Design Document

Client - Professor Alexander Stoytchev

Team 31 - Curriculum for the i281e

Team Members

Ethan Uhrich - Team Lead & Treasurer

Ariana Dirksen - Editor & Note Taker

Tessa Morgan - Graphic Designer & Webmaster

Gigi Harrabi - Client Interaction & Outreach Coordinator

Contact - sdmay25-31@iastate.edu

Website - https://sdmay25-31.sd.ece.iastate.edu/

Revised: 4/15/2025 V2

Executive Summary

This project is a series of labs that will be used as part of a curriculum for undergraduate engineering students. The goal of these labs is to introduce students to digital circuits and microprocessors. The labs are designed to bridge the gap between embedded systems, electronic circuits, and digital logic. Helping students better understand how digital logic is implemented in hardware and how the software interacts with the hardware. Our design supports labs focused on hardware, software and labs that tie the two together. Additionally, our design contains reusable testing circuits that allow students to verify their work in the labs. Our design allows students to complete each lab within three hours. When implementing our design, hardware-focused labs were prioritized to ensure resource availability. These labs build breadboard implementations of components present within the i281e processor. The software labs focus on software like KiCad as well as coding in assembly. The labs are also designed to be convertible into activities that can be used in outreach events. Such that participants with less background knowledge can complete the lab activities. Using our design we finished ten labs, five with a hardware focus, four with a software focus and one that ties the two together. We also created a mini project that builds upon two of our labs to expand student comprehension. The next steps for this project will be to continue any revisions in the labs to refine language and figures to enhance student understanding. Additionally, the lab curriculum we designed has room for an additional lab as well as a final project that can be expanded into their own activities for the lab curriculum to be used alongside a semester long class. There is a goal to build another PCB implementation for the i281e processor and document the development process.

Learning Summary

Development Standards & Practices Used

For standard circuit practices we maintained implementation standardization across our hardware and followed a uniform procedure in our documentation and lab manuals. We also use the following engineering standards for this project.

- IEEE 610.10-1994
- IEEE 610.13-1993
- IEEE 1515-2000

Summary of Requirements

- Undergraduate students must be able to complete labs in three hours.
- Labs must be at an appropriate level of difficulty for undergraduate students.
- Lab instructions must clearly communicate all information needed to complete the lab.
- Hardware lab implementations of the i281e components must remain compatible with the PCB implementation.
- Labs must be easy for the students to test and verify the accuracy of their implementations.

Applicable Courses from Iowa State University Curriculum

Courses
EE 2010: Electric Circuits
EE 2300: Electronic Circuits and Systems
CprE 2810: Digital Logic
CprE 2880: Embedded Systems
CprE 3810: Computer Organization and Assembly Level Programming

Table 1 - Applicable courses for the curriculum for the i281e processor project.

Acquired Skills

Throughout the development of this project the team has learned many new skills while working on design and implementation. Members of the team have learned how to enforce standardization practices, draw schematics and route PCB in KiCad, create simulations in TinkerCAD and Solder.

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Problem Statement

The goal of this project is to use open-source hardware and software designs for the existing central processing unit (CPU), operating system (OS), and simulator to implement a set of lab and outreach activities around the i281e processor. The i218e is a processor built completely on the basic digital logic that makes processors operate. This processor is a minimalistic design intended specifically as a teaching tool for the department.

These labs and activities will help students who have not used the processor adapt to and learn about how to create and work with processors like i281e. Each activity will be of appropriate complexity such that it can be completed by the average undergraduate student in a reasonable amount of time. Each must be tested by the team and documented with detailed step-by-step documentation. For some tasks, video tutorials are provided by the team.

The results will be used to support and enhance the curriculum in Computer Engineering and Electrical Engineering. These documents may also be used as educational materials for existing classes or to support future lectures and labs. These materials can be easily modified and used for outreach activities to get the next generation of engineers excited about computers.

1.2. Intended Users

Our project’s main users are students who are interested in embedded systems and are undergraduates in college. Secondly students in middle/high school that are interested in embedded systems at outreach events. Outside of students this project will also be used by TAs who will need the materials (lab documents and equipment) to conduct the lab or activity to the students. Our final user is the professor who will finalize a version of this course that will bridge the gap between digital logic, embedded systems, and computer architecture.

Our users' needs are different for each user. For example, the students need to understand simpler concepts of digital logic and embedded systems so that they can understand what is going on in the lab that they are executing. The needs for TA’s are a well-written lab document, a rough outline of the answer key and instructions for how to debug once a student is stuck. The professor’s need is for the activities to fill in the knowledge gaps so that students will succeed in future courses.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

Our requirements included physical, functional, and resource requirements as noted below.

2.1.1. Physical Requirements

Our team needed to develop at least 10 interactive lab-based activities that integrated both hardware and software components. Hardware components included breadboards, wires, ribbon cables, DIP switches, LEDs, resistors, capacitors, and a variety of digital ICs, including MUXes, adders, registers, EEPROMs, and seven-segment displays—all primarily sourced through ETG and DigiKey. Some labs also required a programmable EEPROM chip, and testing boards to simulate or verify outputs.

Lab activities were constrained by the physical resources and policies of the ETG-managed lab rooms. In certain labs, students may not be permitted to cut wires, requiring pre-cut jumper kits or alternative wiring strategies. Any issues involving lab hardware or installed software, including programming tools or circuit design applications, must typically be addressed by ETG staff. These logistical factors influenced how labs were set up, tested, and delivered to ensure consistency and accessibility across lab sections.

2.1.2. Functional Requirements

Each lab had to be feasible to complete within a 2- to 3-hour window by undergraduate students with basic digital logic knowledge. Labs had to cover a range of concepts, from basic logic gate design to EEPROM programming and CPU architecture fundamentals. Instructions had to be clear, testable, and suitable for both guided classroom settings and semi-independent outreach use. All activities needed to be designed to reinforce conceptual learning while enabling hands-on implementation, including pre-lab questions, breadboard assembly, and post-lab reflection or code uploads where applicable.

2.1.3. Resource Requirements

2.1.3.1. Lab 1

<table>
<tr><th>Quantity</th><th colspan="2">Item</th></tr>
<tr><td>1</td><td colspan="2">White 830-point Breadboard</td></tr>
<tr><td>Set of</td><td>Breadboard Wire Spools</td><td rowspan="3">Or, you could use a Pre-Cut Wire Kit</td></tr>
<tr><td>1</td><td>Wire Cutters Electronic Grade</td></tr>
<tr><td>1</td><td>Wire Strippers Electronic Grade</td></tr>
<tr><td>1</td><td colspan="2">CD74HC08E Chip (Four AND gates)</td></tr>
</table>

1	CD74HC32E Chip (Four OR gates)
1	CD74HC04E Chip (Six NOT gates)
1	4-Position DIP switch SPST (e.g, 5435640-2 or BPA04B)
1	5mm Red LED
4	1 kΩ THT Resistor
1	Breadboard Power Supply (e.g, YwRobot MB-V2)

Table 2 - Lab 1 materials.

2.1.3.2. Lab 2

<table>
<tr><th>Quantity</th><th colspan="2">Item</th></tr>
<tr><td>2</td><td colspan="2">White 830-point Breadboard</td></tr>
<tr><td>Set of</td><td>Breadboard Wire Spools</td><td rowspan="3">Or, you could use a Pre-Cut Wire Kit</td></tr>
<tr><td>1</td><td>Wire Cutters Electronic Grade</td></tr>
<tr><td>1</td><td>Wire Strippers Electronic Grade</td></tr>
<tr><td>1</td><td colspan="2">4-Position DIP Switch SPST (e.g., 5435640-2 or BPA04B)</td></tr>
<tr><td>1</td><td colspan="2">4-Bit Binary Ripple Counter (CD74HCT93E)</td></tr>
<tr><td>1</td><td colspan="2">BCD-to-Seven-Segment Decoder, Common-Anode (SN7447AN)</td></tr>
<tr><td>1</td><td colspan="2">BCD-to-Seven-Segment Decoder, Common-Cathode (SN74LS48N)</td></tr>
<tr><td>1</td><td colspan="2">Hex Schmitt-Trigger Inverters (SN7414N)</td></tr>
<tr><td>1</td><td colspan="2">Vertical 7-Segment Display, Common-Anode (LSHD-5601)</td></tr>
<tr><td>1</td><td colspan="2">Vertical 7-Segment Display, Common-Cathode (LSHD-5503)</td></tr>
<tr><td>2</td><td colspan="2">Push Button (TS02-66-70-BK-100-LCR-D)</td></tr>
<tr><td>2</td><td colspan="2">1 μF Electrolytic Capacitor (50YXM1MEFR5X11)</td></tr>
<tr><td>7</td><td colspan="2">330 Ω THT Resistor</td></tr>
<tr><td>4</td><td colspan="2">1 kΩ THT Resistor</td></tr>
</table>

1	Breadboard Power Supply (e.g., YwRobot MB-V2)

Table 3 - Lab 2 materials.

2.1.3.3. Lab 3

Quantity	Item	
1	White 830-point Breadboard	
Set of	Breadboard Wire Spools	Or, you could use a Pre-Cut Wire Kit
1	Wire Cutters Electronic Grade	
1	Wire Strippers Electronic Grade	
2	Quad 2-to-1 MUX Chip (SN74HCT257N)	
2	0.1 µF Ceramic Capacitor	
1	5mm Yellow LED	
1	330 Ω THT Resistor	
6	Connector for 16-position ribbon cable, DIP Header Connector (FDP-316-T)	
3	16-Conductor Ribbon Cable, 8 to 12 inches long (AWG28-16/G/300)	
1	Breadboard Power Supply (e.g, YwRobot MB-V2)	

Table 4 - Lab 3 materials.

2.1.3.4. Mini Project

Quantity	Item	
2	White 830-point Breadboard	
Set of	Breadboard Wire Spools	Or, you could use a Pre-Cut Wire Kit
1	Wire Cutters Electronic Grade	

1	Wire Strippers Electronic Grade	
2	Quad 2-to-1 MUX Chip (SN74HCT257N)	
2	0.1 μF Ceramic Capacitor	
1	5mm Yellow LED	
27	330 Ω THT Resistor	
1	Breadboard Power Supply (e.g, YwRobot MB-V2)	
3	CBL RIBN 16COND 0.05 GRAY 300	
6	PLUG FLAT CABLE IDC 16P 2.54MM P	
1	5mm Red LED	
7	5mm Green LED	
2	Dip Switch SPST 8 Position Through Hole Slide	
1	Dip Switch SPST 2 Position Through Hole Slide	

Table 5 - Mini Project materials.

