YRGO

P2-Multimeter

Report Author: Niklas Brodén
Instructor: Erik Jagre
Course Name: Applied Electronics
Report Date: 2024-12-18

YRGO

Summary

The P2 Multimeter Project successfully delivered a functional and accurate tool for measuring voltage, resistance, and battery levels. The design process combined prototyping, testing, and component selection to achieve reliable performance. The final multimeter included a user-friendly interface with visual and auditory feedback through LEDs and a buzzer, as well as mode selection using a rotary switch.

The PCB design optimized space by placing the battery holder on the backside and organizing components efficiently on the front. High voltage and resistance measurement circuits, along with battery monitoring, were implemented with precision, ensuring minimal errors during testing. Breadboard prototyping and simulation guided the selection of resistors and validated the performance of each circuit before final assembly.

During testing, calibration adjustments and real-world validation confirmed that the multimeter performed within acceptable error margins. Minor challenges, such as silkscreen labeling errors and the high voltage divider limitation to 33.7V, were identified and addressed. These insights highlighted opportunities for future improvements, including enhanced ADC resolution, extended voltage range, and overvoltage protection.

In conclusion, the P2 Multimeter met its goals and offered valuable learning experiences. The project highlighted the importance of careful design, thorough testing, and user-friendly features, creating a solid base for future projects.

YRGO

Introduction

The P2 Multimeter Project aims to design and develop a functional multimeter capable of measuring voltage, resistance, and battery levels. The project focuses on creating an accurate, user-friendly, and reliable device using an Arduino Uno as the main controller, combined with a custom-designed PCB layout. This multimeter incorporates key features such as a rotary switch for mode selection, a 7-segment display for output visualization, and LEDs for clear mode indication.

The motivation behind the project was to provide a practical learning experience in circuit design, PCB manufacturing, and testing processes. By designing the multimeter from scratch, the project emphasizes the importance of combining theoretical knowledge with hands-on implementation.

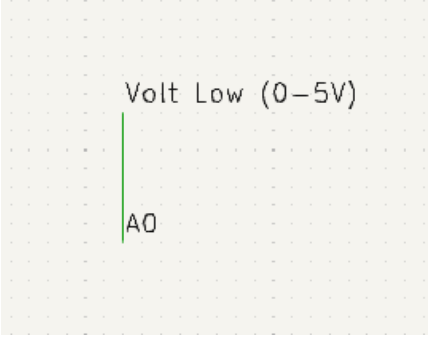
The project followed the requirements from the document “P02 Kravspecifikation - Ex24” [1], which includes important rules for how the multimeter should work, its design, and the materials used. These requirements include:

- **Voltage Measurement:** Ranges include 0–5V for low voltage and 5–35V for high voltage using a voltage divider circuit.
- **Resistance Measurement:** Two modes for measuring low resistance (0–1kΩ) and high resistance (1–100kΩ).
- **Battery Voltage Monitoring:** The device distinguishes between 9V and 12V inputs and provides low battery warnings.
- **User Feedback:** Visual indicators (LEDs) and a piezo buzzer for mode confirmation and continuity testing.
- **Mechanical Constraints:** PCB size capped at 68.6 × 65.0 mm with specified mounting holes for compatibility with Arduino Uno and battery holders.

Through this project, the goal was to produce a fully functional multimeter that meets accuracy requirements while also identifying areas for improvement in design, usability, and performance.

Construction and Design Process

The P2 multimeter's circuit design involves multiple measurement modes while ensuring accuracy, safety, and usability. Each circuit was tailored to its specific measurement type, with resistor selection playing a key role in achieving precision and reliability. The design was developed using KiCad, with all component footprints and 3D models provided by legitimate sources online.

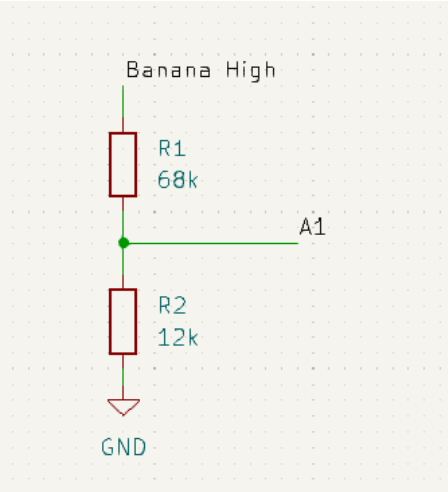


Low Voltage Mode (0–5V):

The low voltage mode connects the input voltage directly to the ADC pin A0. This straightforward configuration uses the Arduino Uno's 10-bit ADC range of 0–5V without requiring additional circuitry. It minimizes complexity and reduces potential sources of error, ensuring reliable measurements for low-voltage inputs.

Figure 1 - Low Voltage circuit.

High Voltage Mode (5–35V):



The high voltage mode employs a voltage divider circuit to scale input voltages down to a safe range for the ADC. The chosen resistors were selected after testing to ensure accuracy near the maximum measurement range.

The resistor values $R1 = 68\,K\Omega$, $R2 = 12K\Omega$ were selected because they provided the best accuracy during testing.(See *table 1*) Comparisons with other pairs, such as and $R1 = 68K\Omega$, $R2 = 10K\Omega$ showed that the selected values produced output voltages closer to 5.0V near the upper measurement range, improving precision.

Figure 2 - High Voltage circuit..

Simulated measurements performed in LTSpice supported the chosen resistor values, demonstrating consistent scaling and alignment with theoretical calculations. Details of these simulations are included in the appendix for reference. (See *figure 24-27*)

R1	R2	IN (V)	Out (V)
100 kΩ (98,5 kΩ)	16,7 kΩ	35 V	33,5 V
		5 V	2,8–4,5 V
100 kΩ (98,5 kΩ)	22 kΩ (21,69 kΩ)	35 V	34,3 V
		5 V	4,0–6,0 V
100 kΩ (98,5 kΩ)	27 kΩ (26,78 kΩ)	35 V	34,3 V
		20 V	27 V
		10 V	13 V
		5 V	5,5–6,7 V
10 kΩ (9,85 kΩ)	2,2 kΩ (2,16 kΩ)	35 V	34,3 V
		20 V	22–24,5 V
		10 V	10,5–12 V
		5 V	4,0–6,0 V
68 kΩ (67,6 kΩ)	12 kΩ (11,7 kΩ)	35 V	34,5–35 V
		30 V	30,5 V
		20 V	18,7–20,5 V
		10 V	7,7–9,5 V
		5 V	3,2–4,8 V
68 kΩ (67,8 kΩ)	10 kΩ	35 V	29,8–31 V
		20 V	16–17,2 V
		10 V	7–8,8 V
		5 V	2,2–4 V

Table 1 - High Voltage resistor test .

