Implement and Test a Curriculum Around the i281e CPU

sdmay25-31

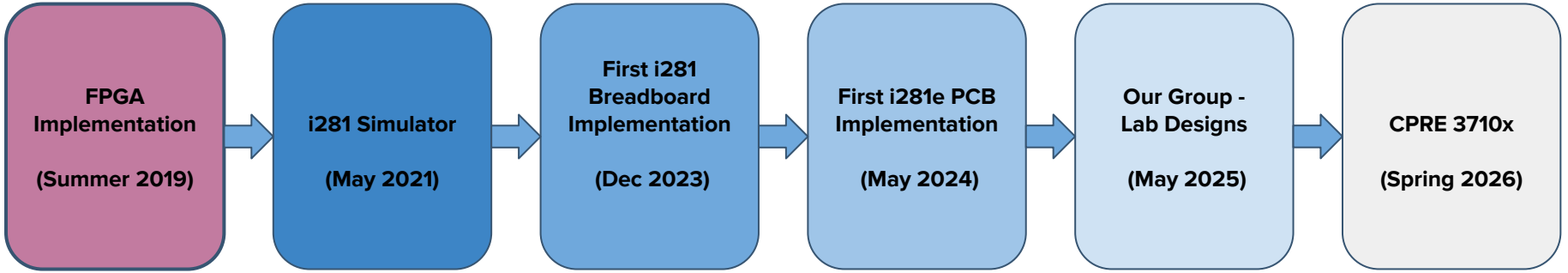
Ariana Dirksen, Gigi Harrabi, Tessa Morgan, Ethan Uhrich
Professor Alexander Stoytchev

