Implementing the ABC Computer with Modern Technology (Phase 2)

Design Document

sdmay26-12
Dr. Alexander Stoytchev
Gavin Bunker – Technical Manual Lead
Ryan McFadden – Client Liaison, Recorder
Samuel Schroeder – Team Leader
Vance Arnold – Team Facilitator, Purchasing
Bryan York – PCB Lead
Kyle Koch – VHDL Lead
sdmay26-12@iastate.edu
https://sdmay26-12.sd.ece.iastate.edu/

Revised: 7 December 2025

Executive Summary

The purpose of this Senior Design team is to restore a piece of Iowa State History using modern technologies to create a replica of the Atanasoff-Berry Computer (ABC), the first digital electric computer made by John Atanasoff and Clifford Berry in the 1930s. Shortly after its construction, the original computer was dismantled due to the rise of World War II, leaving many of its supporting documents to be lost to history, and an important piece of Iowa State history was lost to time.

Our goal is to create a modern version of the ABC using Integrated Circuits (ICs), Printed Circuit Boards (PCBs), and modern computing power, resulting in a working representation that uses the same logic as the ABC. This modern recreation can be used as a teaching aid during ECpE lectures or simply shown to visiting high school students or electronics enthusiasts on Iowa State's campus. The completion of this project will culminate in a single working replica of the original ABC, constructed on PCBs using through-hole components capable of solving a system of 29 linear equations.

Most of the design process for each of the modules created for this project follows a similar process, which includes idea formation, VHDL simulation, breadboard testing, PCB design, and hardware testing. Each of these phases is vital to ensure proper functionality, correct connections inside each module, and the ability to interface all modules together.

Following this design process, the team has already created several modules, including the Full Adder/Subtractor, Decimal to Binary Converter, Decimal Output, and a controllable clock. Implementing these designs will follow a standard created by our team, allowing any user or viewer of the ABC to learn about the inner workings of the computer through the use of silkscreen directions and labels, as well as LEDs for inputs and outputs.

Of all the modules currently being designed, each of them is following the same standards and adhering to the requirements set forth by our advisor, with whom we regularly check in during our weekly meetings. Here, we can raise any questions we have and review the progress we have made to gather his feedback. Following this process, we will continue to work on designing, testing, and implementing the remaining modules necessary for this computer, bringing back an important piece of Iowa State history for everyone, from students to hobbyists, to enjoy.

Learning Summary

Development Standards & Practices Used

In developing the modern recreation of the Atanasoff-Berry Computer, our team relied on a combination of established circuit, hardware, and software practices. Early prototypes were constructed on breadboards to validate the logic of individual modules before committing to permanent designs. To simplify debugging and repair, through-hole and socketed components were chosen, ensuring that faulty chips could be easily replaced. As the designs progressed, PCBs were created following IPC 2221 guidelines, which define trace widths, clearances, and via specifications to guarantee safe and reliable operation. IEEE 1149.1-2013, which provides boundary-scan techniques for verifying interconnections between integrated circuits across multiple PCBs, guided our testing practices as well as our PCB design to ensure that our final system will be easily testable.

For the software side, our team adopted professional standards to maintain clarity and consistency. All logical diagrams and schematics were documented using standardized symbols defined in IEEE/ANSI 91/91a-1991, ensuring that designs could be easily interpreted by other engineers. Then, the simulation and verification of modules were performed in VHDL, adhering to IEEE/IEC 61691-1-1-2023, which specifies the syntax and functionality of the language. This allowed the team to formally describe and test the behavior of each module before moving to hardware implementation.

Beyond technical standards, the project follows standard engineering practices to manage our design process. A waterfall project management approach was adopted, as it suited the sequential nature of hardware development and minimized delays caused by PCB manufacturing cycles. GitLab was used for version control, issue tracking, and collaborative review. Finally, testing was incorporated into the design process at multiple stages to ensure that our project is tested early and often. Thus, each module and subsystem could be validated before full assembly.

Summary of Requirements

- The system should accurately recreate the original functionality and overall design of ABC, including the same modules as the original.
- The system should be able to solve systems of linear equations, up to 29 equations in 29 unknowns, using the same algorithm as the ABC.
- The system should allow users to input data, observe the computation process, and retrieve the output data of each step.
- The system should be composed of modern components, including integrated circuits, printed circuit boards, and semiconductor memory devices.
- The system should be easily carried by one person and should not exceed the dimensions of a typical desk or table.
- The complete system should be modular to allow for easy assembly and disassembly as needed for repairs or long-distance travel.
- The system should have LEDs to indicate the state of the inputs and outputs of each module and be visually appealing to the average user.
- The system should be intuitive to operate, with clear instructions, so that the average user can easily navigate and use it.

Applicable Courses from Iowa State University Curriculum

- CPRE 2810: Digital Logic
- CPRE 3810: Computer Organization and Assembly Level Programming
- EE 2010: Electric Circuits
- EE 3330: Electronic Systems Design
- SE 3090: Software Development Practices
- SE 3290: Software Project Management

New Skills/Knowledge acquired that was not taught in courses

- PCB Design Tools: Learning KiCad for schematic capture and PCB layout
- Soldering and Assembly: Hands-on soldering of through-hole components, socketed ICs, and debugging assembled boards
- Hardware Debugging Techniques: Using continuity testing and modular replacement strategies

Table of Contents

1. Introduction **7**

1.1. Problem Statement 7

1.2. Intended Users 7

2. Requirements, Constraints, And Standards **9**

2.1. Requirements & Constraints 9

2.1.1. Functional requirements 9

2.1.2. Resource Requirements 9

2.1.3. Physical Requirements 9

2.1.4. User Experience Requirements 9

2.2. Engineering Standards 9

2.2.1. IEEE 1149.1-2013 10

2.2.2. IEEE/ANSI 91/91a-1991 10

2.2.3. IEEE/IEC 61691-1-1-2023 10

2.2.4. IPC 2221 10

3. Project Plan **12**

3.1 Project Management/Tracking Procedures 12

3.2 Task Decomposition 12

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria 13

3.4 Project Timeline/Schedule 16

3.5 Risks and Risk Management/Mitigation 18

3.6 Personnel Effort Requirements 19

3.7 Other Resource Requirements 21

4. Design **23**

4.1 Design Context 23

4.1.1 Broader Context 23

4.1.2 Prior Work/Solutions 23

4.1.3 Technical Complexity 24

4.2 Design Exploration 24

4.2.1 Design Decisions 24

4.2.2 Ideation 25

4.2.3 Decision-Making and Trade-Off 26

4.3 Proposed Design 28

4.3.1 Overview 28

4.3.2 Detailed Design and Visuals 29

4.3.3 Functionality 32

4.3.4 Areas of Concern and Development 33

4.4 Technology Considerations 33

4.5 Design Analysis 34

5. Testing **35**

5.1 Unit Testing 35

5.2 Interface Testing 35
5.3 Integration Testing 36
5.4 System Testing 36
5.5 Regression Testing 37
5.6 Acceptance Testing 37
5.7 User Testing 37
5.8 Results 38
6. Implementation 39
6.1 Add/Subtract Mechanism 39
6.2 Binary Conversion Mechanism 41
6.2.1 Functionality 41
6.2.2 Current Implementation Status 41
6.2.3 Remaining Implementation Steps 42
6.3 Decimal Output 43
6.3.1 Current Implementation Status 43
6.3.2 Remaining Implementation Steps 44
6.4 Self Clocking Program Counter 44
7. Ethics and Professional Responsibility 46
7.1 Areas of Professional Responsibility/Codes of Ethics 46
7.2 Four Principles 47
7.3 Virtues 48
7.3.1 Group Virtues 48
7.3.2 Individual Virtues 49
8. Closing Material 52
8.1 Conclusion 52
8.2 References 52
9. Team 53
9.1 Team Members 53
9.2 Required Skill Sets for Your Project 53
9.3 Skill Sets covered by the Team 53
9.4 Project Management Style Adopted by the team 54
9.5 Initial Project Management Roles 54
9.6 Team Contract 54

List of Figures

1. Milestones and Metrics Table 15
2. First Semester Gantt Chart 16
3. Second Semester Gantt Chart 17
4. Risk Management Table 19
5. Expected Effort Table 21
6. PCB Layout Decision Matrix 27
7. High-Level Block Diagram 28
8. Detailed Block Diagram 29
9. Add/Subtract Module 30
10. Address Space Layout 31
11. Add/Subtract Module Schematic 39
12. Add/Subtract Module Pin Layout 40
13. Visualization of Binary Lookup Table 42
14. Binary Conversion Module Schematic 42
15. Decimal Output Module Design 43
16. Seven-Segment Display Prototype 44
17. Code of Ethics Table 47
18. Ethical Principles Table 48
19. Skills Table 53

1. Introduction

1.1. Problem Statement

The Atanasoff-Berry Computer (ABC) was the first electronic digital computer, built at Iowa State University in the late 1930s and early 1940s. Unfortunately, the original ABC was dismantled in 1948. In the 1990s, a team at Iowa State built a replica of the ABC, which was eventually moved to the Computing History Museum in California. The ABC is an important piece of electrical and computer engineering history, which is missing from Iowa State's campus. The goal of this project is to recreate the ABC at the logical level using modern technology. By examining the foundational concepts of the ABC and applying them from a contemporary perspective, we aim to create an accurate and visually engaging recreation of the ABC. This resource will be useful for teaching a diverse audience about computer architecture, design, and history, as well as the legacy of innovation at Iowa State.

1.2. Intended Users

Computing Teachers

1. This user is someone who is teaching computer history, digital logic, or computer architecture. They have expertise in one or more of these topics and are looking for ways to better teach their students, who may not have any previous experience with these concepts, through a more hands-on approach.
2. Computing teachers need a way to accurately demonstrate the logic of the ABC to their students, so that they can help their students better understand the theoretical concepts taught in class. They also need to be able to easily operate this demonstration in a classroom environment and visually show the internal state and operation of the machine.

Computing Students

1. This user is someone who has enrolled in a course about computer history, digital logic, or computer architecture. They are interested in these topics and want to learn more about them in hands-on ways. They also want to better understand and remember what they are taught in class.
2. Computing Students need to see a practical application of what they are learning to better understand the course content. They need to be shown a demonstration that is engaging and visually communicates what the computer is doing, so that they can see how the course concepts are being used.

Computing Enthusiasts

1. This user is someone who has an interest in and some familiarity with computing history, digital logic, or computer architecture, and wants to learn more about the ABC's design, function, or historical role for their personal enjoyment. They may even want to build a replica of their own as a hobby.
2. Computing Enthusiasts need a way to read about the ABC's history and design, so that they can teach themselves about the ABC and its underlying theoretical principles. They need a way to learn how to replicate ABC modules without doing their own deep research through archives into the original design.

Iowa State Visitor

1. This user is someone visiting Iowa State for a university-hosted event, a prospective student tour, or to visit a student or faculty member at Iowa State. They are interested in learning about what makes Iowa State unique and the history of innovation at the university. This user is not familiar with the ABC or the technical issues surrounding its design and implementation.
2. Iowa State Visitors need a way to easily understand what the ABC was and why it is significant, without having to understand the technical details of the machine.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

2.1.1. Functional requirements

1. The system should accurately recreate the original functionality and overall design of ABC, including the same modules as the original.
2. The system should be able to solve systems of linear equations, up to 29 equations in 29 unknowns, using the same algorithm as the ABC.
3. The system should allow users to input data, observe the computation process, and retrieve the output data of each step.

2.1.2. Resource Requirements

1. The system should be composed of modern components, including integrated circuits, printed circuit boards, and semiconductor memory devices (constraint).
2. Only socketed microchips should be used, along with through-hole components to make debugging and repair easier (constraint).

2.1.3. Physical Requirements

1. The system should be a tangible, physical object that users can interact with.
2. The system should be easily carried by one person (constraint).
3. The system should not exceed the dimensions of a typical desk or table, but fit within a space suitable for interactive demonstrations (constraint).
4. The complete system should be modular to allow for easy assembly and disassembly as needed for repairs or long-distance travel.

2.1.4. User Experience Requirements

1. The system should have LEDs to indicate the state of the inputs and outputs of each module.
2. The system should be intuitive to operate, with clear instructions, so that the average user can easily navigate and use it.
3. The system should be visually appealing to the average user.

2.2. Engineering Standards

Engineering standards are an important part of ensuring consistency, safety, and interoperability across the industry. They allow engineers to design systems that work reliably with components from other vendors, based on shared specifications of interaction. By following established safety and testing standards, engineers can ensure that products operate safely and correctly. Standards also reduce errors and streamline the development process, since engineers can focus on improving systems rather than reinventing basic protocols. In this way, standards also promote innovation by providing a shared technical foundation. In education settings like senior design, it teaches students how to think professionally and design with real-world constraints in mind.

2.2.1. IEEE 1149.1-2013

- Defines a method for testing connections between integrated circuits on PCBs and the integrated circuits themselves using boundary-scan techniques. This standard is intended to enhance the testing process of interconnected circuits, improving the safety and reliability of various electronic devices [5].
- This standard is relevant to our project because we will be creating many interconnected PCB modules, each with several integrated circuit chips, which will require testing.
- To incorporate this standard into our project, we will use a modular design that enables the easy insertion of test values and the reading of outputs from individual PCB components. This design will help improve the testing process to ensure the correct and reliable functioning of each component.

2.2.2. IEEE/ANSI 91/91a-1991

- Defines standardized symbols for digital logic gates and functions used in schematics and documentation. The goal of this standard is to ensure that schematics and documentation are clear, consistent, and interpretable by other engineers [6].
- This standard is highly relevant to our project, since we will be reading and writing documentation on the logical design of the ABC and our recreation of it.
- To incorporate this standard into our project, we will ensure that all of the logical diagrams in our documentation or other deliverables use the symbols defined in this standard.

2.2.3. IEEE/IEC 61691-1-1-2023

- Defines VHISC Hardware Description Language (VHDL) syntax and functionality. VHDL is a formal notation designed to aid in all stages of creating electronic systems, from design to testing. This standard is also intended to help engineers develop tools that support the language [7].
- This standard is directly relevant to our project, since we will be creating electronic components to recreate modules of the ABC, which is exactly the process that VHDL is intended to support.
- To incorporate this standard into our project, we will use VHDL, rather than other hardware description languages like Verilog, to develop and verify our component designs.

2.2.4. IPC 2221

- Establishes general design requirements for printed circuit boards (PCBs), including parameters like trace width, clearances, and via design. The goal of this standard is to provide a foundation for designing and manufacturing safe and reliable PCBs [8].
- This standard is directly relevant to our project, since the final product of our project is a recreation of the ABC implemented using PCBs. Much of our work involves converting component designs or breadboard prototypes into PCB designs for manufacturing.
- To incorporate this standard into our project, we will ensure that our PCB design software (KiCad) has been configured with design settings that comply with this standard.

