Jayhawk Engineering Company
3005 Eaton Hall, Lawrence, Kansas 66045

Integrated Electrical Multi-Tool Board

[IEM BOARD]

Final Report for EECS 542, Spring 2025

Tanner Gurley

Humzeh Al-Tamari

Zia Hosainzada

Max Djafarov

ABSTRACT

The Integrated Electrical Multi-tool Board (IEM Board) is a portable circuit prototyping platform designed to replace the need for traditional lab equipment like oscilloscopes, voltmeters, and power supplies in basic circuit analysis. The project aimed to create a compact solution that enables real-time voltage monitoring, power control, and data visualization for breadboard-based circuits, using a Raspberry Pi 5 and Arduino Mega stack.

Our team designed a modular system powered by a 24V 5A supply, integrating adjustable voltage regulation through a DROK buck converter, controlled via physical cabinet buttons and Raspberry Pi–driven MOSFET switching circuits. Voltage measurements were managed using Arduino Mega boards, providing multi-node sensing capabilities. The Raspberry Pi 5 handled graphical data visualization and user interaction through a custom-built Python GUI using Kivy and Matplotlib.

Over the course of the project, we successfully implemented real-time node polling, voltage-over-time graphing, programmable power sweep functionality, and session data management. Key challenges included accurate voltage regulation below 1.25V, optimizing I2C communication, and maintaining graphical responsiveness. The final prototype met our objectives, demonstrating a fully functional IEM Board capable of circuit testing with intuitive user control.

TABLE OF CONTENTS

1. Introduction and Background

1.1 Statement of Purpose

Modern circuit prototyping typically relies on benchtop equipment such as oscilloscopes, multimeters, and programmable power supplies. However, these devices are often expensive, immobile, and impractical for field testing or educational environments where space and resources are limited. As computer engineering students, we recognized the need for a portable, integrated solution that could replicate essential circuit analysis tools in a compact, accessible form factor.

The **Integrated Electrical Multi-tool Board (IEM Board)** was conceived to address these limitations. Our goal was to create a system that combines voltage monitoring, power regulation, and data visualization into a self-contained platform, eliminating the dependence on bulky laboratory instruments. Leveraging **Arduino Mega microcontrollers for voltage sensing** and a **Raspberry Pi 5 for graphical user interaction**, we aimed to deliver a product that allows users to build, test, and analyze circuits on a breadboard with minimal setup.

Throughout the design process, we drew inspiration from existing lab equipment but tailored our approach to prioritize **portability, user-friendliness, and scalability**. The project's evolution was driven by practical constraints—such as the decision to replace a complex transistor network with a more efficient Arduino Mega stack—and focused on delivering core functionality within the available time and resources.

1.2 Statement of Project Objectives

The primary objective of the IEM Board project was to design and build a **portable electrical engineering multi-tool** capable of assisting with circuit prototyping, testing, and data visualization. The system aimed to replicate key functionalities of lab equipment in a compact form, enabling users to prototype and analyze circuits without the need for external oscilloscopes, voltmeters, or power supplies.

Specific Goals:

- **Real-time voltage measurement** of multiple breadboard nodes using an Arduino Mega stack.
- **User-friendly graphical interface** developed on Raspberry Pi OS to display voltage readings and plots.
- **Programmable power supply control**, including adjustable output voltage and sweep mode functionality.
- **Physical interface integration**, enabling both manual and automated control via cabinet-mounted buttons and Raspberry Pi-driven MOSFET circuits.
- **Self-contained operation** powered by a single 24V, 5A wall adapter, minimizing setup complexity.
- **Modular and accessible housing** to securely mount components while maintaining ease of use.
- **Data management capabilities** to log voltage readings over time for analysis.

Final Deliverables:

- Fully functional IEM Board prototype with live demonstration.
- Voltage-over-time graphing via GUI.
- Programmable power sweep from 0V to 24V.

- Independent control of power adjustments through cabinet buttons and Raspberry Pi automation.

2. Theory and Design

2.1 Theory

The IEM Board functions as a **self-contained circuit prototyping and analysis tool**, designed to replicate the essential functions of lab equipment in a compact, portable form. The system integrates **voltage sensing, power regulation, and graphical data visualization**, allowing users to prototype and test circuits directly on a breadboard without external lab instruments.

Voltage Sensing

Voltage readings from user-placed breadboard components are captured using **Arduino Mega Rev3 boards**, which feature multiple analog inputs. The Arduino Mega's 10-bit ADC reads voltages (scaled appropriately using voltage dividers), and transmits this data to the **Raspberry Pi 5** via **Serial communication**.