If Vin=35V, the output voltage would be calculated as:

The resistors installed have the value of $R1 = 67.6k\Omega,\ R2 = 11.76k\Omega$

$$Vout = Vin \cdot \frac{R2}{R1+R2}$$

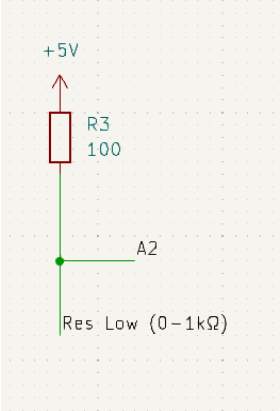
$$Vout = 35 \cdot \frac{11.76}{67.6+11.76} = 35 \cdot 0.148 = 5.18V$$

This exceeds the ADC's safe 5V limit and risks damaging the microcontroller. The measurable range is capped at 33.7V instead of 35V to ensure safe operation. Prioritizing accuracy near the upper limit was considered more important than extending the range.

$$Vin = 33.7V$$

$$Vout = 33.7 \cdot \frac{11.76}{67.6+11.76} = 33.7 \cdot 0.148 = 5.0V$$

Low Resistance Mode (0–1kΩ):



This resistor limits current through the circuit and ensures precise measurements of small resistances while protecting the microcontroller. Voltage drops across the resistor are measured by the ADC to calculate resistance using Ohm's Law.

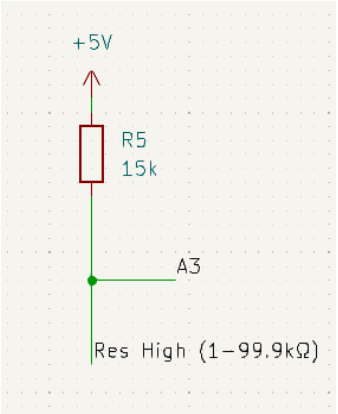
Testing confirmed consistent readings across a range of low resistances.

Figure 3 - Low resistance circuit.

Nominal	Fluke	Arduino
220 Ω	217,6 Ω	217,5 Ω
550 Ω	555 Ω	558,6 Ω
15 Ω	14,8 Ω	14,7 Ω
1000 Ω	980 Ω	978–990 Ω
10 Ω	10,1 Ω	9,98 Ω

Table 2 - Low resistance resistor test (100Ω).

High Resistance Mode (1–100kΩ):



This value balances sensitivity and stability, extending the multimeter's range for high resistance measurements.

Testing revealed minimal deviations and reliable performance for resistances up to 100kΩ.

Figure 4 - High resistance circuit.

Nominal	Fluke	Arduino
100 kΩ	98,5 kΩ	99,17 kΩ
68 kΩ	67,8 kΩ	68,3 kΩ
15 kΩ	14,92 kΩ	14,8 kΩ
5,6 kΩ	5,56 kΩ	5,53 kΩ

Table 3 - High resistance resistor test (15kΩ).

Battery Voltage Measurement

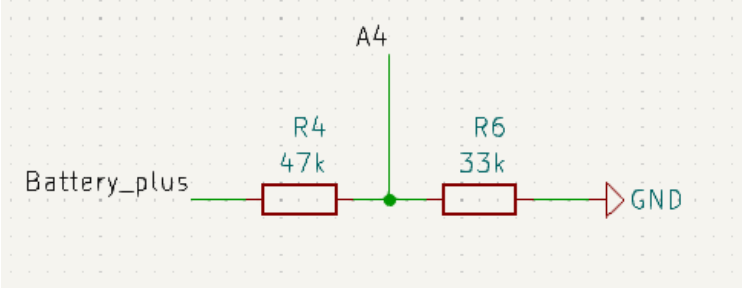


Figure 5 - Battery check circuit.

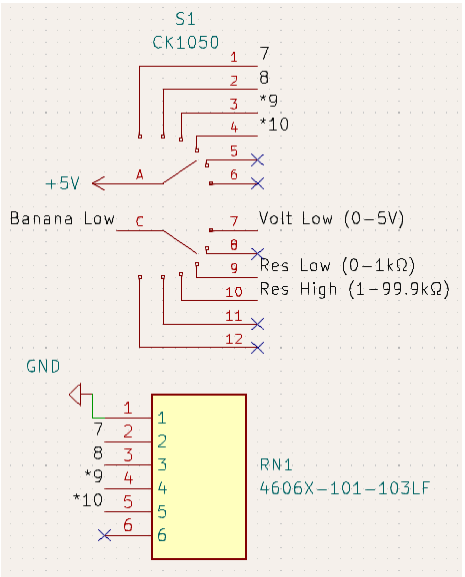
The battery voltage monitoring circuit ensures the multimeter can accurately differentiate between 9V and 12V power sources and provide low battery warnings.

The circuit is scaled after the barrel jack so for a 12V input, the output voltage is:

Installed Resistors: $R4 = 46.45k\Omega,\ R6 = 32.60k\Omega$

$$Vout = 12 \cdot \frac{32.6}{46.45+32.6} = 12 \cdot 0.412 = 4.94V$$

Rotary Mode Selection Circuit



A CK1050 rotary switch(see reference [6]) was used to select the desired measurement mode.

One pole routes the input signal to the appropriate measurement circuit. The second pole outputs binary signals to digital pins D7–D10, allowing the microcontroller to determine the active mode.

A resistor network (RN1: 4606X-101-103LF)(see reference [4]) was integrated into the circuit. This network acts as pull-down resistors, maintaining steady logic levels and preventing floating values during mode transitions.

Figure 6 - Selection Circuit.