2.1.3.5. Lab 6 & 7

Quantity	Items	
3	White 830-point Breadboard	
Set of	Breadboard Wire Spools	
1	Wire Cutters Electronic Grade	
1	Wire Strippers Electronic Grade	
4	CD74HCT283E Chip (4-bit adder)	
2	SN74HCT257N Chip (Quad 2-to-1 MUX)	

1	SN74HCT273N Chip (8-bit Register)
7	0.1 uF Ceramic Capacitor
1	5mm Red LED
7	5mm Green LED
2	5mm Yellow LED
10	330 Ω THT Resistor
2	16-Conductor Ribbon Cable, 8 to 12 inches long (AWG28-16/G/300)
4	Connectors, 16-position ribbon cable DIP Header Connector (FDP-316-T)
1	Breadboard Power Supply (e.g. YwRobot MB-V2)

Table 6 - Lab 6 & 7 materials.

2.1.3.6. Lab 8

Quantity	Item
2	White 830-point Breadboard
Set of	Breadboard Wire Spools
1	Wire Cutters Electronic Grade
1	Wire Strippers Electronic Grade
2	8-Position DIP Switch SPST (418117270908)
2	2-Position DIP Switch SPST (418117270902)
2	EEPROM Memory IC (AT28C64B-15PU)
1	7-Segment Display, Common-Anode (LSHD-5601, 157119S12801)
1	7-Segment Display, Common-Cathode (LSHD-5503, LDS-HTC514RI)
16	330 Ω THT Resistor
18	1 kΩ THT Resistor

1	Breadboard Power Supply (e.g. YwRobot MB-V2)

Table 7 - Lab 8 materials.

2.1.3.7. Lab 9

Quantity	Items
2	White 830-point Breadboard
Set of	Breadboard Wire Spools
1	Wire Cutters Electronic Grade
1	Wire Strippers Electronic Grade
1	2 MHz crystal oscillator (ECS-100A-020)
1	Push Button (TS02-66-70-BK-100-LCR-D)
1	DFF (SN74LS377N)
1	Dial (NR01105ANG13)
3	0.1 microF Electrolytic Capacitors
1	1-position DIP switch (732-3831-5-ND)
1	Quad 2-to-1 MUX (SN74HCT257N)
1	5mm Red LED
3	1 kΩ THT Resistor
2	4-Bit Binary Ripple Counter (CD74HCT93E)
1	Breadboard Power Supply (e.g., YwRobot MB-V2)

Table 8 - Lab 9 materials.

2.1.3.8. Other Resources

Quantity	Items
1	i281 + i281e Manuals

1	Previous Senior Design Team's Documentation
1	i281 Simulator
1	KiCad Version 8.08

Table 9 - Resource requirements.

2.1.4. Requirements for Outreach Events

Set up educational events by contacting outreach organizations. We arranged an outreach event with the Iowa NASA space consortium. Our team facilitated the event which consisted of making a version of the first lab manual that was understandable for middle schoolers and some highschoolers, so the lab used needed to be well fitting to their estimated knowledge. Our users, in this case, were boy scouts that were interested in earning their engineering merit badge at the event. Our first lab content fulfilled part of the requirements, and we created additional material to fill the gaps.

2.1.5. Additional Requirements and Constraints

Project constraints primarily stemmed from the availability of hardware and software resources. All hardware components needed to be sourced through ETG or DigiKey, which limited substitutions or upgrades during development. On the software side, all tools used in the labs needed to be freely available. Students will use KiCad for circuit design and XGpro for EEPROM programming. Both programs require installation privileges on lab computers, and students installing them on personal devices must do so independently. However, XGpro is only compatible with Windows, which restricts its use on macOS and limits programming tasks to Windows-based systems provided in the lab.

Each lab includes four required deliverables: a lab instruction manual, a student lab report, a grading rubric, and an answer key. Labs are evaluated out of 100 points by the course instructor or teaching assistants. Students are expected to complete at least 80% of the pre-lab material and demonstrate an understanding of the lab's purpose and connection to course concepts. Each lab is designed to be completed within a 3-hour session. Students exceeding this time are encouraged to ask for assistance. Teaching assistants will be trained by the instructor to provide guidance without directly giving answers and are expected to uphold academic integrity by reporting suspected plagiarism or cheating.

2.2. Engineering Standards

Engineering standards are important since some, if not most, engineering products depend on the quality of the user's life. While browsing the IEEE Standards website, there are products that are related to healthcare and life sciences that if they do not follow the IEEE standards and other codes, will affect the user's life negatively.

IEEE Standard Glossary of Hardware Terminology (IEEE 610.10-1994)- Describes official definitions related to computer hardware relating to computer architecture, computing storage, processors and components [5].

IEEE Standard Glossary of Computer Languages (IEEE 610.13-1993)- Describes the names and definitions of computer languages, and their historical significance [6].

IEEE Recommended Practice for Electronic Power Subsystems: Parameter Definitions, Test Conditions, and Test Methods (IEEE 1515-2000)- Describes testing methods for electronic circuits and systems [4].

The first standard relates to the specific terms that will be used in the i281e processor from an educational perspective. Meaning that when creating the labs any terms used needed to be elaborated on taught to the participants during the lab activities. The second one is also relevant because in the software labs, we incorporated the use of a computer language on the simulator, for example, and the use of that standard is important. The third one is important because our project involved building breadboard circuits and testing them.

Our team agrees that these standards best apply to our project. These are the main three that apply to our project as the project is intended for internal use within the department rather than commercial use. Additionally, the project is based around a custom design and needs to fit the CPU, course curriculum, and faculty constraints.

3. Project Plan

3.1. Project Management/Tracking Procedures

The team followed an Agile-inspired approach, organizing work around weekly sprints. We met twice a week for stand-up meetings, during which we provided status updates, identified blockers, and set short-term goals. Each lab served as a milestone, and progress was tracked through a combination of internal status reports and printed lab drafts reviewed and marked by our client, Professor Stoytchev.

We defined the requirements for each lab in collaboration with our client, then moved through stages of circuit or software design, part sourcing, testing, documentation and revision. Initial lab manuals were drafted and refined iteratively based on feedback, focusing on improving lab flow, clarity, and educational content. While some of the labs were tested by students outside the team, most feedback came directly from our advisor during review and revision cycles. Task tracking was maintained through status reports, team meetings and version control on the lab manuals.

3.2. Task Decomposition

Due to the nature of our project, the task decomposition was natural. Broadly, we had milestones for each finalized lab. However, each lab had its own milestones for completion which consist of designing the lab, implementing the lab, testing the lab activity, writing the lab manual and report, and revising the lab materials. Due to the contrasting activities of the labs some required a longer design and a research phase. While others required longer building and testing phases. An example of a heavy research lab is the EPROM labs, as our team had a learning curve for programming or using EEPROMs. We also needed to spend additional time on learning how the i281 simulator expected input code, loading assembly programs on FPGA boards and understanding KiCad well enough to create labs using it.

3.3. Project Proposed Milestones, Metrics, and Evaluation Criteria

The primary milestones for this project were the completion of ten fully developed lab manuals, each representing a deliverable aligned with a core instructional goal. Sub-milestones for each lab included the initial draft of the lab activity, completion of hardware or software implementation, internal team testing, revision based on advisor feedback, and delivery of a finalized version including the instruction manual, grading rubric, answer key, and lab report template. These stages reflect the subtasks identified in Section 3.2 and are refined iteratively as part of our weekly sprint process.

Progress on each lab was measured by tracking the time required to complete its circuit construction, code development, documentation and revisioning. Each lab needed to be scoped such that students at the sophomore or junior level could reasonably complete it within a 2-to-3-hour lab session, assuming standard lab conditions. While the labs were designed to be accessible and appropriately challenging for that audience, evaluating student learning outcomes was outside the scope of our project. Our client, Professor Stoytchev, is responsible for integrating the labs into a broader course and curriculum.

Feedback from our advisor served as the primary evaluation method throughout development, with emphasis on clarity, technical correctness, and alignment with course objectives. A student tester also reviewed selected final drafts to provide usability feedback and identify potential student difficulties. Metrics for evaluating each lab included internal time-to-complete for prototyping and documentation, number of revision cycles before approval, and client confirmation of lab readiness.

3.4. Project Timeline/Schedule

September-October (Red - Research): We researched the i281 processor by looking at past documents and lectures provided by our client and previous i281e groups.

October-November (Pink - Lab 3): Our team built two copies of the i281 MUX on breadboards, in groups of two since the labs and outreach events are going to be completed in groups of two, tested the MUX and went through multiple iterations of the documentation process.

November-February (Orange - Lab 6 & 7): Our team built and tested two versions of the program counter. They required a clock for the register, so we added a debouncer to the testing circuit. The initial materials for the first draft were also created including several diagrams and the visuals for the activity section of the lab. This lab was also divided into two parts since we started working on it and determined that it would take students longer than three hours to complete.

November-December (Yellow - Lab 1): We put together lab 1 which walks students through building a 2-to-1 multiplexer setting up groundwork which leads into later labs. This lab also went through multiple stages of documentation.

December-January, March- April (Green - Lab 8): We started the EEPROM lab which includes programming an EEPROM to store the addresses for a 7-segment decoder. We built the first prototype for the circuit, attempted programming the EEPROM and started the testing stage of this lab. When we resumed work in March, the software for the lab was set up and we found a script to run the EEPROM. Due to some issues with the EEPROM software, the EEPROM chip set up for this lab took longer than expected.

January-February (Light Green - Lab 2): This lab is focused on debouncing techniques with basic hardware components such as LEDs, resistors, and reading datasheets. The output of a 4-bit up counter on a 7-segment display with the help of a BCD-to-seven-segment decoder.

February-April (Turquoise - Lab 4, 5 & Mini-Project): This subset of labs is an introduction to KiCad and the labs students need to complete for the mini-project. These labs were done at the same time as they will be the set up for the mini project for the students taking the course. These labs are an introduction to schematics, routing PCBs and using KiCAD. The lab activities and documentation were the focus of part of the team as we pushed to finish them before spring break. Additionally, we finished the two corresponding tutorial videos and documentation explaining the mini-project soldering steps.

February-April (Blue - Lab 9): This lab requires students to build a clock circuit and focuses on real-time programming and registers. The clock circuit acts as a frequency divider consisting of a crystal oscillator which sends a signal through the circuit. When the input is at a certain frequency, the output will be different. This lab went through multiple implementations and occasionally had to be put on hold while waiting on parts.

March-April (Purple - Lab 10): This lab introduces students to assembly level programming. For this lab we created multiple basic C problems and created implementation in assembly of the same functions. Once we decided on this method for lab instruction the documentation came together quickly.

April (Lilac - Outreach): The outreach event was for the BSA Merit Badge University. The event fulfilled an electronic badge requirement for the boy scouts. The outreach event consisted of the team guiding the students through the lab, along with the edited manual that was suited for that age group. Before the event, we converted our documentation for lab 1 to work for it and created supplemental materials for the requirements not sufficiently met with the activity.