Power Regulation

A **24V, 5A wall power supply** feeds into a **DROK adjustable buck converter**, which provides user-defined voltages (up to 24V DC) to the breadboard's power rails. Voltage adjustments are managed through both manual and automated control pathways, ensuring flexibility in operation.

User Interaction & Visualization

A **custom Python GUI**, built with the **Kivy framework**, runs on the Raspberry Pi. This interface allows users to:

- Select breadboard nodes for monitoring.
- Visualize voltage data in real-time graphs.
- Activate programmable sweep modes to analyze circuit response to changing voltages. Voltage readings are timestamped and stored, enabling voltage-over-time analysis similar to oscilloscope functions.

2.2 Design

Hardware Subsystems

- **Arduino Mega Stack (Voltage Sensing)**: Each Arduino Mega supports 16 analog inputs. For broader coverage, multiple boards can be stacked, with each assigned to read specific breadboard nodes. Voltage dividers scale down higher voltages to stay within the Arduino’s 5V ADC range.
- **Power Control Subsystem**: The DROK adjustable buck converter regulates voltage from 24V down to user-selected levels. Control signals for voltage adjustments are delivered in two ways:
    - **Physical Cabinet Buttons**: Hard-wired to the DROK's button terminals for manual user control.

    - **MOSFET Switching Circuit**: Raspberry Pi GPIO pins trigger MOSFETs, simulating button presses for automated control. Both control methods operate **independently**.
- **Power Distribution**: DC-DC buck converters provide regulated 5V and 12V outputs for Arduino and auxiliary circuits, ensuring stable operation.
- **Housing & Integration**: A custom-built enclosure holds the Raspberry Pi, Arduino stack, power modules, LCD, and breadboard. It provides organized port access, durable mounting, and space for cabinet buttons.

Software Subsystem

- **Raspberry Pi GUI**: Developed with Kivy, the GUI offers:
    - Node selection interface for voltage monitoring.
    - Real-time graph plotting using Matplotlib.
    - Power sweep control for dynamic voltage adjustments.
- **Data Handling**: The Raspberry Pi receives voltage data via Serial, stores readings with timestamps, and updates the graph dynamically. This emulates basic oscilloscope functionality.
- **Power Control Automation**: The GUI sends GPIO signals to the MOSFET circuit, enabling programmable sweep modes and digital voltage adjustments on the DROK converter.

2.3 Software Design/Development

The Breadboard Analytics application was developed using Python with the Kivy framework for GUI development, matplotlib for data visualization, and smbus for I2C communication. The software serves as a real-time interface between a Raspberry Pi and multiple Arduino Mega boards managing up to 95 analog voltage nodes.

Key Software Features:

- **Real-Time Voltage Graphing**: The app polls voltage data from user-specified nodes and displays real-time line plots with dynamic updates.
- **Interactive Queue System**: Users can manage a polling queue with clickable labels and trash icons to toggle visibility or remove nodes, respectively.
- **Settings Panel**: Accessible via a floating button, allowing toggling of graph features (log axes, peak annotations), theme switching, export options (PNG/PDF), and batch node entry.
- **Data Management**: The app supports polling single nodes, polling entire queues, real-time data buffering, voltage sweeps, and saving/loading CSV data.
- **Performance Optimizations**: Buffered I2C reads, throttled graph redraws, hardware acceleration (SDL2 backend), and production-mode settings ensure smooth operation on Raspberry Pi.
- **Theme Engine**: Multiple color themes (Light, Dark, Fallout, Electric Jayhawk) dynamically update UI components, including button styles, graphs, popups, and labels.

Software Flow:

- **Initialization**: Loads theme settings, initializes node queue, sets up the graph canvas and UI layout.
- **Polling Handlers**: Separate threads manage polling operations to prevent GUI blocking. Progress indicators inform users of polling status.
- **User Interactions**: Events like node entry, queue manipulation, and settings adjustments are handled through Kivy bindings.
- **Graph Rendering**: Voltage data is plotted using matplotlib, with real-time annotations and hover interactions for precise readings.

3. System Implementation

3.1 Hardware Implementation

The hardware implementation of the IEM Board consists of four key subsystems: **voltage sensing, power regulation, control circuitry, and housing integration**.

Voltage Sensing – Arduino Mega Stack

- Multiple **Arduino Mega Rev3 boards** were used to read voltage from breadboard nodes.
- Each board's **16 analog inputs** were utilized, with additional boards added as needed for node coverage.
- **Voltage dividers** were implemented to safely step-down voltages above 5V for accurate ADC readings.
- Data was sent to the Raspberry Pi via **Serial UART communication**.
- A bidirectional **Adafruit level shifter** was used to safely convert I²C signals between the **3.3V** Raspberry Pi and **5V** Arduino Mega.