At this stage, our project consists primarily of breadboard prototypes and KiCad schematic/PCB designs, each representing slightly different iterations of the ABC logic modules. As we transition toward finalized PCB implementations, we will incorporate the relevant standards identified in Section 2.2 to ensure consistency, reliability, and educational value. Specifically, IPC 2221 will guide our PCB layout decisions by defining trace widths, clearances, and via specifications, thereby

improving manufacturability and signal integrity. IEEE 1149.1-2013 will inform future testing and debugging procedures, particularly as we expand from breadboards to full PCB assemblies. IEEE/ANSI 91/91a-1991 will be applied to ensure schematic symbols and documentation remain standardized and readable across all module designs. Finally, IEEE/IEC 61691-1-1-2023 will be integrated as we implement module behavior in VHDL, aligning our HDL design and verification processes with recognized best practices. Collectively, these standards will help us unify the differing breadboard and PCB versions into a consistent, standards-compliant system, improving reproducibility, maintainability, and instructional clarity.

3. Project Plan

3.1 Project Management/Tracking Procedures

For this project, we have adopted the waterfall project management style. The waterfall approach, with its sequential structure, is a much more suitable approach given the hardware-oriented nature of our project. Taking an iterative approach, such as agile, would not be cost- or time-effective, as each iteration would require purchasing a batch of PCBs and waiting for them to be delivered before proceeding. With the waterfall approach, we will go through the full process of gathering requirements, designing, and simulating our designs before manufacturing anything. To minimize waiting time, we will have multiple "waterfalls", one for each module. With this strategy, we will be able to start the waterfall process on the next module while we wait for the previous module to be manufactured.

We are primarily using GitLab to track and manage our progress. Using the version control system and merge request process of Git, we can track the progress and review of our designs as well as provide feedback to the designer. We are also using GitLab's issue boards to track the status of each step of the design process across all modules.

3.2 Task Decomposition

1. General Module Design Procedure
    a. Create the logical design
        i. Research how the original module worked and how the previous team implemented the module
        ii. Create a logical circuit diagram of the module
        iii. Prototype the design on breadboards (if needed)
    b. Simulate the design in VHDL
        i. Write a structural VHDL representation of the logical design
        ii. Write and run a VHDL test bench to verify the correctness of the logical design
    c. Design the PCB
        i. Implement the logical design as a KiCad schematic with any other needed electrical components
        ii. Convert the KiCad schematic into a PCB layout
    d. Manufacture the PCB
        i. Submit the PCB design to ETG for purchasing
        ii. Solder any necessary components onto the PCB
    e. Unit testing to verify the correct operation of each PCB
2. Perform the General Module Design Procedure for the following modules
    a. Add-Subtract Module
    b. Memory Module
    c. Timing and Control Modules
    d. Binary Conversion Module
    e. Base-10 I/O Modules
    f. Base-2 I/O Modules
3. User Interface
    a. Design the UI

        i. Gather requirements and designs from the previous team
        ii. Combine the previous designs into a single UI
    b. Implement the UI
        i. Write the software for the unified UI
        ii. Connect the software to the hardware
    c. Unit test the UI software
4. Testing
    a. Interface Testing
        i. Test the functionality between each module and its directly related modules
    b. Integration Testing
        i. Test the data subsystem
        ii. Test the I/O subsystem
        iii. Test the control subsystem
    c. System testing
5. Top-Level Assembly
    a. Design the final assembly to connect all of the modules together into a single product
    b. Create the assembly
    c. Assemble the final product

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone	Metric
Logical Design	
Completion of historical research and prior implementation review	Summary document with references and design notes for each module
Logical circuit diagram finalized	Diagram reviewed and approved by the team and the client
Structural VHDL representation completed	Code compiles without errors and matches the logical diagram
VHDL test bench verifies correctness	All expected outputs match actual outputs for test inputs

PCB Design	
KiCad schematic completed	Schematic includes all required components and passes ERC (Electrical Rule Check) with zero critical errors
PCB layout finalized	DRC (Design Rule Check) passes with zero critical errors
PCB submitted to ETG	Submission confirmation received and logged
PCB manufactured and received	Physical board matches design specs and passes visual inspection
Components soldered	All components are correctly placed, and solder joints pass a continuity test
Unit testing of PCB	≥ 95% of functional tests pass; timing matches expected behavior within ±5% tolerance
UI Development	
Requirements gathered and documented	UI spec document completed and approved
Unified UI design completed	Wireframes/mockups reviewed and approved
UI software implemented	Code compiles and basic functionality is demonstrated
Hardware interface connected	UI successfully triggers hardware responses in test mode

Unit testing of UI	All buttons and indicators respond correctly; latency < 100 ms per
Integration and Testing	
Interface testing between modules	Each module communicates correctly with adjacent modules; signal integrity verified
Subsystem integration testing	Data, I/O, and control subsystems pass ≥ 90% of integration tests
Full system testing	ABC replica solves a system of equations correctly and visibly demonstrates each step
Final assembly completed	All modules are physically connected, and the system operates as a unified unit

Figure 1. Milestones and Metrics Table

3.4 Project Timeline/Schedule

Task Name	Duration	Start	Finish
Data Subsystem Done	0 days	3/21/26	3/21/26
Control Subsystem Done	0 days	2/20/26	2/20/26
All PCBs Ordered	0 days	3/1/26	3/1/26
1. Add-Subtract Module	**71 days**	**9/8/25**	**11/17/25**
1.1 Logical Desgin	8 days	9/8/25	9/15/25
1.2 VHDL Simulation	12 days	10/3/25	10/14/25
1.3 PCB Design	31 days	9/16/25	10/16/25
1.4 PCB Manufacutring	28 days	10/17/25	11/13/25
1.5 Unit Testing	4 days	11/14/25	11/17/25
3. Timing and Control Module	**82 days**	**10/29/25**	**2/20/26**
4.1 Logical Design	45 days	10/29/25	12/12/25
4.2 VHDL Simulation	12 days	12/13/25	1/26/26
4.3 PCB Design	12 days	12/13/25	1/26/26
4.4 PCB Manufacturing	21 days	1/27/26	2/16/26
4.5 Unit Testing	4 days	2/17/26	2/20/26
4. Binary Conversion Module	**55 days**	**10/29/25**	**1/24/26**
5.1 Logical Design	14 days	10/29/25	11/11/25
5.2 VHDL Simulation	4 days	11/12/25	11/15/25
5.3 PCB Design	16 days	11/12/25	11/27/25
5.4 PCB Manufacturing	21 days	11/28/25	12/18/25
5.5 Unit Testing	4 days	12/19/25	1/24/26
5. Base-10 I/O Module	**69 days**	**10/29/25**	**2/7/26**
6.1 Logical Design	24 days	10/29/25	11/21/25
6.2 VHDL Simulation	4 days	11/22/25	11/25/25
6.3 PCB Design	20 days	11/22/25	12/11/25
6.4 PCB Manufacturing	21 days	12/12/25	2/3/26
6.5 Unit Testing	4 days	2/4/26	2/7/26

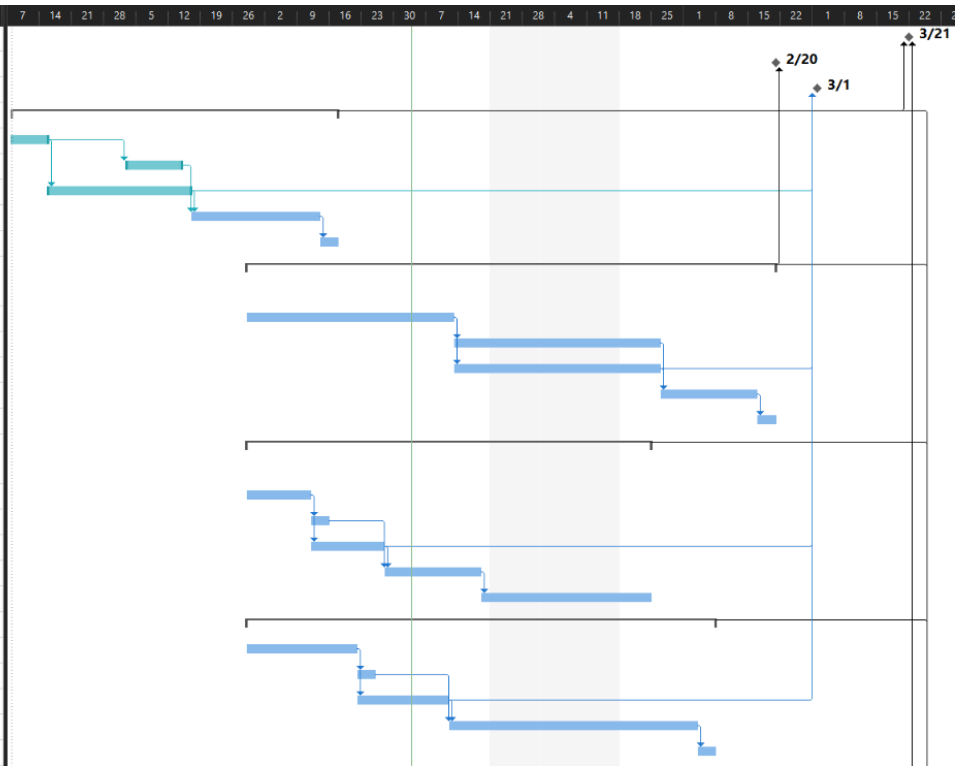


Figure 2. First Semester Gantt Chart

Task Name	Duration	Start	Finish
5. Base-10 I/O Module	**69 days**	**10/29/25**	**2/7/26**
I/O Subsystem Done	0 days	3/26/26	3/26/26
All Modules Done	0 days	3/26/26	3/26/26
Project Complete	0 days	5/4/26	5/4/26
2. Memory Module	**57 days**	**1/24/26**	**3/21/26**
3.1 Logical Design	20 days	1/24/26	2/12/26
3.2 VHDL Simulation	4 days	2/13/26	2/16/26
3.3 PCB Design	12 days	2/13/26	2/24/26
3.4 PCB Manufacturing	21 days	2/25/26	3/17/26
3.5 Unit Testing	4 days	3/18/26	3/21/26
6. Base-2 I/O Module	**47 days**	**2/8/26**	**3/26/26**
7.1 Logical Design	10 days	2/8/26	2/17/26
7.2 VHDL Simulation	4 days	2/18/26	2/21/26
7.3 PCB Design	12 days	2/18/26	3/1/26
7.4 PCB Manufacturing	21 days	3/2/26	3/22/26
7.5 Unit Testing	4 days	3/23/26	3/26/26
7. User Interface	**62 days**	**2/20/26**	**4/22/26**
8.1 Design UI	20 days	2/20/26	3/11/26
8.2 Implement UI	30 days	3/12/26	4/10/26
8.3 Unit Testing	12 days	4/11/26	4/22/26
8. Testing	**100 days**	**1/25/26**	**5/4/26**
8.1 Interface Testing 1	2 days	1/25/26	1/26/26
8.2 Interface Testing 2	2 days	3/22/26	3/23/26
8.3 Interface Testing 3	2 days	3/27/26	3/28/26
8.4 Integration Testing	4 days	3/24/26	3/27/26
8.5 System Testing	10 days	4/25/26	5/4/26
9. Top-Level Assembly	**34 days**	**3/22/26**	**4/24/26**
10.1 Assembly Design	14 days	3/22/26	4/4/26
10.2 Assembly Manufacturing	18 days	4/5/26	4/22/26
10.3 Product Assembly	2 days	4/23/26	4/24/26

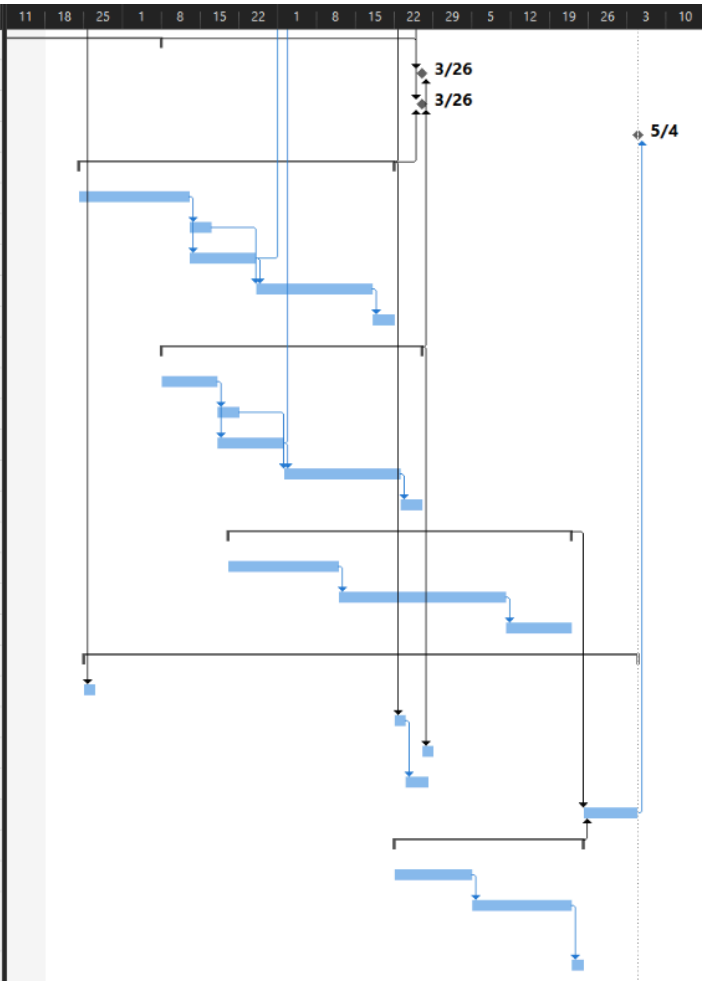


Figure 3. Second Semester Gantt Chart

The first module that we will work on is the Add-Subtract module. Since the previous team was able to fully implement this module, we will each individually design the module to gain familiarity with all of the tools we will use and the design process. Once this module is complete, we will split into groups, with each group working on a single module. With this method, we can schedule tasks for modules simultaneously, as shown in Figure 2, for example, tasks 3, 4, and 5. To minimize time lost waiting for PCB manufacturing, each group will start work on the next module. For example, in Figure 2, task 4.1 begins once task 1.4 (manufacturing the previous module's PCB) has begun. With all modules completed, we will proceed to various levels of testing, unify the UI application developed last year, and assemble the final product.