Team Composition

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

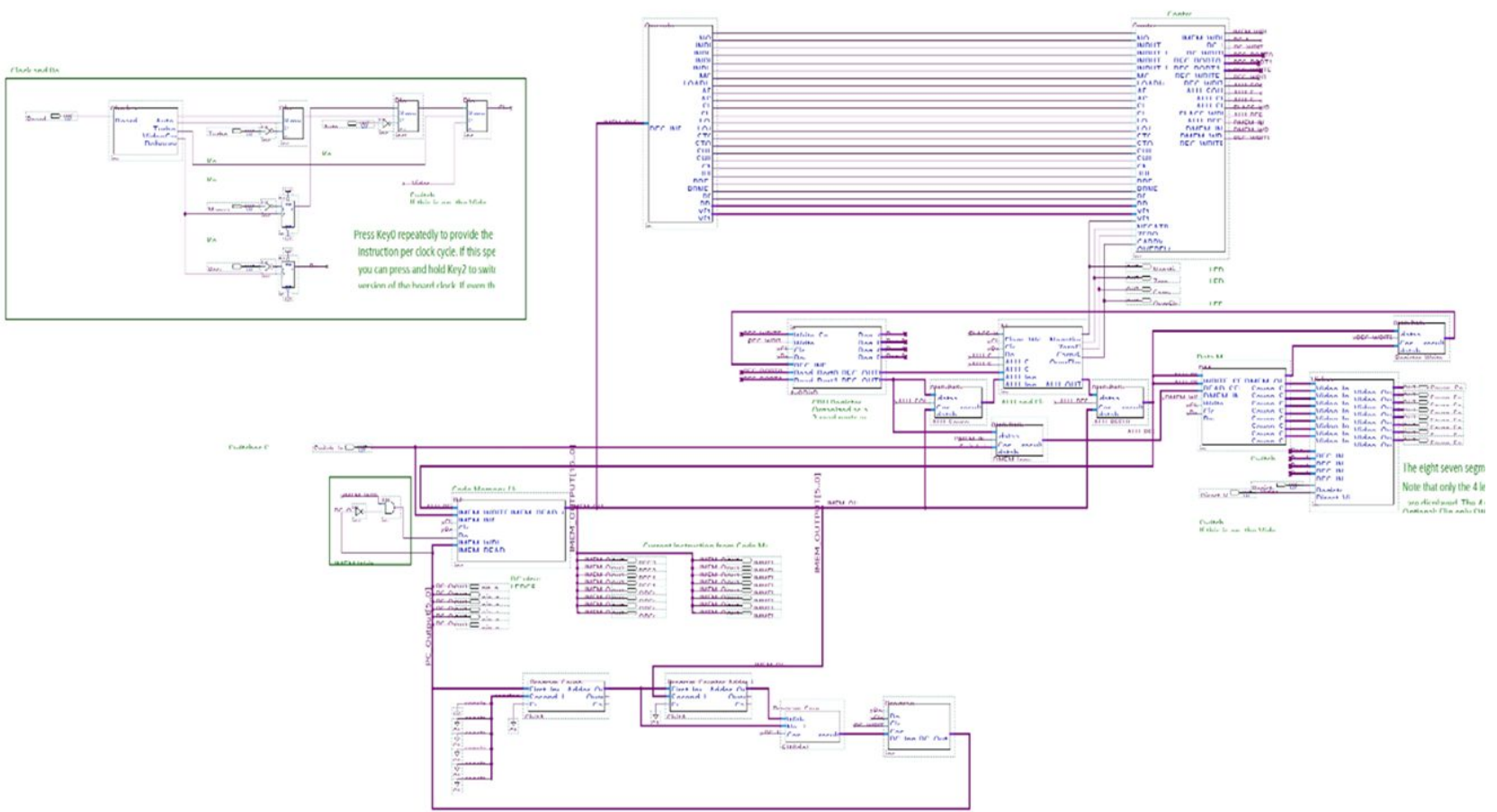
Background

Historic Timeline

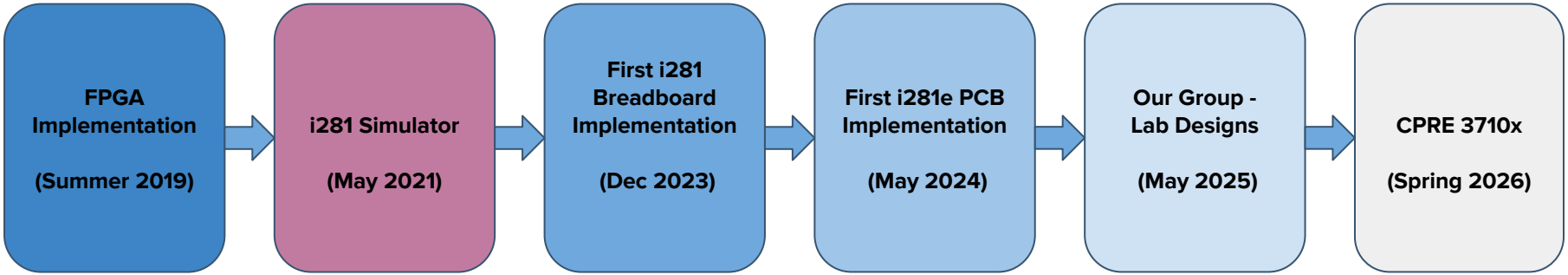


FPGA Implementation- Summer 2019

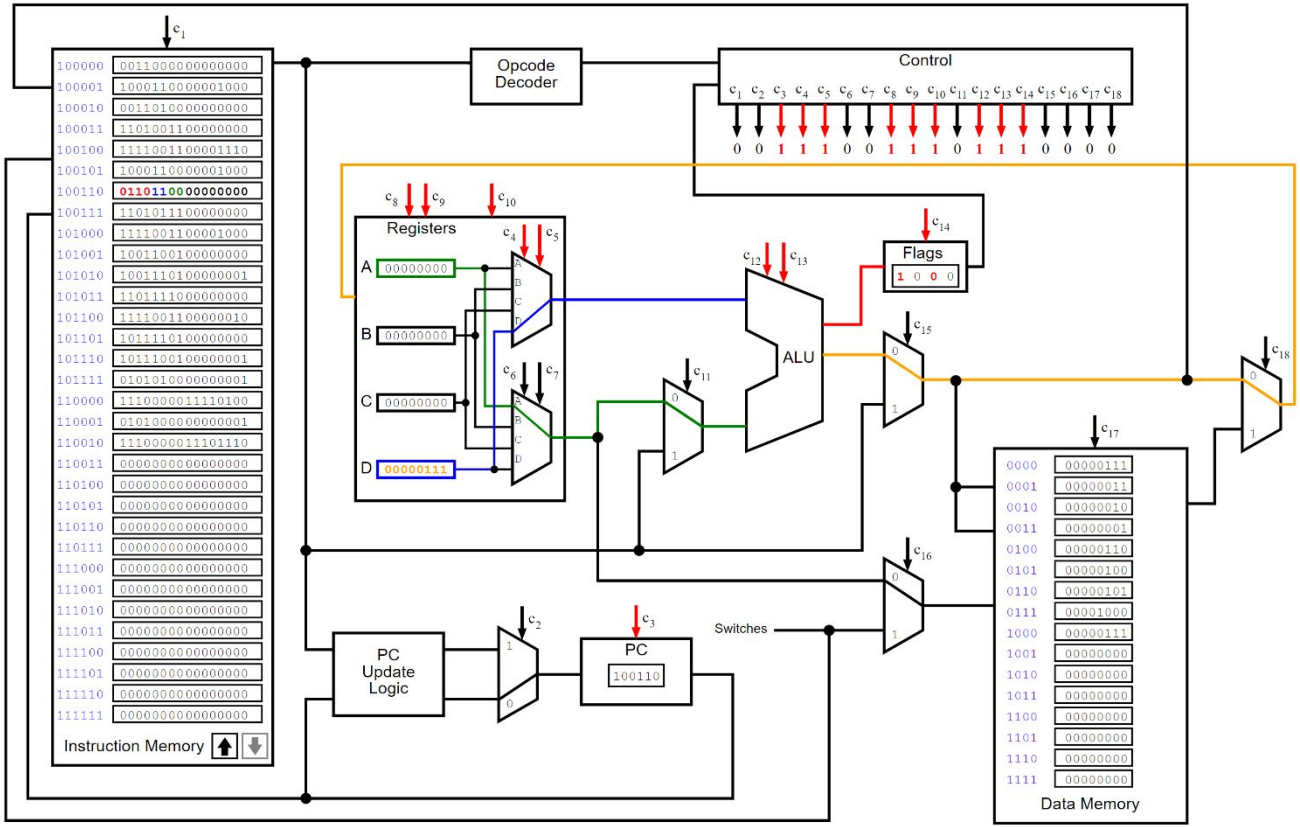
i281CPU

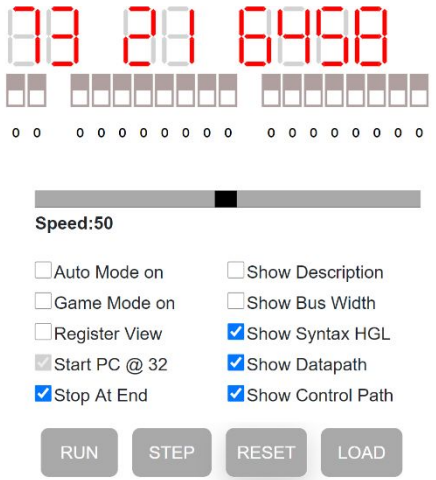


Historic Timeline

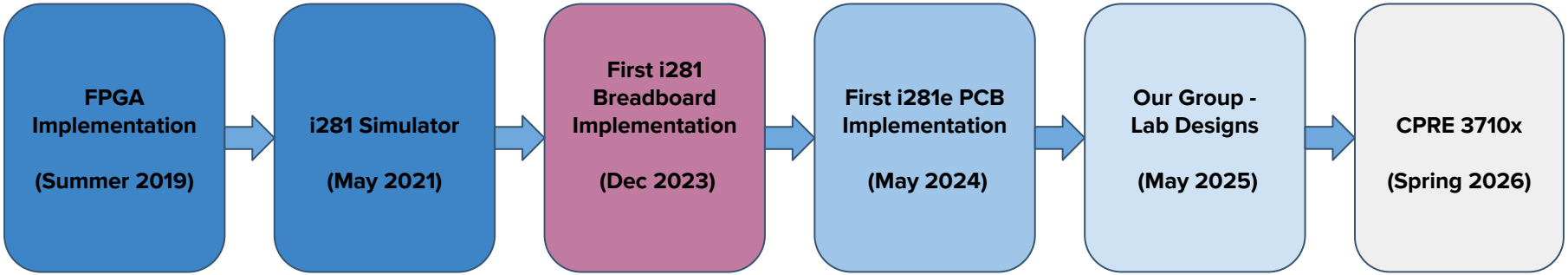


Simulator - May 2021

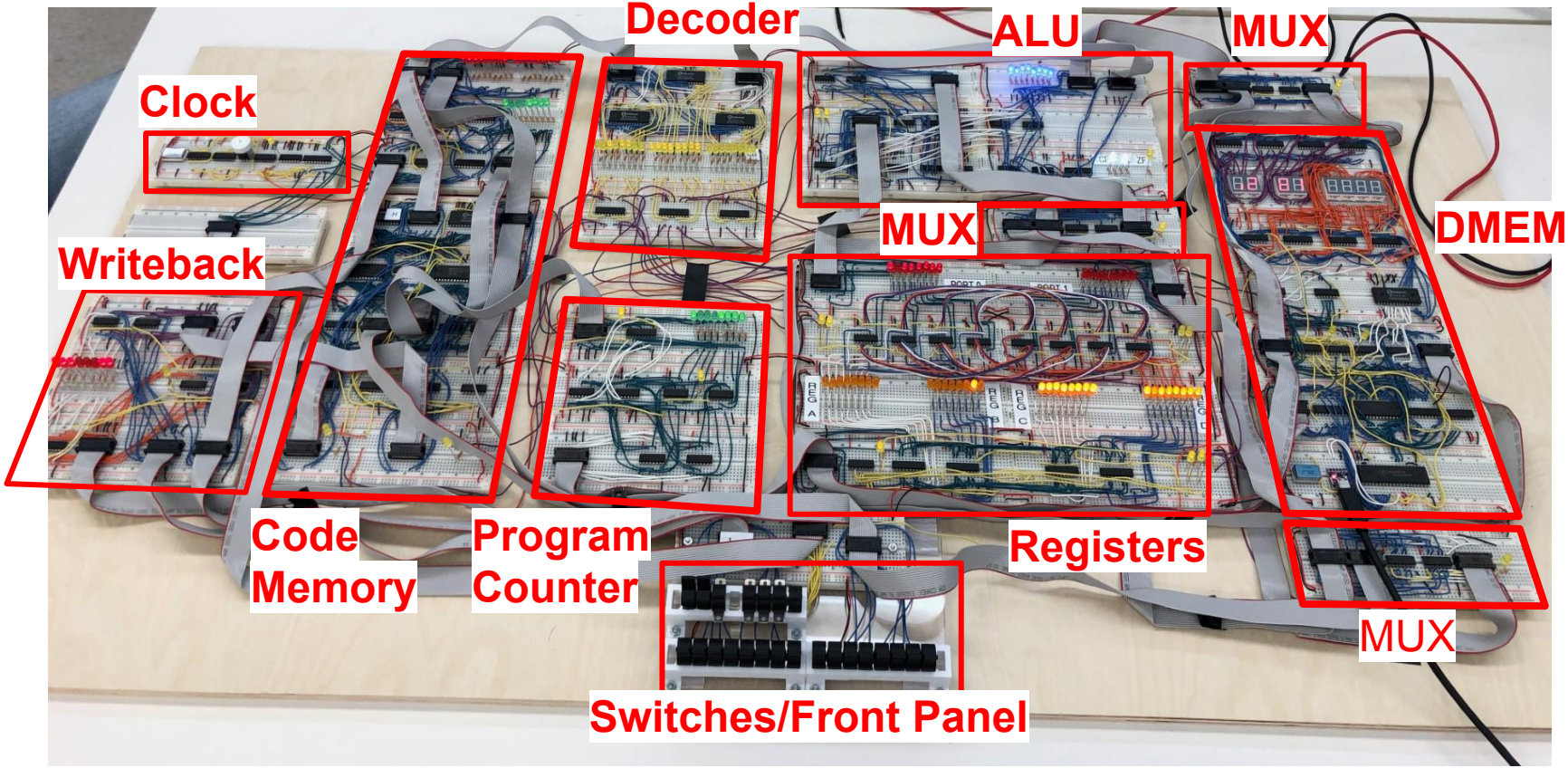




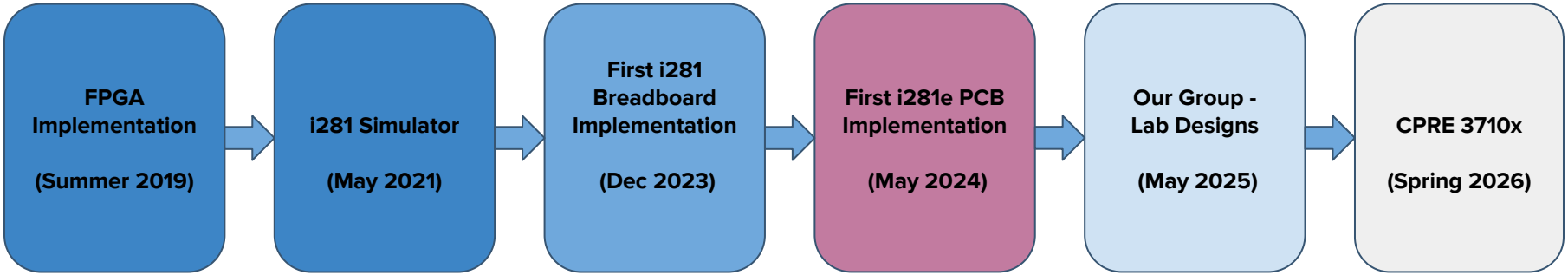
Historic Timeline



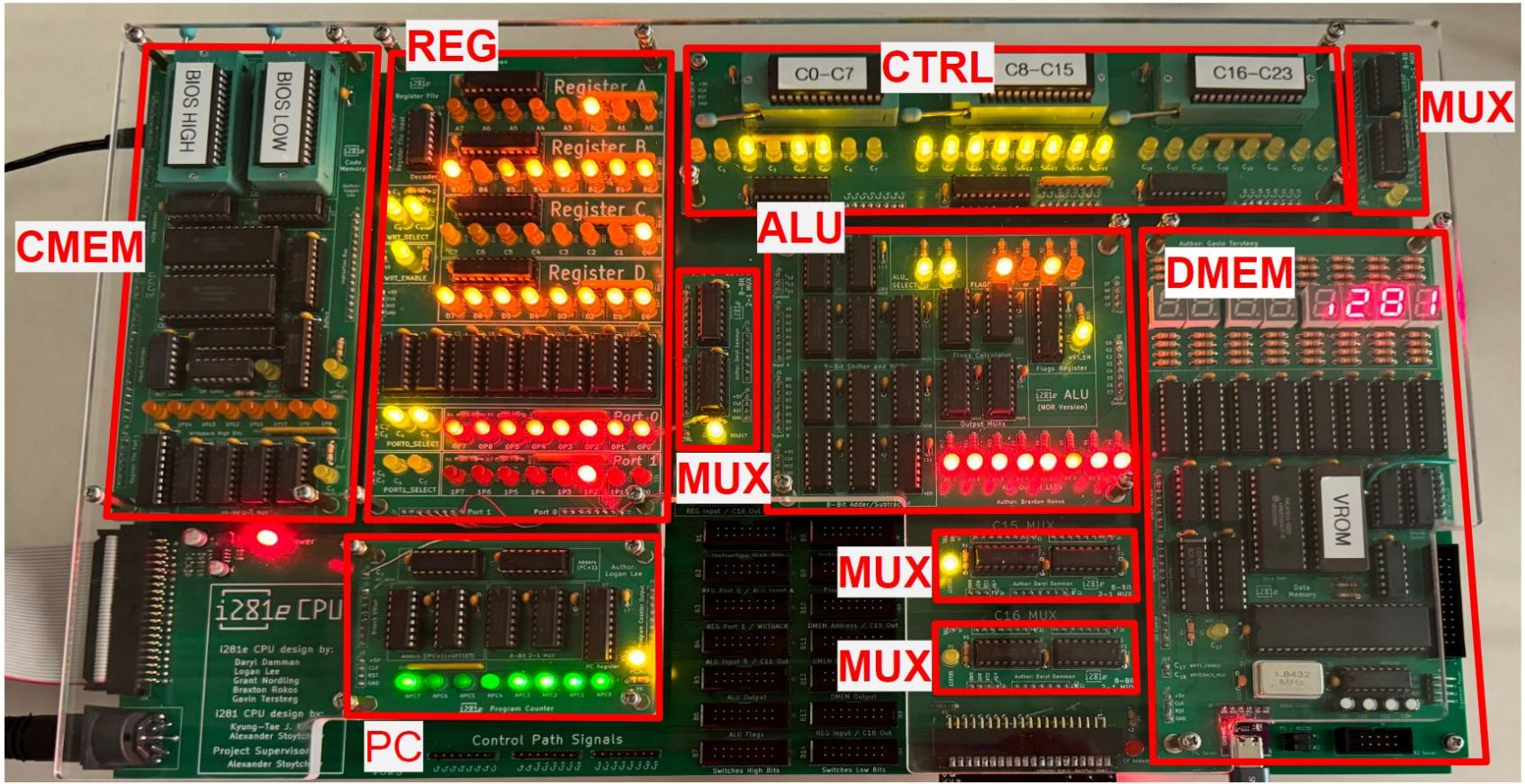
Breadboard Implementation - Dec 2023 to Feb 2024



Historic Timeline

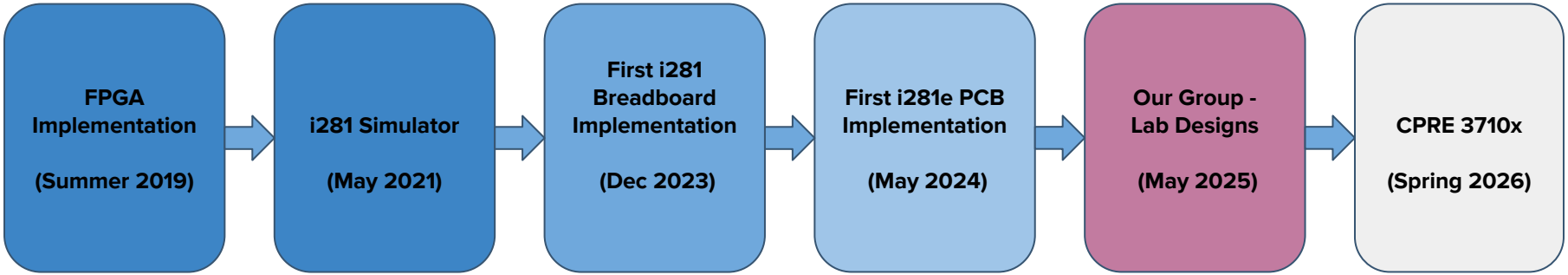


PCB Implementation - May 2024

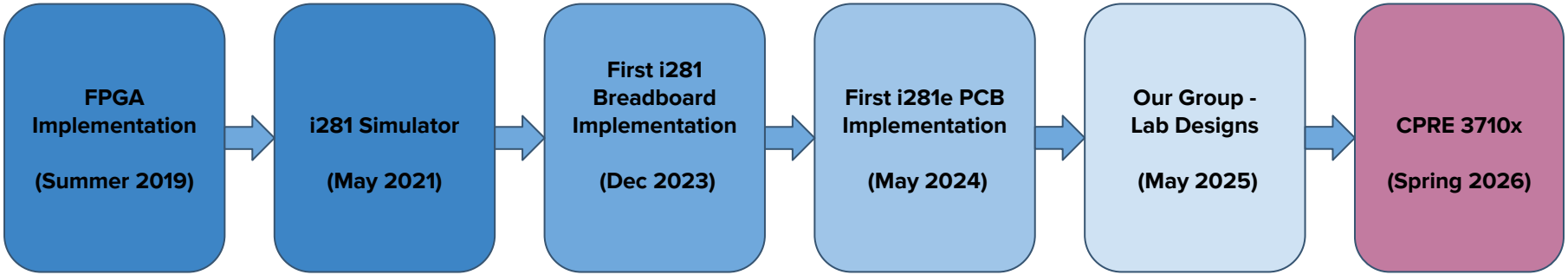


Source: https://sdmay24-14.sd.ece.iastate.edu/

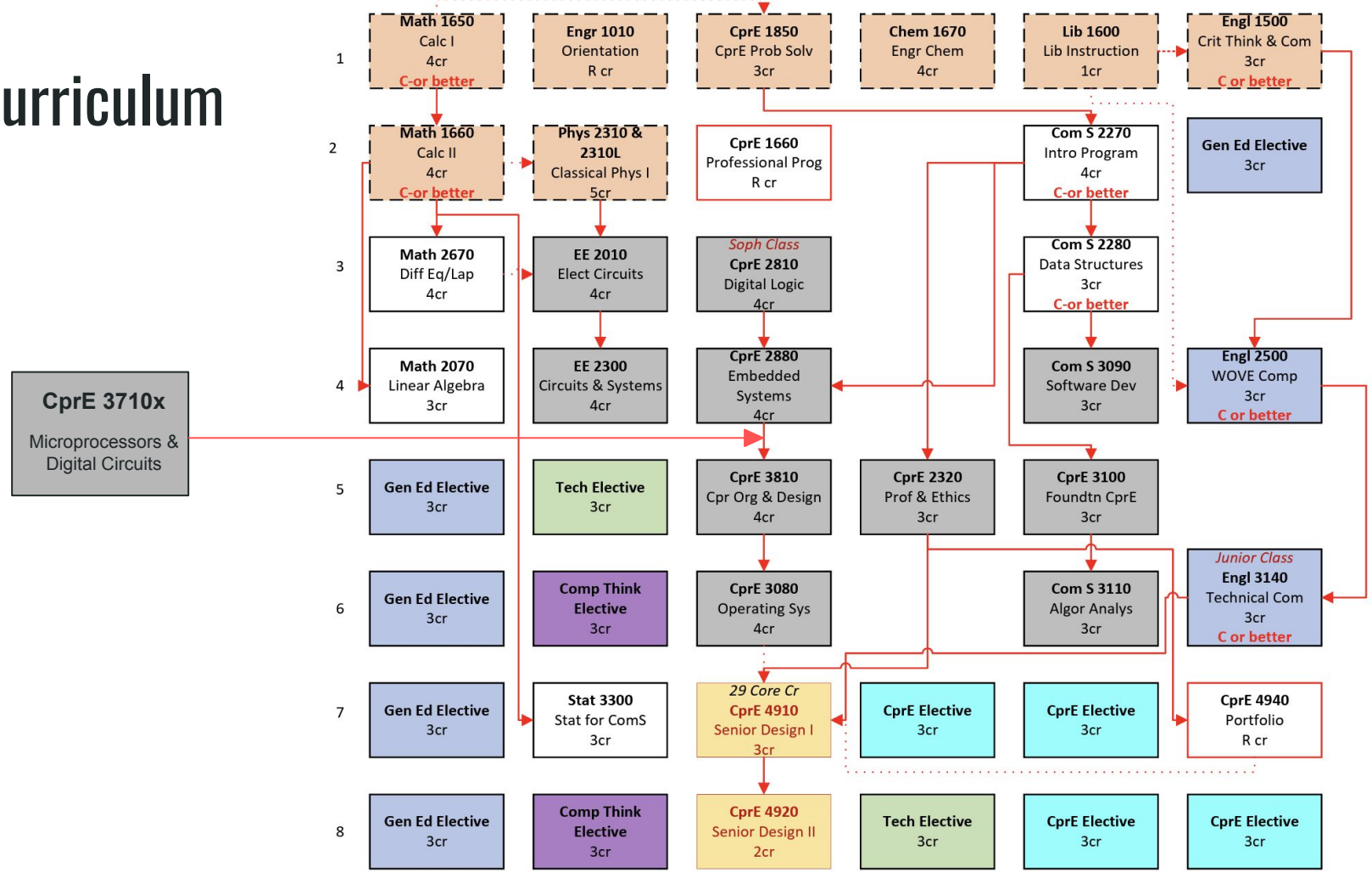
Historic Timeline



Historic Timeline



Curriculum


CprE 3710x
Microprocessors & Digital Circuits
1
Math 1650 Calc I 4cr C-or better
Engr 1010 Orientation R cr
CprE 1850 CprE Prob Solv 3cr
Chem 1670 Engr Chem 4cr
Lib 1600 Lib Instruction 1cr
Engl 1500 Crit Think & Com 3cr C or better
2
Math 1660 Calc II 4cr C-or better
Phys 2310 & 2310L Classical Phys I 5cr
CprE 1660 Professional Prog R cr
Com S 2270 Intro Program 4cr C-or better
Gen Ed Elective 3cr
3
Math 2670 Diff Eq/Lap 4cr
EE 2010 Elect Circuits 4cr
Soph Class CprE 2810 Digital Logic 4cr
Com S 2280 Data Structures 3cr C-or better
4
Math 2070 Linear Algebra 3cr
EE 2300 Circuits & Systems 4cr
CprE 2880 Embedded Systems 4cr
Com S 3090 Software Dev 3cr
Engl 2500 WOVE Comp 3cr C or better
5
Gen Ed Elective 3cr
Tech Elective 3cr
CprE 3810 Cpr Org & Design 4cr
CprE 2320 Prof & Ethics 3cr
CprE 3100 Foundtn CprE 3cr
6
Gen Ed Elective 3cr
Comp Think Elective 3cr
CprE 3080 Operating Sys 4cr
Com S 3110 Algor Analys 3cr
Junior Class Engl 3140 Technical Com 3cr C or better
7
Gen Ed Elective 3cr
Stat 3300 Stat for ComS 3cr
29 Core Cr CprE 4910 Senior Design I 3cr
CprE Elective 3cr
CprE Elective 3cr
CprE 4940 Portfolio R cr
8
Gen Ed Elective 3cr
Comp Think Elective 3cr
CprE 4920 Senior Design II 2cr
Tech Elective 3cr
CprE Elective 3cr
CprE Elective 3cr

Overview and Requirements

Existing Resources

- GitLab
  - PCB Schematics
  - User Manual
- Electronics and Technology Group (ETG)
  - Matthew Post, Leland Harker, Student Workers
- Breadboard Implementation
- Previous Team Websites
- CprE 2810 Lectures
- Professor Stoytchev

Chapter 4: i281e Programming

4.1: Structure of an Instruction