Power Regulation – DROK Buck Converter & MOSFET Control

- A **DROK adjustable buck converter** received 24V input and provided regulated output (1.5-24V) to the breadboard power rails.
- Voltage adjustments were controlled in two ways:
    - **Physical cabinet-mounted buttons** wired directly to DROK button terminals.
    - **MOSFET switching circuit**, triggered by **Raspberry Pi GPIO pins**, to emulate button presses digitally.
- Both control paths function independently, allowing manual user input or automated adjustment from the Pi.

Power Distribution & Safety

- The primary **24V 5A wall supply** fed into **DC-DC buck converters** to create isolated **5V** and **12V rails**.
- Arduinos and auxiliary circuits were powered through these regulated outputs.
- Circuit protection elements (e.g., fuses, current limiters) were included to prevent damage during overcurrent scenarios.

Logic Level Shifting

- An Adafruit **bidirectional level shifter** was used to interface the **5V** logic of the Arduino Mega with the **3.3V** logic of the Raspberry Pi.
- Positioned on the I^2C lines (SDA and SCL), the module enabled safe, bidirectional voltage translation.
- This ensured stable communication and protected the Raspberry Pi's GPIO pins from overvoltage.

Enclosure & Physical Integration

- All components were mounted within a **custom-fabricated enclosure** designed for ease of use.
- Cabinet buttons were panel-mounted for user accessibility.
- Proper routing of wires and connectors ensured organized integration.
- Ports for USB peripherals, power, and the LCD screen were made easily accessible.

3.2 Software Development

The software system was developed on the **Raspberry Pi 5**, utilizing **Python** and the **Kivy framework** to create an intuitive, responsive graphical user interface (GUI). The system's primary functions involve **data acquisition over I2C**, **voltage visualization**, and **power control automation**.

Graphical User Interface (GUI)

- Built using **Kivy** for touchscreen-friendly interaction.
- Provides functionality for:
  - **Node selection** from available breadboard nodes.
  - **Real-time voltage graphing** using **Matplotlib**.
  - Activating **power sweep mode** with user-defined parameters.

Data Acquisition & Processing

- The Raspberry Pi communicates with the **Arduino Mega stack via I2C protocol**.
- The Pi functions as the **I2C master**, polling each Arduino Mega (slave devices) to retrieve voltage data.
- Voltage readings are:
  - Received in response to I2C read requests.
  - Timestamped and stored in arrays for visualization and logging.
- Data is dynamically plotted, emulating oscilloscope-like real-time voltage monitoring.

Power Sweep & Control Logic

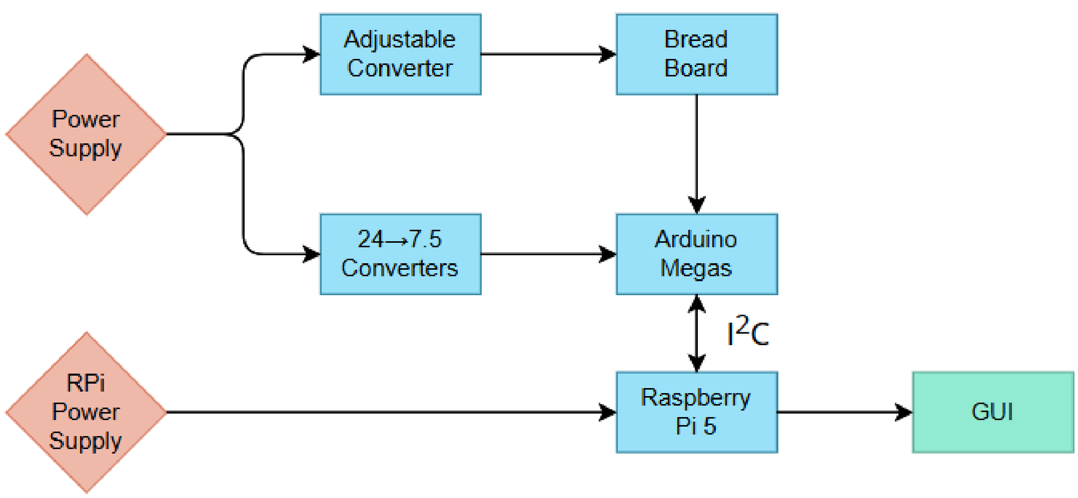
- GUI options allow users to initiate a **voltage sweep mode**.
- The Raspberry Pi sends **GPIO signals** to a **MOSFET switching circuit**, emulating DROK buck converter button presses.
- Sweep parameters (start voltage, end voltage, step intervals) are configured through the GUI.
- This enables automated, programmable control of the DROK's output voltage.