3.5 Risks and Risk Management/Mitigation

Task	Potential Risk	Probability	Impact	Mitigation Strategy
Logical Design & Simulation	Incorrect logic implementation leading to system malfunction	Medium	High	Perform extensive VHDL simulation before hardware implementation, review logic diagrams against the original ABC documentation, and conduct peer design reviews.
PCB Design & Fabrication	PCB errors (incorrect routing, footprint mismatch, or design rule violations)	Medium	High	Use KiCad's ERC/DRC checks, perform design reviews before fabrication, prototype with small test boards, and order extra PCBs for rework if necessary.
Component Availability	TTL or discrete logic components may be difficult to source	Low	Medium	Identify alternative or pin-compatible components early; maintain multiple supplier options.
Hardware Integration	Timing or signal integrity issues between modules	Medium	High	Use oscilloscope and logic analyzer testing to validate interconnections, ensure consistent power and ground planes, and document signal interface specifications clearly.
Power Distribution	Excessive current draw or unstable voltage regulation	Low	Medium	Simulate expected power consumption; design power rails with margin; include current-limiting resistors and decoupling capacitors.

Software/Firmware Verification	Misalignment between simulated logic and hardware implementation	Low	Medium	Compare VHDL simulation waveforms with real hardware behavior, develop test scripts for each module, and automate regression testing whenever possible.
Schedule/Resource Constraints	Delays due to component lead times or team workload	Medium	Medium	Begin ordering components early; prioritize critical path tasks; schedule weekly progress reviews; reallocate labor to high-risk modules if needed.
Documentation & Demonstration	Insufficient documentation or hardware failure during demo	Low	High	Maintain live documentation on GitLab; perform multiple pre-demo dry runs; prepare backup demonstration video.

Figure 4. Risk Management Table

3.6 Personnel Effort Requirements

Task	Subtasks	Projected Person-hours	Description
Initial Discovery Work	Reviewing the work of last year's SD team that started this ABC Project	24	It is essential for our team to have a thorough understanding of both the original ABC's design and the progress made by last year's senior design team in recreating it with modern components, in order to undertake this ambitious project effectively.
	Researching and reading about the original ABC	30	
Add-Subtract Module	Logical Design	10	The Add-Subtract Module will be the first module of the ABC that we will recreate. To broaden each team member's experience and appreciation for the module, we will each recreate the logic of the ABC's Add-Subtract mechanism on a breadboard. Then, we will
	Structural VHDL Design	5	
	PCB Design	16	
	Ordering PCBs & Logistics	2	
	PCB Assembly & Testing	8	

			develop structural VHDL code for the model and design the module's PCB using the KiCad software.
Memory Module	Logical Design	15	This unit will be responsible for storing all 30 coefficients of the equations to be solved.
	Structural VHDL Design	6	
	PCB Design	15	
	Ordering PCBs & Logistics	2	
	PCB Assembly & Testing	4	
Binary Conversion Module	Logical Design	6	This unit will be responsible for converting decimal numbers from the decimal input module into binary numbers to be stored on the memory drums.
	Structural VHDL Design	8	
	PCB Design	12	
	Ordering PCBs & Logistics	2	
	PCB Assembly & Testing	4	
Timing and Control Module	Logical Design	40	This unit is very important as it sets all the timings with the clock and control signals.
	Structural VHDL Design	20	
	PCB Design	24	
	Ordering PCBs & Logistics	4	
	PCB Assembly & Testing	10	
Base-10 I/O Module	Logical Design	20	This unit will take base-10 input and use an add/subtract module operation to store it in the memory unit.
	Structural VHDL Design	10	
	PCB Design	14	
	Ordering PCBs & Logistics	2	
	PCB Assembly & Testing	6	
Base-2 I/O Module	Logical Design	10	This is the last component of the ABC we will be making, and is used to directly input and output the contents of the memory drums.
	Structural VHDL Design	10	
	PCB Design	12	
	Ordering PCBs & Logistics	2	

	PCB Assembly & Testing	4	
User Interface	UI Design	20	This unit will serve as the UI to control the punch card inputs and will also display the outputs from the machine's calculations.
	UI Implementation	40	
	UI testing	10	
Top-Level Assembly	Motherboard Design	30	This will be the final step in assembling all the units on a motherboard. This will take a significant amount of time, as debugging will be difficult and the overall design of all the wires will be complex.
	Motherboard Testing	20	

Figure 5. Expected Effort Table

3.7 Other Resource Requirements

1. Technical Resources
    a. Breadboards and TTL Chips
        i. To reference and test.
    b. PCB Design Software
        i. Tools like KiCad designing custom printed circuit boards.
    c. Soldering Equipment
        i. For assembling PCBs with through-hole components.
    d. Logic Analyzers and Oscilloscopes
        i. For debugging signal flow and verifying timing.
    e. Power Supplies
        i. Bench power supplies or regulated DC adapters for powering modules.
    f. LEDs, Switches, and Buttons
        i. To replicate the ABC's original user interface and provide visual feedback.
    g. FPGA Development Boards
        i. For implementing the ABC logic in hardware.
2. Knowledge and Expertise
    a. Historical Documentation of the ABC
        i. Schematics, logic descriptions, and operational principles from the original and replica projects.
    b. PCB Fabrication Knowledge
        i. Understanding of layout constraints, signal integrity, and manufacturability.
    c. FPGA Programming Skills
        i. Familiarity with VHDL or Verilog for implementing logic modules.
3. Human Resources
    a. Client Expertise
        i. Weekly meetings with Dr. Stoytchev for guidance and historical accuracy.
    b. Peer Collaboration

        i. Team coordination for modular development and integration.
4. Software Resources
    a. Simulation Tools
        i. For verifying logic before hardware implementation, such as ModelSim
    b. Version Control Systems
        i. GitLab for managing code and documentation.
    c. Documentation Tools
        i. Google Drive and Google Docs for writing technical reports and user guides.

4. Design

4.1 Design Context

4.1.1 Broader Context

Our recreation of the ABC exists within the context of historical preservation, engineering education, and the origins of digital technology. The design aims to make the ABC's principles, such as binary arithmetic, electronic switching, and capacitive memory, tangible to modern learners. We are designing for educational audiences, primarily university students and instructors, as well as historians and anyone interested in the origins of digital computing. The project contributes to societal needs for STEM education, historical literacy, and preservation of technological milestones. By reconstructing the ABC, we are helping people appreciate the ingenuity of early computing pioneers and understand how their ideas have shaped today's digital world.

The project prioritizes safe, low-voltage electronic design suitable for classrooms. While it does not directly affect public health, it promotes public welfare through educational engagement and technological awareness. We will ensure all electrical components operate at relatively low voltages and are safe for public interaction. The documentation we will develop to explain how to operate our recreation will help make sure it is usable across classrooms and other interested engineers.

The ABC is an important piece of global computing history and a key part of Iowa State University's history of innovation. The project honors the contributions of Atanasoff and Berry while highlighting their work in the computing field. We will ensure our supporting documentation will be educational for learners and visitors of all backgrounds. This project will help create institutional pride and teach users about local contributions to world-changing technology.

Given its educational and small-scale nature, the project's environmental impact is minimal. Efforts will be made to minimize waste in construction and design the replica to be modular and repairable, extending its useful life in educational settings and reducing wasted parts.

The recreation will be designed to be as cost-effective as possible, given the technological complexity of staying true to the ABC's logical design. It will be reproducible by other institutions, schools, or museums thanks to our open-source documentation and construction with widely available electronic components. Its economic value lies in supporting education, research, and cultural tourism rather than commercial profit.

4.1.2 Prior Work/Solutions

Our recreation of the ABC is a continuation of a project from one of the senior design teams last year (sdmay25-29). Their project focused on doing a lot of discovery work to determine the logic behind each of the ABC's modules and how they operated together. The second semester of their project concluded with them making a small-scale recreation of the ABC on breadboards, successfully implementing most of the components of the original ABC [3]. Our work this year will utilize the discovery work this team did in order to recreate all of the components of the ABC

machine on PCBs and bring it up to its full computational scale (solving systems of 30 linear equations).

In the 1990s, a team of researchers at Iowa State attempted to create a one-to-one replica of the ABC, using vacuum tubes, rotating drums, and other mechanical devices to match the technology used by the original ABC [9]. While this recreation was primarily considered a success, it had an incredibly high budget ($300,000), and the final product was seemingly only able to solve carefully chosen sets of linear equations that are "easier" for the machine to solve. This recreation also severely lacks proper documentation and a technical manual describing how each component was implemented. This significant deficit in documentation of the details of the ABC's operation is something we will be addressing through our project and the technical manual we will develop.

Our project will also have many parallels with the i281 Hardware senior design project from 2014. In this project, the team designed hardware to support the i281 CPU for Iowa State's CprE 281: Digital Logic course [1]. While this CPU is very different from the ABC, we will follow a similar design workflow as this team: prototyping each component to understand how it functions before designing and fabricating PCBs for each module. In their case, this prototyping was done primarily through creating their logical designs on breadboards. We plan on doing most of our prototyping in structural VHDL to ensure thorough testing, reduce human error, and improve efficiency.

4.1.3 Technical Complexity

Our project is sufficiently technically complex due to the high degree of technical research it will require and the large number of interdependent components we will design. For each component of the original ABC, we will need to investigate its behavior to determine the logical diagram behind it. Then, we will develop our designs in VHDL and test them to make sure they operate as expected before designing and manufacturing PCBs for each module.

This mix of logical design, programming, and PCB fabrication give our project a wide breadth of technical complexity. The challenge of staying true to the ABC's original design while designing for modern components and bringing everything together to an operational, full-scale recreation will not be easy. However, we believe that through this unique challenge, we will be able to develop an operational final product that will improve on and exceed the current ABC replicas in existence today.

4.2 Design Exploration

4.2.1 Design Decisions

The unique structure of this project allows for many design decisions to be made that consider both modern-day computing practices and the computing practices of the ABC. Since its original inception, many standards from logic gate symbolism to counting bases have changed. For example, the ABC was implemented to hold 30 sets of numbers: 29 variables, and 1 constant term. In today's computing world, data is stored in base two, which is not a perfect match when it comes to representing or storing 30 sets of numbers. The best option to address this mismatch in technology is to increase the hardware needed in order to cover 32 equations. Of course, adding more hardware means increasing the overall size and cost of this system, but provides the most accurate rendition of the ABC. This option also gives us the option to either implement 30 equations exactly, or scale up to 32 equations and utilize the full hardware capabilities that modern-day components offer.

Another quirk of the 1930s computing powers deals with the punch cards they used. These punch cards were able to hold the information for 5 variables. Once again, due to the nature of base 2 counting, many chip or gate packages are assembled in groups of 2, 4, or 8. This once again leads to the same dilemma mentioned above where the team will need to decide whether the modern system should be implemented exactly like the original or modified to meet today's technology standards.

When it comes to making the actual PCBs, some consideration about the overall placement of different interfacing components, such as connector pins or ports to other modules or power supplies. This decision involves a lot of discussion from our group about standardization throughout all PCB modules that would be created, and affects the spacing and sizing of all the modules created. Ultimately, the team will need to decide if interfacing through breadboards would be important for testing once each PCB is created. This also means each input and output pin would need to be oriented facing downwards and spaced to fit into the channels of a breadboard. While spacing itself will not affect the overall functionality of the full ABC design, having an easier option to test and supply signals to the PCBs via a breadboard will make the debugging process much easier and quicker, leading to a lower probability of resulting errors.

One decision that the team made early on in the project was not to directly solder any chips to the PCBs. Instead, each chip should interface with the board through a socket, which would be soldered to the board instead. This practice goes against the normal convention of PCB design, but is practical for an experimental project such as this one. If a chip were to go bad on the board, having the ability to swap components in and out of the sockets quickly allows for a shorter time of maintenance for the machine or any sub-module. This decision was also made with the guidance of previous senior design teams at Iowa State who were implementing similar projects. These sockets allow for easier debugging to take place if a module is not working properly. If rewiring needs to take place to a pin, it is easier to remove a pin from the socket than it is from a PCB.

Finally, components used on the modules created for this project are restricted to through-hole packages, which allows for easier soldering and bigger visuals for the project as well. Since one of the end goals for this project is to be a visual aid for those learning about the ABC or digital logic, having larger components or blinking lights is a must. These systems will allow a presenter to point to a certain module or chip and describe what that chip is doing. Similarly, for the LEDs, having sizable bulbs that can be seen distinctly from across a table or presentation area, not only gives people feedback about what the system is doing, but also draws curiosity from bystanders.

4.2.2 Ideation

All design considerations for this project are made by the whole team, with advice from our faculty advisor, together in joint brainstorming sessions during weekly team meetings. Many factors go into making these decisions, and they rely on weighing the positives and negatives of each option and drawing on information gathered from technical documents from Atanassof and Berry during the creation of the original ABC or previous senior design teams' projects.

In order to create the Adder Subtractor Module, a lot of discussion was had about how each of the inputs and outputs will need to be laid out on the board in order to interface each module together. Listed below are some examples of designs considered.

- Create a PCB tester module to connect the Adder Subtractor Module to:

Creating another PCB specifically for each subcomponent would allow the team to easily connect and disconnect different modules together and run tests independently of any other external wires, breadboards, or components. This option would also allow the placement of the input and output pins to be more flexible in their placement on the main modules, and would result in a smaller PCB area. However, this would also add an incredible amount of cost to this project, as many more PCBs and components would need to be purchased to test each module.

- Connect all PCBs together via breadboards:

  Instead of connecting modules to PCBs for testing, modules could be connected to breadboards, with signals coming in and out via the breadboard channels. This would allow for more in-depth testing and applying of inputs of various input voltages and currents

- Standardizing CLK and RST_N on all modules:

  One standard that was implemented by the previous senior design team for this project was to implement inputs for the CLK and RST_N signals on all modules created, regardless of whether they used these signals or not. While having extra connectors to the board may help to add stability in mounting, this will also open up to possibility of adding noise to our system and interfering with more important signals on the PCBs.

- Input and output pins are located on opposite sides or the same side of the board:

  In most applications, inputs to a system are usually shown on the left side of the system, and outputs on the right side. This makes it easier for whoever is watching the system to understand the natural flow of how information is being processed and output. In order to save space routing bus wires to these modules, having all of the inputs and outputs located on one side of the board would allow for shorter wires to be run between modules. Additionally, this would cut down on the number of breadboards needed if they were used to interface modules together.

- Connect all PCBs together on a singular motherboard PCB:

  Connecting all of the submodules together on one huge motherboard PCB would allow a cleaner finish to the overall project and allow for an easier way to show off the project. One major downside of this would be the overall cost to manufacture a board of this size. PCBs alone would also not be able to support the weight of all of the modules combined, and would need to be reinforced by another material as well.