Between different instruction set architectures, there is a wide diversity in how machine code instructions are encoded. Different processors employ a wide range of techniques to balance performance, memory consumption, and technical limitations. Some, like the MIPS processor, try to make instruction formats as simple as possible to reduce decoding times. Others, like the x86 family of processors, include a wide variety of valid instructions to make the most of each execution cycle. A modern x86-64 processor can execute 981 unique instructions, ranging from one to fifteen bytes long.

The i281e does not come close to that in terms of complexity. Each instruction is exactly 16 bits long. It is fetched, executed, and terminated in exactly one clock cycle. Not every instruction ends up using all 16 bits but keeping them at that length helps keep the i281e conceptually simple. An instruction can be broken up into two parts.

- **Opcode**: The highest 8 bits, which are sent to the control table and is used to generate the control signals for that instruction. This, along with the flag inputs, determines what the processor does during a specific clock cycle. The control table is the only part of the processor to receive the opcode.
- **Operand**: The lower 8 bits, which are sent to C_{11} and C_{15} to be used in the data path of the processor. The operand has no influence on what the control signals are during the instruction execution cycle. On the hardware, the value is known as the "Immediate Value."

The opcode itself can be broken up further. The top 4 bits of the opcode determine what "group" of instruction executes. Groups are a vague category that lump similar instruction together to ease decoding logic complexity. Some instruction groups contain several similar functions, while most contain only one. The bottom 4 are used as arguments to augment the function of that instruction further.

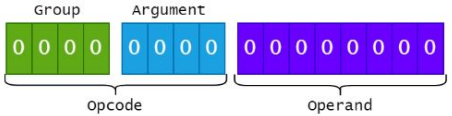


Fig 4.1.1: Basic structure of an i281e instruction

Project Proposal Summary

- Goal: Take these open-source hardware and software designs and implement a set of curriculum and outreach activities around them
- Each activity will be tested and documented in detail
- These documents could also be used as educational materials for existing classes or to support future lectures and labs
- A subset of these materials will be used for outreach activities

Project Proposal Summary Cont.

