing conflicts constructively by using the 3 W's such as what is the problem, how is it impacting, and why it's important. Also, we promise to adhere to the schedule we created as a team to ensure that each of our tasks is completed within the agreed timeframe. Finally, each team member has a unique skill set, therefore we will ensure that collaboration on difficulties and barriers occurs. In summary, by honoring these commitments, we aim to create a positive environment that promotes successful collaboration, foster mutual respect, individual and team growth, and ensures timely completion of our project. We strive to achieve our goals and deliver exceptional results.

External Specification

The goal of our project is to ensure the visual display of ARM CPU signals. Our design will implement ARM CPU emulation on an FPGA, allowing students to program the CPU and observe its electrical signals through a logic analyzer. Crucial displayed signal information will include IMEM and DMEM access, as well as the movement of data within the ARM CPU, providing a comprehensive view for analysis.

To ensure a functional system, we will verify the proper simulation and operation of the system through FPGA and logic analyzer testing. This involves exposing the expected data through software outputs and ensuring that all signals are accurately captured and displayed. The interface design will include memory mapping for I/O ports, enabling students to execute programs and observe the corresponding signal changes.

Comprehensive documentation will be created to support the system's design and operation. This documentation will include detailed project reports, posters, flowcharts, and presentations. These materials will provide clear instructions and information, enabling others to understand and replicate the project results. The documentation will be essential for future students and educators to effectively use and build upon the system we develop.

Bibliography

We would like to give special recognition to James Oelund, a member of the prior capstone team, for his continuous response to all of our questions. We also wish to express our gratitude to Majeeda, Angelina, Nigel, and Mohamed from the former ARM CPU team, whose work we are inheriting.

Appendix

- Previous ARM CPU Team's Documentation
- Ltd., Arm. "What Is FPGA?" *Arm*, www.arm.com/glossary/fpga#:~:text=Field%20Programmable%20Gate%20Arrays%20(FPGAs,requirements%20after%20the%20manufacturing%20process. Accessed 08 June 2024.
- Ltd., Arm. "Keil MDK – ARM®." *Arm*, www.arm.com/products/development-tools/embedded-and-software/keil-mdk#:~:text=Arm%20Keil%20MDK%20is%20a,and%20Arm%20Ethos%2DU%20processors. Accessed 08 June 2024.