4.2.3 Decision-Making and Trade-Off

During weekly team meetings, we conducted joint brainstorming sessions with our faculty advisor to evaluate multiple layout and testing strategies for our individual modules. These sessions drew on historical documentation from Atanasoff and Berry, as well as insights from a previous senior design team. Our goal was to balance historical accuracy, cost, modularity, and ease of testing.

We identified five primary options:

1. Create a dedicated PCB tester module
2. Connect modules via breadboards
3. Standardizing CLK and RST_N on all modules
4. Input and output pins are located on opposite sides or the same side of the board.
5. Connect all modules via a singular motherboard PCB

To evaluate these options, we used a weighted decision matrix because it allowed us to compare multiple decision options across criteria that mattered most to our project goals. Each option was scored from 1 (poor) to 5 (excellent) on each criterion.

Option	**Cost**	**Debug**	**Modularity**	**Fidelity**	**Demo**	**Weighted Score**
PCB Tester Module	2	4	5	3	4	3.6
Breadboard Interconnection	5	4	5	4	3	4.2
Standardize CLK and RST_N	5	4	4	4	4	4.2
Pins Locations	5	5	4	3	5	4.4
Singular Motherboard PCB	2	2	1	2	4	2.2

Figure 6. PCB Layout Decision Matrix

We determined breadboard-based interconnection as the best testing strategy, with the addition of selecting the suboptions of standardizing CLK and RST_N signals on all modules and having input/output pins on the same side of the board. These options offer the best balance of cost efficiency, modularity, and debugging flexibility. It allows us to test modules independently and quickly identify issues without the overhead of manufacturing additional PCBs. The decision to use breadboards also influences our PCB layout choices, including downward-facing pin orientation and consistent spacing across modules. These decisions make it easier to interface with breadboards and streamline the testing process.

4.3 Proposed Design

4.3.1 Overview

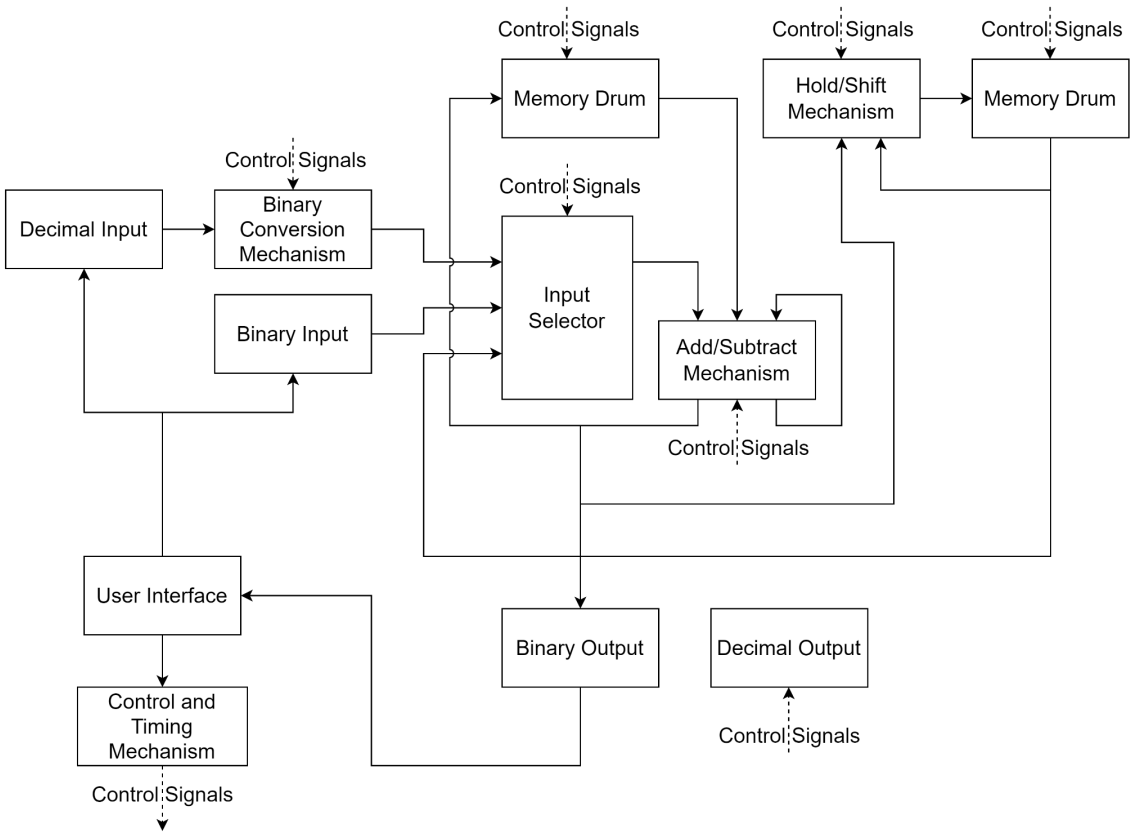


Figure 7. High-Level Block Diagram

The design of our ABC replica is intended to accurately represent the original ABC design. The core component of our design is the Add/Subtract Mechanism, which is responsible for adding or subtracting two equations to eliminate a variable from the system of equations being solved. The inputs to this component can come from multiple sources, which are controlled by the Input Selector. If two equations are being operated on, both inputs come from the two Memory Drums, which store the two linear equations. There are times in the elimination process when one equation needs to be divided by two. This is handled by the Hold/Shift Mechanism. The Input Selector can also send decimal or binary user input to the Add/Subtract Mechanism. Since the computer operates on binary numbers, Decimal Input must be sent through the Binary Conversion Mechanism before reaching the Input Selector. Binary Input can be sent directly to the Add/Subtract Mechanism. The result of the addition or subtraction is sent back to one of the Memory Drums to use in the next operation. Both the Binary Output and Decimal Output components are used to view the contents of the Memory Drums. Outputting the value in binary format makes it easier to enter as an input in later operations, since it does not need to be converted. The final two components, the User Interface and Control and Timing Mechanism, are used to view outputs, send inputs, and send all control signals to the other components.

4.3.2 Detailed Design and Visuals

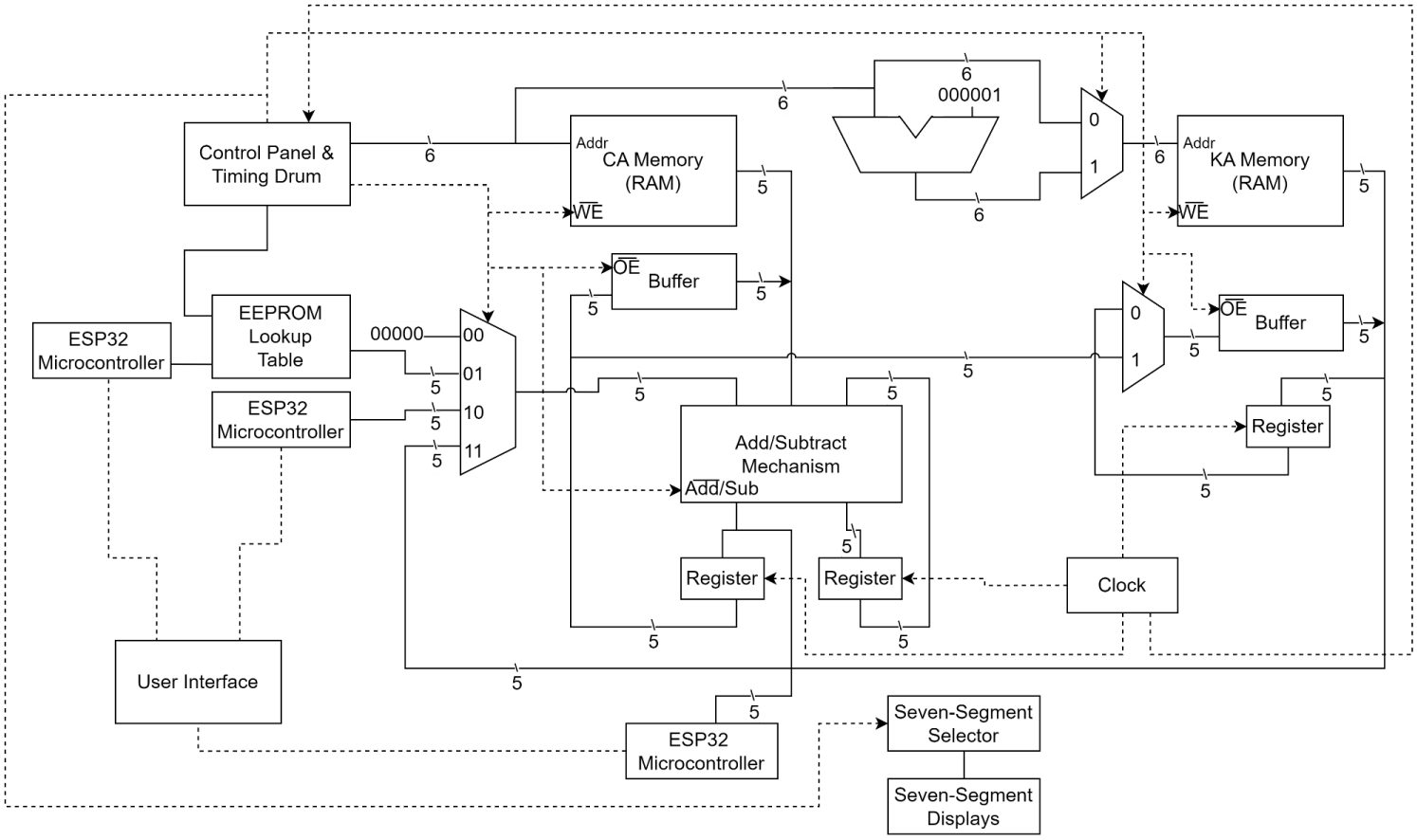


Figure 8. Detailed Block Diagram

Figure 8 provides a more detailed block diagram of our replica for a five-equation prototype. The original ABC was able to solve a system of thirty equations, but it performed some operations, such as decimal input, five equations at a time. The final full-scale recreation would contain six of these components, with the bus widths scaled appropriately.

Add/Subtract Mechanism

This component is responsible for performing arithmetic operations on the coefficients of the two linear equations being operated on. The component consists of five one-bit adders/subtractors (Figure 9) that work in parallel on each of the five coefficients. The computed sum or difference is stored in a register to be written back to memory. The carry or borrow bits are also stored in a register for use in the next addition or subtraction.

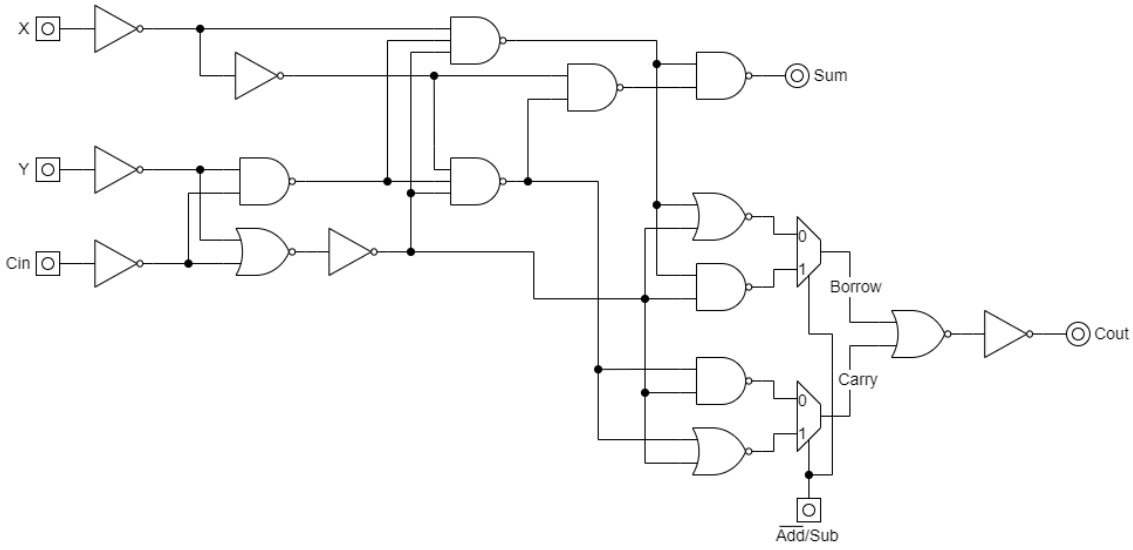


Figure 9. Add/Subtract Module

Memory Drums

Each of the ABC's memory drums stored the fifty-bit numbers for up to thirty coefficients as thirty bands of capacitors around the circumference of the drum. The drums were read across the length of the drum, meaning one bit from each coefficient was read simultaneously. After being read, the capacitors were written to while the next row was being read. Since this level of parallel reading and writing is not possible with modern memory, each memory chip is connected to a buffer so that writes occur after reads. The KA memory buffer also has a multiplexer to aid in transferring data between drums. The memory layout of our RAM chips is not in a standard format. Instead of storing each coefficient at its own memory address, each memory address will store the nth bit of each number. To store five, fifty-bit numbers using this method, we use a six-bit address space as shown in Figure 10 with the least-significant bit at address zero. All of the cells highlighted in red are unused space. The original drums had sixty rows per band, with the last nine rows being empty for control signals. These empty rows are represented by the cells highlighted in blue. To allow for sign extension when performing a bit-shift, the ABC's drums connected the 50th and 49th row together so that they always held the same value. This is represented by the cells highlighted in green.

	Bits							
Address	0	1	2	3	4	5	6	7
000000 (0)								
...								
...								
...								
...								
110001 (49)								
110010 (50)								
...								
111011 (59)								
...								
111111 (63)								

Figure 10. Address Space Layout

Hold/Shift Mechanism

The modified Gaussian elimination algorithm that the ABC uses to solve systems of equations sometimes requires dividing an equation by two, which can be performed with a right bit shift. With the memory layout described earlier, this bit shift can be performed by reading from the next address and writing to the original address. This is implemented with the adder and multiplexer connected to the address line of the KA memory. If a shift is needed, the multiplexer selects the output of the adder, which adds one to the address input. All other times, the multiplexer selects the raw address line.

Input Selector

This component selects which input to combine with the values from the CA memory and is a simple multiplexer. For adding or subtracting two equations, the multiplexer selects the output from the KA memory. To input values into CA memory, the multiplexer selects either the binary or decimal input, which will add that value to the existing CA memory. For new values, CA memory should be cleared before inputting data. The CA memory is the only memory that can be directly written to. To move data to KA memory, the data must first be written to CA memory then transferred. Once the data is in CA memory, the input selector chooses the zero line so that the value from CA memory is not changed and the multiplexer on the KA memory's buffer selects the output from the add/subtract mechanism.