April-May (Light Pink - Lab 11): This lab implements a basic version of the video game flappy bird. This game runs on the simulator and uses various components of the CPU and knowledge from previous labs. As we were approaching the end of the semester, we focused on finalizing the labs and revisioning them over developing new labs.

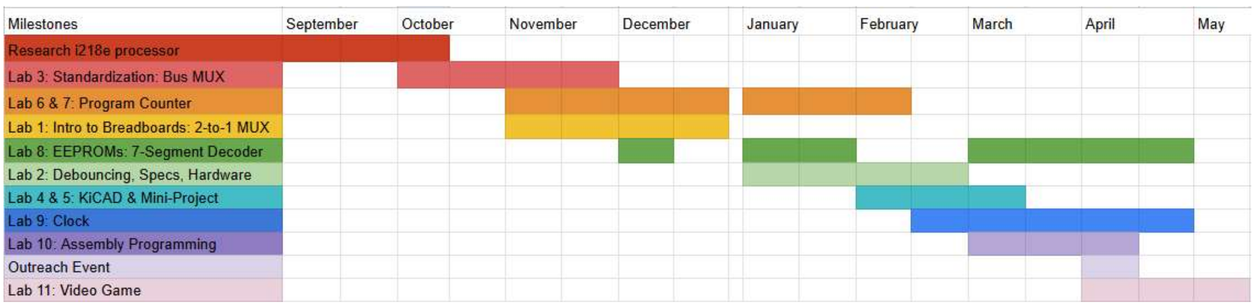


Figure 1 - Gantt Chart

3.5. Risks and Risk Management/Mitigation

This project faced multiple risks across hardware development, lab design, and process coordination. The most significant and recurring risk involved shifting requirements and evolving expectations throughout development. Since each lab functioned as independent projects, we refined best practices—such as formatting, testing protocols, or documentation standards—only after completing the early labs. This resulted in backtracking, reworking, and significant context-switching, which reduced efficiency and limited the number of new labs we could concurrently develop. This occurred frequently and ultimately shaped our sprint planning. We mitigated this by standardizing templates and reusable components mid-project and applying lessons learned to the labs still in progress.

Another key risk was hardware availability. Early in the project, part shortages and delayed shipments required us to substitute components or redesign circuits to accommodate what was available. To mitigate this, we began source parts early in the lab drafting process, using generic logic chips whenever possible, and selecting components with multiple compatible substitutes to reduce future disruption.

Tool limitations also posed risks. For example, XGpro— which is used to program EEPROMs—only runs on Windows, which restricts its use to lab machines and other Windows systems. This created bottlenecks in testing and prevented some team members from working on EEPROM-related labs outside scheduled lab times. While no cross-platform alternative exists for our specific programmer, we mitigated this by reserving in-lab testing time and consolidating EEPROM programming tasks into specific sprints.

Finally, there is the risk that the CprE 3710x course, for which these labs were developed, does not receive full departmental approval or that some labs are not adopted. While this is largely outside our control, we mitigated this by maintaining alignment with department expectations, integrating feedback from Professor Stoytchev, and ensuring that the labs met academic and technical standards for junior-level coursework.

3.6. Personnel Effort Requirements

Our Labs were created in four stages. First, we designed the lab. Next, we implemented the activity using hardware, software, or both. Once implemented, we tested any hardware or programs. Then, we document the expected process for the students and create lab manuals, lab report templates, answer key and grading rubrics. Through the course of creating these labs the labs increase in complexity. Some labs, such as Lab 4, required additional research. Others, like Lab 6 & 7, needed more time for documentation as the processes described in the labs are more time-consuming.

Below is a table with the time each lab was estimated to take. The times listed are cumulative across all members of the team. The design section includes gathering requirements for the overall lab, creating parts lists, ordering components, and deciding on circuit layouts. Implementation consists of building circuits, writing code, and programming components with various software, including later re-working after receiving feedback on previous deployments. Testing consists of all stages of testing circuits, individual components, final testing on students, and gathering feedback from the client. Documentation includes documenting the overall process, assembling lab drafts, generating diagrams/visuals, and taking pictures. This covered multiple drafts and stages of editing.

Task	Design	Implementation	Testing	Documentation
Lab 1: Intro: MUX	30 minutes	30 minutes	15 minutes	4 hours
Lab 2: Debouncing and Hardware	30 minutes	1 hour	30 minutes	1 hour
Lab 3: Bus MUX	1 hour	3 hours	1 hour	8 hours
Lab 4: Intro to KiCAD	1 hour	3 hours	1 hour	6 hours
Lab 6 & 7: Program Counter	2 hours	6 hours	9 hours	12 hours
Lab 8: EEPROM Programming	30 minutes	2 hours	2 hours	6 hours
Lab 9: Clock	1.5 hours	5 hours	3 hours	10+ hours
Lab 10: Assembly	2 hours	6 hours	4 hours	10+ hours
Lab 11: Video Game	3 hours	10+ hours	4 hours	10+ hours
Lab 12: Peripherals	2 hours	10 hours	6 hours	10+ hours
Final Project	3 hours	10 hours	6 hours	15+ hours
Test Circuit	30 minutes	3 hours	4 hours	1 hour

Table 10 - Personal effort estimates by task breakdown.

The next table is the time it ended up taking to create each lab. The design, implementation and testing columns reflect what was described above. Drafting is the time it took to create the rough draft for the lab manual and report. Revisioning is the time it took to make all of the changes necessary to make the final drafts for the lab manual and reports. Point of Contact is the team members that were responsible for working on the lab.

Task	Design	Implementation	Testing	Drafting	Revisioning	Point of Contact
Lab 1: Intro: MUX	30 mins	30 mins	30 mins	1 hour	4 hours	Ariana & Ethan
Lab 2: Debouncing	2 hours	3 hours	30 mins	3 hours	4 hours	Tessa & Gigi
Lab 3: Bus MUX	1 hour	3 hours	1 hour	8 hours	7 hours	Ariana, Ethan, Tessa, & Gigi
Lab 4: Intro to KiCAD	2 hours	1 hour	30 mins	3 hours	2 hours	Ariana
Lab 5: Routing a PCB	1 hour	2 hours	30 mins	2 hours	3 hours	Ethan
Lab 6 & 7: Program Counter	2 hours	10 hours	5 hours	4 hours	6 hours	Ariana, Ethan, Tessa, & Gigi
Lab 8: EEPROM Programming	1 hour	9 hours	2 hours	5 hours	6 hours	Tessa
Lab 9: Clock	4 hours	10 hours	2 hours	4 hours	9 hours	Gigi
Lab 10: Assembly	2 hours	7 hours	2 hours	3 hours	2 hours	Ariana & Ethan
Lab 11: Video Game	1 hour	3 hours	1 hour	2 hours	2 hours	Ariana & Ethan
Mini Project	30 mins	2.5 hours	1 hour	1 hour	1 hour	Ethan

Table 11 - Actual personal effort by task breakdown.

3.7. Other Resource Requirements

As mentioned in section 2.1.3 each lab requires a set component list to complete. For the full list of resource requirements refer to section 2.1.3. Generally, the main components we used for our hardware-oriented labs include:

- Breadboards
- Assorted Resistors
- Assorted LEDs
- Assorted Wires
- 0.1 uF Ceramic Capacitors
- CD74HCT283E Chips
- SN74HCT273N Chips
- CD74HCT86E Chips
- CD74HCT377E Chips
- CD4078BE Chips

- Ribbon Cable
- EEPROM Programmable Chip
- SRAM Memory

When completing the labs, we also utilized existing materials from the previously compiled GitHub repository containing PCB schematics for each CPU component. We also referenced the physical breadboard implementation to review circuit layouts as well as instructional videos and articles on various topics relating to the labs to make sure we understood the topic we covered. In addition, we used the online i281 simulator and FPGA boards to simulate and test programs for the software labs.

4. Design

4.1. Design Context

4.1.1. Broader Context

Our team was tasked to take open-source software and design of the i281e processor and implement them into labs. Undergraduates who are around the sophomore or junior level in relevant majors at Iowa State University are the main users who would benefit from the labs. In addition, anyone who may be interested in our software can replicate our labs, with the list of parts provided in our lab reports and the lab manuals created. The professor(s) and TAs teaching the CPRE 371x course and outreach coordinators will also benefit as they will be provided with the material and the answer keys.

Area	Description	Examples
Public health, safety, and welfare	Our stakeholders who are the TAs, professors, and outreach coordinators would likely be concerned about the tools and improvement of teaching effectiveness. For students, the project leads to improved learning outcomes through more engaging and effective teaching methods, as well as increased access to academic resources. This creates a more supportive learning environment that fosters student success.	Students will understand and build the knowledge gap between embedded systems (software and hardware) and computer architecture, which includes a range of basic digital logic to more complex components in a RISC-V processor.
Global, cultural, and social	Our project, a community-based engineering education program, reflects the values and aspirations of the local cultural group by prioritizing community engagement and representation. By involving professors, TAs and receiving feedback from students in the planning process, we ensure that the program aligns with the community's emphasis on education as a means of empowerment. Additionally, by recruiting mentors from similar cultural backgrounds, we provide relatable role models who inspire students to pursue careers in engineering fields, ultimately fostering a sense of pride and ownership within the community.	Students and other interested participants will be capable of accessing our open-source software and replicating the labs to further understand our course.
Environmental	One concern is wasting wire connections in electrical kits since they are often too short or long. A way to combat this is to customize and cut specific wires with certain measurements.	This method would decrease the use of plastic and decrease disposing of unneeded wires.
Economic	Our project is at low cost. A singular electrical lab kit costs at least about $15. But,	The resources that are provided are at relatively low cost to make those labs but are at no cost to

	at outreach events, students can participate at no cost.	participants who are willing to learn about the labs. Individuals who come across our open-source software will need to provide their own lab kit. In addition, university students who register to attend CPRE 371x will be provided with a prepared lab kit by the university.

Table 12 - Design considerations.

4.1.2. Prior Work/Solutions

Past teams have worked on the i281 CPU. The teams worked on breadboard implementations [8][10], a simulator implementation [9] and a PCB implementation [8]. The gap our team fills is an educational implementation, which is a combination of software and hardware labs. An advantage our team provides is a further and more detailed understanding of the i281 CPU architecture and lab activities that benefit whoever wants to understand the i281 CPU. Another advantage is our activities are focused on the unique CPU that doesn't have competing designs, if anything, like it on the market.