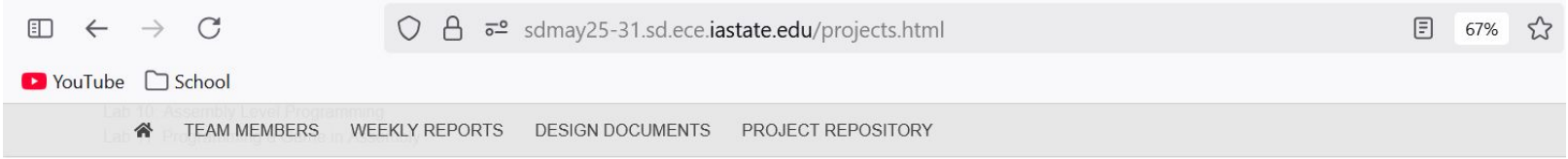
- Design, create and test at least 10 labs/activities based around the i281e processor completable within a lab period (about 2 hours)
- To be incorporated into a new class tentatively referred to as:
  - CprE 3710x : Microprocessors and Digital Circuits

Deliverables

Deliverables

- Lab Instruction Manual
  - Background Information & Images
  - Step-by-Step Activity Instructions
  - Testing Guides
- Lab Report
  - Pre-Lab Activities
  - Lab Questions
- Grading Rubric & Answer Key

- 43 Minutes of Instruction Videos
- 150+ Pages
- 450+ MB
- 650 Hours



Lab 1

Lab Manual
Lab Report Template

Lab 2

Lab Manual
Lab Report Template

Lab 3

Lab Manual
Lab Report Template

Lab 4

Lab Manual
Lab Report Template
Tutorial

Lab 5

Lab Manual
Lab Report Template
Tutorial
Mini Project

Lab 6 & 7

Lab Manual
Lab Report Template

Lab 8

Lab Manual
Lab Report Template

Lab 9

Lab Manual
Lab Report Template

Lab 10

Lab Manual
Lab Report Template

Lab 11

Lab Manual
Lab Report Template
Up-Down Example
Left-Right Example

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

Lab 3 Report

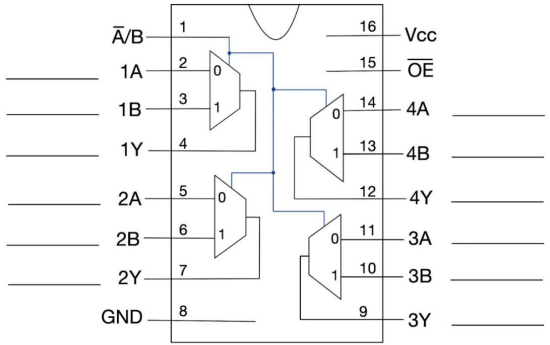
Name: _____ Student ID: _____

Lab Section: _____ Date: _____

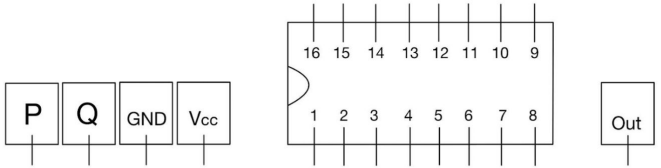
Prelab

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.



3. On the diagram below, connect inputs P and Q to any MUX of the SN74HCT257N chip. Connect the output of the MUX to the box labeled "Out". Use Vcc and GND to select line P as the output, to power the chip, and to enable the output.



Lab

4.1
Verify that you placed your chips, connectors, power and ground wires correctly. Show your progress on the breadboard implementation to the TA before you proceed.

TA Initials: _____

4.3
Verify that you connected the inputs to the SN74HCT257N chips correctly. Show your progress on the breadboard implementation to the TA before you proceed.

TA Initials: _____

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

Bus Multiplexer and Intro to Standardization and Connectors

1.0 Objectives

In this lab you will recreate the 8-bit 2-to-1 bus multiplexer from the i281e processor on a breadboard. This lab also covers the concepts of hardware buses, hardware circuit standardization, and connectors.

2.0 Parts List

Quantity	Item	
1	White 830-point Breadboard	
Set of	Breadboard Wire Spools	Or you can 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)	

3.0 Background

3.1 8-bit Buses

We will implement an 8-bit bus with a 16-conductor ribbon cable. This cable has twice as many wires than we need, so half of them will remain unused. Figure 1 shows the mapping of the wires to the 16-position connector. The unused wires are labeled with G or GND because they are typically set to ground. By convention, the red wire indicates the least significant bit. Figure 2 shows the pins of the end connector that is designed to plug into a breadboard. The pin numbers indicate the position of the bits in the 8-bit bus. Figure 3 shows the finished cable, with the connectors attached.

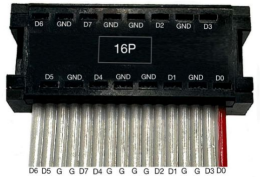


Figure 1: Standardized mapping of an 8-bit bus to a 16-conductor ribbon cable.

Figure 2: Side view of the connector pins.

Figure 3: An 8-bit bus with connectors on both sides.

3.2 Implementation

An 8-bit 2-to-1 bus MUX is typically drawn as shown in Figure 4. Figure 5 expands this diagram to show the individual 2-to-1 multiplexers, which have the same select input.

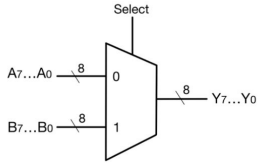


Figure 4: Graphical symbol for a 2-to-1 bus MUX (8-bits wide).

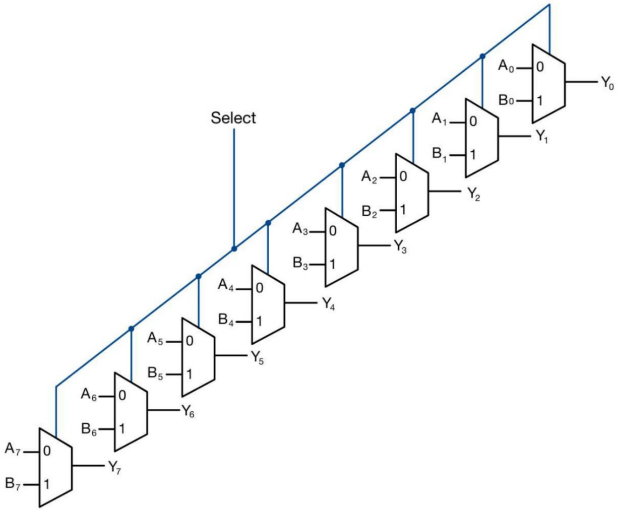


Figure 5: Expanded wiring diagram for the bus MUX from Figure 4.

To implement the 8-bit 2-to-1 bus multiplexer, you will use two SN74HCT257N chips. As shown in Figure 6 each of them contains four 2-to-1 multiplexers that share the same select line ($\overline{A}/B$). To complete the entire circuit, you need to combine the two chips and configure them to choose between the two input buses A = (A_7 … A_0) and B = (B_7 … B_0), depending on the value of the common select line. The output bus is called Y = (Y_7 … Y_0). Don't forget to set the output enable line ($\overline{OE}$) of each chip to 0. Please refer to the datasheet for additional information.

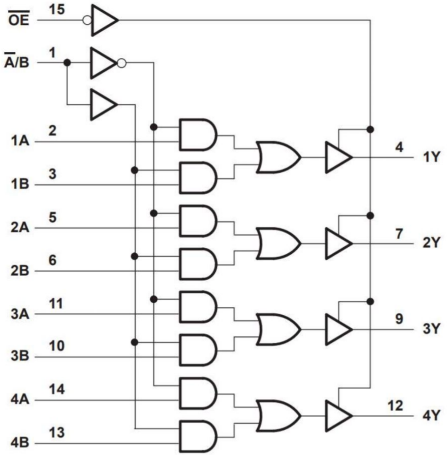


Figure 6: Pin layout for the Quad 2-to-1 MUX Chip (SN74HCT257N). From the spec sheet.

3.3 Standardization

When implementing circuits for this class, please follow these breadboard conventions:

- Each breadboard has an orientation, imposed by the numbers printed on it. In the correct orientation you must be able to read the numbers. (i.e., they are not upside down). Also, the positive power rail must be on top.
- Red wires are only for powering components, while black wires are for grounding.
- The red wire on each ribbon cable represents the least significant bit of the bus (i.e., it should be on the right side when looking at the ribbon cable in the correct orientation).
- When visualizing an 8-bit binary number with LEDs, the least significant bit should always be on the right side when you look at the breadboard from the correct orientation.

4.0 Activity

Please complete the pre-lab before starting with the lab. Specifically, you need to understand the purpose of the chips and the labels of the pins.

4.1 Place the Bus Connectors and the Chips

- Place the chips as shown in Figure 7.
- Connect them to power (pin 16) and ground (pin 8).
- Connect $\overline{OE}$ (pin 15) to ground. This enables the output of the chip.
- Connect the top power rail of the breadboard to the bottom power rail with a red wire. Also, connect the two ground rails with a black wire.
- Place the three bus connectors as shown in Figure 7. Don't insert the ribbon cables and don't press together the two parts of the connector yet.
- Study Figure 8, which is a simulated version of Figure 7. The pins of the connectors are labeled with small numbers. The red 0 indicates the position of the least significant bit of the corresponding bus.

Before continuing to the next step, have your circuit checked by the TA.

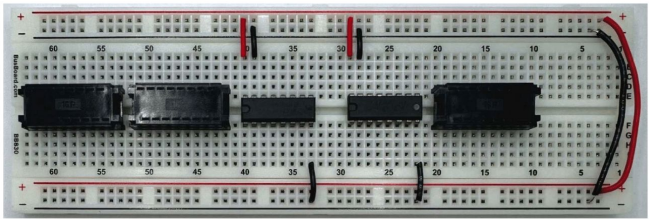


Figure 7: Image of a real breadboard with connectors and chips inserted in place.

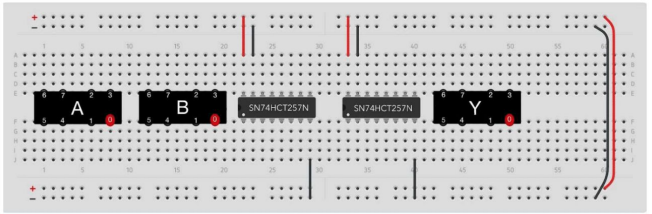


Figure 8: Image of a simulated breadboard that is equivalent to Figure 7.

4.4 Connect the Chips to the Output Bus

Start by connecting the four most significant bits (left chip) to the output connector:

- Top: Connect pins 9 and 12 of the chip to bits Y6 and Y7, respectively.
- Bottom: Connect pins 4 and 7 of the chip to bits Y4 and Y5, respectively.

Next, connect the least significant bits (right chip) to the output connector:

- Top: Connect pins 9 and 12 of the chip to bits Y2 and Y3, respectively.
- Bottom: Connect pins 4 and 7 of the chip to bits Y0 and Y1, respectively.

Your circuit should be similar to the one shown in Figure 11.

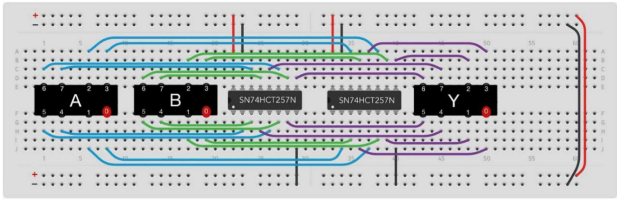


Figure 11: Connect the chips to the output bus (purple wires).

4.5 Finish the Circuit

- Place an LED on the right side of the circuit and ground its cathode (shorter leg).
- Connect the anode of the LED to a 330 Ω resistor.
- Connect the other side of the resistor to both chips at pin 1 (yellow wires in Figure 12).
- Connect that same side of the resistor to the select line, which may be grounded for now but will come from the test circuit later.
- Finally place two 0.1 μF capacitors to connect pin 16 of each chip to ground. Because in this case pin 15 is already grounded, the capacitor can connect pin 16 to pin 15.

Show your completed circuit to the TA before continuing.

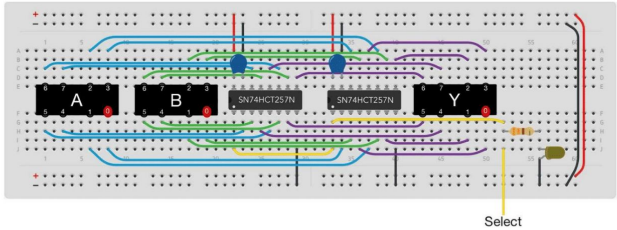


Figure 12: Add the select line and visualize it with an LED.

4.6 Make Three Ribbon Cables with Connectors

- Place a connector on an empty breadboard.
- Next, insert a ribbon cable into the top part of the connector. Ensure that each pin of the connector is in contact with exactly one wire from the ribbon bundle.
- Once the ribbon is positioned correctly, firmly press down with a flat object until the two parts of the connector click into place.

4.7 Connect the Ribbon Cables to the Bus MUX Circuit

- Figure 13 shows the completed real circuit that corresponds to Figure 12. It uses 3 connectors as placeholders. They are not connected to anything at this point.
- Start with the breadboard upright (the numbers are upright facing you) and replace each connector with a connector that already has a ribbon cable attached to it.
- The three ribbons should go out the bottom of the breadboard with the red wires on the right-hand side.
- The result should look like Figure 14.

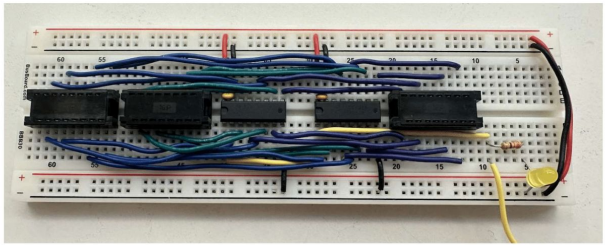


Figure 13: The finished bus MUX with place-holder connectors.

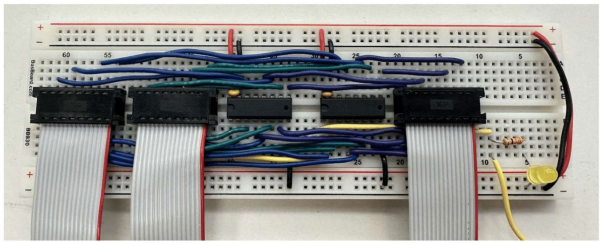


Figure 14: The finished bus MUX with ribbon cables attached.

5.1 Connect the Ribbon Cables to the Tester Circuit

- Place your bus MUX circuit above the tester circuit. Connect the leftmost ribbon cable of the tester to the A input of the MUX.
- Repeat this for bus B and the output bus Y.
- Next, connect the select line of the MUX to the tester circuit. To do this, use the yellow select wire that was grounded in step 4.5 and connect it to the small DIP switch.
- Finally, connect the power and ground rails of the tester circuit to the rails of the breadboard with your MUX circuit (see Figure 16). This will ensure that both circuits are powered.

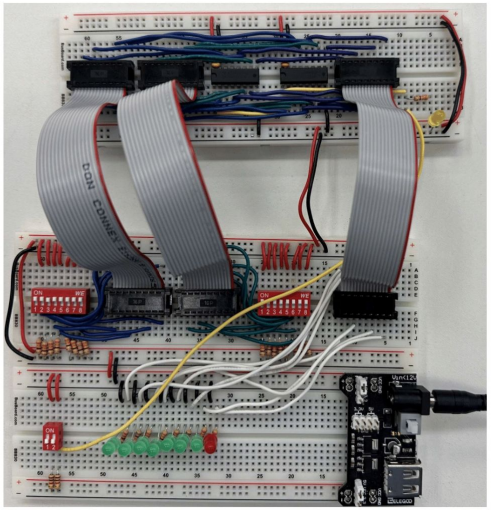


Figure 16: Bus MUX circuit connected to the tester circuit.

5.2 Test the Bus MUX

- Power the tester circuit using a breadboard power supply.
- Ensure that when the select line is low, the output shows the value of input A from the first eight switches. Similarly, when select is high, the output must show the value of input B from the second eight switches.

<u>Once you have completed and recorded the results of each test, perform each test for the TA.</u>

Considerations

User Considerations

Primary

- Prof. Stoytchev (Client)
  - Curriculum
  - Past i281 CPU Work
- Undergraduate Students
  - Previous knowledge
  - Time constraints
  - Lab room constraints

Secondary

- Teaching Assistants
- Outreach Coordinators
- Middle and High School students

Hardware

Chips:

- Need to be available with plenty of stock for the labs
- Some chips used in i281e processor no longer in production (EEPROM)
- Need to find replacements that are pin compatible to processor
- Shipments to US are paused in China so it's hard to get necessary parts.

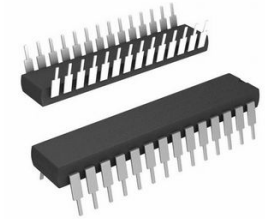


Image shown is a representation only. Exact specifications should be obtained from the product data sheet.