Decimal Input and Output

The original ABC used IBM punch cards for decimal input and an odometer-like mechanism for decimal output. Since our goal is to create a replica using modern technology, we will not use punch cards for decimal input, but will instead use an ESP32 module connected to an Android application that will simulate the signals from the ABC's punch card reader. For the decimal output, we will use a seven-segment display along with a series of counters to mimic the odometer-like output.

Binary Conversion Mechanism

When decimal punch cards were entered into the ABC, the decimal numbers first had to be converted to binary. To do this, the ABC had another drum that held a conversion table between decimal and binary. Instead of a physical drum, our recreation uses an EEPROM chip to hold the binary conversion tables.

Binary Input and Output

For intermediate steps that did not need to be human readable, the ABC used sheets of paper with the binary representation of the memory contents punched through the paper with arcs of electricity. For our recreation, we will use ESP32 modules connected to an Android application to simulate the signals from the binary card reader and interpret the signals from the binary card writer.

Control Panel & Timing Drum

Each of the ABC's components require different signals to control the different operations they perform. Some of these signals were manually provided by the operator, using different buttons, switches, and jumpers, and others were automated with a timing drum, which emitted different signals based on the rotation of the drums. For our recreation, the manual signals can still be entered into a physical control panel with switches and buttons and the automated signals from the timing drum come from an EEPROM which contains the appropriate signals for each position.

4.3.3 Functionality

Our design is intended to fit onto a large PCB or a large plywood board that holds all of the PCBs and circuits. The design will feature a large screen or monitor for setting inputs, since we will not use physical punch cards, but instead use digital ones. We will have a binary and decimal punch card on the screen, as well as an output card and decimal display of the output value from the ABC. The device will also feature a number of LEDs to indicate exactly what is happening in the circuits, allowing you to view some of the intermediate values.

An example of a typical user experience with the ABC begins with the user loading/programming the first decimal card on the display. They would then press a button, and that value would be converted to a binary code and stored in the correct register for each variable. This input process would be repeated as many times as needed to store all values of the linear system of equations they want to solve. Once all the values are inputed, the user will choose a value to start eliminating using Gaussian elimination and enter it into the first drum of the machine. Then, the user will select another equation and store it in the second drum, after which they will start the elimination process. After the process is complete, the variables will be removed from the equation, and the new update value will be stored back in the registers. This is the basic process of the machine, which will continue until the entire system is solved and the values can be outputed back to the display. Once back on the display, each equation will require a single division operation to get the final results. This value will be calculated by the program and displayed to the user.

4.3.4 Areas of Concern and Development

The current design done by the group before us works mostly as intended, except that they have not implemented much of the user controls for the machine. The design is currently limited to inputting four inputs at a time using digital punch cards, but the original ABC used a system with

five inputs from the punch card. Therefore, we need to redesign the ABC recreation to accommodate this. One area of concern that we have is the time required to implement the full-scale recreation of the ABC from the start. To address this concern, we plan to create a design that can operate on five variables at once, and extend our design if this works. Given the parallel nature of the ABC, we should be able to easily connect multiple of our small-scale designs together to reach the full scale. We are also concerned about the historical accuracy of the ABC and how accurate and true to the original we can be while using modern design and technologies. Our immediate plan is to develop new ways to enhance the previous team's design, allowing for more inputs, and utilize various components and ideas to solve these problems in innovative ways, enabling the system to operate like the original ABC. One challenge we face is that many of the solutions that the original ABC used are not feasible with modern technologies, since it was designed before most modern electronics were developed. We also will need to ensure that our design assumptions are correct. Since many facts from the original ABC have been lost over time, we have had to make some assumptions about the design of the new modern ABC.

4.4 Technology Considerations

This project combines modern hardware with historical logic to rebuild the Atanasoff-Berry Computer in a form that is accurate, durable, and educational. It uses TTL logic chips, through-hole components, socketed ICs, and KiCad-designed PCBs verified through VHDL simulation before fabrication. TTL logic provides stable performance and keeps the system faithful to early digital designs. Through-hole parts and sockets make the boards easy to assemble, repair, and understand during demonstrations. The trade-off is space and power use, since discrete logic takes more room and current than integrated options, especially when we recreate it as initally designed, which is innefficent compared to modern methods.

VHDL simulation allows the team to confirm circuit logic before committing to hardware. This reduces wasted materials and ensures functionality before manufacturing. KiCad is the main tool for schematic and PCB layout, chosen for its open-source flexibility and compliance with IPC-2221 standards. Electrical and design rule checks built into KiCad help prevent layout errors before fabrication.

ESP32 modules manage user input and output by replacing physical punch cards with a Bluetooth-connected mobile interface. This preserves the interactive behavior of the ABC while keeping it practical for classroom or exhibit use. The main drawback is synchronization, since the ESP32 operates on a separate clock, requiring careful timing alignment with the logic system.

Alternative technologies like FPGAs were evaluated but not chosen. While they would simplify the build, they would remove the physical logic that makes the project valuable for teaching and historically accurate. The design remains focused on clarity and visibility, with LEDs showing signal states and all major components large enough for hands-on demonstration and classroom viewing.

4.5 Design Analysis

The Add/Subtract, Carry-Borrow, and Memory modules have been successfully simulated and verified in VHDL. The Add/Subtract breadboard prototype operates as expected, and its PCB design has been submitted for production. Testbench results confirm correct arithmetic logic and the initial design performs according to plan, with simulation results closely matching physical testing. Breadboard validation has confirmed that control and data paths are functioning as intended.

The next steps are final PCB assembly, inter-module testing, verifying communication between units under shared timing control, and creating the rest of the ABC's modules. Once modules are confirmed to operate together, the focus will shift to system-level timing, control, and interface refinement. Additionally, the whole computer will be constructred in VHDL to avoid unexpected problems.

Progress so far supports the project goal: a modular, operational ABC recreation that maintains historical accuracy while using modern components. Future efforts will concentrate on improving power distribution, signal timing, and inter-module reliability to prepare for full-system integration.

5. Testing

5.1 Unit Testing

Each module and its PCB implementation will be considered a single unit for testing. We will be testing our proposed logic for each module of the ABC in structural VHDL before designing and fabricating PCBs in order to ensure our approach is functional. This will enable us to verify our designs quickly - significantly faster than breadboard prototyping and testing every unit - and to reduce our development costs by catching design flaws before fabrication. We will still perform preliminary breadboard testing on some units, such as the binary conversion drum and decimal output counter, to ensure our designs work with the chips we will use in the PCB and functions as expected. We will also test our fabricated PCBs individually using breadboards and control wires to simulate how each unit of the ABC will function in its respective place in the full machine. Where feasible, we will use an exhaustive truth table to test each unit. For modules with more complex input where this is infeasible, we will use a combination of boundary value and equivalence class testing to cover as many cases as possible. This unit testing will ensure that each module of the ABC works on its own, and will narrow down the possible issues that we may encounter when testing the modules together and doing full system testing.

Tools used for physical unit testing include oscilloscopes for analyzing timing signals, and function generators for supplying controlled input patterns. Bench power supplies and multimeters are used to validate voltage levels, continuity, and connector pinout accuracy. Collectively, these methods ensure that all electrical and logical components of the unit will operate reliably when integrated into the final system.

5.2 Interface Testing

The final product for this design consists of multiple PCBs, or submodules, mounted onto a larger motherboard PCB. Each board connects through defined interfaces such as shared data lines, address lines, control signals, clock inputs, and regulated power rails. The submodules are designed for hot-swapping, which allows them to be inserted onto the main motherboard or tested individually on a breadboard. This capability supports verification that interface signals, such as binary outputs from the EEPROM being sent to the add/subtract module or timing pulses from the control circuitry updating the decimal output module, maintain correct logic levels and align with expected timing. By connecting multiple submodules together, various input combinations can be tested to ensure they propagate correctly through the system and match expected behavior based on the ABC architecture. Interface validation is also performed before PCB fabrication by modeling each subsystem in VHDL. Simulating logic behavior, timing interactions, and architectural dependencies helps verify that the design is feasible using the selected electronic components. This early-stage testing prevents conflicts between submodules and reduces the risk of errors before committing to PCB manufacturing and assembly.

Interface testing will largely use the same tools used in the unit testing phase, but for different purposes. Because the system consists of multiple digital submodules placed in close proximity, noise and signal integrity will also evaluated as part of the interface testing process. Switching noise and other interference will be monitored using an oscilloscope at key locations such as the data bus, clock distribution lines, and the EEPROM outputs. Interface testing will also be used to ensure that long trace lengths on the motherboard do not introduce unintended signal coupling or degrade logic thresholds.

5.3 Integration Testing

Although there are several different integration paths, three are the most critical as they provide the essential functionality required to solve a system of linear equations like the original ABC. The first path is the input-to-memory path, which is required to input data into the ABC for computation. This path utilizes the decimal card input, binary conversion drum, add/subtract module, one of the memory drums, and the various routing components that correctly direct the signals between these modules. The second path is the computation path for performing the Gaussian elimination algorithm, which includes both memory drums, the add/subtract module, the hold/shift mechanism, and the connecting components. The final critical path is the memory-to-output path which is essential for users to see the results of computation. This path is made up of one memory drum, the add/subtract module, the decimal output module, and the routing components.

Testing these integration paths will use similar tools and methods as the other testing phases. Each integration path will be tested using a set of test values determined using boundary value and equivalence class methods to ensure a wide range of cases are covered. For virtual integration testing, the VHDL simulation of each component in a critical path will be connected together, and waveforms will be used for the test values and results. For physical integration testing, each PCB module will be connected together with breadboards, and the tools mentioned above will be used to generate and read the electrical signals for the input and output.

5.4 System Testing

System testing will be performed once all modules have passed unit, interface, and integration testing. The purpose of this stage is to validate the complete ABC replica as a unified system, ensuring that the hardware modules, control logic, and user interface operate together to meet the functional and user experience requirements. At this level, testing focuses on the end-to-end behavior of the system, verifying that it can correctly solve systems of linear equations, display intermediate states through LEDs, and provide accurate outputs through the I/O modules and user interface.

The methodology for system testing will involve executing representative workflows that mirror how the ABC was originally intended to operate. This includes entering equations through the input modules, observing the computation process step by step, and confirming that the final solution matches expected results. Timing and synchronization between the control and memory modules will be measured against VHDL simulation benchmarks to ensure that the physical implementation remains within acceptable tolerances. Extended demonstrations will also be conducted to evaluate reliability.

In addition to functional correctness, system testing will assess performance, usability, and safety. Performance testing will determine whether the system can handle the maximum problem size of equations without exceeding timing constraints. Usability testing will involve users interacting with the system using only the provided instructions, ensuring that the interface is intuitive and visually engaging. Safety testing will confirm that the system responds appropriately to invalid inputs or incorrect operating conditions. The system will be considered successful if it demonstrates accurate computation, reliable operation, and clear communication of its internal state to users. Results from this stage will be used in the final acceptance testing phase to provide evidence that the modern ABC replica meets its design requirements.

5.5 Regression Testing

The progressive nature of our project and testing process will help to ensure that new additions do not break old functionality. Due to constraints discussed in section 3.1, our project is not taking an iterative approach for creating our PCB modules. The result of this decision is that we will gradually accumulate completed PCBs, which has two implications for testing. First, previously tested units and interfaces are not modified. This, by nature, reduces the likelihood of regression. Second, the different stages of testing (unit, interface, integration, etc.) will be completed as the PCB fabrication process finishes. After finishing the fabrication and unit testing of a new module, we will perform interface and integration testing with this new module and a set of already tested previous modules. Regression testing will be incorporated into this step of the testing process by including previously successful test cases, in addition to the test cases for the new module. Because regression testing will be incorporated into the other testing stages, it will be performed using the same tools and techniques as those other stages.

5.6 Acceptance Testing

Since many of our design requirements were provided by our client and since he will be the primary user of our project's results, our client will play an important role in our acceptance testing process. Acceptance testing will be conducted at the end of the logical design, PCB design, and fabrication phases. At the end of each of these phases we will present or designs or PCBs to our client to ensure that they meet all functional and non-functional requirements before proceeding to the next development phase. Performing this acceptance testing early and often will ensure that our design and implementation will meet the requirements and ensure that decisions that will move our project away from meeting the requirements are quickly identified and corrected.

5.7 User Testing

User testing will focus on evaluating whether the ABC replica is understandable, operable, and useful to its intended audience: computing instructors, students, and campus visitors. Since the project's purpose extends beyond raw functionality, user testing will evaluate clarity, ease of interaction, and the educational value of the system.

Testing will be conducted in three phases:

Phase 1: Early Interaction Testing (UI Prototype and Module Demonstrations)
We will recruit small groups of computing students (100 and 200 level courses) and faculty members to interact with early UI prototypes and individual modules. Users will perform tasks such as reading memory values, activating control signals, and navigating the UI. We will observe how quickly they understand the display, whether they can identify the state of LEDs, and where confusion occurs. Feedback will be collected through observation forms and a short survey.

Phase 2: Subsystem Walkthrough Testing
Once multiple modules are functioning together, we will invite users to execute partial workflows (such as loading decimal input or performing one add/subtract operation). Users will be asked to follow the written instructions without team guidance. Observers will take notes on hesitation points, unclear terminology, and any mismatches between expected and actual system behavior. A follow-up interview will identify usability issues and suggestions.

Phase 3: Full Workflow Testing (End-to-End Problem Solving)
When the system can execute multi-step operations (for example, eliminating a variable), users will solve small systems of equations using the full interface. Metrics collected will include:

- Time required to complete major steps
- Number of user mistakes or misinterpretations
- Points where instructions were insufficient
- Perceived clarity of the LED indications and UI layout
- Overall confidence in understanding how the ABC operates

This testing will validate whether the replica meets the educational and usability requirements. Collected feedback will guide refinements to documentation, UI layout, labeling conventions, and physical design decisions (LED positioning, button labeling, etc.).

5.8 Results

The Add/Subtract module is the only fully completed and fabricated PCB at this stage. The board has undergone electrical verification, including continuity checks, power-rail validation, and LED indicator confirmation. Functional testing used the Boolean tables documented in the ABC reference materials. These tables define the correct arithmetic behavior for addition, subtraction, and carry/borrow propagation exactly as the original machine performed them.

Prior to fabrication, a VHDL simulation of the logical design was tested in Questa using waveform versions of this table. Across all tested input combinations, the VHDL design and PCB output matched the expected values from the reference Boolean tables. This confirms that the logical design, VHDL simulation, and physical hardware implementation are consistent and accurate. The successful operation of this PCB demonstrates that the design methodology for the ABC modules is sound and can be applied to the remaining modules.

Further results will be generated as additional PCBs are manufactured and integrated, allowing our team to begin subsystem and user-oriented testing.