4.1.3. Technical Complexity

This project involved the design and development of multiple hardware and software-integrated labs, each targeting a distinct concept within computer architecture—such as the ALU, program counter, and seven-segment display decoding. Each lab functions as a standalone module, requiring its own implementation, testing, and documentation. The combination of digital circuit construction, EEPROM programming, and assembly-level simulation contribute to the overall technical depth of the project.

While the project is built with an educational focus, this increases its technical complexity. Communicating complex hardware behavior in a way that is accessible to a range of audiences requires not only clear instructional design but also a deep and accurate understanding of the underlying technical systems. Designing labs that are scalable from outreach activities for K–12 students to university-level instruction demands careful abstraction, troubleshooting, and iterative refinement. This teaching-oriented framing requires a solid grasp of both the "how" and the "why" of each component, adding a unique layer of complexity to both the design and implementation phases of lab creation.

4.2. Design Exploration

4.2.1. Design Decisions

We created a timeline for what a semester of the class would look like in terms of labs. This led to us designing a lab curriculum for fifteen weeks of time. It was proposed that there would be a lab for each week with the exception of three. Those three would account for the semester break and the final project. Also, it was decided that we would focus on hardware labs in the beginning as we wanted plenty of time for ordering parts if there were issues with implementation.

In the first half of our project, we focused on completing four labs. The introduction lab, which required a custom circuit to introduce students to breadboards, wiring, and standardizations. The MUX and Program Counter (PC) labs, which implemented components used in the actual i281e CPU. As well as an EEPROM lab which was originally the first of two EEPROM labs. The second of which was scrapped to allow for a more complete lab curriculum.

Because we focused on the hardware labs and ordering parts, we discovered that the EEPROM (W27C512) chip, used in the existing design, is no longer in production. Thus, we came up with a replacement that would meet as many specs as possible and decided on the AT28C64B-15PU which has less storage but is cheaper and maintains an adequate speed unlike other replacement options.

When implementing our labs there were many decisions that came with how we would go about teaching the material covered in our labs and all the different material we would cover. For each of the labs there were unique challenges in their implementation.

4.2.1.1 Lab 1

Our first lab was created to allow students to re-familiarize themselves with concepts they should already know. Our labs were made with the assumption that students had previous knowledge of breadboards, resistors, capacitors, and LEDs from an electrical perspective. This lab introduces them to digital chips and breadboard implementations built from a digital logic perspective. While this lab was completed relatively fast to implement, creating thorough documentation to ensure that the students received adequate knowledge and background information took some time. During implementation, the most significant decision was the order to place the chips to ensure minimal wires overlapping within the 2-to-1 MUX. As the biggest challenge for developing labs was time, it was decided that one team member would focus on creating this lab's documentation while the others moved on to other labs. This lab also had challenges with the documentation's graphic quality, and the graphics had to be edited multiple times to ensure compliance during revisioning.

4.2.1.2 Lab 2

This lab covers counting, decoding, and debouncing. The main goal was to introduce students to basic sequential logic with a 4-bit ripple counter and display its output using both common-anode and common-cathode 7-segment displays. A key decision was to include both manual input using a DIP switch and automated input using a CD74HCT93 counter, allowing students to see both direct logic control and timed state transitions. One of the design challenges was introducing hardware switch debouncing early in the curriculum. We chose to implement a hardware debounce circuit using a Schmitt trigger inverter and polarized capacitor to smooth out the clock signal. This required careful documentation and a strong focus on explanation in the lab manual, as students new to analog components may struggle with the behavior.

4.2.1.3 Lab 3

This lab is the first lab in which students build a component of the i281e processor, the multiplexer. This lab was the first lab we implemented; thus, it was in this lab we learned that having the entire team work on the same lab took too much time for development and wasn't going to allow us to accomplish the ten labs we set out for. However, the decision to split the team in two allowing two labs to be developed at once completely happened during lab 6 & 7. This lab also led to the creation of a tester circuit for testing our hardware labs and the outline for lab manuals.

4.2.1.4 Lab 4

This lab is presumably the student's first introduction to KiCad. It covers how to create schematics in KiCad and all the necessary steps to prepare for the next lab. The biggest decision for this lab was how to best teach students to use KiCad. This process involved consulting many videos and informational websites on how to use KiCad. It was decided that both a lab manual and tutorial video were needed to ensure students had all the tools for understanding. Additionally, as a part of the pre-lab students are expected to look at the KiCad article from the website build-electronic-circuits [11]. This ensures that students have all the resources that they may need to complete the lab.

4.2.1.5 Lab 5

This lab is the second lab involving KiCad and was set to teach students an important skill, how to route a PCB. The design challenges with this one were how to best describe routing since there are so many different ways of routing the same board with a lot of different things to consider. We ended up going with a simple design starting with the easy, short connections and then creating paths around those connections for the more complicated connections. Another thing we needed to consider was testing the circuit. After they build and solder the circuit, they need to be able to test it so included in the lab was how to get the PCB to fit into a breadboard which would allow for very easy testing of their design.

4.2.1.6 Lab 6 & 7

This lab brought interesting challenges with it because it was the most complicated circuit, we were going to have the students implement. At first, we were planning to give the students a basic circuit diagram and have them create their own design and layout for the circuit on their breadboards. However, we quickly realized that it would be too complicated for an average student to get done in a single lab period. Even our own implementations took several hours to build and countless more were spent testing and troubleshooting the resulting circuits. To help remedy this we decided to give the students a set circuit design to build like we had done in previous labs and also extend the lab to be over the course of two lab periods rather than one. This would allow the students ample time to implement and test the circuit rather than being rushed to complete the circuit in only a single class period. This decision benefits the project as a whole because it allows us to mitigate a problem in our design with the complexity of the circuit by giving the students more time to implement the lab. This takes into account the needs of the students since we had difficulty with implementing them in only a couple of hours and it's assumed that it would take the typical student longer. As such building that circuit in only a couple of hours would be a difficult thing to ask of a student in a single lab period, let alone the TA that would need to be constantly helping them with troubleshooting.

4.2.1.7 Lab 8

This lab introduces students to EEPROM programming using Python and the XGpro software. The primary challenge in designing this lab was balancing the complexity of memory addressing and EEPROM behavior with students' limited background in scripting. We decided to have students write a Python script that generates a binary file with display encodings for two modes: counting mode (repeating 0–F hex digits) and game mode (where each bit corresponds to a segment on the display). The EEPROMs are programmed outside the circuit using the XGpro software, which required us to provide extensive instructions due to the Windows-only nature of the software and the extra setup steps. Another challenge was conveying the idea of memory-mapped outputs in a static decoder without overwhelming the students with unnecessary abstraction. We chose to emphasize the directness of memory-to-display mappings and carefully broke down the circuit-building process to support this.

4.2.1.8 Lab 9

This lab is about building an analog clock frequency divider. One of the few challenges was that there were a couple of versions of the circuit's schematic. So, deciding on which schematic to use was initially based on what the desired outcome of the circuit should be. Another piece that was a part of the decision was based on which parts in the schematic matched the most with the circuit. In addition, a final part of that decision was due to whether or not the schematic's logic matched with the problem statement, which is to output different frequencies from the crystal oscillator source. Another challenge faced was after implementing the original circuit successfully, the circuit was required to be optimized. The optimization was to replace the comparator portion, which is consistent of a few parts, with a single component. That optimization made the instructions of the lab much simpler with the same goals for the lab being fulfilled.

4.2.1.9 Lab 10

This lab was designed as the students first introduction to the i281 assembly and possibly assembly programming in general, so it was important to us that we teach the concepts in blocks so that a student can understand the code that they are coding. This would make it so they could potentially do something with assembly as their final project. One of the sections that we wanted to include was one around a sorting algorithm. Unfortunately, the i281 simulator includes a couple of sample sorting algorithms, all of the common iterable algorithms like bubble sort, insertion sort and selection sort. We didn't want the students to have direct access to the code they would be implementing so it was decided that we'd try to implement a recursive algorithm like merge sort. This brought on its own problems since as it was recursive a lot more data needed to be stored. And the simulator only had 15 bytes of data storage as well as four registers which made it extremely difficult to implement. We did get something to almost work but found that it was ninety plus lines of assembly instructions, which given that the i281 could only store 63 lines of assembly. It was decided that we would have them just implement the merge section of the merge sort algorithm, which we did get working within the constraints of the i281 processor. However, it was very complex assembly code and took several hours to get working. With this in mind we decided to change the merge section into a section on conditionals since it was something important to the merge code. We believe this will greatly improve the student experience with the course since they are not expected to implement complicated code as a single section of a lab that's teaching them assembly.

4.2.1.10 Lab 11

For lab 11, it was initially decided that we would have the students create a video game purely in digital components using hardware. It was going to be a version of rock Paper Scissors. However, we ended up with more hardware labs than software labs. So, this lab became a software lab, and we had to rework the design. Since this lab got picked up after the assembly lab, we decided that the game would be programmed for the simulator. It was quickly decided that we needed a simple game as the processor had all the limitations mentioned in 4.2.1.9. Our client suggested the game Flappy Bird, and we decided it was a good idea. However, after implementing the game, it was discovered that even a simple movement cycle of a rock on top followed by a rock on bottom took 22 lines of code without any input code added. After adding the input code and the collision detection, the code took 58 lines and reused the four registers multiple times. While the code was manageable, the game couldn't be any more complicated. The students already had to figure out how to reuse registers to comply with the processor specifications. Any more added difficulty would be challenging for students to complete without giving them step-by-step instructions, which would contradict the purpose of teaching students.

4.2.2. Ideation

When deciding what labs to do, we came up with two ideas. This included an introductory lab, visualizing flip-flops, rock paper scissors, memory storage on breadboards, Tic-Tac-Toe, a maze using joysticks, and nim. Based on these options, we further fleshed out what labs Professor Stoytchev had in mind and prioritized the labs based on that. This introduced us to the idea of EEPROM labs. Additionally, we decided a good, interactive lab would be Rock Raper Scissors, which would be more software-heavy to offset the first labs. It can also be visualized using the work done in previous labs, including the EEPROM 7-segment display code. With that game in mind, we had most of our labs filled up and planned to continue further fleshing them out. It was only a matter of figuring out where in the planned agenda of the course each idea fit and what labs agreed with what the client expected.