W27C512-45Z

DigiKey Part Number	W27C512-45Z-ND
Manufacturer	Winbond Electronics
Manufacturer Product Number	W27C512-45Z
Description	IC EEPROM 512KBIT PARALLEL 28DIP
Customer Reference	
Detailed Description	EEPROM Memory IC 512Kbit Parallel 45 ns 28-PDIP
Datasheet	Datasheet
EDA/CAD Models	W27C512-45Z Models

Wiring:

- The lab room for the class might not allow for cutting wires
- Wire kits have limited lengths and colors
- Some hardware, like the power supply may short, causing delays in testing

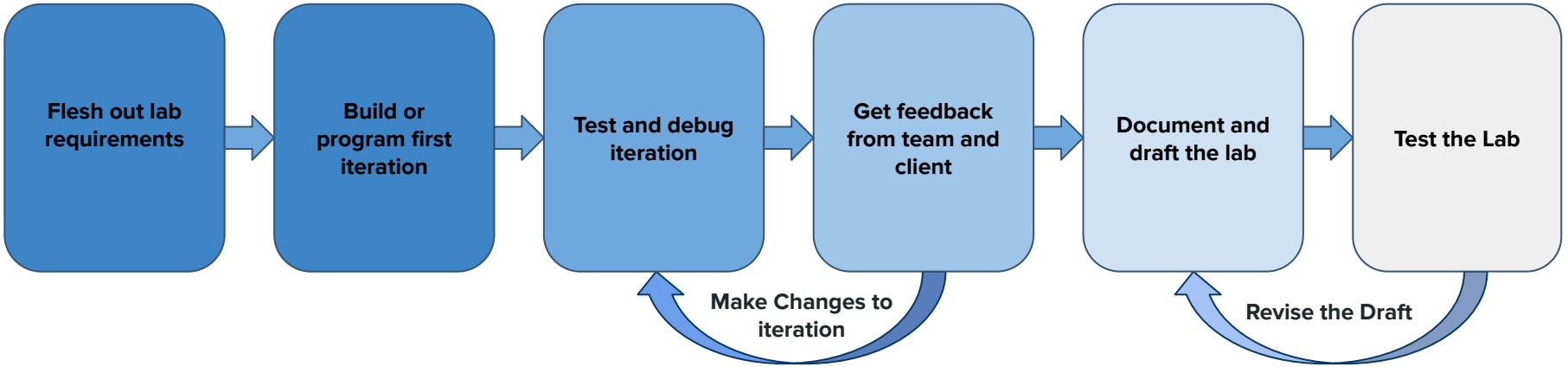
Software

AUTODESK®
TINKERCAD®

XGecu®

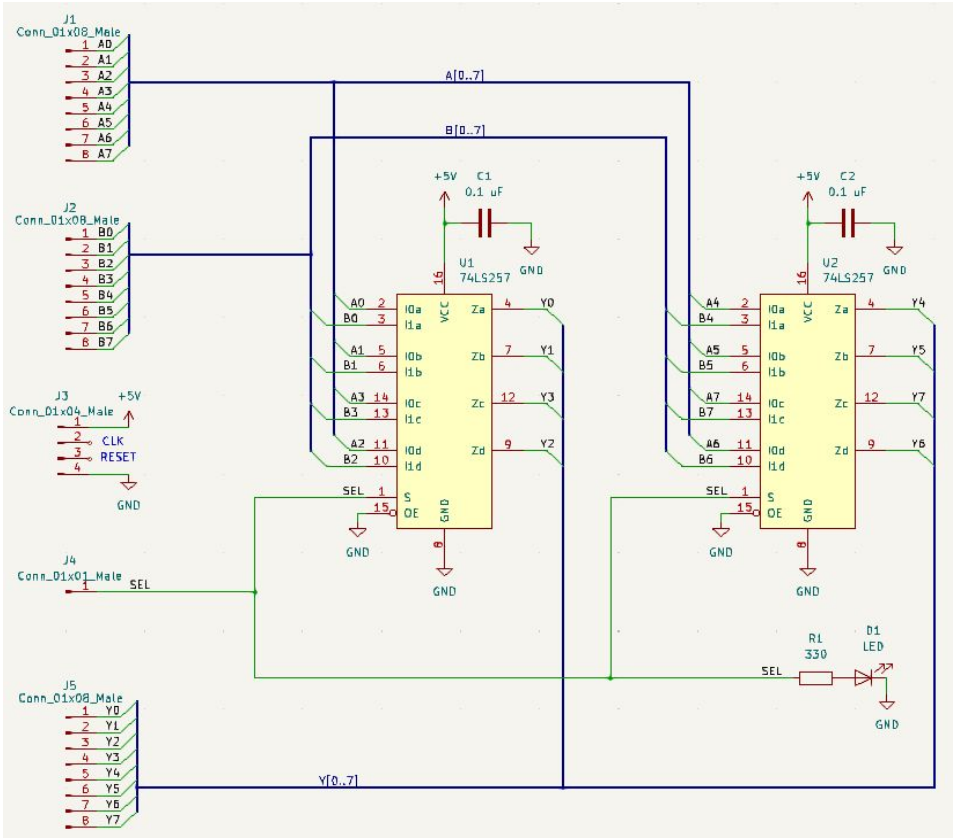
Design Process

Design Iterations



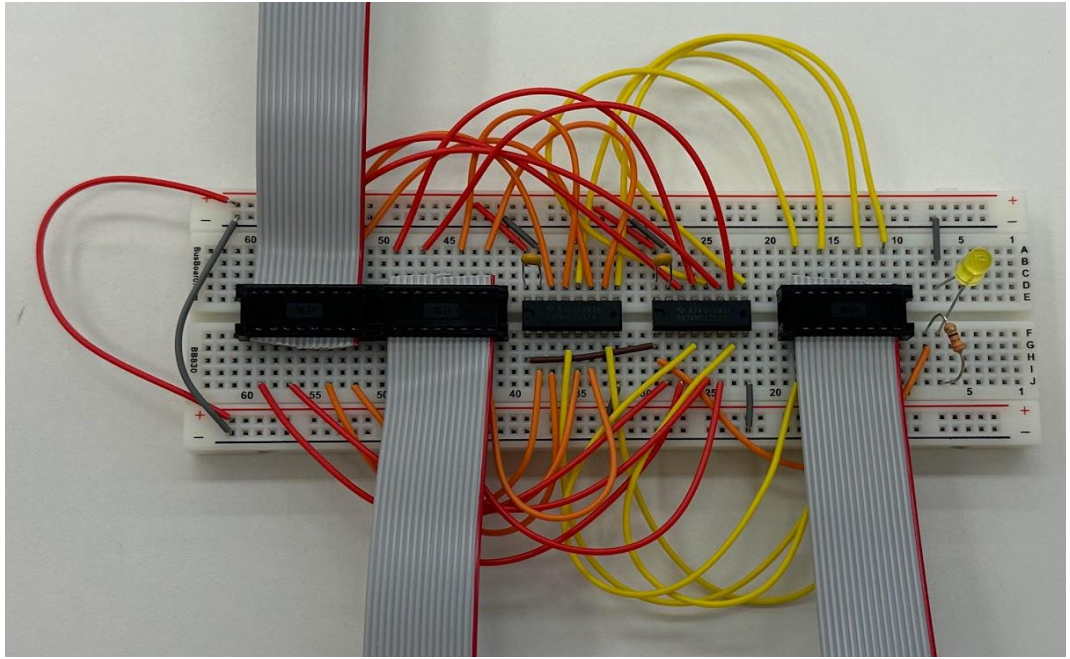
Flesh Out Lab Requirements

- What do we want to accomplish with this lab?
- What do the students need to learn?
- How can we accomplish this?
  - What activities would help the students learn?
  - What clarifications need to be made?
- Examples are from Lab 3.



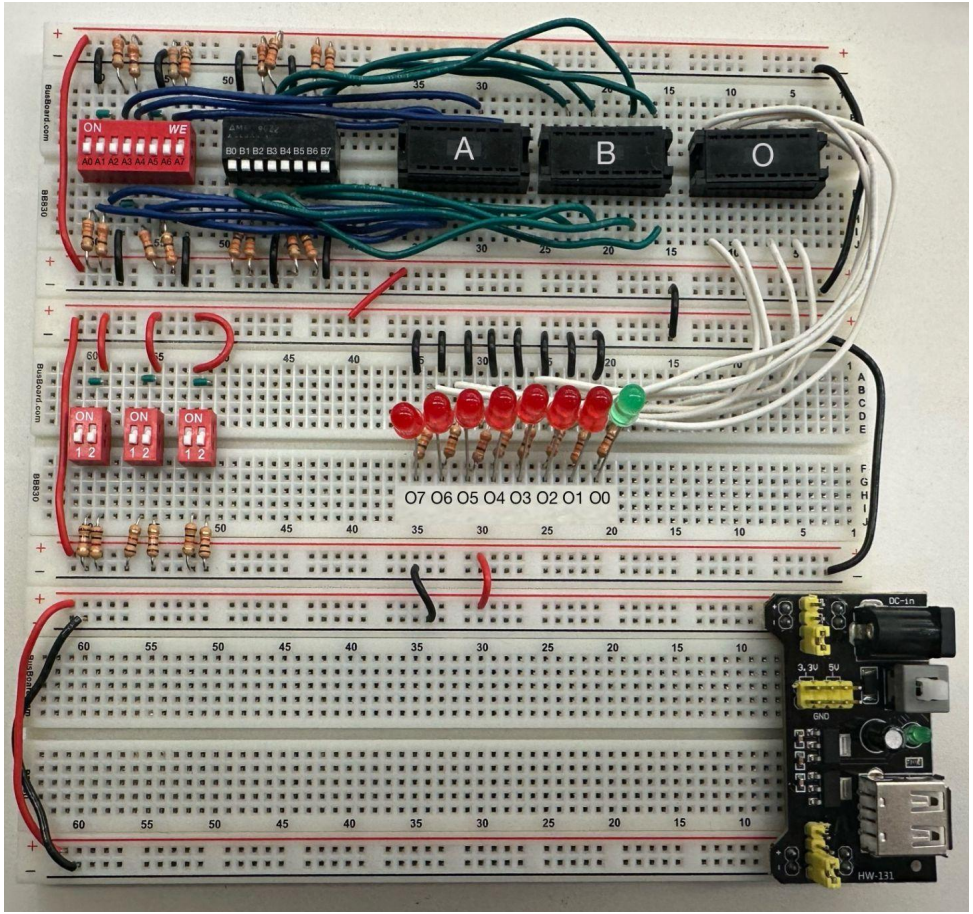
Build or Code Our First Iteration

- For Hardware Labs
  - Use pre-cut wires to build.
  - Verify that the parts on hand are correct.
- For Software Labs
  - Attempt the activity and note what steps or considerations are made.



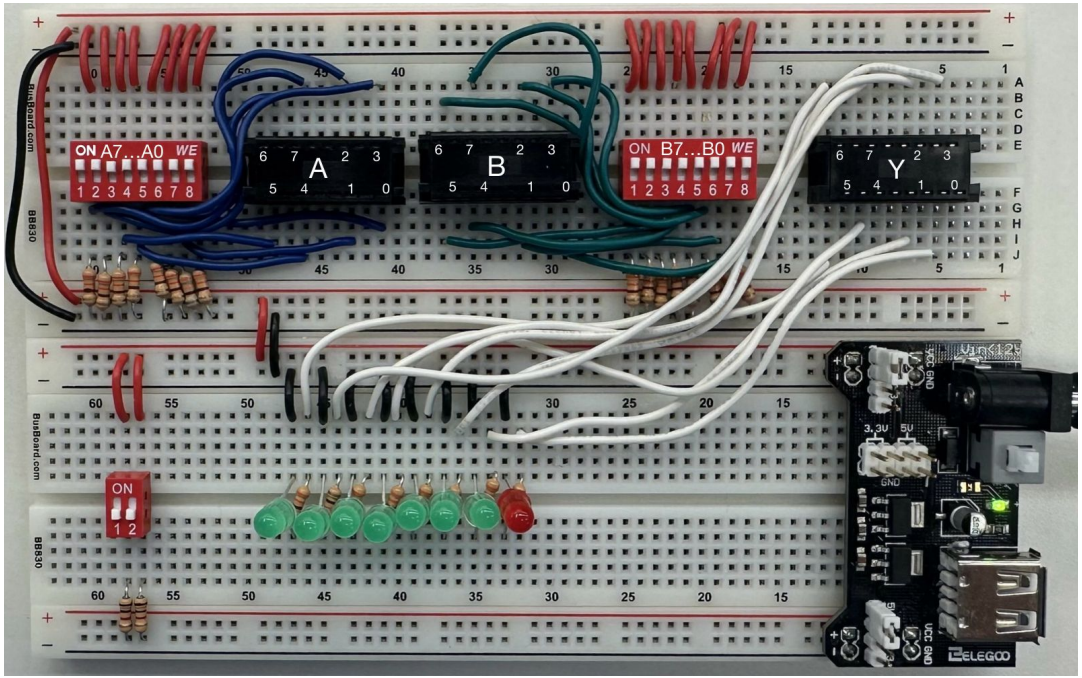
Test and Debug Circuit

- How can the students verify that their implementation is correct?
- For Hardware Labs
  - Created a tester circuit.
- For Software Labs
  - Created test cases for each activity.



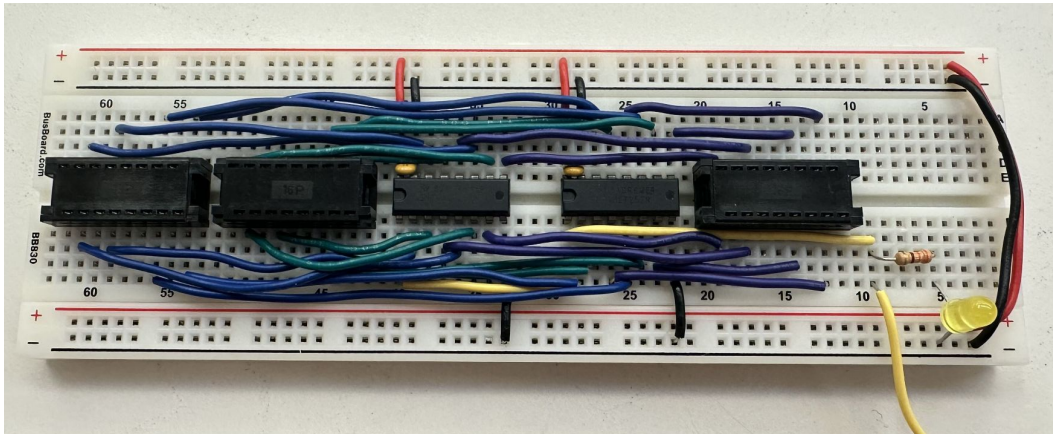
Test and Debug Circuit

- Through testing and development of more labs we refined the design of our tester circuit to adjust for changing needs.



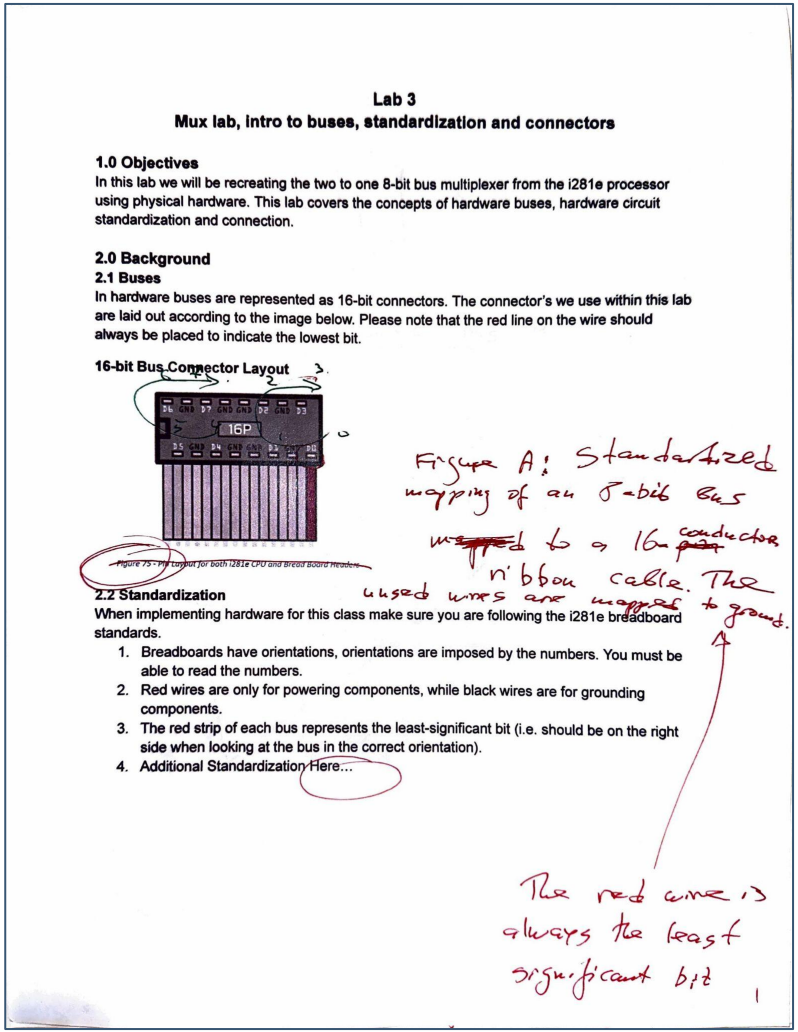
Get Feedback from Team and Client

- What improvements can be made to the implementation?
- Does the implementation need any changes?
- Are there any logic errors or flaws in the implementation that need to be addressed?