Visual Indicators:

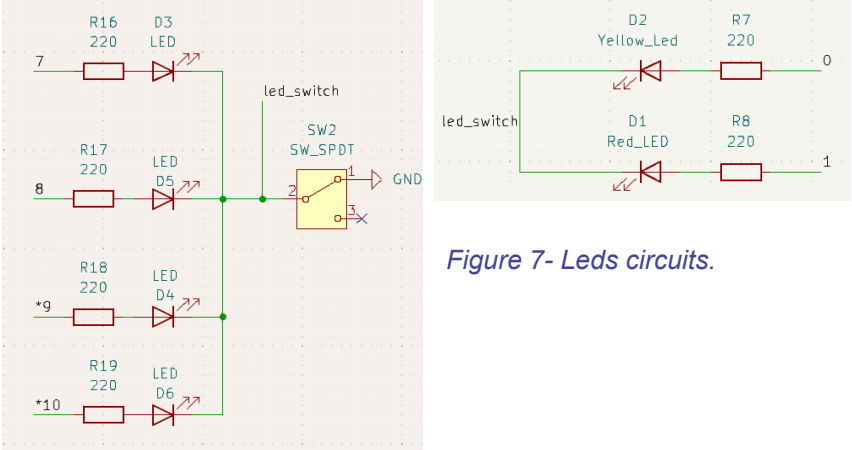


Figure 7- Leds circuits.

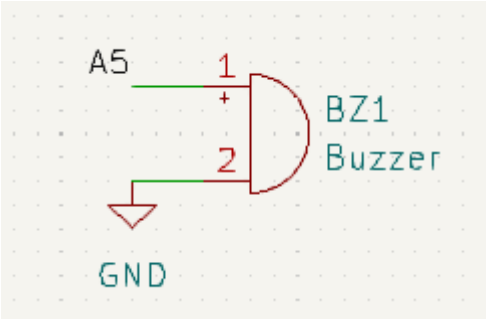
LEDs provide clear feedback for active modes:

Green LEDs(D3-D6): Represent the different measurement modes. Controlled by hardware.

Yellow LED: High Voltage mode. Controlled by software.

Red LED: Low Voltage mode.Controlled by software.

Auditory Feedback:



An active buzzer connected to pin A5 serves dual functions:

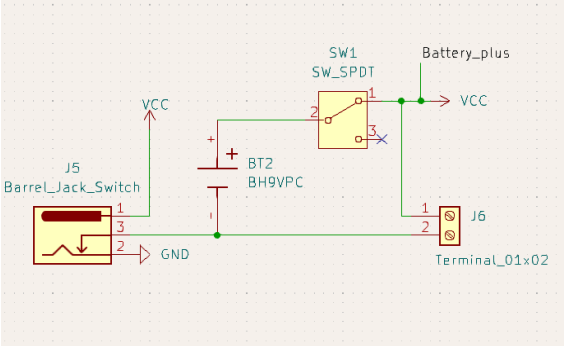
Signaling continuity detection in resistance measurements.

Triggering an alert for low battery conditions.

The buzzer's state is controlled by software.

Figure 8 - Buzzer circuit.

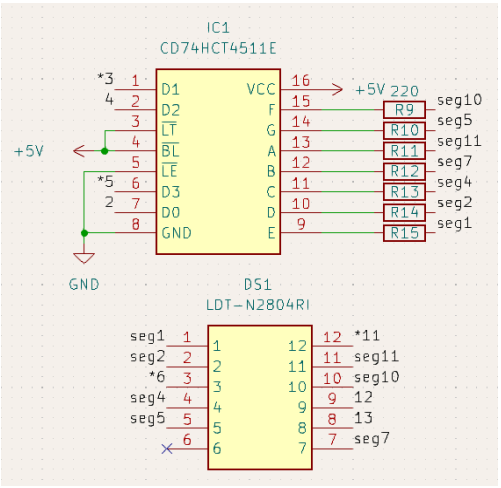
Power supply:



The power supply circuit provides flexible and reliable power options for the multimeter, supporting both a 9V battery and a 12V barrel jack input. The barrel jack (J5)(see reference [8]) allows connection to an external DC power source, while the battery holder (BT2) holds a 9V battery for portable use. A single-pole double-throw (SPDT) switch (SW1) controls power delivery, enabling the user to turn the circuit on or off, which helps conserve battery life. The output is routed through a terminal block (J6), which can also be used by a power source.

Figure 9 - Power supply circuit.

Display:



The CD74HCT4511E BCD-to-7-segment decoder(see reference [2]) controls the 3-digit 7-segment display (LDT-N2804RI)(see reference [3]) using a shared set of segment lines (A–G) and individual digit control. The decoder takes a 4-bit BCD input (D0–D3) and activates the corresponding segments to display numbers from 0 to 9.

The microcontroller rapidly switches between the three digits by controlling their common cathodes (DIGIT1, DIGIT2, DIGIT3), lighting one digit at a time. By cycling through the digits quickly, the display appears continuous to the human eye.

Figure 10 - Display circuit.

The DP pin controls the decimal point, enabling fractional measurements when needed. This design simplifies control, requiring only 4 input pins for segment data and 3 pins for digit selection, efficiently displaying values up to 999.

Connectors:

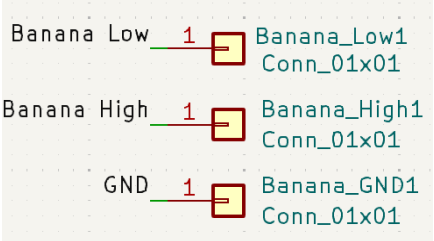


Figure 11 - Connector circuit.

This section shows the banana connectors used for input and ground connections. The Banana Low and Banana High inputs are used to measure voltages or resistances in different modes, while the GND connector provides a common ground reference.

Pin Headers:

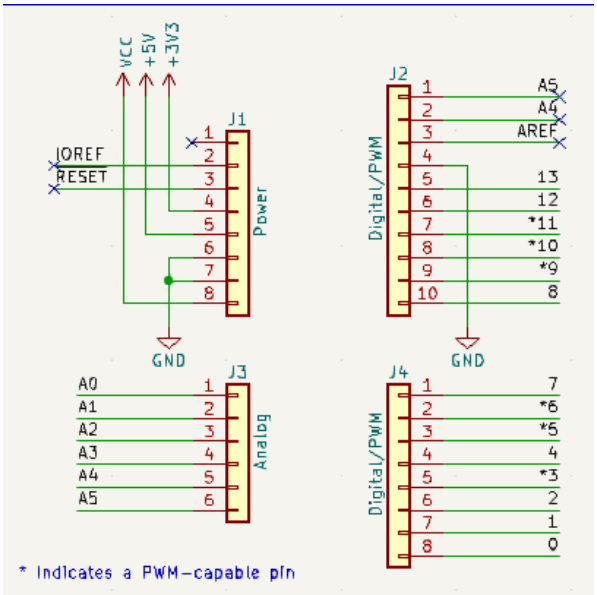


Figure 12 - Pin Headers.