4.2.3. Decision-Making and Trade-Off

After discussing our first lab timeline with the client, it became clear that most labs were already scoped out, including the introduction lab, a MUX, PC, ALU labs, two EEPROM labs, device drivers, and a final project. This left us with two labs of our choice, including a game, which Professor Stoytchev had suggested in the proposal, for which we decided on the Rock Paper Scissors game since it will be straightforward to represent on the CPU and can incorporate the previous 7-segment display lab. Over the course of the project, the timeline was adjusted. We added KiCad and soldering to the planned labs as another two-part lab and the mini project. One of the EEPROM labs was removed along with the ALU lab as they were too complicated. The device drivers lab was changed to RAM Chips and Buffer. Overall, this led to the creation of our final lab timeline based on a typical Fall semester:

- Intro to Breadboards: 2-to-1 MUX
- Counting, Decoding, & Debouncing

- Standardization and Connectors: Bus MUX
- Introduction to KiCAD (Start of Mini Project)
- Mini-Project: Routing a PCB
- Program Counter Part 1
- Program Counter Part 2
- EEPROMs: Program 7-Segment Decoder
- Clock Circuit + Final Project Proposal
- Assembly Level Programming
- Video Game in Assembly
- RAM Chips + Buffer
- Thanksgiving Break
- Final Project Pt 1
- Final Project Pt 2

4.3. Final Design

4.3.1. Overview

Our final design is a set of 10 hands-on lab activities that help students learn how computers and processors work by building and testing real circuits. These labs are designed for students studying computer engineering, but they are written so that someone with basic experience in digital logic can follow along. Each lab focuses on a different part of how a computer operates, allowing students to connect what they've learned in class to real hardware.

The labs include key components such as breadboards, switches, LEDs, logic chips, and programmable memory (EEPROMs). For example, in one lab, students build a program counter circuit that keeps track of instructions, similar to what happens in an actual CPU. In another lab, students use a Python script to create a file that programs an EEPROM chip, which then drives a 7-segment display. Some labs are simpler and focus on building logic circuits like multiplexers (MUXes), while others introduce skills like designing a printed circuit board (PCB) or writing assembly instructions for a basic CPU.

These labs are tied to concepts from three core computer engineering courses:

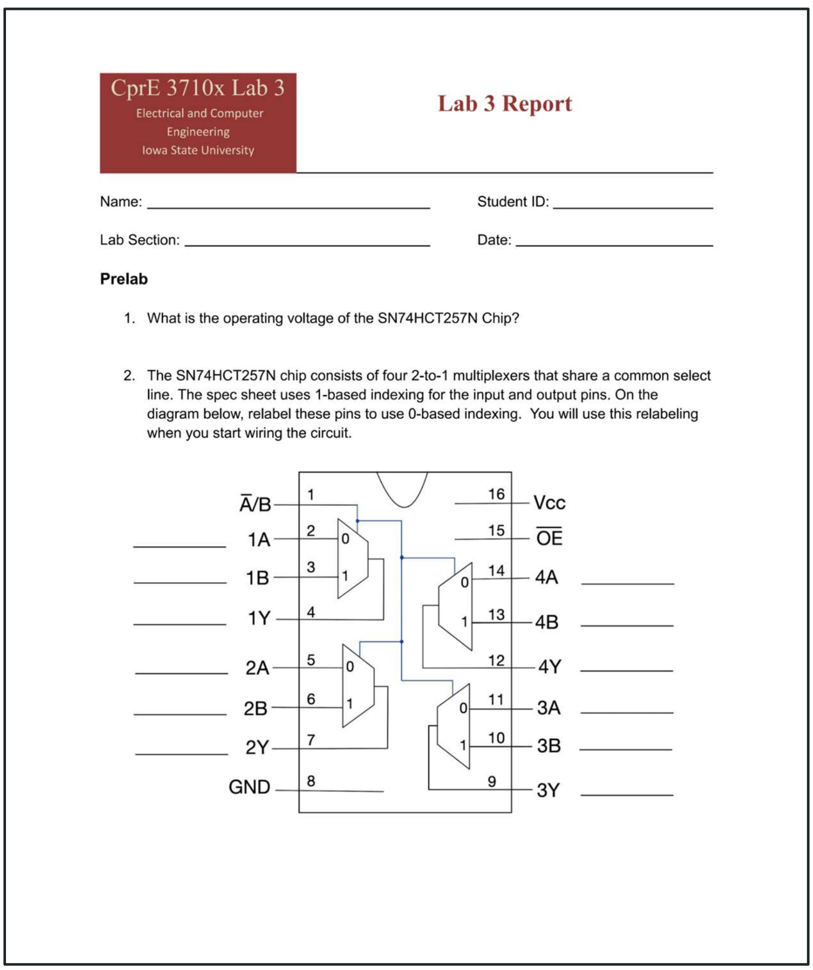
- CprE 2810: Digital logic, including gates, flip-flops, counters, and registers.
- CprE 2880: Embedded systems, including hardware-software interaction.
- CprE 3810: Computer architecture, such as how processors perform operations.

Each lab includes a circuit to build, a visual diagram to help students wire it correctly, and a written manual to guide them through the process. The goal is for students to not just read about computer systems, but to see how the pieces work together in practice. A simple visual (see Figure X) can help show how different labs connect to topics like input, logic, memory, and output.

4.3.2. Detailed Design and Visuals

The key components of a lab are the lab manual, the lab report template, the answer key, the grading rubric and any additional resources such as provided skeleton code or tutorial videos. The main component, the lab manual, has the following sections:

- Prelab: Basic activities and questions for students to answer that set a baseline of expected knowledge the student is expected to know before they start the lab.
- Objectives: The purpose of the lab.
- Parts List: This section includes all the materials that are needed for the lab.
- Background: Information that the student needs to know for the lab that the student might not already know. This section helps students answer pre-lab questions, along with providing information on what kind of components and concepts will be used in this lab.
- Activity: Step by step instructions on how to complete the lab. Or a general list of checkpoints the students need to accomplish to complete the activity. As well as a problem statement for what they need to do and which documents and hints they can refer to for additional information.
- Testing: How to test whatever implementation the students completed within the lab.



CprE 3710x Lab 3
Electrical and Computer Engineering
Iowa State University

Lab 3 Report

Name: ______________________ Student ID: ______________

Lab Section: ______________________ Date: ______________

Prelab

1. What is the operating voltage of the SN74HCT257N Chip?

2. The SN74HCT257N chip consists of four 2-to-1 multiplexers that share a common select line. The spec sheet uses 1-based indexing for the input and output pins. On the diagram below, relabel these pins to use 0-based indexing. You will use this relabeling when you start wiring the circuit.

Figure 2 - First page of lab 3 report template.

4.3.3. Functionality

Our design operates as a series of labs attached to an experimental class that students who are interested in computer engineering can use to attain a better understanding for the previous classes they took if they are a college student at Iowa State University. For other students who are not familiar with this, the labs can serve as a basic understanding of computer engineering overall.

4.3.4. Areas of Challenge

Limited resources and requirements change as building each lab like some circuit parts would be discontinued. In addition, our design challenges included filling the team's knowledge gap to catch up with the information that past teams left behind. After the labs were fully implemented, another design challenge was revisioning the lab drafts which needed to fit into the requirements.

The requirements that needed to be met were that the team needs to fulfill the client's expectations as well as the TA and the students' needs. The client's expectations are the team needed at least ten fully functioning and implemented labs along with the revised version of the labs. The labs needed to be well-documented and along with a prelab to give a clearer idea of what the lab should entail. The instructions for the lab should consist of a background about the concept of the lab's topic, the premise of the activity with sufficient instructions and a testing section that is tailored for the lab. In addition, the TA should be able to give the students sufficient hints to the students from reading from the answer key.

For combating limited resources, our team found alternative replacements for those circuit components. In addition, asking the advisor and using other resources to fill the knowledge gaps were satisfied. With feedback from the users, the labs were adjusted, which were to simplify the lab instructions and clarify on the testing section.

4.4. Technology Considerations

The leading technology our design uses is breadboard circuits. The advantage of using breadboards is that they are easy to test circuits and designs inexpensively and quickly. This allows us to change the original circuit designed for the i281e processor. This also allows us to prototype what we ask the students in our design to do, where we can get an idea of how long a circuit takes to build on a breadboard. The main weakness of breadboards is that parts are not connected with soldered joints and tend to come loose if not handled with care. In addition, parts placed on a breadboard are generally quite fragile and can become damaged without an easy way of testing them out. In addition, breadboards themselves can break from time to time and cause issues that way. We decided to go with breadboards because they provide the most accessibility to physically building circuits.

The second technology we are using in our design is KiCad. KiCad allows us to design circuits that could later be printed onto PCBs. The main advantage of using KiCad is that the previous team's designs were made in KiCad so no translation of circuits into a different program needs to be done. In addition, it makes it very easy to see what values of the chip's inputs are and which are outputs. One weakness of KiCad is that it can't standardize the schematic to follow which side the pins are on, so, in KiCad, all inputs are to the left and all outputs to the right. This makes it difficult to visualize how the circuit will look on a breadboard. In addition, the images produced from KiCad are not easy to read with small numbers, so it can take some educated guesses to figure out which pin the diagram meant. Overall, we went with KiCad because of its previous use on the team, and possible future lab portions will use KiCad to show PCB design.

The third technology we have used in our design is TinkerCad. This website allows us to generate cleaner images of each circuit than we can achieve on actual breadboards so students who follow the pictures have a clear view of where the circuits connect. As mentioned before, the strength of TinkerCad is its clarity in circuit wiring compared to just an image of the circuit. A weakness of TinkerCad is that it doesn't have all the chip layouts we need, but we spent some time custom-made them. Overall, TinkerCad was chosen because it generates clear images that make the labs as easy to follow as possible.

The fourth technology we have used is XGpro. The software programs instructions into the EEPROM by writing the data and configuring it into a suitable memory module to perform the desired task. In addition, Xgpro allows the bytes written into the EEPROM to be erased and reprogrammed and see how an EEPROM is currently programmed. However, this program has difficulty detecting which programmer model is connected and is not compatible with macOS. Despite this, Xgpro is a very straightforward program for students to use and interact with.

The final technology that we used was the i281 simulator. The simulator was chosen as it is the only visualization tool that can run assembly code on the i281 processor, as the students will not have direct access to the i281e processor PCB. The challenge with the simulator is that it expects the assembly code used in programs loaded on it to follow a specific format and restricts the instructions that can be used on it to a limited set of instructions. Overall, once students can understand the format and with the assistance of the Java compiler built for the simulator, it's a relatively straightforward tool.

5. Testing

For testing our project with the uniqueness of its requirements traditional testing outlines do not apply to our project. Not included are integration and system testing. These were excluded because our project as a whole is a set of labs and activities that are all around a similar unifying theme; they are not designed to be directly integrated together. Because of this quirk, system testing does not make much sense in this setting either because that kind of testing would be done when the labs are run in conjunction with the course itself which is outside of the scope of the project. Our testing plan included unit testing when designing labs for both hardware and software labs, interface testing, regression testing, acceptance testing and user testing.

5.1. Unit Testing for Hardware

For each of our breadboards our main methods of testing included multimeter voltage testing and unit testing with our test board. With multimeter testing we probed voltages at different points in the circuit allowing us to check if a component worked as expected.