6. Implementation

6.1 Add/Subtract Mechanism

The first part of the modernized ABC computer that the team successfully implemented was the Full Adder/Subtractor (FAS) module, which is used to compute the addition or subtraction of two one-bit numbers with a carry-in bit. This module will handle all the computing that the full computer will perform. Thankfully, the team was equipped with multiple different design files from Atanasoff, Berry, and the previous Senior Design team outlining the logical and electrical implementation of one of these FAS modules. Because of these designs, our team was able to start by verifying the logical design specifications provided by the previous senior design team [3] against the logic table in Figure A.18 in the book "The First Electronic Computer" [2] using a combination of breadboard prototypes and VHDL simulations.

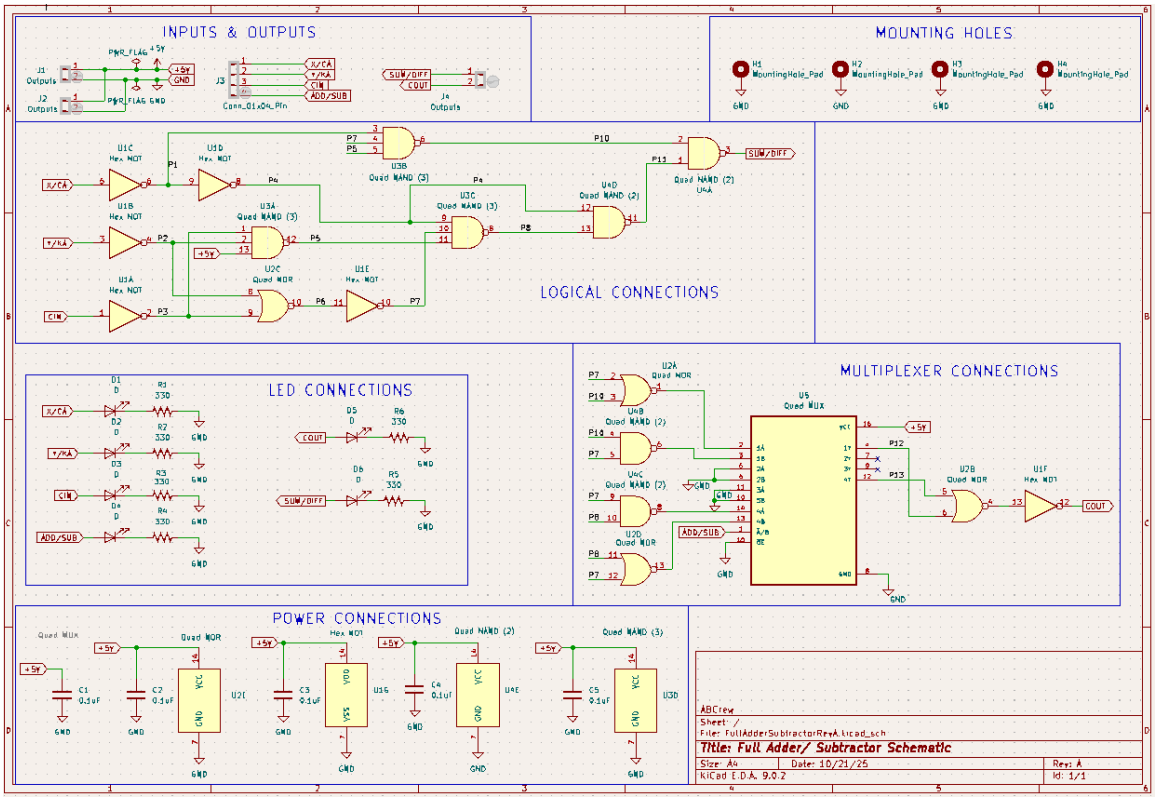


Figure 11. Add/Subtract Module Schematic

Once verified, the logical design was implemented using KiCAD, along with all the necessary electrical components. The layout process became quite tedious to ensure proper functionality, while also trying to fit the entire module on a single breadboard for testing once printed. This was the first experience for many of our team members with PCB design and manufacturing, so the process provided a great opportunity to learn and dive deeper into these topics.

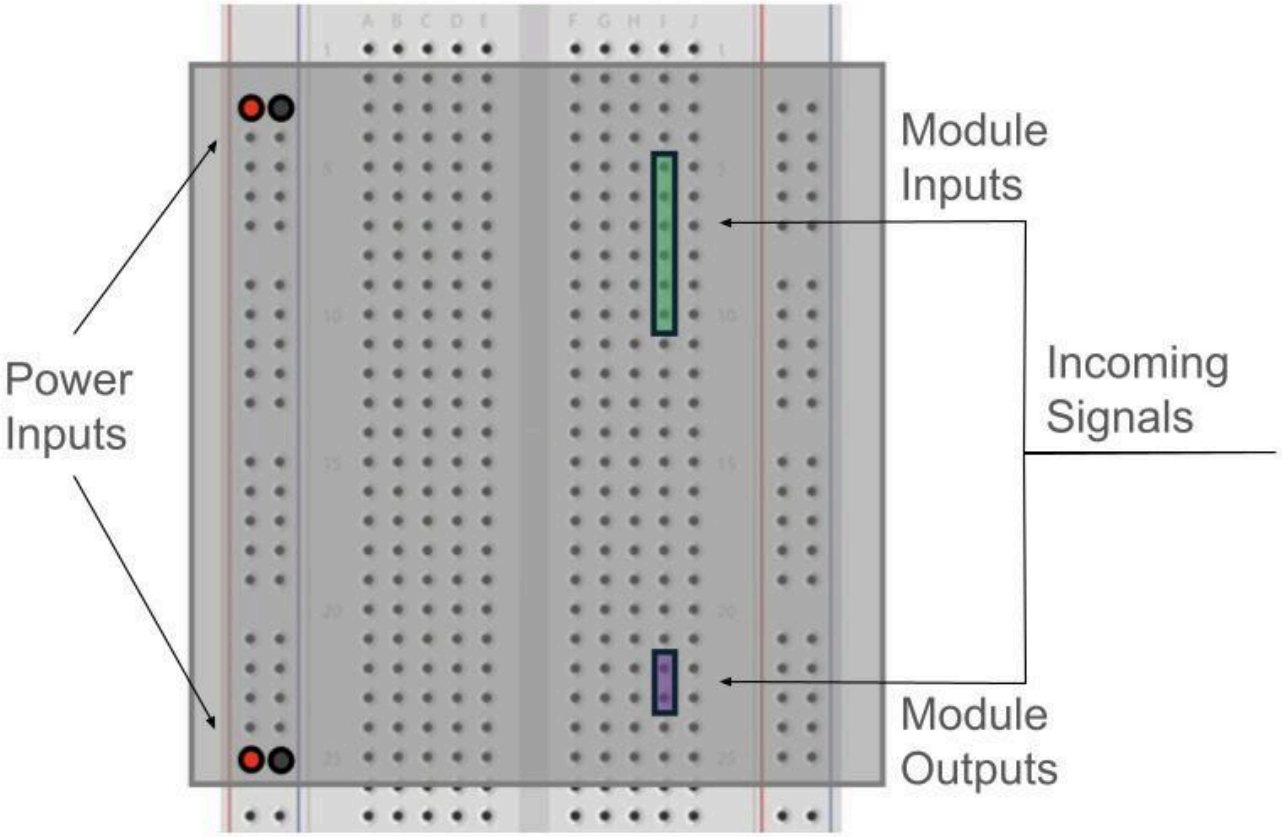


Figure 12. Add/Subtract Module Pin Layout

One feature that emerged from this design and implementation process was the placement of input and output pins to facilitate easy connection with a breadboard. For example, the VCC and GND pins for the FAS module are spaced out to fit perfectly on the power rails of a breadboard. Lining up the power pins in this manner will put all input and output pins on the Ith column, making for easy wiring of the inputs and outputs from a different module or breadboard coming in from the right. These design choices for the pins of the FAS module enable us to keep the overall area of the module small, allowing us to fit two FAS modules on a single breadboard. Moreover, with just three breadboards, four FAS modules can be connected and wired together, allowing all inputs and outputs to be supplied from the middle breadboard. Scaling up this principle for the full 30 FAS modules required will not only conserve the resources needed for implementation, but it is also an example of a design choice that maintains the modernized ABC's resemblance to the original ABC, which had panels of groups of these modules.

Once the modules were manufactured, assembly and testing could then begin. To start, the team only ordered 5 modules and enough parts for them. Knowing that the logic on the PCB modules and the logic on the breadboard modules were the same, testing would show any problems with the wiring or assembly of the board. After soldering the modules, testing commenced, and they were subjected to the same conditions and tests as those used for the breadboard circuits. In our testing, all input combinations were tested, and all observed outputs matched what was recorded by the given logic table. With the logical and electrical connections of these modules verified, the

remaining four FAS modules were soldered together by the end of the semester. All five modules, manufactured and soldered together, were tested and verified to be working as expected. As the team continues to work throughout the second semester of this project, it is expected that the remaining 25 modules will be manufactured and soldered in the same manner as the first five. Eventually, all 30 modules will be assembled together on the full modernized ABC.

6.2 Binary Conversion Mechanism

6.2.1 Functionality

The purpose of the binary conversion mechanism is to convert the decimal numbers read from decimal punch cards into binary numbers that the ABC can perform calculations with. In the original ABC, this mechanism was a large mechanical drum, which effectively contained a binary lookup table of all the digits from one to nine in their binary form, along with the binary forms of each of those digits up to the 15th decade (1, 10, 100, 1000, etc.).

Reading a number from a base 10 punch card then consisted of first reading the most significant digit, finding the corresponding ring of pegs on the conversion drum, and then adding that number to zero and storing the result on the CA drum. Each subsequent conversion was then added to the previous results to get the full binary representation of the decimal number. Since each base 10 punch card could contain up to five coefficients, the drum was capable of converting all these numbers at the same time in a parallel fashion.

For our design, we chose to implement the binary lookup table using an EEPROM chip that could be pre-programmed with the proper values across a predetermined encoding scheme for its address. The four most significant digits of the EEPROM's address were reserved for the specific decimal digit being converted, the next four digits were used to represent the decade scaling factor as a power of 10, and the last six bits were set to the same rotational address used for the memory drums. Our design will contain five of these EEPROMs to emulate the parallelism demonstrated by the original machine for this process.

6.2.2 Current Implementation Status

We have completed the logical design of this module and have done breadboard testing to confirm that we can accurately program the module's EEPROM chips and access the programmed values. We use a Python script developed by last year's senior design team to generate the binary lookup table as a binary file and program it on each EEPROM chip. The contents of each of the five EEPROM chips are identical, but all five are necessary to convert all five coefficients of the decimal punch card in parallel.

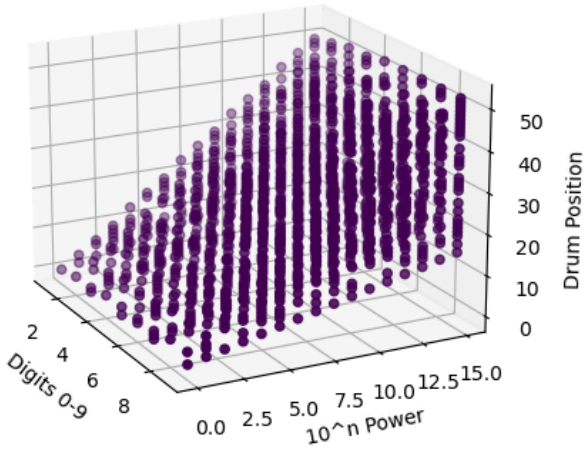


Figure 13. Visualization of Binary Lookup Table

We have also completed the KiCad schematic for the Binary Conversion Module.

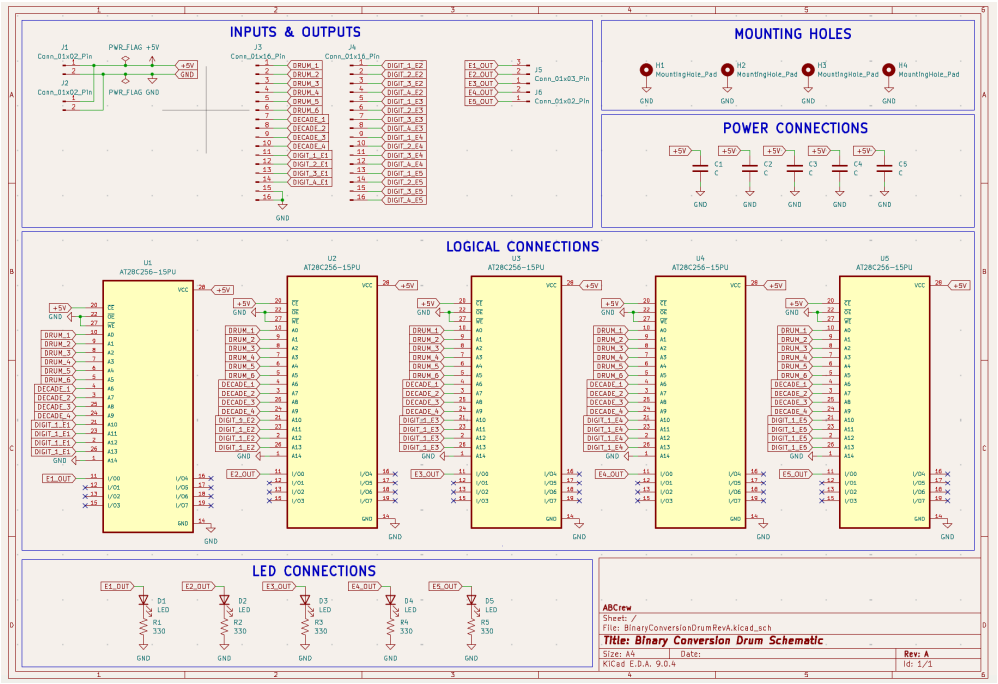


Figure 14. Binary Conversion Module Schematic

6.2.3 Remaining Implementation Steps

The next step in implementing this module is to design, fabricate, and test the PCB. We have discussed factors that will be important for the design of this PCB, such as optimizing routing paths and ideal placement of input and output pins for intuitive operation. Therefore, we are prepared to design the PCB with these factors in mind.

6.3 Decimal Output

6.3.1 Current Implementation Status

For this module, we have completed the logical design and prototyping phases of the design process. First, a Python simulation of the algorithm that was used to display decimal output as described in Berry's manual [4] was used to test several high-level component designs. Once a high-level design was selected that correctly displayed all outputs and was as historically accurate as possible, the design was translated into a hardware design with two separate modules: the seven-segment display module and the seven-segment selector module. The seven-segment display module was also prototyped on a breadboard to verify that the chosen counter chips work as expected.