This diagram shows the pin headers for power, analog, and digital connections.

- Power (J1): Provides VCC, 5V, 3.3V, and GND connections for powering the circuit.
- Analog Header (J3): Contains analog input pins (A0–A5) for voltage measurements.
- Digital/PWM Headers (J2 and J4): Include digital I/O pins, some of which are PWM-capable, enabling control of components like LEDs and the buzzer.

These headers allow easy interfacing with external components and ensure flexible connectivity for measurements and outputs.

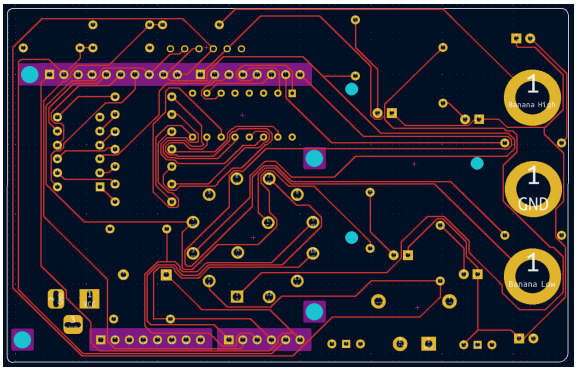
YRGO

PCB Layout

Dimensions and Adjustments:

The PCB was finalized at 99 x 64 mm. The board was shifted to the left to create space for a 9V battery holder on the backside. This adjustment allowed room on the front for silkscreen labeling and mode indicator LEDs.

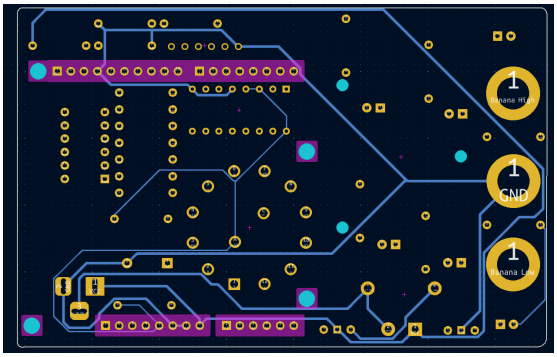
Top Layer (F.Cu):



Signal traces were routed with a width of 0.25 mm to minimize interference and maintain signal integrity. Pads width sizes were adjusted to help with the soldering.

Figure 13 - Top Layer (F.Cu).

Bottom Layer (B.Cu):



Power supply and ground traces were routed with a width of 0.5 mm since they should handle higher voltages and currents.. +5V uses 0.25mm.

Figure 14 - Bottom Layer (B.Cu)

Silkscreen Labeling:

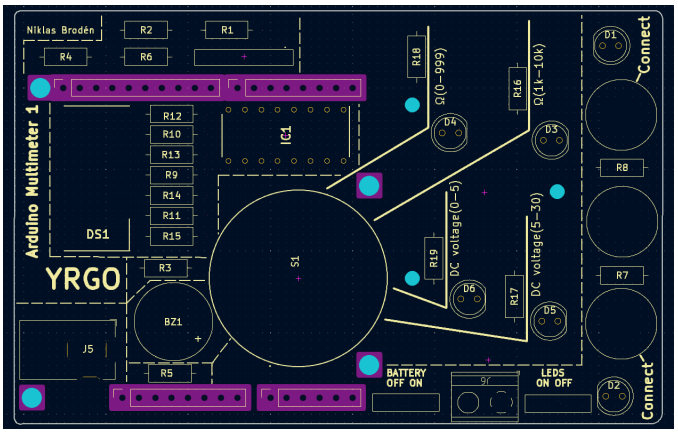


Figure 15 - Silkscreen.

The silkscreen layer was designed to provide clear labeling for all components, connectors, and switches, improving usability. Key labels include measurement modes, banana connector inputs, and switch functions such as Battery ON/OFF and LEDs ON/OFF. The silkscreen also marks the positions of resistors, the rotary switch, and the 7-segment display, ensuring easy assembly and user guidance.

3D-View:

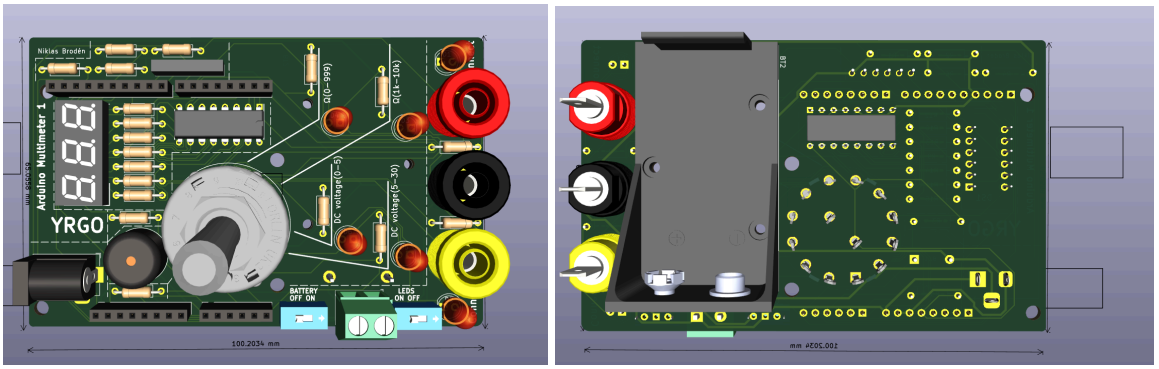


Figure 17 - 3D view.

The 3D view of the PCB layout was invaluable during the design process, providing a clear visualization of component placement and spatial constraints. It helped ensure that all components, such as the battery holder, rotary switch, banana connectors, and display, were optimally positioned for both functionality and usability. The battery holder was placed on the backside to free up space on the front for silkscreen labels and LEDs, improving user interaction. The 3D view also allowed verification of mounting hole positions and alignment, ensuring the board could be securely installed. Overall, this visualization tool helped refine the layout for a compact, user-friendly, and efficient design.

Board Assembly and Finalization

After receiving the PCB, all through-hole components were soldered onto the board. The 9V battery holder was placed on the backside to free space for components and labels. The Arduino Uno R3 was mounted on the back. Minor silkscreen errors were corrected with handwritten labels for clarity.