In addition to multimeter testing, we also tested circuits used in labs with a test board. Our test board consists of

- 2 sets of 8-switch DIP switches which allow us to provide two 8-bit inputs.
- 3 sets of 2-switch DIP switches which allow for up to 6 various select lines.
- 2 sets of debounce switches to allow for a debounced input to a clock input.
- 8 LEDs that are used to display the output of the circuit.

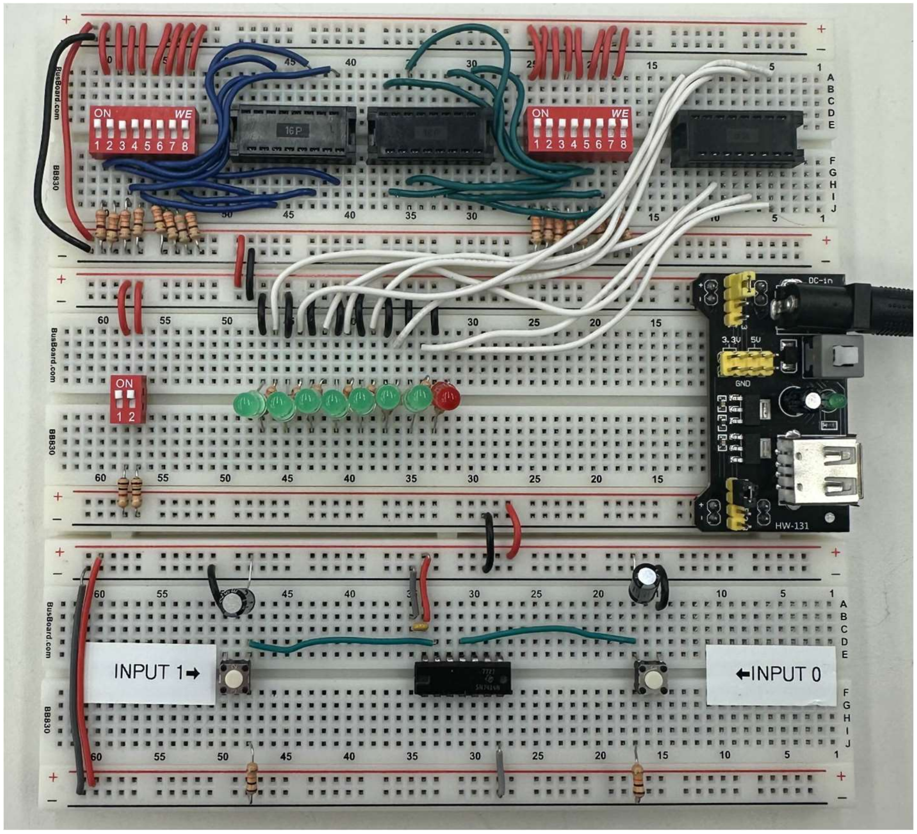


Figure 3 - Breadboard Mockup of the test board.

5.2. Unit Testing for Software

For our labs primarily involving software, such as our KiCad or assembly labs, we had a different process for unit testing. The unit testing is that once an activity is designed and created, we can run it against simulators in the case of the assembly lab or check it against past designs and the digital logic for the case of our KiCad lab. These kinds of tests allow us to first see if what we are asking is possible and second if it is at a reasonable difficulty for a student to complete within a lab period.

5.3. Interface Testing

The main way of interfacing used in our designs is through our labs and through our test board described above in 5.1 and 5.2. The labs are how we demonstrate our design ability and testing for the labs was done through a rigorous revision process. As well as testing with students who have the experience expected of a student taking the class. With the testing board we were able to test if the breadboard units designed will interface properly with the i281e processor.

5.4. Regression Testing

We ensured that new labs and breadboard don't break old functionality by following a strict set of standardization when it comes to interfacing between components and following a set guideline when creating labs to make sure that even if different people worked on the same part of a lab a student wouldn't be able to tell. Most of these standards we are following were laid out by the previous i281e senior design groups. The standard we use when creating labs was initially outlined by us but further refined by our client to better fit his needs.

5.5. Acceptance Testing

In the creation of circuits and labs whenever we have a mockup or a draft, we involve the client by having him go over our designs or drafts and leave feedback. With the circuits this mainly involves checking if the placement of the components makes sense, if standards are being followed and if the circuit is clean enough to be understood. In terms of the labs, our client and us work together in red pen to make changes that the client wants to the lab whether it be adding more sections or making edits. This ensures that the quality of the product is where the client wants it and that all requirements are met both from what the client wants and what is outlined for us to do in the project proposal.

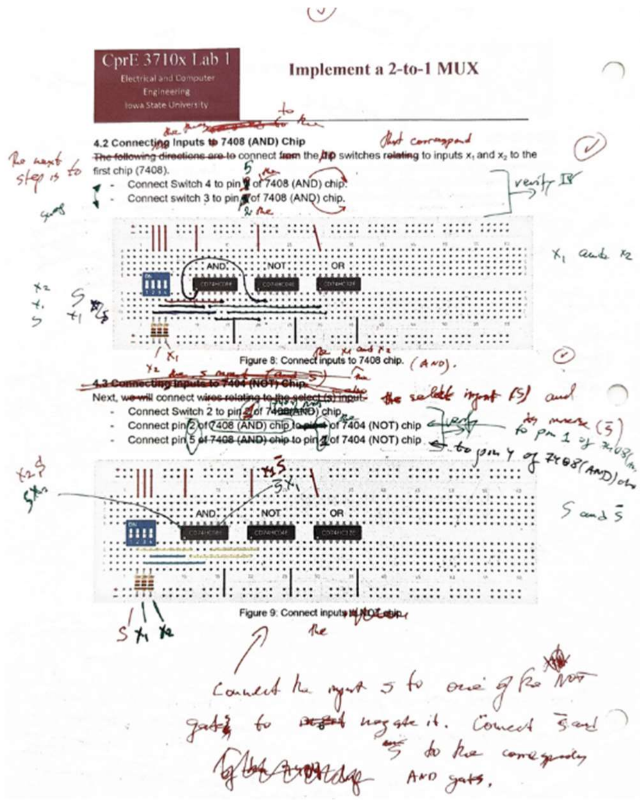


Figure 4 - Sample first round of edits from lab 3.

5.6. User Testing

Our design had three forms of user testing: student testing, outreach event and client editing reviews. The first form of user testing, student testing, allowed us to get feedback on our activities from someone who would be at about the right level of knowledge for what is expected of the students who would be taking the class. This allowed us to get a time estimate on how an average to more experienced student would approach the lab as well as identifying difficulties that students could face. The outreach event gave us insight into how a student with little to no knowledge of a lab would approach our activities. This is important because when combined with the student testing it gives us a comprehensive estimate for the full range of knowledge and experiences of our labs. The final user testing we did was client editing reviews. Whenever we got a lab into a finished state with the rough draft. We gave a copy to our client, for review. This gave us insight both into how the client thinks the activities should be run and outlined but also put a second pair of eyes on the product that can catch possible mistakes in the process of our design.

5.7. Results

The results of our testing internally and also externally with our outreach and student testers brought up issues that would need to be addressed. Our internal testing and reviewing of our activities and documents found many grammatical and image errors that were not caught in the creation of our activities. This was by far one of the most common results of testing by having our client review the labs and activities we created. This alone had almost every lab going through at least two or three rewrites to get them into a state that is acceptable.

The student testing helped us determine spots in our labs and activities where the directions were unclear or could be interpreted in different ways. This was things like mentioning to a new student that the color of the wire doesn't matter and they will all function the same regardless of color. Another thing pointed out was inconsistency with color in our actual photos of our circuit and those we generated using tinkercad. Although these kinds of oversights and mistakes were not thought of initially, the testing revealed them to be a consideration to take into account as it would help possibly save our users time with these kinds of inconsistencies caught.

The outreach event allowed us to observe students completely new to circuit building and computer engineering as they try to tackle our circuit. This allowed us great insight into how a student may approach our lab, one of the things that was evident was when supervised the students would go step by step, but unsupervised would just build the circuit shown in the final image. This told us that as important as the intermediary steps are, the final circuit was the most important one to get right since it would be the one most referenced by students.

The testing allowed us to meet our primary user's needs, our client, by including him directly in the design and testing process. This allowed for an agile-like testing cycle where we would get constant reviews and feedback on our work so that we could have the best final product that our client would be satisfied with.

6. Implementation

6.1. Implementation Process

For our implementation process we approached our labs one at a time. First, we selected and designed a lab from our designed timeline and curriculum. Then we order the parts, if any, needed to implement any hardware components for the lab. Once the parts arrive, we begin implementing our designed activity for the lab. For implementation we work through the activity that the students will be completing in the lab and build any testing components or code any programs needed. This lets us gauge roughly how long the activity will take to complete. It also informs us of potential challenges students may face if we make the lab according to our initial design.

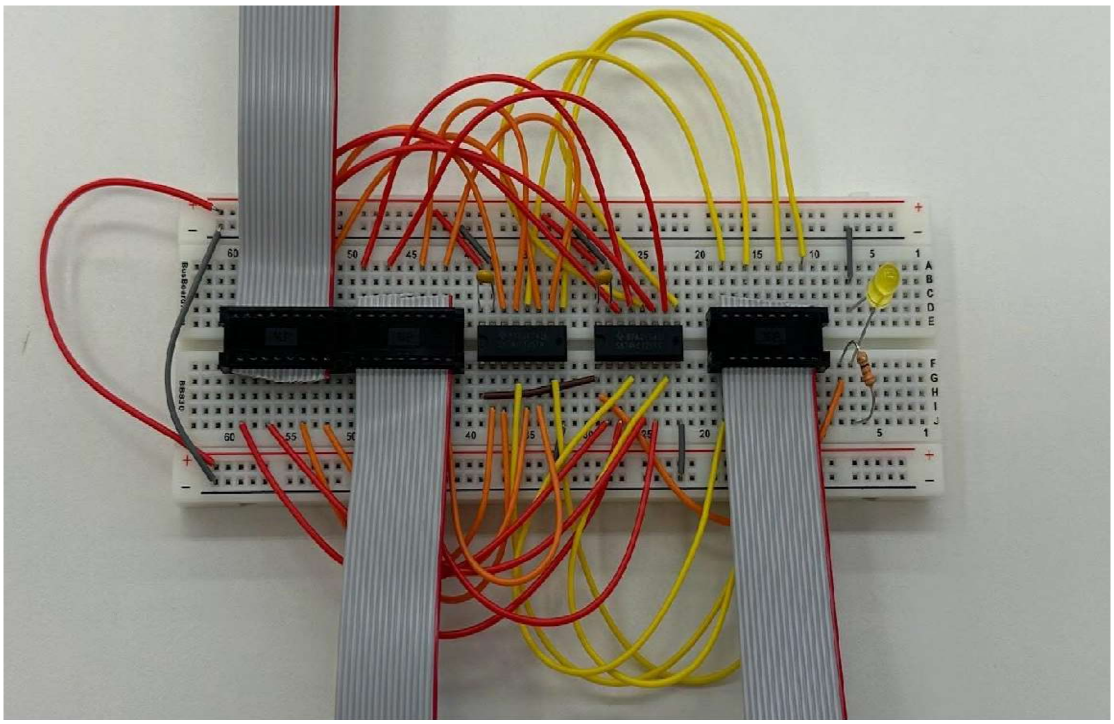


Figure 5 - Initial implementation lab 3's 2-to-1 8-bit bus multiplexer.