Performance Considerations

- Efficient **I2C communication routines** were developed to minimize bus congestion and ensure timely data retrieval.
- Graph update rates were throttled to balance **smooth visual updates** with **CPU resource management**.
- Data buffering methods were implemented to prevent loss of voltage readings during high-frequency polling.

3.3 System Interconnections

The IEM Board's functionality is achieved through the integration of four primary subsystems: **Voltage Sensing**, **Power Control**, **Data Processing & Visualization**, and **User Interface & Control**. These systems interact through carefully planned electrical connections and communication protocols.



Subsystem Interactions

- **Power Distribution**:
  - A **24V, 5A wall power supply** feeds into the **DROK adjustable buck converter**.
  - The buck converter regulates voltage (1.5-24V) to the **breadboard power rails**.
  - Cabinet buttons and a **MOSFET switching circuit** provide control inputs to the DROK's adjustment pins.

    - The MOSFETs are controlled by **Raspberry Pi GPIO signals**, enabling automated voltage adjustments.
- **Voltage Sensing**:
    - **Breadboard nodes** are tapped through **voltage divider circuits** to ensure safe ADC readings.
    - These nodes are read by the **Arduino Mega stack**, each functioning as an **I2C slave device**.
    - Voltage data is transmitted over the **I2C bus** to the **Raspberry Pi 5 (I2C master)**.
- **Data Processing & Visualization**:
    - The Raspberry Pi receives voltage data and processes it using **Python scripts**.
    - Data is visualized on the **LCD touchscreen** via a **Kivy-based GUI** with real-time graphing using **Matplotlib**.
    - Data arrays are maintained for historical voltage-over-time analysis.
- **Control Interface**:
    - The GUI provides user controls for:
        - Node selection for voltage monitoring.
        - Activating **voltage sweep mode**.
    - GPIO outputs from the Pi control the **MOSFET switching circuit**, simulating DROK button presses for automated voltage changes.

3.4 Technical Standards Employed

Communication Protocols

- **I2C Bus**: Handles voltage data acquisition from the Arduino Mega stack.
- **GPIO Control Signals**: Manage power sweep automation via the MOSFET circuit.
- **USB Peripherals (optional)**: Allow keyboard/mouse input for non-touchscreen control.

4. System Evaluation

4.1 Test Plan

To verify the functionality and performance of the IEM Board, we established a comprehensive test plan focused on validating each subsystem independently before full system integration.

Voltage Sensing Tests:

- Verified accuracy of **Arduino Mega ADC readings** against a calibrated multimeter.
- Tested multiple node voltage readings simultaneously to ensure **I2C communication reliability**.
- Evaluated voltage divider scaling to confirm safe and accurate readings within 0-5V ADC range.

Power Control Tests:

- Confirmed DROK buck converter output voltage consistency with target setpoints.
- Tested **manual control** via cabinet-mounted buttons.
- Validated **automated control** through Raspberry Pi GPIO-driven MOSFET switching, simulating button presses accurately.
- Ensured **independent operation** of manual and automated control methods.

GUI Functionality Tests:

- Evaluated responsiveness of the **Kivy-based GUI** on the Raspberry Pi touchscreen.
- Verified real-time plotting of voltage-over-time data with Matplotlib.
- Tested user interactions for **node selection** and **power sweep activation**.
- Monitored system performance under typical usage to ensure smooth graphical updates without lag.

Full System Integration Tests:

- Conducted end-to-end tests from powering on the system to monitoring voltage across nodes while adjusting output voltage.
- Performed **power sweep demonstrations**, verifying voltage changes reflected in real-time graphs.
- Tested **data logging** of voltage readings with accurate timestamps.

.

4.2 Performance Evaluation

Voltage Measurement Accuracy:

- Achieved voltage reading accuracy within **±10mV** after calibration.
- Minimal data loss observed during I2C polling at standard refresh rates.

Power Regulation Stability:

- DROK converter maintained output voltage within **±50mV** of target value under typical breadboard load conditions.
- Sweep mode functioned smoothly across 0V to 15V range without overshoot or instability.

Software Responsiveness:

- Real-time graph updates remained smooth up to **8 simultaneous node readings**.
- GUI remained responsive with typical user interactions, including during active power sweeps.

System Limitations:

- Voltage readings below **0.1V** exhibited minor noise due to hardware limitations of the voltage divider scaling.