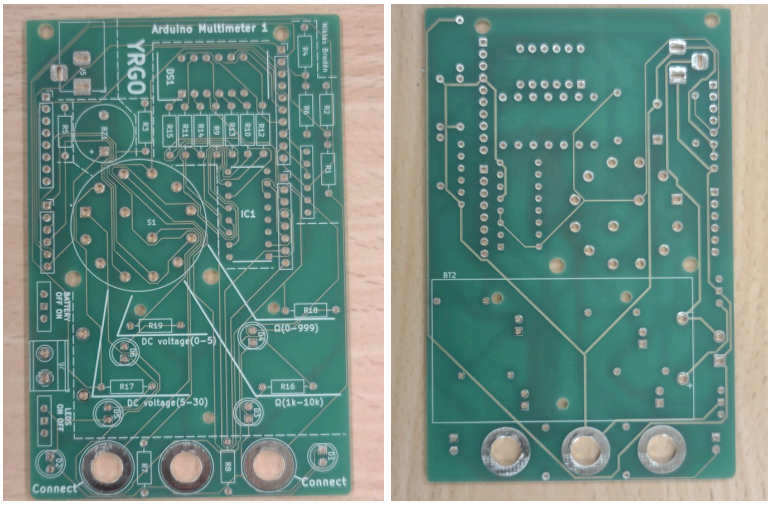


Figure 19 - PCB.

After assembly, the board was tested to confirm that all components and connections were functioning as intended. The solder joints were inspected for quality, and no faults or short circuits were identified. This smooth assembly process ensured the device met its design requirements with reliable performance and a clean, professional finish.

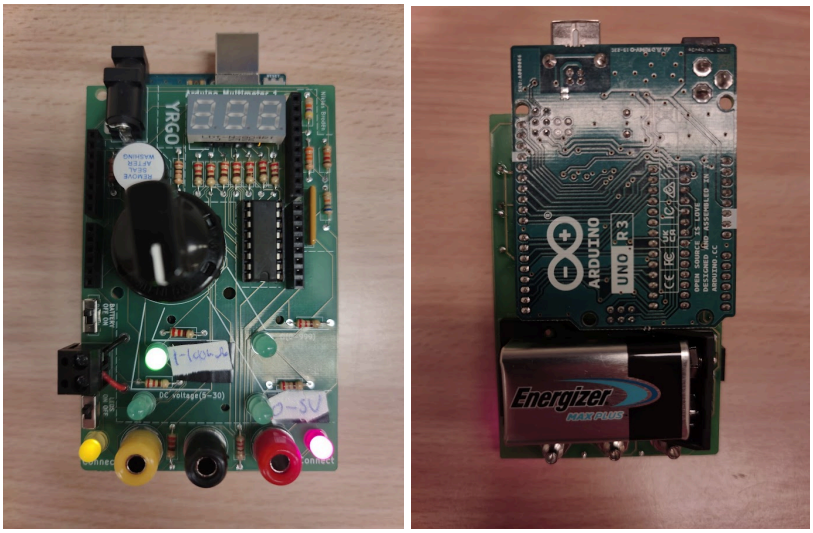


Figure 21 - Final assembly.

YRGO

Component list

Reference	Value	Datasheet	Footprint	Qty	ROHS
Banana_GND1, Banana_High1, Banana_Low1	Conn_01x01	https://www.electrokit.com/upload/quick/aa/00/867a_41010869-74-TDS.pdf	Connector:Banana_Jack_1Pin	3	YES
BT2	BH9VPC	https://www.electrokit.com/upload/quick/db/85/ce52_41003388-TDS.pdf	BH9VPC:BAT_BH9VPC	1	YES
BZ1	Buzzer	https://www.mouser.se/datasheet/2/1628/cmi_1295_05th.pdf	Buzzer_Beeper:Buzzer_12x9.5RM7.6	1	YES
D1	Red_LED	https://www.mouser.se/datasheet/2/216/APTD2012LSURCK.pdf	LED_THT:LED_D5.0mm	1	YES
D2	Yellow_Led	https://www.mouser.se/datasheet/2/216/apt2012pyw.pdf	LED_THT:LED_D5.0mm	1	YES
D3,D4,D5,D6	LED	https://www.mouser.se/datasheet/2/216/apt2012pyw-1173287.pdf	LED_THT:LED_D5.0mm	4	YES
DS1	LDT-N2804RI	https://www.ti.com/lit/gpn/cd74hc4511	N2804RI:DIPS762W45P254L2253H610Q12N	1	YES
IC1	CD74HCT4511E	http://www.ti.com/lit/gpn/cd74hct4511	CD74HCT4511E:DIP794W53P254L1930H508Q16N	1	YES
J1	Power		Connector_PinSocket_2.54mm:PinSocket_1x08_P2.54mm_Vertical	1	N/A
J2,J4	Digital/PWM		-- mixed values --	2	N/A
J3	Analog	~	Connector_PinSocket_2.54mm:PinSocket_1x06_P2.54mm_Vertical	1	N/A
J5	Barrel_Jack_Switch	https://www.electrokit.com/upload/product/41013/41013585/18742.pdf	Connector_BarrelJack:BarrelJack_Horizontal	1	YES
J6	Terminal_01x02	~	p2lib:terminal_battery	1	N/A
R1	68k	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*13ie4i7*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES
R2	12k	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*oioteu*_gcl_au*NDc4MDEyOTQyLjE3Mjc3Nz	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES

		QzOTQ.			
R3	100	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*1d5qbu1*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES
R4	47k	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*1mkff6g*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES
R5	15k	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*gtr0y3*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES
R6	33k	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*1mkff6g*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	YES
R7,R8,R9,R10,R11,R12,R13,R14,R15,R16,R17,R18,R19	220	http://www.farnell.com/datasheets/2710513.pdf?_gl=1*1mkff6g*_gcl_au*NDc4MDEyOTQyLjE3Mjc3NzQzOTQ.	Resistor_THT:R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	13	YES
RN1	4606X-101-103LF	https://www.bourns.com/pdfs/4600X.pdf	4606X:4606X102103LF	1	YES
S1	CK1050	https://www.mouser.se/datasheet/2/242/CK_ROTARY_Nov_2020_1-2353427.pdf	CK1050:SW_CK1050	1	YES
SW1,SW2	SW_SPDT	https://www.we-online.com/components/products/datasheet/452404020202.pdf	Button_Switch_THT:SW_Slide-03_Wuerth-WS-SLTV_10x2.5x6.4_P2.54mm	2	YES
PCB		No datasheet but ROHS confirmed by Cogra			YES

Table 4 - Component list.