While we work through the activities, we closely monitor any difficulties that we face in completing the activity. Difficulties include unusual results when testing components, problems with IDEs or design programs, designs that do not give us accurate results and things we learn by going through the activity that are helpful for completing it. If the designed activity has any major problems or it does not give us the intended results, we adjust the design to remedy the problems. Once a design is thoroughly tested and all the major and a majority of the minor problems are worked out. We bring the lab before the team and client for approval. Once a lab has been thoroughly tested and approved by both the team and client we begin drafting the first draft of the lab documents.

6.2. Drafting Labs

Each lab features two main and additional supporting documents. The two main documents are the lab manual and the lab report. While supporting documents range from grading rubrics and answer keys to specification sheets and supplemental resources. Additionally, the KiCad labs include tutorial videos.

Each lab manual features five main sections.

- Lab Objective
- Parts List
- Background
- Activity
- Testing

The contents of the lab reports change depending on the content of the lab. The only part that will always be present is the pre-lab.

Once the drafts are created, they are sent to the client for review. After receiving feedback, the team makes any required, clarifying and formatting changes necessary to the lab. Then the process is repeated until both the team and client are satisfied with the lab.

CprE 3710x Lab X

Electrical and Computer Engineering

Iowa State University

Lab Name

1.0 Objectives

What do we want to accomplish with this lab? What should the students get out of the lab? What does this lab cover?

2.0 Background

What information do the students need to understand before working through the lab? Does this lab require any logic? How does the circuit work? What will the student's find difficult to figure out if it's not included? Subsections might be dedicated to different parts of the lab or included prior to the activity step where the information is needed.

3.0 Activity

What steps do the students need to do to complete the lab? This is where the points are, what do the students need to accomplish to get these points? Offer general guidance, but how much direction is necessary for a student to draw the necessary conclusion on how to complete the activity?

3.1 Specific Activity Step

What is the first step to the overall goal of this lab?

Figure 6 - Beginning of lab manual template.

6.3. Design Analysis

As previously mentioned, we create our labs in stages. These stages are designing the lab, ordering parts for the lab, building the hardware or coding the software to mockup the activity for the lab, testing the design for the lab, and then drafting and revising. Of our designed lab timeline, we have completed all of the material for labs 1 through 11 and the mini project. Additionally, we created tutorial videos for labs 4 and 5. Due to the time constraints, while we designed lab 12, we did not receive the parts for it in time to complete its implementation.

Given that our task was to develop ten activities, we reached our goal. We finished over 150 pages of material in finalized lab manuals, lab reports, grading rubrics, and answer keys. As well as twenty minutes of recorded tutorial content. At least one revision for each lab with the maximum being five versions. Around one hundred different images and figures with some redone multiple times. Additionally, the students testing our labs could understand the instructions and follow along with limited clarifications within and often under the expected lab period.

7. Ethics and Professional Responsibility

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment".

While our understanding of ethical considerations remains important, nothing in this section has changed because the scope of our project is limited to developing and delivering the labs to our client, Professor Stoytchev. It is ultimately his responsibility to integrate these labs into the course and evaluate their effectiveness as teaching tools. Our focus is on ensuring the labs are accurate, well-documented, and responsibly designed—rather than measuring learning outcomes directly.

7.1. Areas of Professional Responsibility/Codes of Ethics

It's important that we effectively communicate design progress and feasibility to each other as well as the client. Based on this communication, we can further improve our labs and speed up our workflow. Our team upholds ethical standards since our project is an educational one, it is important to follow the IEEE code of ethics and the university's code of ethics by setting a good example for the students who will be taking this course. Our ethics for this project most likely line up with Benjamin Franklin's virtues. Since our project is for educational purposes, and our team operates on honesty and integrity. In addition, our team is upfront about any potential issues that concerns the i281e processor.

Area of Responsibility	Communication Honesty
Definition	Reporting work, truthfully without any form of deception to advisors, professors or any faculty who is supervising the project. In addition, that includes truthfully reporting to the public.
Relevant Item from Code of Ethics	Our team reports truthful work with our project advisor and senior design advisors. If there is a concern in our project, our team would bring attention to

	that concern right away to find ways to address this issue as soon as possible.

Table 13 - Professional area of responsibility.

7.2. Four Principles

	Beneficence	Nonmaleficence	Respect for Autonomy	Justice
Public Health, Safety & Welfare	Students will gain experience and learn from our labs	Students will learn in a safe and well-guided environment by the TAs and professor	We must make sure students are able to gain experience rather than just following instructions	We need to make a fair grading standard
Global, Cultural & Social	People from different backgrounds and universities can use our open-source software	We need to make sure that our open-source software is safe to use	We need to make sure that our instructions of our open-source software are user-friendly and clear to read for all users	Making sure that our open-source software is not plagiarized
Environmental	What students learn in this course may enable some of them to leverage their careers positively later	We need to keep in mind that we are using a lot of plastic materials and not use more than needed	We need to make sure that our project does not harm the environment and find optimal methods for better use of our resources without also being wasteful	Make sure the materials we source is from a respectable company and not wasted
Economic	Students are going to have to pay for materials, we need to make sure what they learn is worth the cost	Make sure students don’t pay more than necessary	Keep in mind the cost burden to the department and individual students	Make sure students pay for what they use

Table 14 - Four principles.

7.3. VIRTUES

Our team believes that communication, honesty, financial responsibility and sustainability are important for many reasons. It is important to report honestly in a timely and professional manner since our attitude towards the project and faculty as a team reflects and may influence how students and participants in the labs treat the classwork and lab work. It is also important to our team to be financially responsible and sustainable since the project is also made for future students and individuals who are interested in replicating those labs.

Team Member	Ethan	Ariana	Tessa	Gigi
Virtue Demonstrated	Honesty	Clear and Through Documentation	Attentiveness	Commitment to Quality
Virtue Importance	It's important to be honest in reporting so that all team members are working with the same information and that what you say can be trusted to be true.	It's important to clearly and effectively communicate information when it's included in project documentation, so it can be understood by readers.	It's important to be responsive to the needs of teammates, the project, and the client and be able to prioritize them and come up with workable solutions to them.	It's important to have a quality documented lab, and that also means that it is important to have constructed lab work of the same quality to ensure that student(s) comprehend the information being given to them.
How was it Demonstrated?	Honest reporting of what was worked on and when.	Thoroughly creating documentation for the labs and editing group documentation.	Working with the client to improve several drafts of a lab and get it to a final state.	Going through detailed and thorough testing of our labs to make sure that the concepts are correctly implemented, along with updating lab documentation whenever there is a change in implementation method.
Virtue to Improve	Industry	Frugality	Cooperativeness	Completeness
Virtue Importance	It's important to make sure that little time is wasted in meetings and work sessions so those meetings can be as productive as possible.	It's important to make sure that components are not wasted to keep costs manageable. It's also important that components are used with care and not recklessly gone through.	It's important to be flexible and open to other teammates' ideas. After communicating our ideas, and those of the client, we need to resolve any creative differences.	It's important to make the lab reports feel more like they are made for an engineering course. It's also important to provide full, detailed information about what the labs would entail and what the questions are really asking the students to do.

Team Member	Ethan	Ariana	Tessa	Gigi
How Can it be Improved?	Following the Gantt chart much closer next semester to make sure all labs get done on time and not to fall behind on what is planned.	By being more careful when using components and not being reckless with materials.	Our ideas for some labs are not completely fleshed out and I sometimes focus too much on how I want the lab timeline to work.	It is important to go through many iterations of the lab documents and involve our advisor and other participants in outreach events to provide input about the instructions made in the labs.

Table 15 - Virtues and Improvements within the team.

In conclusion, our commitment to honest communication, financial responsibility, and sustainability not only enhances the integrity of our project but also sets a positive example for future students and participants. By incorporating these principles into our work, we aim to create a legacy that encourages responsible practices and fosters a culture of collaboration and respect within the academic community.

8. Closing Material

8.1. Summary of Progress

The primary goal of our project was to develop ten interactive labs that help students understand core concepts in digital logic, embedded systems, and processor design through hands-on circuit building and programming. We successfully completed this goal by producing ten fully drafted labs, each with working circuits/programs, clear diagrams, and detailed instructional manuals. These labs span topics such as breadboarding basics, multiplexers, program counters, PCB design, EEPROM programming, and introductory assembly programming using the i281e processor.

Throughout the project, we followed a structured process of design, testing, and iterative documentation, with consistent feedback from our client, Professor Stoytchev. While some labs are more polished than others, all ten have been tested or reviewed to a functional level and are ready for refinement and classroom use. Alongside lab development, we explored assembly and testing procedures for additional i281e processors to help scale lab availability. These accomplishments align with our project objectives and provide a strong foundation for both course integration and future expansion.

8.2. Value Provided

Our project delivers value on multiple levels—educational, technical, and practical. These labs provide a hands-on learning experience that helps bridge the gap between theoretical coursework and real-world system design. Each lab is built to support key learning outcomes from CprE 2810, 2880, and 3810, and our design choices reflect an intentional effort to make abstract concepts visible and testable. For example, the program counter lab gives students an opportunity to trace instruction flow through physical wiring, while the EEPROM lab introduces memory-mapped decoding with real outputs to a seven-segment display.

In terms of usability, all labs are written to be feasible to complete within a 2- to 3-hour window, designed with resource constraints in mind, and formatted for ease of grading and delivery. Our advisor has reviewed each lab draft and helped ensure alignment with course expectations. Although our direct responsibility does not include measuring student learning, the clarity and organization of our deliverables make it easier for future instructors and TAs to deliver these labs effectively. Long term, this project supports both outreach and formal instruction—offering engaging tools for middle school and high school events as well as university-level engineering labs.

8.3. Next Steps

Future work includes finalizing the remaining lab drafts that are currently in progress, developing new labs to expand the curriculum, and adapting select activities for outreach programs such as Women in Science and Engineering (WiSE). Additionally, some labs may require revision as the course continues to evolve, and small issues are identified during classroom use. These revisions will likely be carried out by Professor Stoytchev and future teaching assistants once the course is officially approved and running.