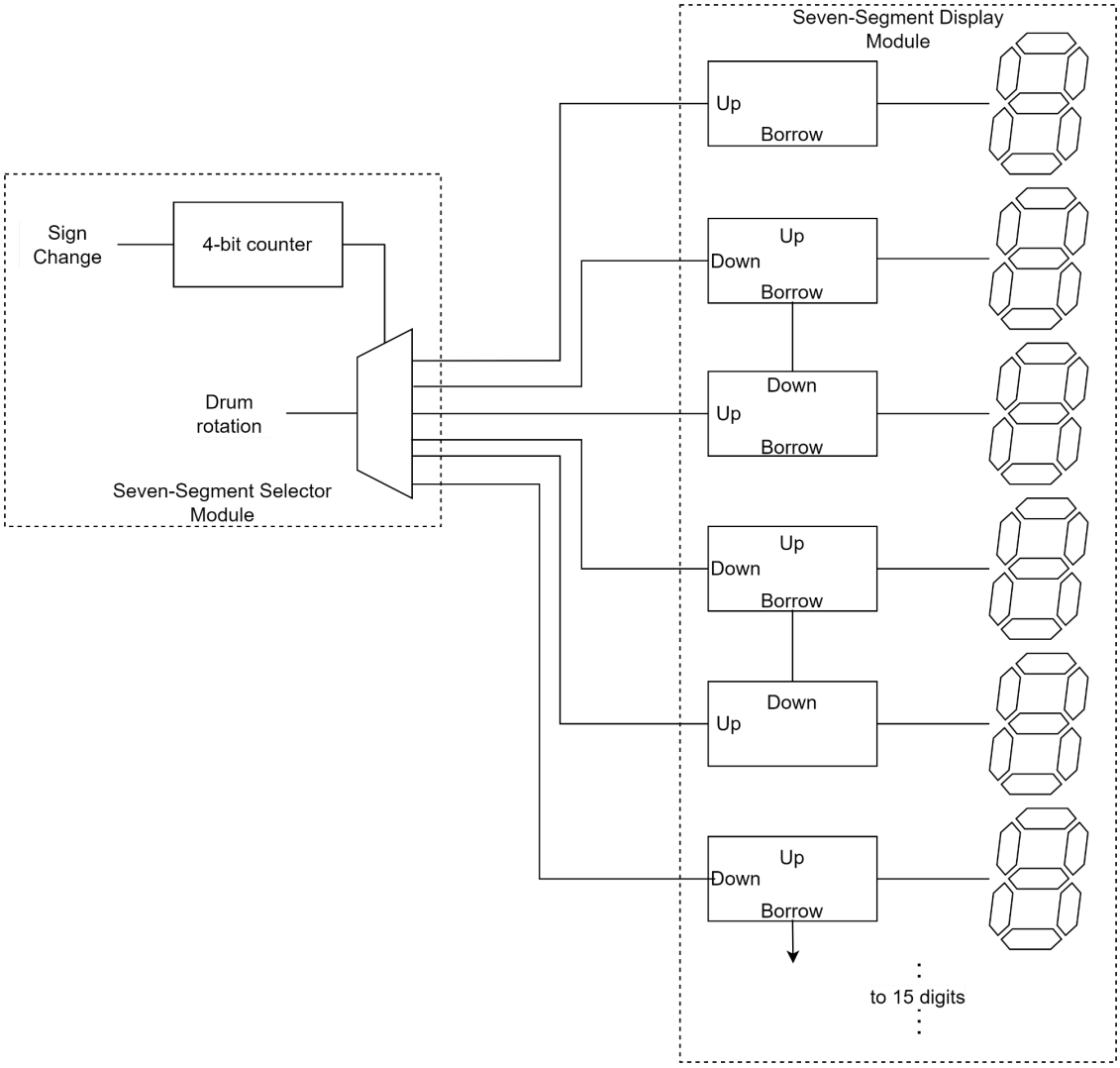


Figure 15. Decimal Output Module Design

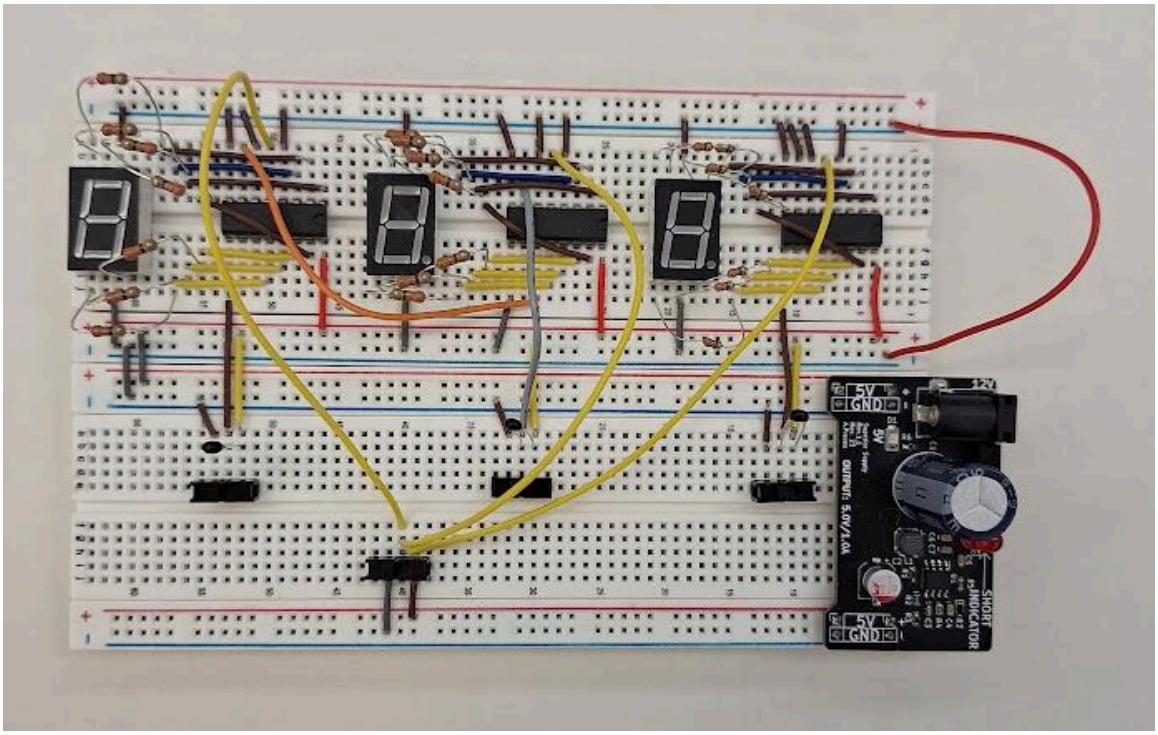


Figure 16. Seven-Segment Display Prototype

6.3.2 Remaining Implementation Steps

The next step in the implementation process for this module is to create a VHDL simulation of it to verify the hardware design. Once the hardware design is verified, it can be turned into a PCB and manufactured.

6.4 Self Clocking Program Counter

The clock circuit is responsible for generating all timing signals required by the modernized ABC and for synchronizing the control signals that drive computation, memory access, and shifting operations. In the original ABC, these time-based control patterns were provided mechanically by the rotating timing drum. In our system, the equivalent timing behavior is produced electronically through a dedicated clock-generation chain combined with a program counter that steps through the 120 control positions associated with a full ABC timing cycle.

The current clock design is composed of three major subsystems:

1. The selectable master clock source,
2. The Self-Clocking Program Counter (SCPC), and
3. The control-signal decoder that maps program-counter states to individual module enable lines.

Together, these components act as a drop-in replacement for the timing drum.

Selectable Master Clock

The clock source is implemented in top_clock.vhd (likely to be refactored/renamed prior to the finalized design), which accepts either the 2 MHz oscillator divided down through two cascaded 12-stage counters or an externally provided manual clock input. The selectable frequency taps range from approximately 120 kHz down to near 0.25 Hz, including the 120 Hz and 60 Hz taps that correspond closely to the ABC's operational speeds. The user may choose between automatic (continuous) operation and manual stepping, enabling fine-grained observation of intermediate system behavior.

Self-Clocking Program Counter

At this stage of implementation, the Program Counter subsystem is partially complete. The long-term design uses a custom 120-state counter (pc120.vhd) driven by the selected master clock. This component has been written and structurally verified in VHDL, but it is not yet fully debugged or validated in hardware.

Until debugging is complete, the SCPC uses the previously developed gate120 module to generate the 0–119 cycle. This fallback module has been stable in testbench operation and allows the remainder of the Timing and Control subsystem to be integrated and tested while work continues on the true pc120 implementation. Because the SCPC is responsible for advancing the timing position autonomously, it supports both continuous stepping and single-step advancement via the ADV input, reproducing the behavior desired of the ABC timing drum.

Control Signal Mapping

Once the Program Counter outputs a value in the range 0–119, the control decoder maps each count to the corresponding enable, read/write, shift, and routing signals required by the memory drums, add/subtract modules, and I/O subsystems. These mappings reflect patterns documented in the ABC reference materials and will eventually be stored in EEPROM to mirror the original drum's physical control tracks.

Current Status and Next Steps

The clock-generation path from oscillator to frequency selection is fully operational and has been verified on breadboards, but has not been verified after that point. The SCPC operates correctly in VHDL when driven by the temporary gate120 source, and its interface has been tested against simulated timing patterns expected for normal use. The remaining work consists of finishing the debugging and validation of the true pc120 module, replacing the temporary component, and formally validating the complete timing sequence against both VHDL waveforms and testing anything needed on a breadboard prior to fabrication.

Once complete, the clock circuit will supply all ABC modules with stable, historically faithful addresses and clock signals while also offering selectable speeds for demonstration, teaching, and debugging.

7. Ethics and Professional Responsibility

7.1 Areas of Professional Responsibility/Codes of Ethics

Area of Resposibility	**Definition**	**Relevant Item from IEEE Code of Ethics**	**Relation to Our Work**
Work Competence	Perform work of high quality with integrity	1.6: "...undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations"	Assign tasks inline with our individual competencies (i.e. hardware design, VHDL development) and ensure proper mentorship when someone is working on a task in an area they do not have previous experience in
Financial Responsibility	Design at reasonable costs, delivering a quality return on investment	1.1: "...to strive to comply with ethical design and sustainable development practices..."	Keep fabrication costs in mind when making PCB design decisions, and solder carefully to avoid wasting parts
Communication Honesty	Report work truthfully and understandably	1.5: "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others"	Document our work and designs accurately through this design document and the technical manual we are developing
Health, Safety, & Well-being	Minimize risks to the safety, health, and well-being of users, designers, and clients	1.1: "to hold paramount the safety, health, and welfare of the public..."	Ensure safe hardware design, designing at low voltages, and soldering solid connections
Property Ownership	Respect the property and ideas of others, giving credit where credit is due	1.5: "...and to credit properly the contributions of others"	Properly credit the work of previous groups who attempted to recreate the ABC, particularly the work of sdmady25-29,

			whose work and research we are building much of our design off of
Sustainability	Protect the environment and natural resources	1.1: "...to strive to comply with ethical design and sustainable development practices..."	Thoroughly test our designs before fabricating PCBs for them to minimize wasted PCB development and shipping
Social Responsibility	Produce a product that will benefit society	1.2: "to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies..."	Ensure our recreation implements the logical design of the original ABC as accurately as possible, even when it would be more convenient to change the original design, to ensure historical accuracy

Figure 17. Code of Ethics Table

An area on this chart where our team is performing well is in Communication Honesty, as we have been honest in all our documentation of designs and in our communications with the client. Our team could improve in the category of Property Ownership next semester by compiling our citations together in one place and further discussing with our project advisor how we can best credit the sdmay25-29 team in our final presentation and deliverables.

7.2 Four Principles

	Beneficence	Nonmalificence	Respect for Autonomy	Justice
Public health, safety, and welfare	Design safe circuits that operate at low voltages to promote user safety	Design does not support unsafe or harmful practices (e.g., sourcing parts)	We listened to our client and avoided making choices for them during design	Our open-sourced design promotes education equality
Global, cultural, and social	Educating others about computing history and logic, and logically recreating this device that has been lost to	Avoiding misinforming others by keeping our design logically accurate to the ABC	Enable others to recreate and customize our designs by documenting and sharing them well	Give credit to all sources used and work from previous teams

	history			
Environmental	Ordering parts intentionally and not fabricating more parts than we need	Avoid wasting PCB boards by designing intentionally and soldering carefully.	We will not directly disturb any natural ecosystems through our design	Implementation would not disturb one habitat over another
Economic	Design could promote the success of Iowa State University's admissions department, aiding the university financially	Design would not disrupt the economy	Users recreating this design could fabricate a different number of modules based on their budget	Implementation would not unduly financially affect some people

Figure 18. Ethical Principles Table

An area of this table that is important to our project is beneficence in the global, cultural, and social context, because our project's end goal of promoting computing education and recreating lost history aligns well with this principle-context pair. One area in which our project has a weakness is environmental nonmaleficence, as it utilizes natural resources without a positive or mitigating effect. The scale of our environmental impact is relatively minimal, though, so we believe it is outweighed by the overall positive impact of this project.

7.3 Virtues

7.3.1 Group Virtues

- Determination
    - At the very start of this project, all members understood the scope of the project and knew that hard work would be required to fully implement it. Understanding these requirements, all team members are committed to implementing the full ABC system and documenting each step of the design process. Providing regular updates and encouragement, as well as helping other team members, supports the virtue of determination within our team.
- Collaboration
    - Due to the large nature of this project and the complexity of each individual component, collaboration is essential. Collaboration, both internally as team members and externally with our advisor and first-hand resources, has been and will continue to be an important part of our project. Collaboration not only speeds up our design process but also ensures a better understanding of the project by all members involved. To support this virtue within our team, we hold multiple weekly meetings and

maintain open communication, which provides many opportunities for collaboration.

- Documentation
    - Upon reviewing the limited documentation and replicas of the ABC created in the past, it became essential for us to learn from these findings and document all steps in our design process, including research, troubleshooting, and implementation. This way, our team, as well as future teams, will be able to pick up where we left off, spend less time finding the same information, and more time unlocking new clues and findings about the original ABC. Viewing documentation as a required final deliverable and peer reviewing documentation will help promote this virtue.

7.3.2 Individual Virtues

Vance Arnold - Team Facilitator, Purchasing

- Virtue Demonstrated: Collaboration
    - Collaboration ensures that the project benefits from diverse skills and perspectives, and it prevents bottlenecks by distribution responsibilities.
    - I have facilitated communication between team members, coordinated purchasing needs, and ensured that everyone has the resources required to complete their modules. By anticipating partner needs and clearly framing progress, I've contributed to keeping the team aligned.
- Virtue Not Yet Demonstrated: Confidence
    - Confidence allows me to take initiative, trust my technical decisions, and contribute more fully to design discussions. It also helps reduce hesitation when facing unfamiliar tasks.
    - I plan to build confidence by studying the original ABC's design more deeply, asking clarifying questions when I feel unsure, and practicing schematic design with feedback from teammates. Over time, this will help me feel more capable and engaged in the technical aspects of the project.