Detailed Compliance for CD74HCT4511E

REACH: The component is verified as REACH compliant, with no substances of very high concern (SVHC) exceeding the 0.1% threshold.(see reference [11])

***Material Composition**:*

Bond Wire: 99.99% Copper, trace Silver

Mold Compound: 88% Fused Silica, 11% Epoxy

YRGO

Software Implementation

The multimeter's software, based on an Arduino code(See appendix for source code) provided by the instructor, required additional implementation to achieve full functionality. Additions included:

- **High Voltage Mode:** Developed to scale input voltages using a voltage divider and calculate results for display.
- **Continuity Detection:** Enabled with a threshold to activate the buzzer for low resistance values, improving usability for circuit testing.
- **Battery Check:** Implemented to monitor battery levels and trigger warnings for low or no battery conditions.
- **LED Control:** Added functionality to manage yellow and red LEDs for mode-specific visual feedback.
- **Smoothing Function:** A smoothing algorithm was implemented using a buffer to stabilize measurements. However, this was later removed as it caused delays in providing correct values, reducing the responsiveness of the multimeter.

YRGO

Testing and Verification

The testing and validation phase ensured its functionality and accuracy through a systematic approach. The process began with breadboard testing for choosing components and finished with real-world testing of the final PCB assembly.

During testing, the Arduino was connected to a PC via USB, utilizing the serial monitor for real-time feedback and debugging. This enabled quick validation of measurement results. Validation against a Fluke multimeter confirmed reliability, with consistent accuracy across all modes. High voltage and resistance measurements were verified within acceptable margins.

Initial breadboard testing allowed potential issues to be identified before soldering. High voltage dividers were tested with various input voltages to confirm accurate scaling. Calibration followed, with resistor values measured using a precision multimeter and software adjustments made to account for tolerances.

After assembly, the fully soldered PCB was tested under real-world conditions to evaluate its overall functionality. All measurement modes were tested systematically:

- Voltage measurements (low and high) were confirmed across their respective ranges, aligning closely with the Fluke multimeter reference values.
- Resistance measurements demonstrated reliable accuracy for both low and high ranges.
- Battery monitoring accurately differentiated between 9V and 12V inputs, with low and no battery conditions triggering appropriate warnings.

Calibration significantly improved accuracy across all modes. Minor noise was observed at higher resistance values, though deviations remained within acceptable limits. The high voltage divider's limitation to 33.7V highlighted a potential area for improvement, as did the 10-bit resolution of the Arduino's ADC, which posed challenges for precision at extreme ranges.

YRGO

Final Measurement Results

Low Voltage Mode (0–5V):

Testing revealed minimal deviation, with errors consistently below 0.5%. Measurements closely aligned with reference values from a Fluke multimeter.

Input (V)	Displayed (V)
0	0
0.5	0.47
1	0.97
2	1.97
3.5	3.48
5	4.99

Table 5 - Low Voltage final test.

High Voltage Mode (5–35V):

Calibration ensured accuracy within 2%, with the voltage divider scaling input voltages reliably. Maximum input of 33.7V produced an output voltage of 5.0V at the ADC. An input value above 33.7V will produce a capped 33.7V on the display.

Input (V)	Displayed (V)
5	4.9
10	9.9
15	14.9
20	19.9
25	25
30	30
33.63	33.6
33.66	33.7
34	33.7

Table 6 - High Voltage final test.

Low Resistance Mode (0–1kΩ):

Resistance values showed errors under 1.1%, with continuity tests providing clear audio feedback.

Nominal (Ω)	Fluke (Ω)	Displayed (Ω)
820	812	820
680	670	674
390	384	388
220	217	218
22	22.1	22

Table 7 - Low Resistance final test.

High Resistance Mode (1–100kΩ):

Minor deviations occurred at the lower end of the range due to ADC resolution limits.

High resistance measurements remained within acceptable error margins.

Nominal (kΩ)	Fluke (kΩ)	Displayed (kΩ)
100	98	99
82	81.1	82
47	46.88	47
18	17.82	17
8.2	8.1	8.1
1.2	1.214	1.1

Table 8 - High Resistance final test.

Battery Voltage Measurement:

Software thresholds were validated to trigger warnings for low and no battery conditions.

Low Battery Warning: Triggered when the voltage drops below 6.5V. Activating the beeper.

No Battery Detection: Activated when the input voltage falls below 1.0V. Turning off the function.

The multimeter accurately detected voltage levels for both 9V and 12V sources. Software thresholds effectively distinguish between low and no battery conditions, with auditory warnings triggered for low battery.

Discussion and conclusion

The P2 Multimeter Project successfully achieved its design objectives, delivering a functional and accurate tool for measuring voltage, resistance, and battery levels. The project highlighted both the strengths of the design and areas for potential improvement.

The multimeter demonstrated reliable accuracy across all measurement modes, as verified by testing against a Fluke multimeter. The chosen resistor values and calibration adjustments ensured that the voltage and resistance measurements performed within acceptable error margins. The rotary switch together with clear visual and auditory feedback systems significantly enhanced the usability of the device. Furthermore, the compact PCB design optimized space while maintaining accessibility for all components.

Some challenges arose during the project. The silkscreen labeling error required a manual correction, emphasizing the importance of thorough design reviews before manufacturing. Additionally, the high voltage divider limited the measurable range to 33.7V instead of the intended 35V, reflecting a trade-off made to prioritize accuracy near the upper range. The Arduino's 10-bit ADC resolution also constrained precision for high resistance and high voltage measurements at extreme ranges, slightly impacting overall performance.

Future iterations of the multimeter could address some of the current limitations. Incorporating a higher-resolution ADC would improve measurement precision, especially for high resistance and high voltage ranges. Expanding the voltage divider's scaling capability could extend the measurable range beyond 35V, meeting the initial design specifications. Or using other resistor combinations that were not available. Adding Zener diodes for overvoltage protection would enhance the durability and safety of the device, particularly for high voltage measurements.

In conclusion, the P2 Multimeter Project met its primary goals while providing valuable insights into the challenges of designing accurate and user-friendly measurement tools. The lessons learned and identified improvements lay a solid foundation for future development and innovation.