Document and Draft the Lab

- Document circuit designs or program requirements and create instructions.
- Add relevant background information and begin the pre-lab for the report.
- Finalize testing procedure for the lab.
- Get feedback from client on the draft.
- Begin revisions on the draft until it is satisfactory.



Lab 3

Mux lab, intro to buses, standardization and connectors

1.0 Objectives

In this lab we will be recreating the two to one 8-bit bus multiplexer from the i281e processor using physical hardware. This lab covers the concepts of hardware buses, hardware circuit standardization and connection.

2.0 Background

2.1 Buses

In hardware buses are represented as 16-bit connectors. The connector's we use within this lab are laid out according to the image below. Please note that the red line on the wire should always be placed to indicate the lowest bit.

16-bit Bus Connector Layout

Figure A: Standardized mapping of an 8-bit bus to a 16-conductor ribbon cable. The unused wires are mapped to ground.

2.2 Standardization

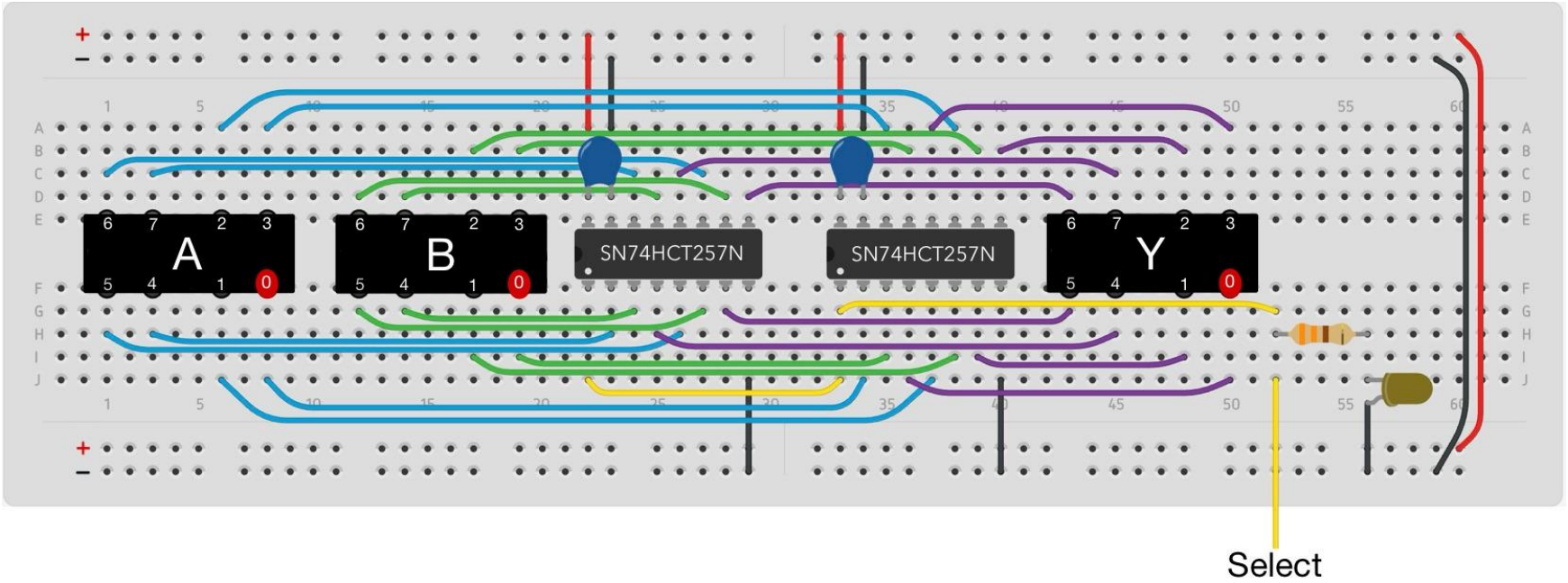
When implementing hardware for this class make sure you are following the i281e breadboard standards.

1. Breadboards have orientations, orientations are imposed by the numbers. You must be able to read the numbers.
2. Red wires are only for powering components, while black wires are for grounding components.
3. The red strip of each bus represents the least-significant bit (i.e. should be on the right side when looking at the bus in the correct orientation).
4. Additional Standardization Here...

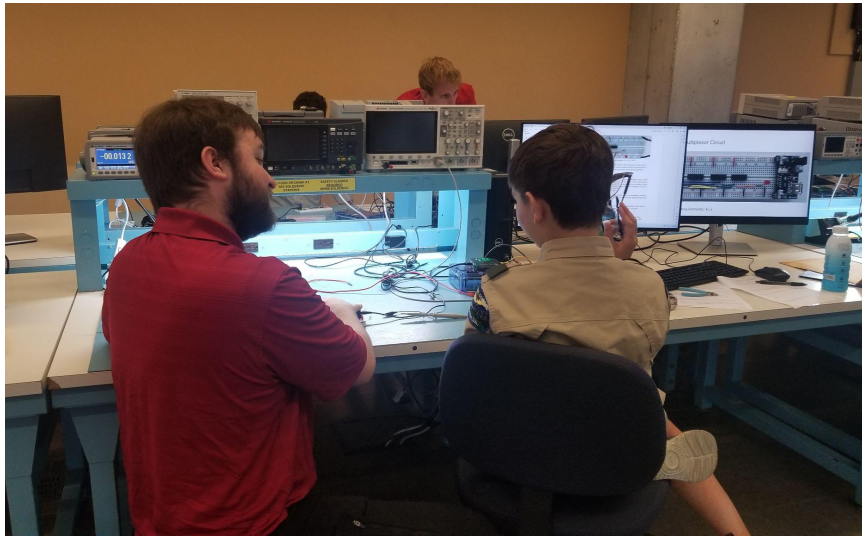
The red wire is always the least significant bit

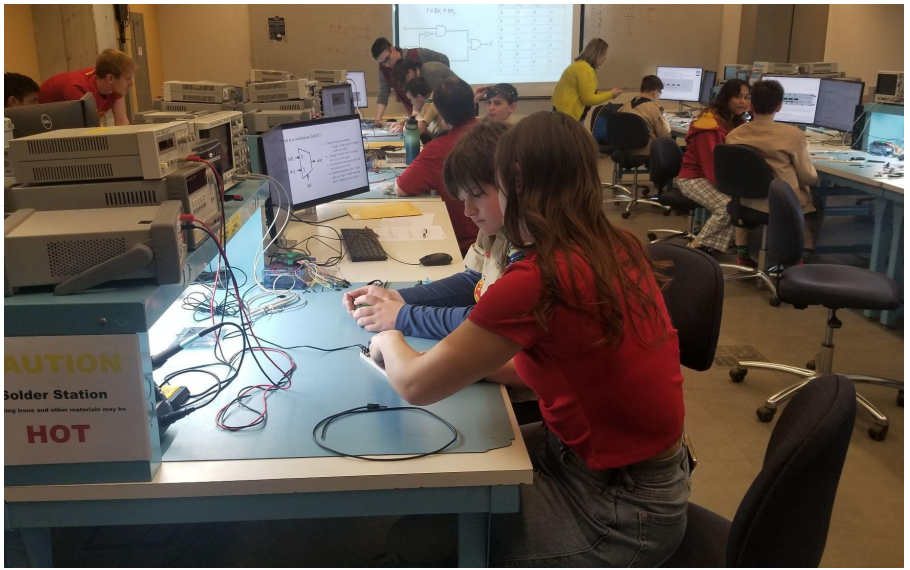
Test the Lab

- Have some volunteers follow along the lab
- See how long it takes them
- Get feedback



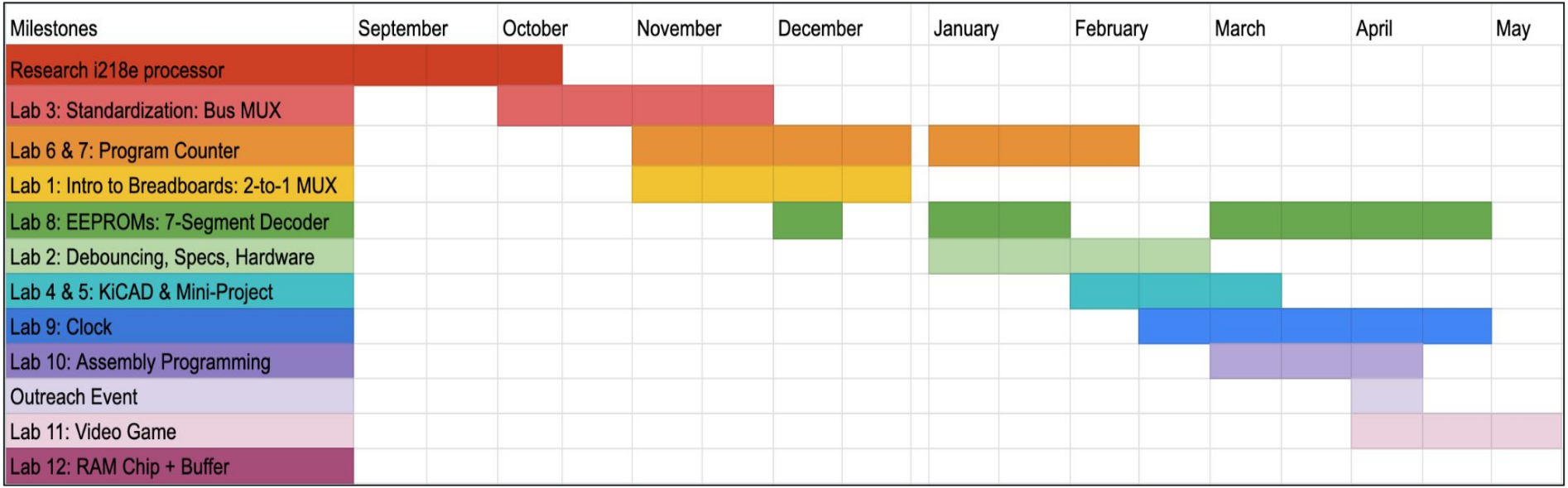
Outreach: BSA Electronic Merit Badge University, Iowa Space Grant



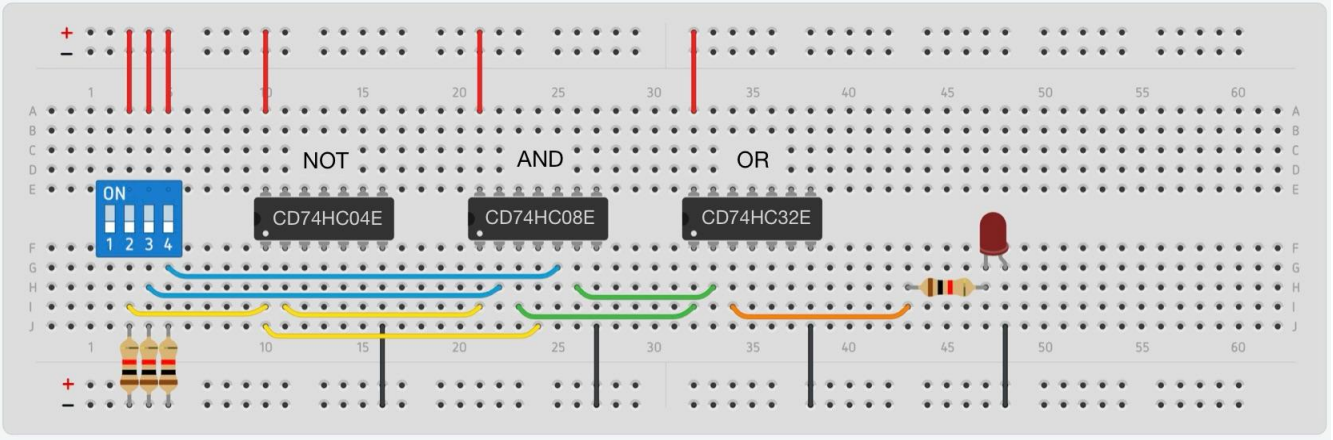


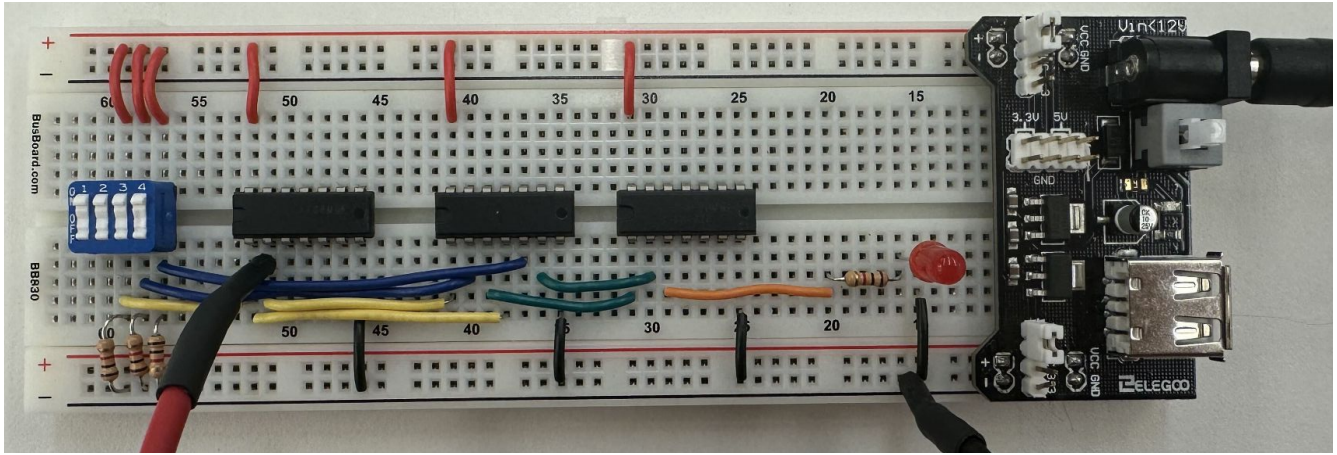
Accomplishments

Gantt Chart

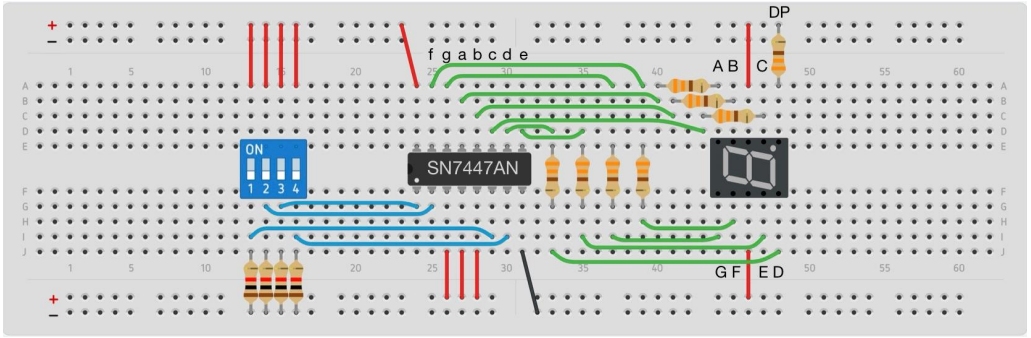

Milestones
September
October
November
December
January
February
March
April
May
Research i218e processor
Lab 3: Standardization: Bus MUX
Lab 6 & 7: Program Counter
Lab 1: Intro to Breadboards: 2-to-1 MUX
Lab 8: EEPROMs: 7-Segment Decoder
Lab 2: Debouncing, Specs, Hardware
Lab 4 & 5: KiCAD & Mini-Project
Lab 9: Clock
Lab 10: Assembly Programming
Outreach Event
Lab 11: Video Game
Lab 12: RAM Chip + Buffer