Samuel Schroeder - Team Leader

- Virtue Demonstrated: Commitment to quality
    - This virtue is important to me because I value doing things right the first time, and I know that reliably delivering high-quality work reflects on my standards and creates better long-term solutions.
    - I have demonstrated a commitment to quality through my reviews and revisions of all course deliverables, such as status reports and design document sections. I have also demonstrated this virtue in my feedback to team members when designing and reviewing modules, which helps us create a high-quality end product.
- Virtue Not Yet Demonstrated: Clear and thorough documentation
    - The virtue of clear and thorough documentation is important to me because I believe it is my responsibility to adequately prepare those who come after me, ensuring that any knowledge or progress is not lost. Good documentation is the means to accomplish this.

    - To better demonstrate this virtue, I should take the time to document the outputs of each design phase, incorporate documentation into my workflow by always considering what I should document and how it should be documented, and peer-review my documentation to ensure it is clear and understandable.

Ryan McFadden - Scribe, Client Liason

- Virtue Demonstrated: Honesty
    - I have consistently been honest about our team's progress throughout my communications with our client and in my discussions with team members regarding my individual work.
- Virtue Not Yet Demonstrated: Clear and Thorough Documentation
    - When designing our Decimal to Binary Conversion Module, I was not thorough in documenting my design and the discovery work I conducted on the module. I plan to improve this by better documenting how the module works and how it should interact with other modules as I finalize its design.

Bryan York - PCB Lead

- Virtue Demonstrated: Hard Work
    - I believe that through designing and building the Full Adder/Subtractor module, I have demonstrated my ability to complete all tasks effectively and on time, while also working collaboratively with team members.
    - In review of the designs for the different modules I have been working on, I have made sure to implement changes efficiently while also ensuring proper logic connections and restoration of the original ABC.
- Virtue Not Yet Demonstrated: Effective Communication
    - When problems arose during component sourcing and reviewing documentation, I often found myself waiting until our next team meeting to address these issues with the full team. Doing this ultimately wasted precious time needed to complete this project.
    - To implement this virtue in the future, I plan to utilize our team's Discord server to raise these questions and solve any issues faster.

Gavin Bunker - Manual Lead

- Virtue Demonstrated: Critical Thinking
    - This is important to me as one of the key takeaways of a senior design project is to critically think about how to meet a requirement set forth by a client.
    - I have demonstrated critical thinking in various parts of the project, with the most prevalent example being the Clock Generation and Program Counter Module. I designed this module from scratch and continue to find new optimizations and improvements in its design as requirements evolve.
- Virtue Not Yet Demonstrated: Clear and Thorough Documentation

    - This is important as it will allow us to effectively turn over our work to others to either improve upon it or operate it. We will only be working with and operating this prototype until May, after which we will need to ensure that there is thorough documentation so others do not have to reverse engineer it, as we have had to do for many parts of last year's project.
    - This will be demonstrated after I finalize the clock generator module and write operating instructions for it, as well as explain how it functions in the technical manual.

Kyle Koch - VHDL Lead

- Virtue Demonstrated: Determination to learn
    - This virtue is very important, as for this project, we were thrown into an already started project and had to learn about it, while also picking up where the last group left off.
    - I had to spend a considerable amount of time learning about how the ABC works and how it is designed, as well as reviewing the previous group's work and picking up where they left off in the design process.
- Virtue Not Yet Demonstrated: Effective Communication
    - I spent a lot of time working on the background knowledge of the ABC and not a whole lot of time on some of the new deliverables that were expected of me. So, I need to ensure that I both complete these deliverables and communicate to my teammates what is needed and expected of me as a team member.
    - I plan to work on this virtue more next semester, as I will be less busy with other classes, and I now have the necessary background knowledge to move forward with the next steps of the project.

8. Closing Material

8.1 Conclusion

This semester, our team established the core building blocks of a modern recreation of the Atanasoff–Berry Computer. We completed and verified the Add/Subtract Module in both VHDL and hardware, developed the logical design and EEPROM structure for the Binary Conversion mechanism, prototyped the Decimal Output system, and implemented the early version of the Self-Clocking Program Counter. Together, these pieces demonstrate that our design workflow is reliable and that the ABC's logic can be recreated cleanly using modern components.

Our goal remains to produce a faithful, modular, and educational replica of the ABC. The results so far show steady progress toward that objective. The primary constraints this semester were time and PCB fabrication lead-times, which limited how many modules could be fully assembled and tested. Several components, such as the full 120-state counter and memory drums, will continue development next semester.

Looking ahead, earlier PCB staging, expanded simulation test harnesses, and increased parallel prototyping would help accelerate progress. With the project's foundation now in place, the remaining modules can follow the same pattern, bringing the completed modern ABC closer to reality.

8.2 References

[1] B. Rokos, D. Damman, G. Tersteeg, G. Nordling, L. Lee, "i281 Hardware" ece.iastate.edu, [Online]. Available: https://sdmay24-14.sd.ece.iastate.edu/. [Accessed Oct. 28, 2025]

[2] Burks, A. R., & Burks, A. W., The First Electronic Computer: The Atanasoff Story. Ann Arbor, MI: University of Michigan Press, 1988. [Accessed 17 Sep. 2025]

[3] C. Hand, Z. Scurlock, W. Mayer, N. Butler, P. Hurd, "Implement the ABC Computer with Modern Technology," ece.iastate.edu, [Online]. Available: https://sdmay25-29.sd.ece.iastate.edu/. [Accessed Oct. 28, 2025]

[4] C. Berry, Manual for the ABC, Iowa State Pyhsicis Department, 1941. [Accessed 20 Sep. 2025]

[5] IEEE Std 1149.1™-2013, *IEEE Standard for Test Access Port and Boundary-Scan Architecture*, IEEE Computer Society, May 13, 2013.

[6] IEEE/ANSI Std 91/91a-1991, IEEE/ANSI Standard Graphic Symbols for Logic Functions, IEEE Standards Board and American National Standards Institute, 1991.

[7] IEEE/IEC 61691-1-1:2023, Behavioural Languages – Part 1-1: VHDL Language Reference Manual, IEC/IEEE, Oct. 2023.

[8] IPC-2221, Generic Standard on Printed Board Design. Northbrook, IL: Association Connecting Electronics Industries (IPC), 1998.

[9] J. Gustafson, "Reconstruction of the Atanasoff-Berry Computer," johngustafson.net, [Online]. Available: www.johngustafson.net/pubs/pub57/ABCPaper.htm. [Accessed 28 Oct. 2025].

9. Team

9.1 Team Members

- Gavin Bunker
- Ryan McFadden
- Samuel Schroeder
- Vance Arnold
- Bryan York
- Kyle Koch

9.2 Required Skill Sets for Your Project

- **Digital logic and electrical circuit design** - Creating logic designs from available functional and technical descriptions will be essential for accurately recreating the ABC with modern technology. In addition to logical elements, our modules will also contain various other electrical components, like LEDs and seven-segment displays, for clearly showing the state and operation of the ABC.
- **PCB design with KiCAD** - All of our logical and electrical designs will need to be turned into PCBs for manufacturing and final assembly.
- **Soldering** - Once we receive our manufactured PCBs, individual components will need to be soldered on to complete the manufacturing process.
- **VHDL simulation** - Individual components and our design as a whole will be simulated in VHDL to verify correctness before PCB implementations of them are purchased.
- **Software development** - A user interface application will be developed that both accurately represents the ABC's user interface and simplifies some of the manual operations for easier demonstrations.

9.3 Skill Sets covered by the Team

Skill	Covering Team Member(s)
Digital logic and electrical ciruit design	Everyone
Soldering	Bryan York, Gavin Bunker
PCB design with KiCAD	Bryan York
VHDL simulation	Kyle Koch, Vance Arnold, Ryan McFadden, Gavin Bunker
Software development	Samuel Schroeder

Figure 19. Skills Table

9.4 Project Management Style Adopted by the team

For this project, we have adopted a modular waterfall project management style, where each component or module follows the waterfall model, but there may be several active waterfalls at any given time. For more information about the selected project management style, see section 3.1.

9.5 Initial Project Management Roles

For this project, we have two initial project management roles: project owner and project manager. Dr. Stoytchev, our client and faculty advisor, will serve as the project owner by specifying the broader goals and deadlines and prioritizing which modules should be developed first. Samuel Schroeder will serve as the project manager by performing more of the day-to-day coordination of team members and individual tasks.

9.6 Team Contract

Team Members

1) ____Gavin Bunker_____ 2) ____Ryan McFadden_____

3) ____Samuel Schroeder_____ 4) ____Vance Arnold_____

5) ____Bryan York_____ 6) ____Kyle Koch_____

Team Procedures

Team meetings will occur after our senior design class ends on Tuesdays for an hour in person at the library. Other weekly sessions will be held between all students and Dr. Stoytchev, faculty advisor, in the senior design lab in Coover 1301 at 2:00 pm on Thursdays. These times are subject to change on the agreement between all parties involved to allow for better attendance or as otherwise noted by the student team.

Any communication that needs to occur outside of these team meetings will be conducted over our preferred method of communication, Discord for student team questions and comments, or via email when needing to involve the Faculty advisor.

Decisions will be made on various criteria depending on the subject matter that the decision will affect. A breakdown of the hierarchy of this process is outlined below and agreed upon by all student team members by signing this contract at the bottom of the document.

Changes requested by the client shall be enacted and worked into the project

a. Decisions regarding an individual's or a subgroup's (ex, hardware or software) responsibility are decided by the members in that subgroup (not involving other subgroup votes).
b. A majority vote is needed for decisions requiring a group or subgroup, with tiebreakers being decided by the project leader.
c. Only members present may vote.
d. Any individual may call for a vote.
e. Every individual present must vote.

The student team member designated as the recorder will record all action items, business conducted, and any other important legislation brought up during any meetings. All notes will be taken and shared on Google Docs within our shared Google Drive folder so all students can view them.

Participation Expectations

A student team member may miss meetings when necessary, provided they give at least 12 hours' notice. Members are allowed a maximum of three absences without extenuating circumstances and cannot have consecutive absences without a valid reason.

All students are expected to meet deadlines set by the project leader. Each individual is responsible for communicating if they believe a deadline is unrealistic or anticipate a delay.

If a topic requiring a student's attention is mentioned in the preferred communication method (Discord), the student must respond within 24 hours.

Students are committed to fulfilling the requirements established by the client and the project leader as outlined.

Leadership

1. Student team members have all been assigned unique roles to fulfill, which will all benefit the group throughout this project. These students and their roles are outlined below:
    a. Team Leader: Samuel Schroeder
    b. Technical Manual Lead: Gavin Bunker
    c. PCB Lead: Bryan York
    d. VHDL Lead: Kyle Koch
    e. Client Liaison, Recorder: Ryan McFadden
    f. Team Facilitator, Purchasing: Vance Arnold

2. Strategies for supporting and guiding the work of all team members:

To maintain a productive learning environment and group dynamic, all criticism is to be given constructively. There will be no non-constructive criticism given without subject to penalties in the "Consequences for Not Adhering to Team Contract" section of this contract, or a majority of the members present at the given meeting see cause to inflict. Constructive criticism should be given when appropriate and promptly.

During team meetings, each team member will state their most significant contribution (1) since their last meeting, with a brief overview. If they made no noticeable progress, they will discuss why that was.

Collaboration and Inclusion

An outline of each student team member, along with the assets brought to the project team, is given below:

a. Gavin Bunker: Non-traditional student with over 7 years of active duty in the Navy and experience operating a nuclear reactor onboard an operational submarine.
a. Ryan McFadden: A variety of experience in multi-disciplinary engineering, incorporating mechanical design with software and electrical hardware. Good communicator and team worker.

b. Vance Arnold: Computer engineering student with hands-on experience in the hardware and software industry, with skills including coding, embedded, and general problem-solving. Prioritize communication and cooperation.
c. Bryan York: Electrical Engineering student who specializes in VLSI and Hardware design. I like to be hands-on and always ensure I work towards the end goal.
d. Kyle Koch: Computer Engineer with hands-on experience in electrical and Robotic engineering with a focus on Automation. I enjoy working on hardware design on circuits, and computer components.
e. Samuel Schroeder: Software Engineer with experience in software development for embedded software to high-level languages. I have an interest and expertise in cybersecurity, system architecture, and scripting.

Each member will have the opportunity to bring up subjects of their choice at the conclusion of the team meeting schedule and before concluding the meeting. Issues regarding collaboration or inclusion should be brought up for discussion by the relevant team member during the team meeting, as stated in the section "Participation Expectations" above. Any individual found to be in breach of the outlined contract, and any amendments voted and passed by the group, will have the following occur:

1. The group will take an initial vote on whether to warn the individual.
2. The group may also vote on whether to notify the instructor following a warning to the individual.
3. The instructor may not be notified unless the vote is unanimous or they have at least two warnings.

Goal-Setting, Planning, and Execution

Below is a list of goals set forth by the faculty advisor assigned to this project, the project outline document given to the student team members when signing up for projects at the beginning of the year, and other team and individual goals set forth by the student team members.

1. Team goals for this semester:
    a. Complete the project as outlined by the client on the schedule defined by the project leader.
2. Strategies for planning and assigning individual and team work:
    a. The team leader's main task will be coordinating the team and setting forth meeting plans. They are to assign individuals deadlines and timelines, and collaborate with them to ensure the deadlines are reasonable.
3. Strategies for keeping on task:
    a. The team facilitator will ensure that the meetings are time-effective. The project leader will follow up with individuals on project deadlines.

Consequences for Not Adhering to Team Contract

Any individual found to be in breach or in violation of the outlined contract, and any amendments voted and passed by the group, will have the following occur. Continuous violations of the agreement will be subject to the same penalties as the other student team members and faculty advisor, as needed, working their way through the given leadership structure outlined in the surrounding EE/CPRE 491/492 class structure until any problems are resolved.

1. The group will take an initial vote on whether to warn the individual.
2. The group may also vote on whether to notify the instructor following a warning to the individual.

3. The instructor may not be notified unless the vote is unanimous or they have at least two warnings.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Bryan York _____ DATE: 03 September 2025

2) Vance Arnold_____ DATE: 09 September 2025

3) Ryan McFadden_____ DATE: 09 September 2025

4) Samuel Schroeder _____ DATE: 09 September 2025

5) Gavin Bunker_____ DATE: 11 September 2025

6) Kyle Koch _____ DATE: 13 September 2025