YRGO

References

[1] E. Jagre, "P02 Kravspecifikation - Ex24," Rev. 3, Oct. 21, 2024. [Online]. Available: https://docs.google.com/document/d/1Md-3sL7DFWtVUQRTn1kG0PjF5JSB97vi9y1uiQYecPA/edit?usp=classroom_web&authuser=0. [Accessed: Dec. 12, 2024].

[2] **Texas Instruments**, "CD74HCT4511E BCD-to-7-Segment Decoder Datasheet." [Online]. Available: https://www.ti.com/lit/ds/symlink/cd74hct4511.pdf. [Accessed: Dec. 12, 2024].

[3] **Lumex**, "LUMXS13609 3-Digit 7-Segment Display Datasheet." [Online]. Available: https://www.lumex.com/spec/LUMXS13609.pdf. [Accessed: Dec. 12, 2024].

[4] **Bourns**, "4606X-101-103LF Resistor Network Datasheet." [Online]. Available: https://www.bourns.com/docs/product-datasheets/4600x.pdf. [Accessed: Dec. 12, 2024].

[5] **CUI Devices**, "CMI-1295 Buzzer Datasheet." [Online]. Available: https://www.cuidevices.com/product/resource/cmi-1295.pdf. [Accessed: Dec. 12, 2024].

[6] **CK Components**, "CK1050 Rotary Switch Datasheet." [Online]. Available: https://www.ckswitches.com/media/1477/rotary_ck.pdf. [Accessed: Dec. 12, 2024].

[7] **Keystone Electronics**, "BH9VPC 9V Battery Holder Datasheet." [Online]. Available: https://www.keyelco.com/product-pdf.cfm?p=1988. [Accessed: Dec. 12, 2024].

[8] **Würth Elektronik**, "Barrel Jack Connector Datasheet." [Online]. Available: https://www.we-online.com/components/products/datasheet/694108103102.pdf. [Accessed: Dec. 12, 2024].

[9] **Amphenol**, "Banana Jack Connector Datasheet." [Online]. Available: https://www.amphenolrf.com/media/downloads/2493/bj_jack.pdf. [Accessed: Dec. 12, 2024].

[10] **Texas Instruments**, "CD74HCT4511E BCD-to-7-Segment Decoder Quality, reliability & packaging data download”." [Online]. Available: https://www.ti.com/quality-reliability-packaging-download/report?opn=CD74HCT4511E [Accessed: Dec. 12, 2024].

Appendix

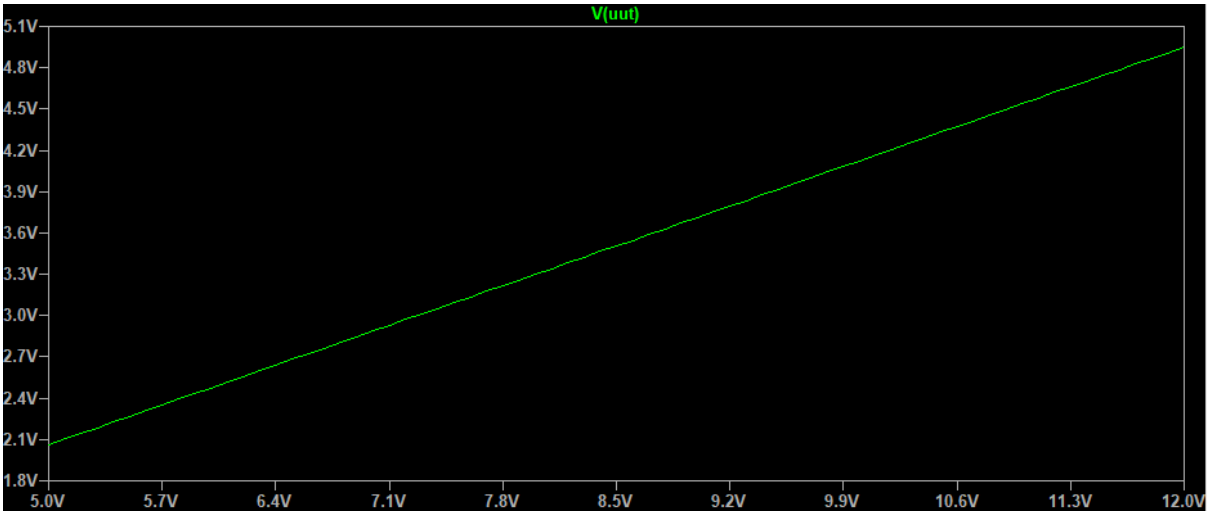


Figure 22 - LT-spice Battery check circuit.

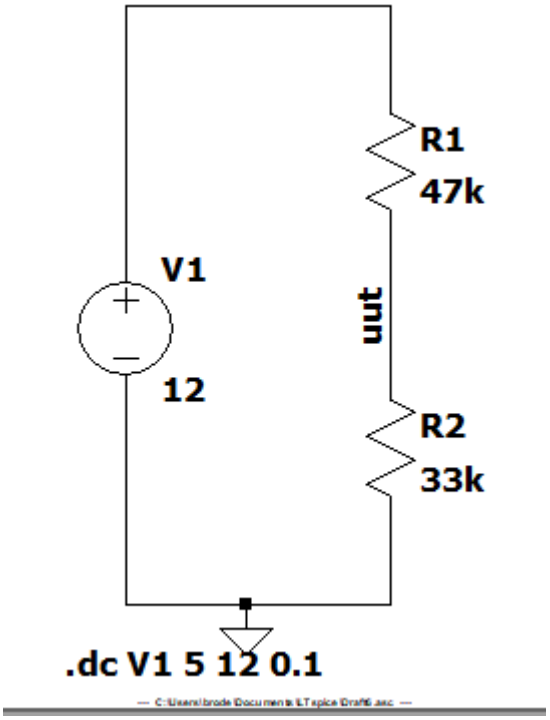


Figure 23- LT-spice Battery check circuit.