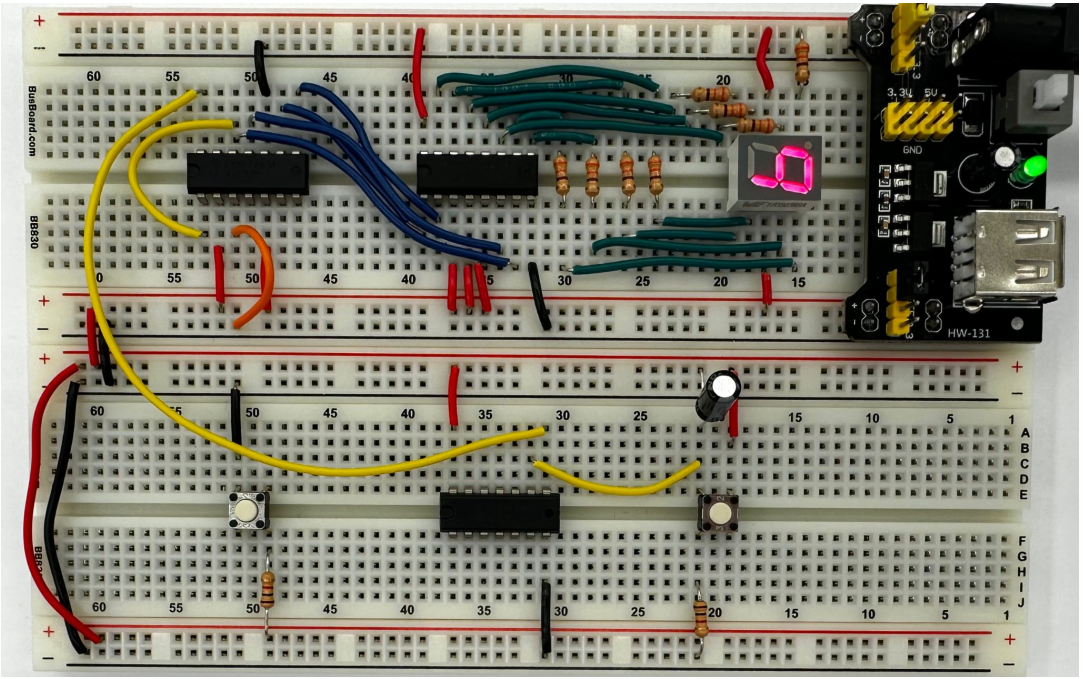
Lab 1: Bitwise Multiplexer



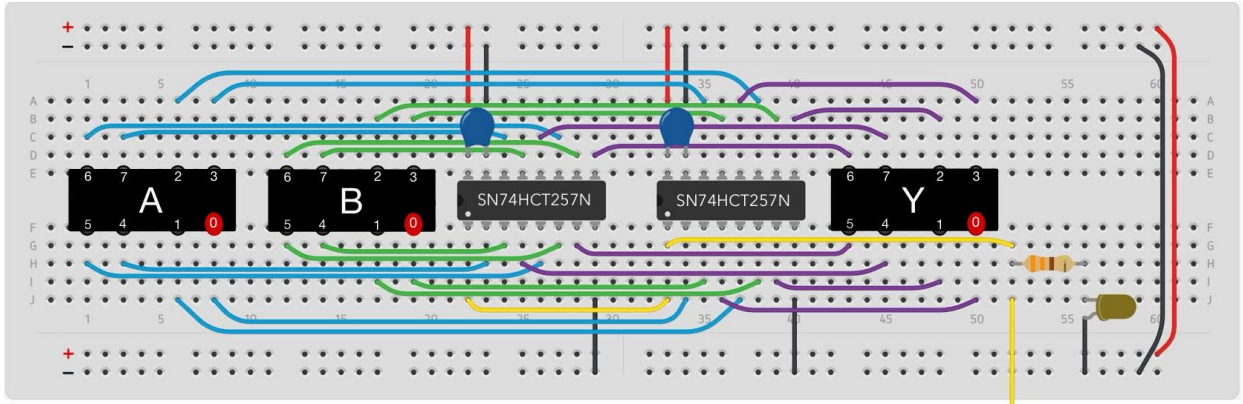


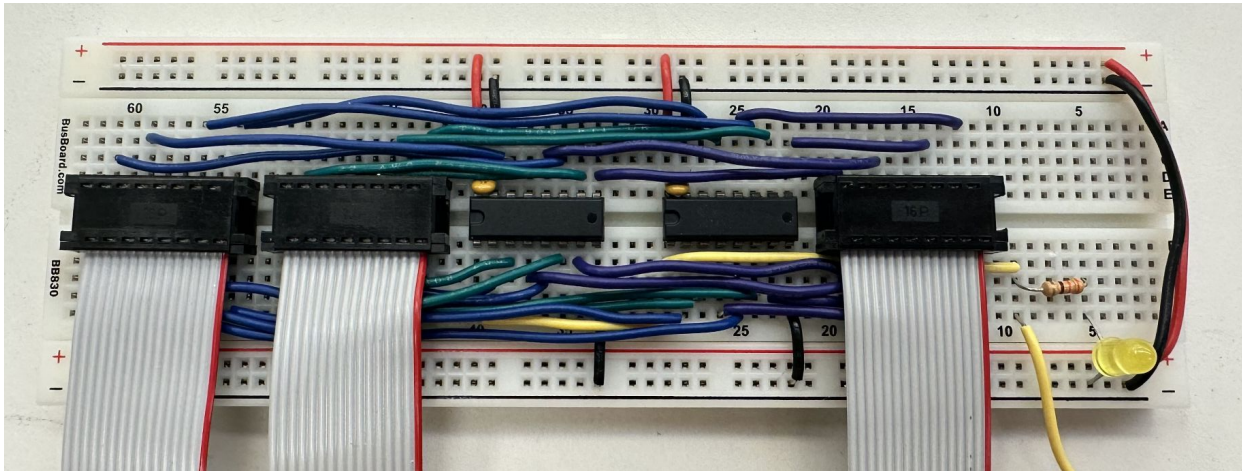
Lab 2:
Counting
& Debouncing



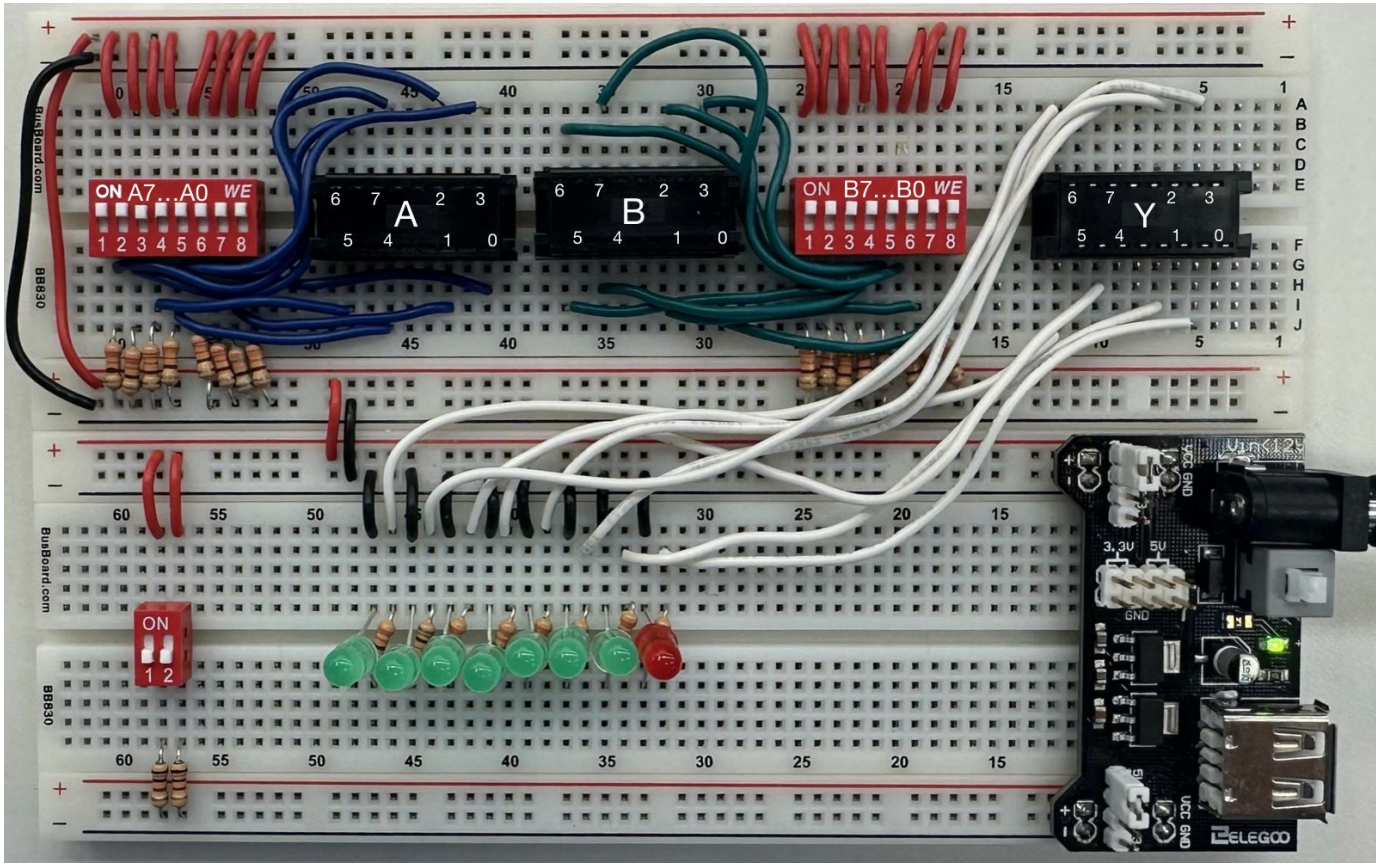


Lab 3: 8-Bit Bus Multiplexor

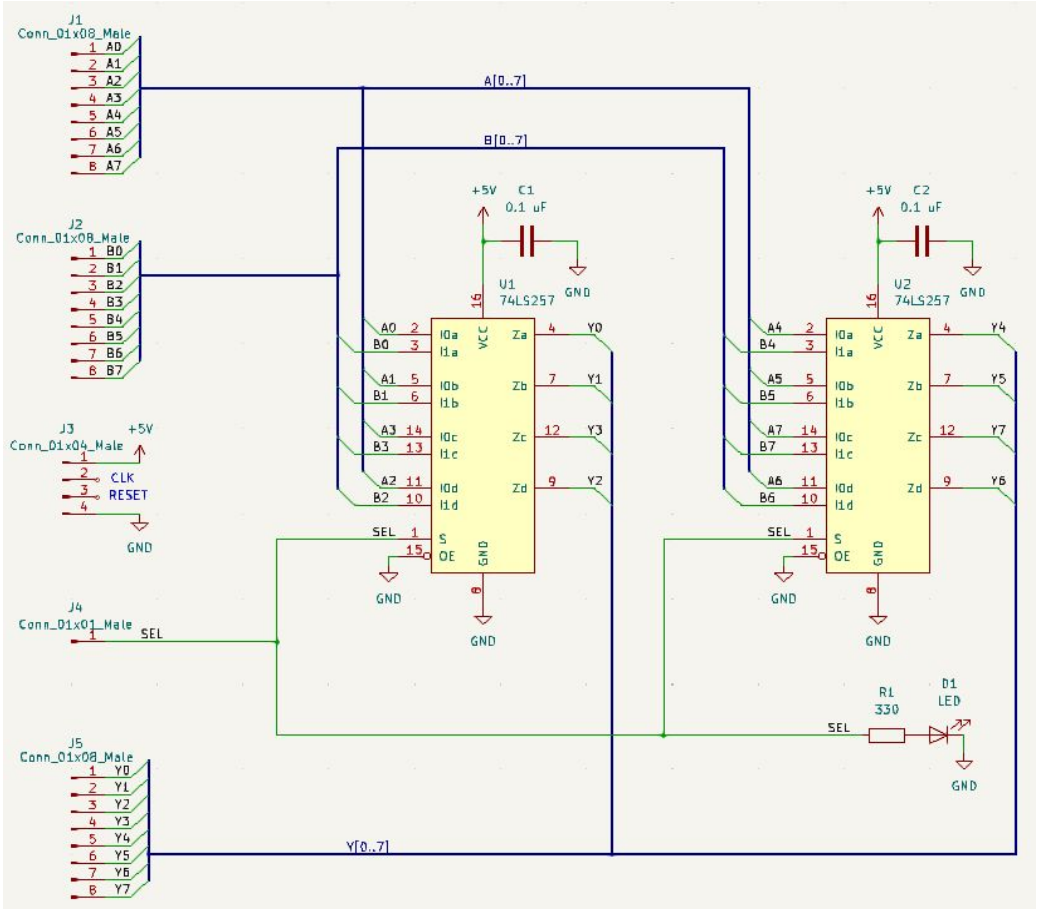




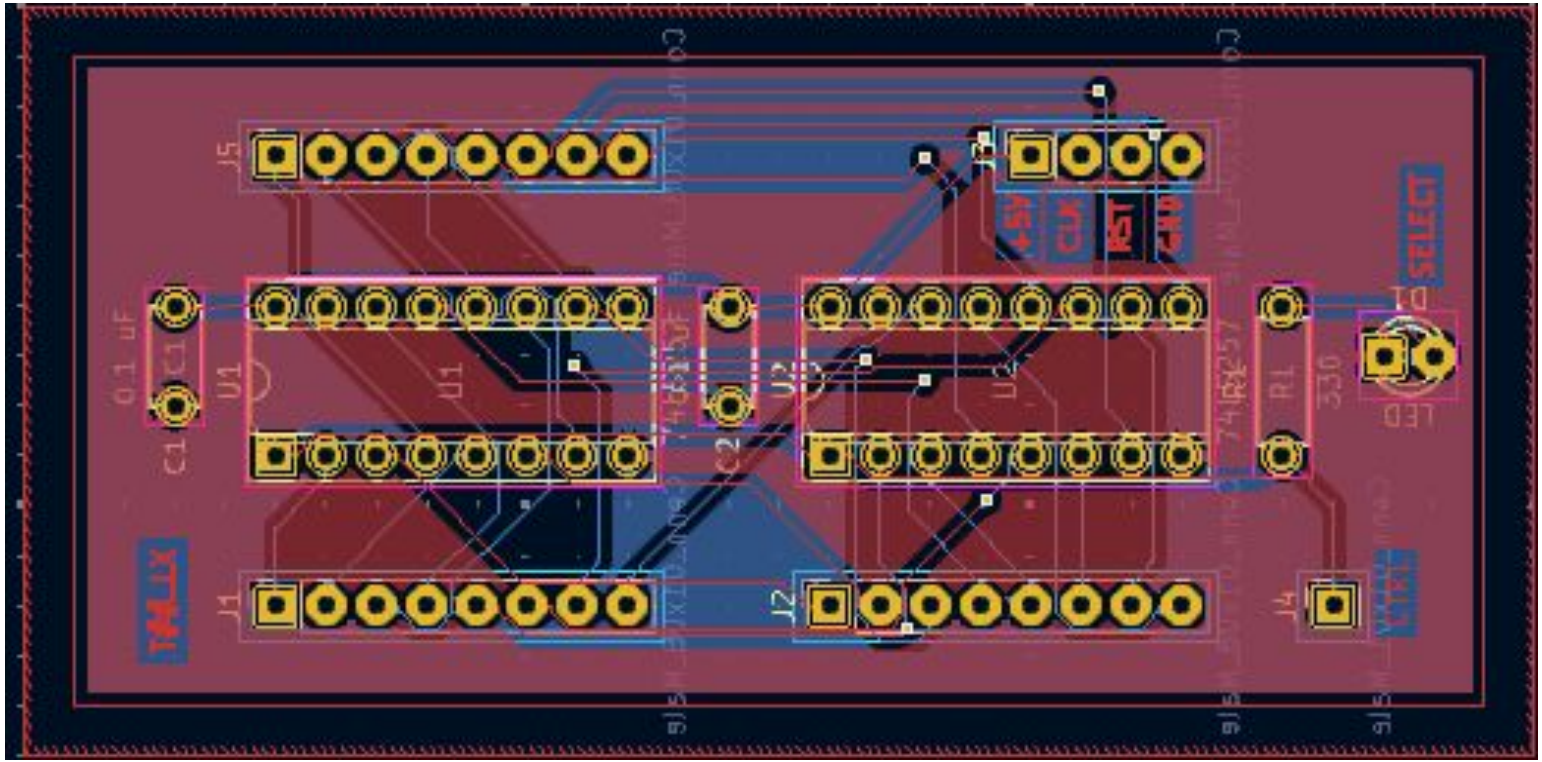
Test Circuit



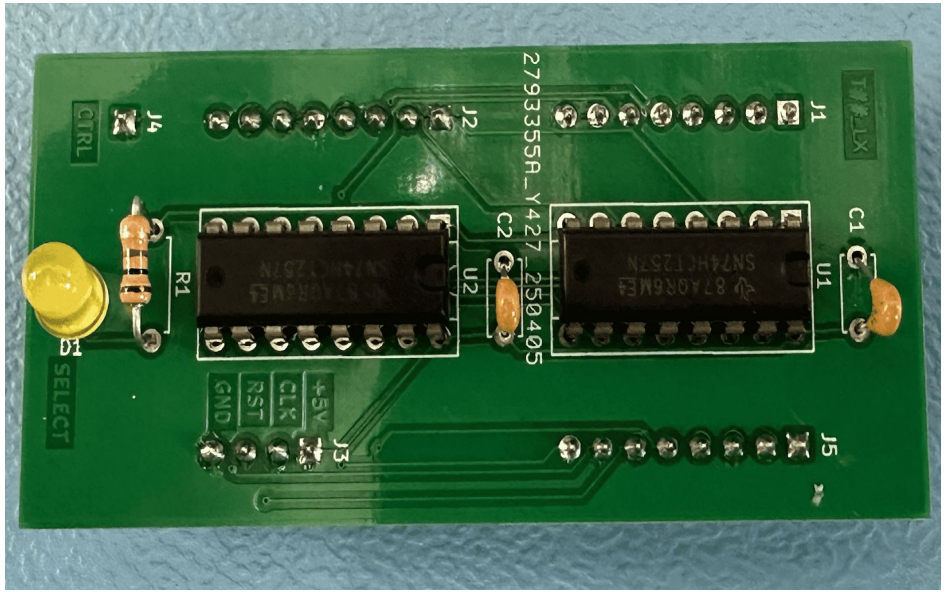
Lab 4:
Introduction to KiCAD
(Start of Mini-Project)