To support that transition, it is essential that we organize and deliver all supplemental materials used during development, including original circuit simulations, source files, and editable documentation. This will help ensure that future instructors and TAs can efficiently modify labs, maintain clarity, and continue aligning the content with instructional goals. While our direct work on this project is nearing completion, its long-term success depends on careful handoff and ongoing collaboration within the department.

9. References

[1] A. Stoytchev, “CprE 281: Digital Logic (Fall 2023),” Dec, 2023. [Online]. Available: https://www.ece.iastate.edu/~alexs/classes/2023_Fall_281/.

[2] B. Damman, “i281 CPU,” May, 2024. [Online]. Available: https://github.com/brandtdamman/i281e-cpu.

[3] E. Mirecki, “7 Segment Controller via EEPROM,” July, 2022. [Online]. Available: https://elijah.mirecki.com/blog/7-segment-eeprom.

[4] "IEEE Recommended Practice for Electronic Power Subsystems: Parameter Definitions, Test Conditions, and Test Methods," in IEEE Std 1515-2000, vol., no., pp.1-84, 15 Sept. 2000, doi: 10.1109/IEEESTD.2000.91922.

[5] "IEEE Standard Glossary of Computer Hardware Terminology," in IEEE Std 610.10-1994 , vol., no., pp.i-, 1995, doi: 10.1109/IEEESTD.1995.79522.

[6] "IEEE Standard Glossary of Computer Languages," in IEEE Std 610.13-1993, vol., no., pp.i-, 1993, doi: 10.1109/IEEESTD.1993.119224.

[7] “maze-game,” June, 2022. [Online]. Available: https://github.com/cheblankenshipUTD/maze-game/commits/main/.

[8] sdmay24-14,”Team14 Final Presentation”, ece.iastate.edu. Available: https://sdmay24-14.sd.ece.iastate.edu/final/2024/492_Team14_FinalPresentation.pdf

[9] sdmay21-38,”i281 Processor Simulator”, ece.iastate.edu. Available: https://sdmay21-38.sd.ece.iastate.edu/docs/DDV1.pdf

[10] sdmay22-20, ”Implementing the i281 CPU”, ece.iastate.edu. Available: https://sddec22-20.sd.ece.iastate.edu/

[11] “KICAD tutorial: Make your first printed circuit board,” Build Electronic Circuits, https://www.build-electronic-circuits.com/kicad-tutorial/

10. Appendices

10.1. Research

Product Services and Design	CprE 381 Lab 1 Documentation	Qatar University 261 Lab 2: Logic Circuits	Iowa State University CprE 288 Lab 5: Interrupts	CprE 281 Counters Lab Documentation
Unique Value Proposition	This lab is the first stage of CprE 381 which culminates in designing and building a MIPS processor. In this lab, students test the output of several provided components as well as design, build, and test a full adder and an N-bit adder.	This lab explores different ways of creating logic circuits using different logic chips (LC) and different ways of implementing the same design.	This lab is tailored to the Tiva TM4C123GH6PM Processor and requires knowledge of the processor's inner workings or the ability to read its documentation to complete. The purpose of this lab is to demonstrate how interrupts work within an embedded system.	This lab is about basic digital logic and builds up to gaining knowledge about how to combine certain logic gates to create efficient components.
Product Advantages	All the components designed and tested in this lab are subcomponents of the processor students will put together later so this helps students understand each component for later.	This product includes pin diagrams and helpful pictures to follow step-by-step instructions while exploring the basic concepts of logic circuits.	This lab provides skeleton code and resources so that to complete it a student would only have to mess with the relevant parts of the code while understanding the concepts presented in the lab.	This lab provides a basic understanding of how a part of our product works and our team could tweak the code if we wanted to, instead of having to build it from the ground up.
Product Disadvantages	There are not a ton of instructions on getting started or steps to follow throughout the lab making certain questions misleading or confusing to understand.	This lab requires the ability to read circuit diagrams although the pictures can help circumvent the need to read these diagrams.	This lab requires extended readings and prior knowledge not included in the documentation to complete. There is a learning hurdle in understanding how to read microcontroller documentation.	The lab may require knowledge about what flip flops and clock speeds are. In addition, this lab requires knowledge of how each flip flop is different when connected or disconnected.
User Pros	Gives them a chance to understand the individual components and start designing some as well.	Easy to understand and follow. All the information is contained within the lab. Gives helpful tips to stay organized while building the design.	The concepts are easy to understand and implement if you can read the datasheet and pay attention to class.	The concept is easy to understand with enough practice and does not require complex calculations.

User Cons	This lab is very time consuming and makes it difficult to understand where to start.	The whole lab takes a lot of time to complete if the time isn't taken previously to fill out some of the diagrams and truth tables.	The lab is time consuming and takes a lot of time outside of the lab to complete. The provided equipment occasionally glitches and doesn't work as it is supposed to. If you don't understand the datasheet, it's really challenging to complete.	The lab may require a truth table that is accurate, meaning that the user needs to track each input and output of the counter and the flip-flops contained in the counter's circuit.

Table 16 - Product research.

10.2. Team

10.2.1. Team Members

- Ethan Uhrich
- Ariana Dirksen
- Tessa Morgan
- Gigi Harrabi

10.2.2. Required Skill Sets for Your Project

Skill Set	Rationale
Circuit building	The first set of labs we are creating require skills to build and modify breadboard circuits based off of previous KiCad Designs.
Detailed writing and editing	In the creation of labs, it is important to create detailed and understandable directions for students and others to be able to follow along with.
Photo editing and image generation	It's important for labs to be visually appealing so having engaging but professional images to go along with our labs.
EEPROM programing	One of the labs in progress is EEPROM programming which the i281e processor uses for its main and code memory.
Circuit testing	In addition to circuit building, it's important to know how to test if a circuit is functioning as intended and troubleshooting steps involved with fixing a circuit.
Digital logic	Most of our labs are centered around digital logic so it's important to have the knowledge to write labs around it.
KiCad	Most of the designs left by the previous team were made in KiCad. We had to learn how to read these designs and how to edit and implement them onto breadboards.
Assembly	Labs 10 and 11 require coding in assembly.
Reading Data Sheets	All of our hardware labs require components that have logical data sheets.

Table 17 - Skills needed for the project.

10.2.3. Skill Sets covered by the Team

Skill	Team Member
Circuit Building	Ariana, Ethan, Tessa, Gigi
Detailed writing and editing	Ariana, Ethan, Tessa, Gigi
Photo editing and image generation	Ariana, Tessa
EEPROM programing	Tessa, Gigi
Circuit testing	Ariana, Ethan, Tessa, Gigi
Digital logic	Ariana, Ethan, Tessa, Gigi
KiCad	Tessa, Ethan, Ariana
Assembly	Ariana, Ethan
Reading Data Sheets	Ariana, Ethan, Tessa, Gigi

Table 18 - Skills covered by the team.

10.2.4. Project Management Style Adopted by the Team

As a team we are operating under Agile project management style as our process for creating the labs requires us to revise our designs and the labs multiple times before they are ready. Throughout our planning and implementation processes, we have already scrapped multiple lab ideas, so we've been using Agile to adapt to changes as they appear. It's also imperative that we're able to jump from one task to the next whenever one is stalled to make sure that we are still working on our project. The stand-ups involved in the agile project management style also allow us time to communicate on what everyone is working on and will be working on through the week and gives time to talk about any challenges we are facing in our own assignments.

10.2.5. Project Management Roles

- Ethan Uhrich - Treasurer, Team Lead
- Ariana Dirksen - Note Taker, Editor
- Tessa Morgan - Webmaster, Graphic Designer
- Gigi Harrabi - Client Interaction, Outreach Coordinator

10.2.6. TEAM CONTRACT

Team Name sdmay25-31

Team Members:
1) Ethan Uhrich 2) Ariana Dirksen
3) Gigi Harrabi 4) Tessa Morgan

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    a. Thursday, 2:15pm Coover 1301 - Team Meeting
    b. Monday 10-11am Durham 303 - Advisor/Client Meeting
    c. Wednesday 11am Coover 1301- Optional Additional Team Meeting
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. Discord
    b. E-mail - Outside of team correspondence
3. Decision-making policy (e.g., consensus, majority vote):
    a. Majority vote if disagreement. If tied, the Advisor/Client is the tie-breaker.
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    a. Ariana will keep notes stored in a Folder in shared Google Drive.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
    a. Attendance is expected but exceptions will be made with notice.
    b. Punctuality: Be on time if not early. If running late, notify the group via discord.
2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    a. Get things in on time, if not early.
    b. Ask for help early, not at the last minute.
    c. Communicate with the team if you will miss a deadline.
3. Expected level of communication with other team members:
    a. Emote to important messages that don't require a text response.
    b. Respond to important messages within a day
4. Expected level of commitment to team decisions and tasks:
    a. Decisions should be promptly decided on
    b. Tasks need to be done by deadline, but MUST be completed by drop-deadline.
    c. If subtask is needed by another team member due by deadline, finish in time for deadline.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    a. Ethan Uhrich - Treasurer, Team Lead
    b. Ariana Dirksen - Note Taker, Editor
    c. Tessa Morgan - Task Manager, Webmaster
    d. Gigi Harrabi - Client Interaction, Outreach Coordinator
2. Strategies for supporting and guiding the work of all team members:
    a. Routine checks during weekly meetings (weekly standup)
3. Strategies for recognizing the contributions of all team members:
    a. Kudos, physically represented with Gold Star Stickers
    b. Personal and team affirmations

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
    a. Ethan - Only Male, Lots of team-leading experience.
    b. Ariana - Tutoring experience, Ran a high school science club.
    c. Tessa - Taught in "Girls Who Code", CprE 185 TA
    d. Gigi - Experience with academic correspondence
2. Strategies for encouraging and supporting contributions and ideas from all team members:
    a. Weekly Stand-ups
    b. Round Robin
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
    a. Go to an advisor with systemic problems.
    b. Try to resolve smaller problems as a group.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    a. Create 3 of the labs/activities w/ documentation, videos
    b. Do at least 1 outreach event
    c. Outline the last 7 activities.
2. Strategies for planning and assigning individual and team work:
    a. Tessa will assign tasks via git.
3. Strategies for keeping on task:
    a. Standups, team lead making sure we don't go too far off topic during meetings/stand-ups

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    a. Gold stars revoked.
    b. 3 strike system - 2 warnings on 3rd escalate
2. What will your team do if the infractions continue?
    a. After 3 strikes the issue will be escalated to the advisor.

****

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) ______________________________ DATE 9/12/24

2) ______________________________ DATE 9/12/24

3) ______________________________ DATE 9/12/24

4) ______________________________ DATE 9/12/24