- Sweep speed limited by DROK converter button emulation latency (mechanical button timing constraints).

  Final Outcome:
- The IEM Board successfully demonstrated all planned functionalities, including **multi-node voltage monitoring**, **programmable power control**, and **real-time data visualization**.
- The system proved stable, reliable, and aligned with project objectives for a portable circuit prototyping tool

5. Analysis and Conclusions

5.1 Cost

Unit	Unit Quantity	Price Per Unit	Total Cost
Raspberry Pi 5	2	80.00	160
RPi Power Supply	2	6.75	13.50
32 Gb microSD	2	53.70	107.40
RPi 7” LCD Screen	2	60.00	120.00
RPi Keyboard	2	17.52	35.04
RPi Mouse	2	10.44	20.88
Micro to HDMI cable	2	1.26	2.52
Full Breadboard	2	6.50	13.00
24V 5A Power Adapter	1	19.99	19.99
Arduino Mega Rev3	9	48.40	435.60
Resistor 56K Ohm 2W	160	0.0862	13.79
Resistor 5K6 Ohm 1W	160	0.5066	81.06
Resistor 4K7 Ohm 3W	250	0.09632	24.08
Switch – Push Button	2	9.56	19.12
Switch – Rocker	1	0.71	0.71
DC – DC Buck Converter 24V to 7.5V	10	4.95	49.50
DROK Adjustable Buck Converter	1	16.99	16.99
Voltage Divider PCBs	10	33.00	330.00
Inner/Outer Case Structure (3D Printing Materials)	N/A	N/A	~300.00
TOTAL COST			~1763.18

5.2 Reliability

- **Robust Power Distribution**: The system maintained stable voltage outputs with current limiting measures to protect sensitive components.
- **Arduino Mega Stack Modularity**: Independent voltage sensing modules allowed for scalable node coverage and easy replacement if needed.
- **Dual Control Redundancy**: Manual cabinet buttons ensured reliable operation even if Raspberry Pi automation failed.
- **Mechanical Limitations**: DROK button emulation via MOSFETs introduced some latency, though manageable for educational and prototyping use.

5.3 Risk and Safety, Health and Environment

Risk	Impact	Mitigation Strategy
Voltage measurement inaccuracies	Medium	Calibration against multimeters; voltage divider tuning.
Power sweep timing mismatch (DROK emulation)	Low	Adjusted GPIO timing to match button press intervals.
Overcurrent damage to breadboard circuits	High	Implemented current limiting and fuse protection.
I2C communication delays with multiple devices	Medium	Optimized polling frequency; staggered read requests.
GUI performance bottlenecks on Raspberry Pi	Low	Throttled graph refresh rates; optimized Python routines.

5.4 Conclusions

The IEM Board successfully met its design objectives as a **portable electrical multi-tool**, integrating **real-time voltage monitoring, programmable power control, and user-friendly data visualization**. Key achievements included:

- Functional **multi-node voltage sensing** with I2C communication.
- Dual-mode **manual and automated power adjustment**.
- A responsive **Python-based GUI** for live voltage plotting and control.
- A **compact, standalone platform** powered from a single 24V wall supply.

The project demonstrated the feasibility of replicating essential lab equipment functions in a portable, cost-effective system suitable for students and engineers prototyping in non-lab environments. While certain limitations (e.g., DROK emulation latency, ADC noise at low voltages) were encountered, they remained within acceptable margins for intended educational and prototyping applications.

6. References

Include citations for material used in preparation of the experiment or report. Number each citation and provide author, title, publication and date.

[1] Arduino, *Arduino Mega 2560 Rev3*, Arduino.cc, 2024. [Online]. Available: https://store.arduino.cc/products/arduino-mega-2560-rev3
[2] Adafruit Industries, *TXB0108 8-bit Bidirectional Logic Level Shifter*, Adafruit, 2024. [Online]. Available: https://www.adafruit.com/product/395
[3] DROK, *DC-DC Boost Converter, Adjustable Voltage Regulator with LCD Display*, Amazon.com, 2024. [Online]. Available: https://www.amazon.com/Converter-DROK-Adjustable-Regulator-Transformer/dp/B08RBN31DT.

[4] Raspberry Pi Foundation, *Raspberry Pi 5 – Product Documentation*, RaspberryPi.com, 2024. [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi-5.html

[5] Kivy Developers, *Kivy: Cross-platform Python Framework for NUI Development*, Kivy.org, 2024. [Online]. Available: https://kivy.org/

[6] Hunter, J. D. et al., *Matplotlib: Visualization with Python*, matplotlib.org, 2024. [Online]. Available: https://matplotlib.org/