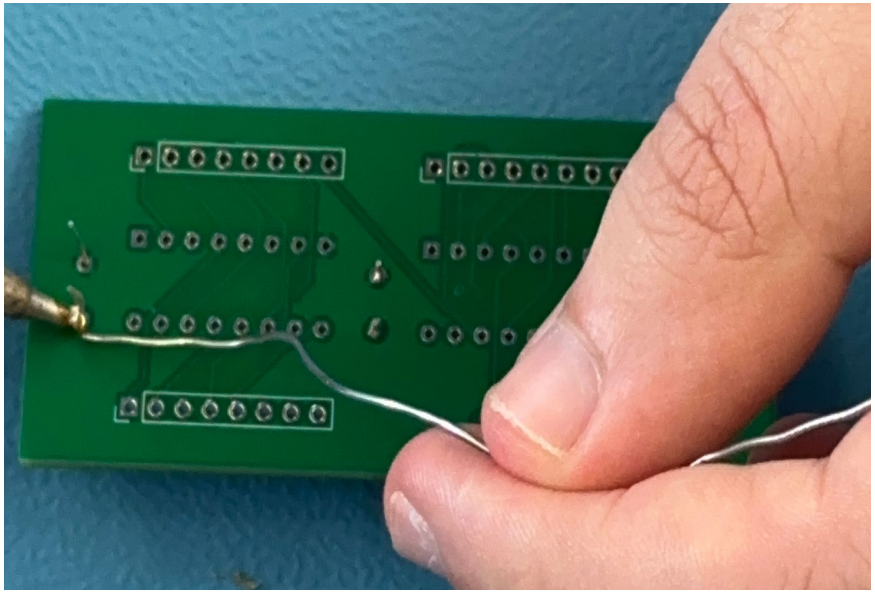


Lab 5: Routing a PCB

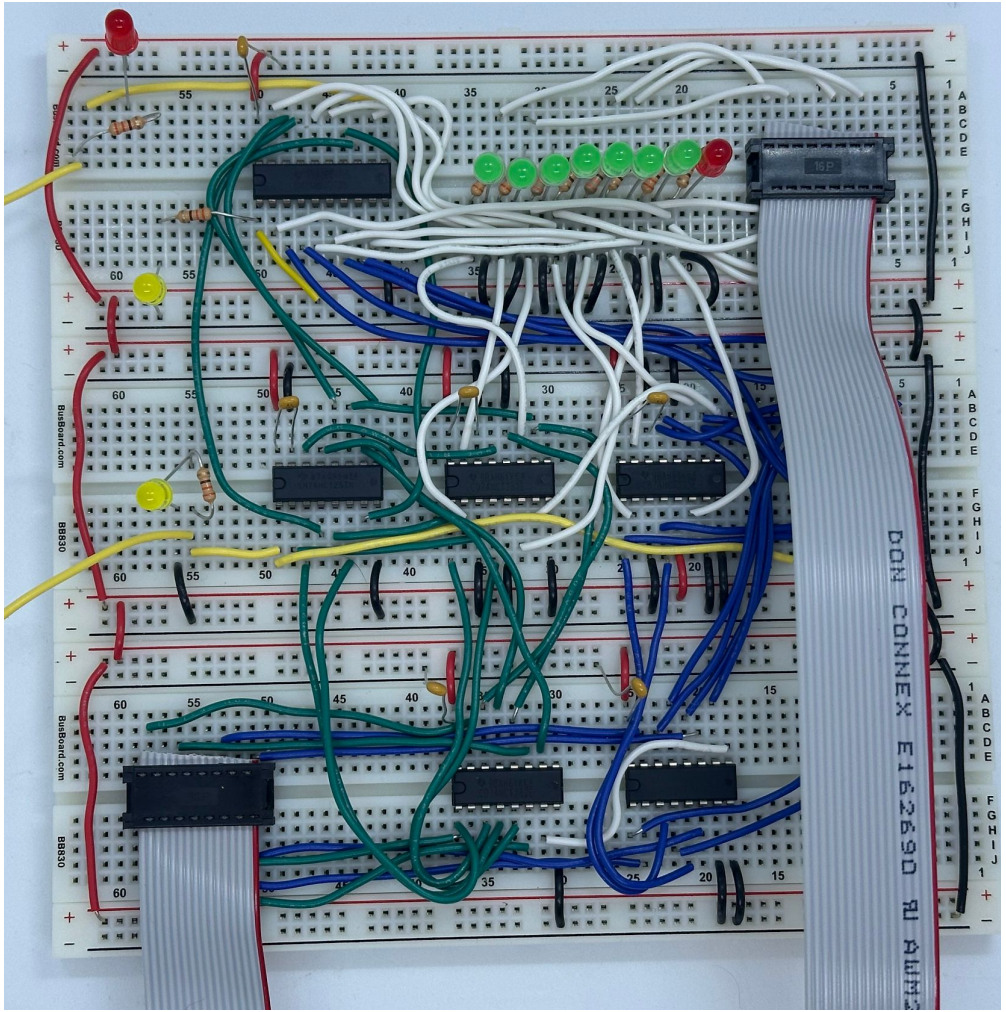


Mini-Project

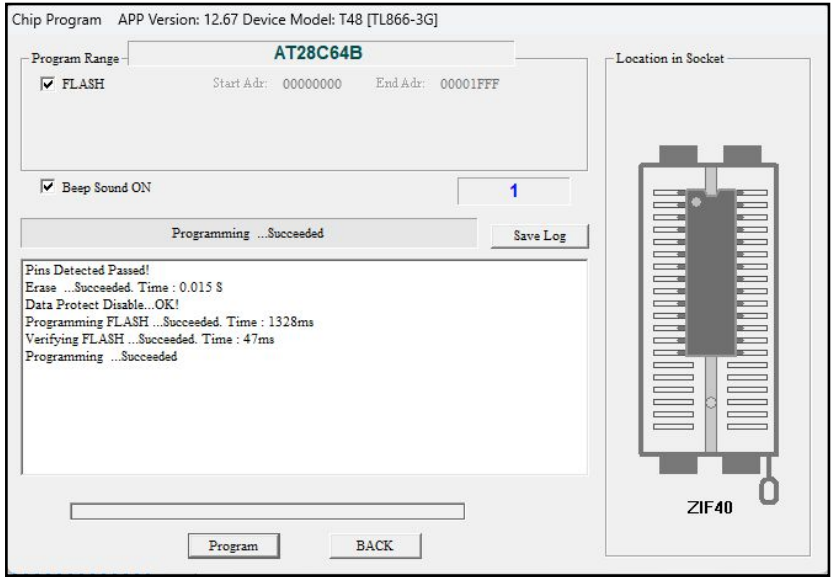


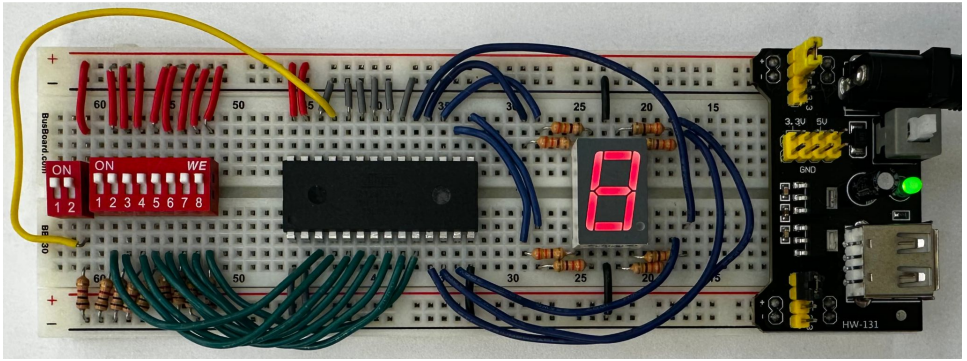


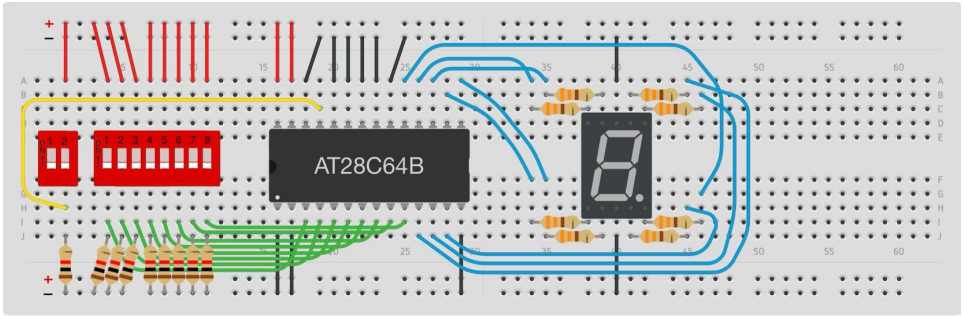
Lab 6 & 7:
Program Counter



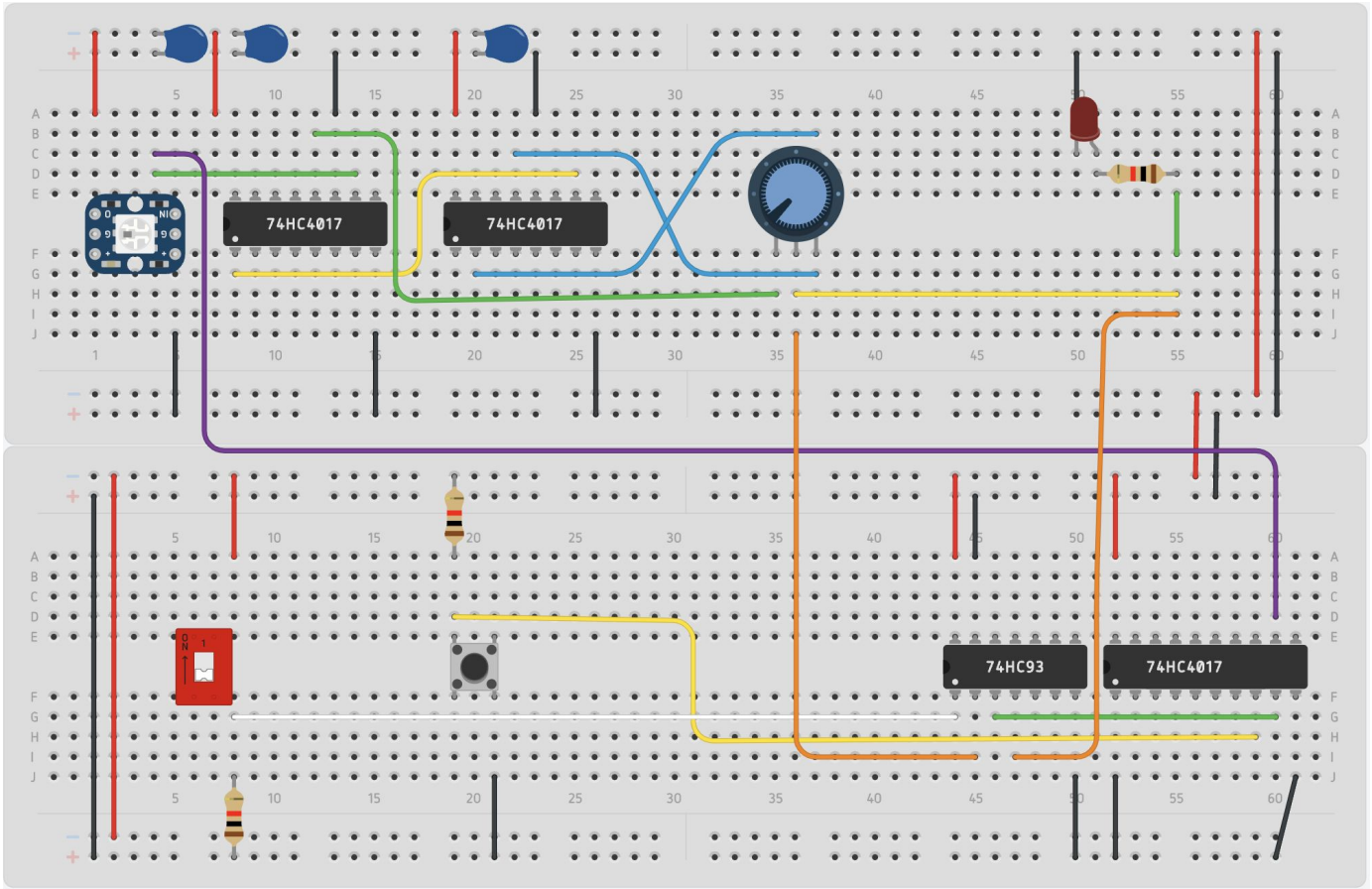
Lab 8:
EEPROM 7-Segment Decoder





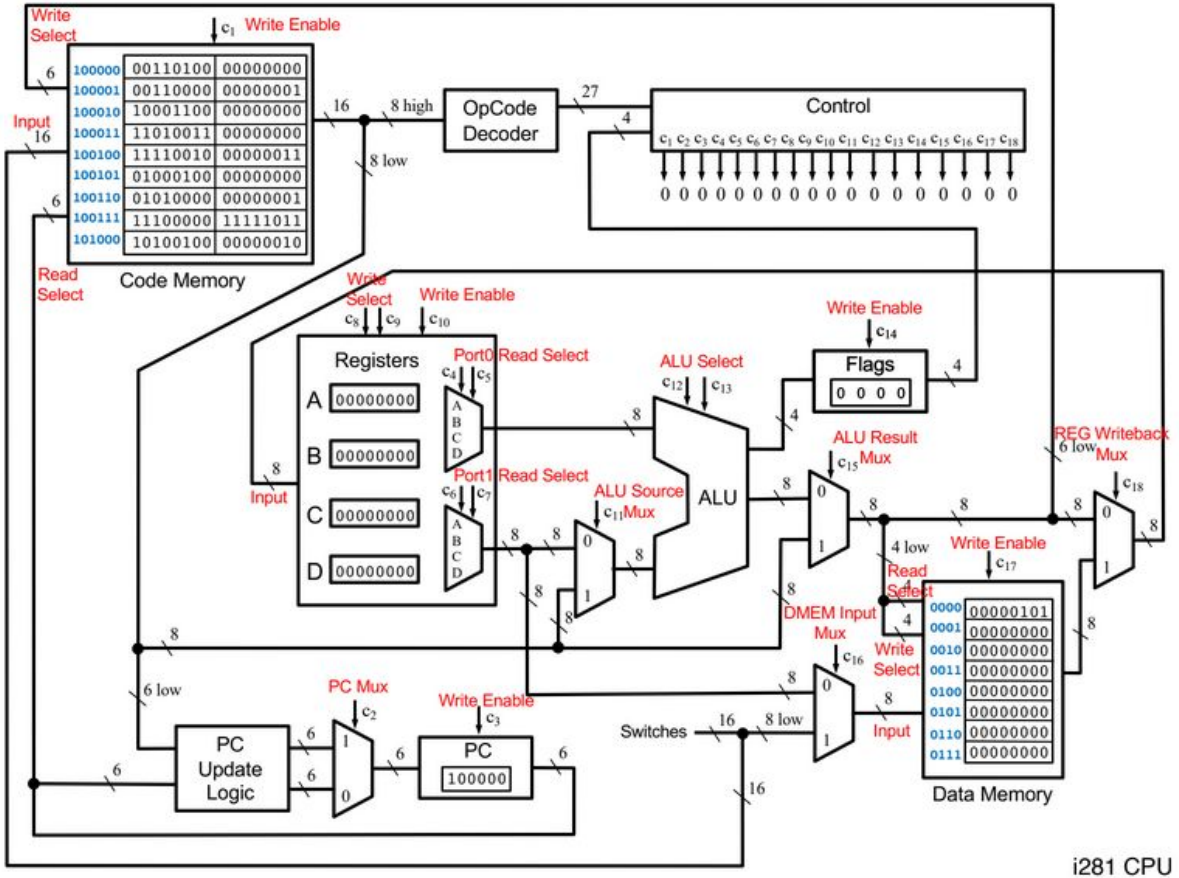


Lab 9: Clock



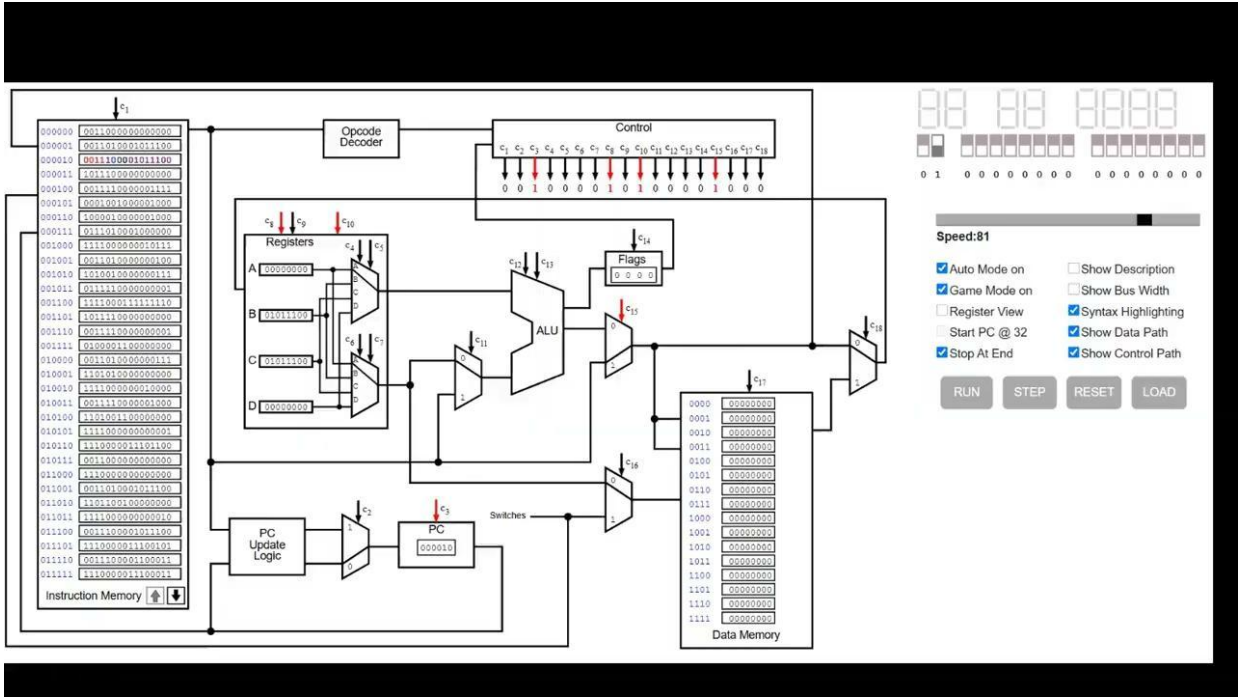
Lab 10: Assembly Level Programming

- Learn how to translate code from C to assembly that can run on the i281 simulator.



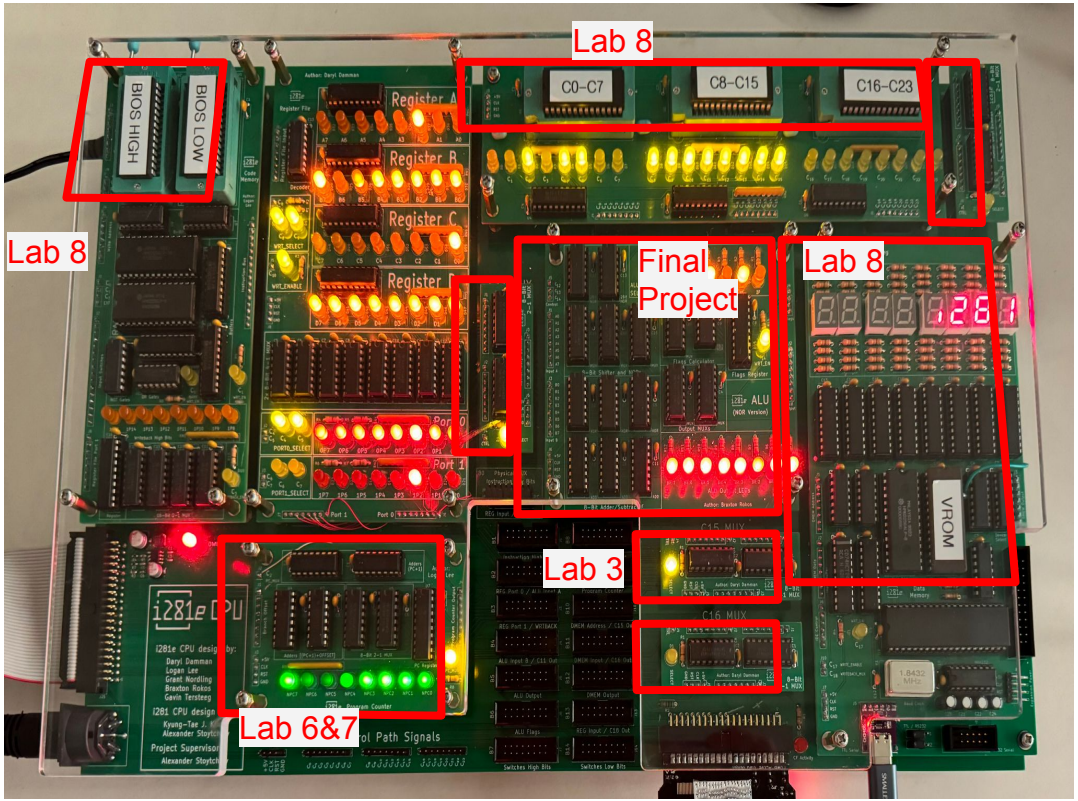
Lab 11: Video Game in Assembly

- Implement a version of the video game flappy bird in assembly.
- Teaches students to be mindful of the system architecture and available resources.



Conclusion

CprE 3710x Timeline by Week



1. Intro to Breadboards: 2-to-1 MUX
2. Counting, Decoding, & Debouncing
3. Standardization and Connectors: Bus MUX
4. Introduction to KiCAD (Start of Mini Project)
5. Mini-Project: Routing a PCB
6. Program Counter Part 1
7. Program Counter Part 2
8. EEPROMs: Program 7-Segment Decoder
9. Clock Circuit + Final Project Proposal
10. Assembly Level Programming
11. Video Game in Assembly
12. RAM Chips + Buffer
13. Thanksgiving Break
14. Final Project Pt 1
15. Final Project Pt 2

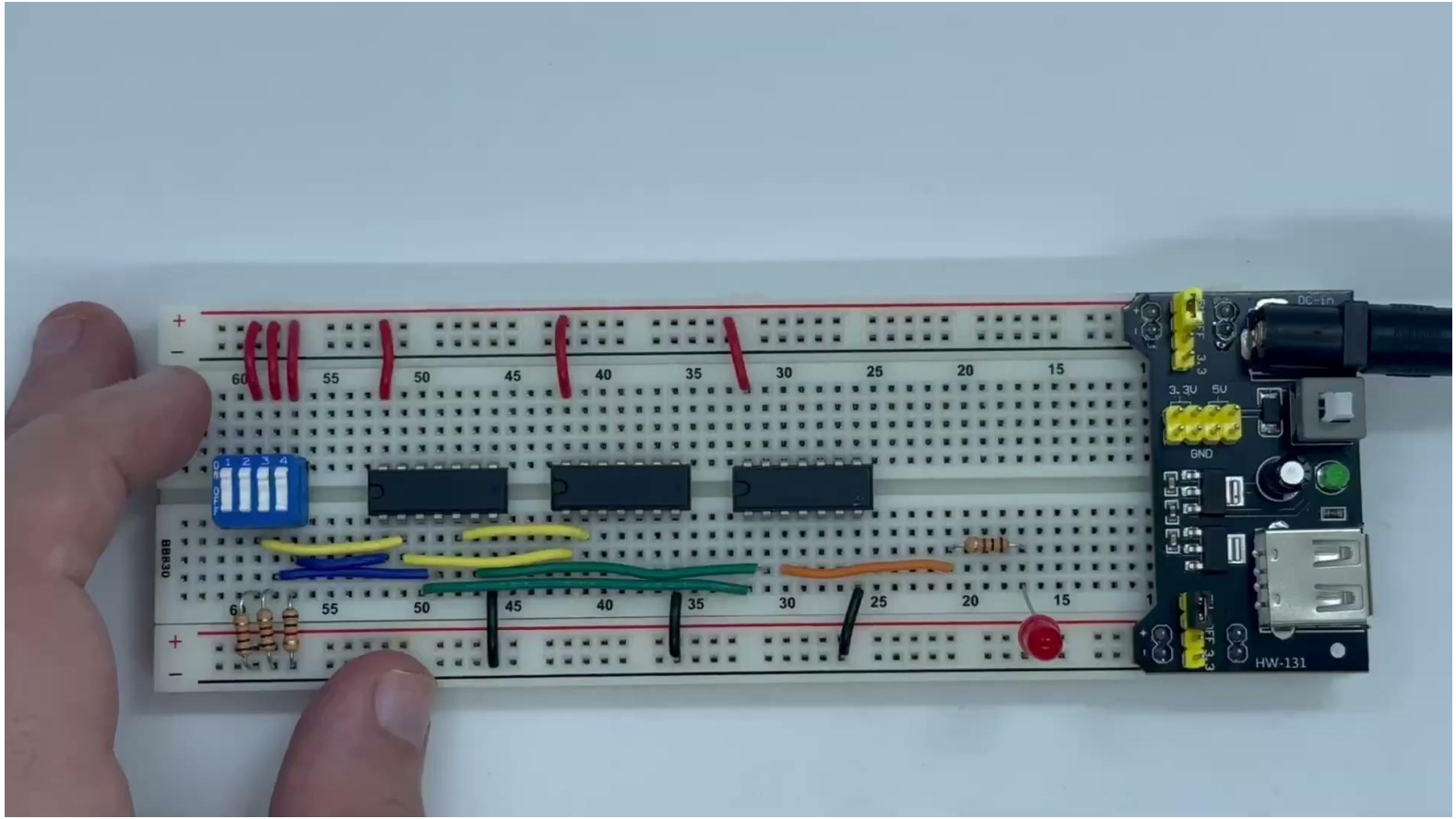
Source: https://sdmay24-14.sd.ece.iastate.edu/

Questions?

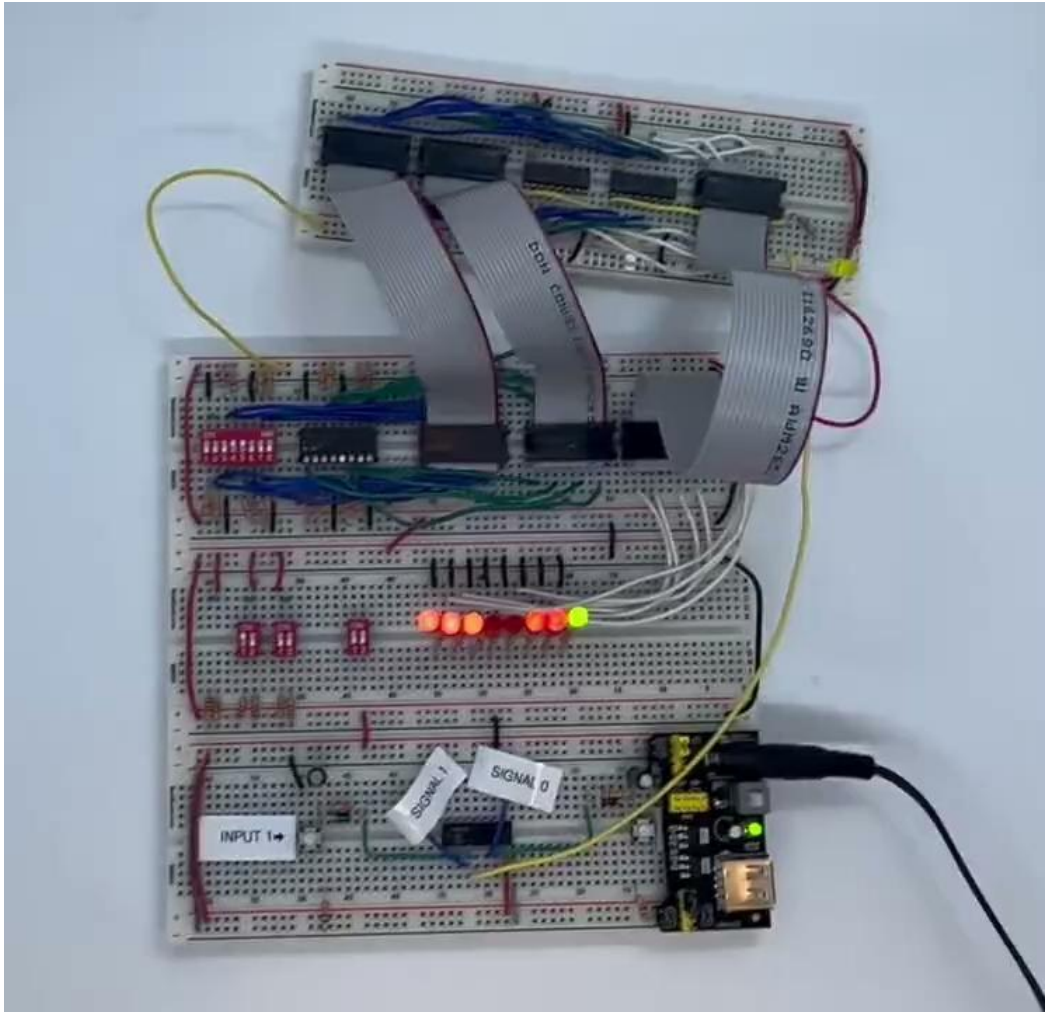
i281e Processor Specifications

- Clock Speed: 1Hz up to 2MHz (up to 2.5MHz overclock)
  - Processor fails around 2.75MHz
- Power Requirements: 0.8A @ 5VDC (Input 5-12VDC)
  - Fuse for overcurrent protection @ 2A
- Memory: 32 KW (64 KB) of Code RAM, 32 KB of Data RAM
  - Also includes 128 words of Code ROM for booting the system
- Compact Flash: extended memory for long-term storage
  - Acts as the “hard disk” and stores the OS and File System for DOS/281

Lab 1: Demo



Lab 3:
Bus MUX Demo



Lab 6 & 7:
Program Counter Demo