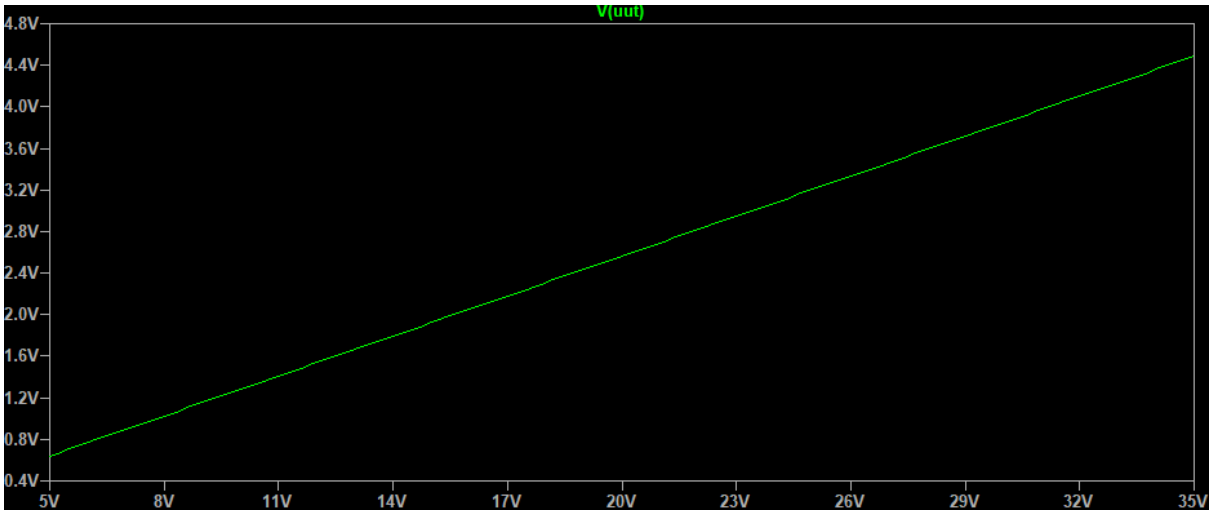


Figure 24 - LT-spice High Voltage test (68k and 10k).

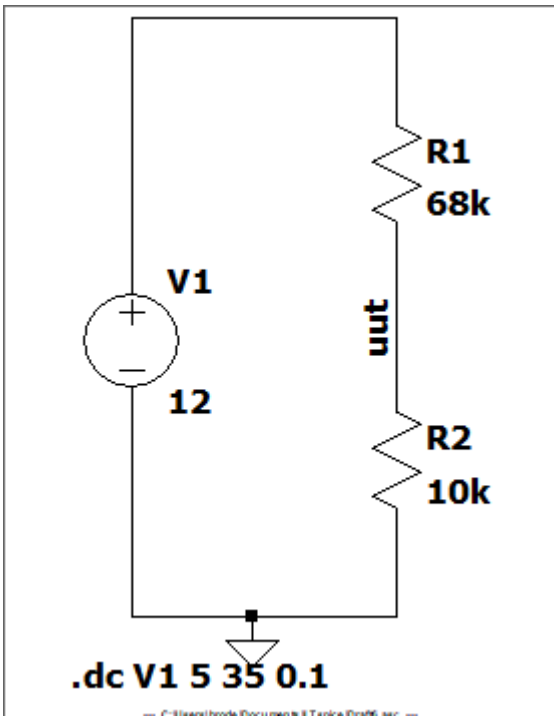


Figure 25 -LT-spice High Voltage test (68k and 10k).

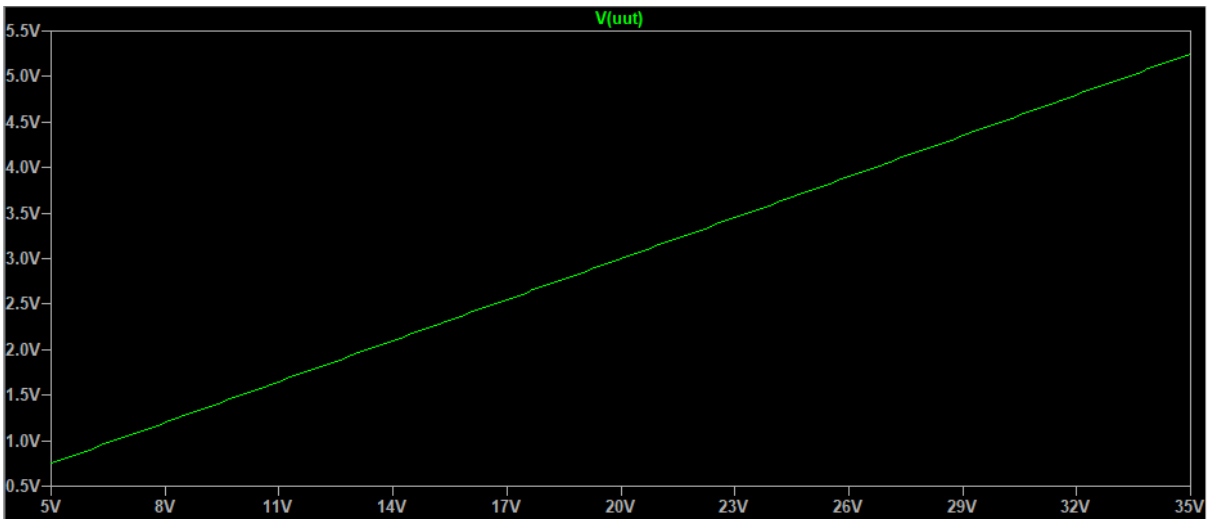


Figure 26 LT-spice High Voltage test (68k and 12k).

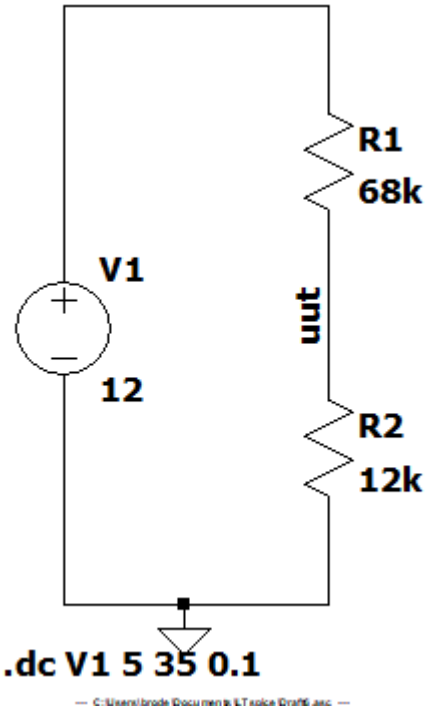


Figure 27 LT-spice High Voltage test (68k and 12k).

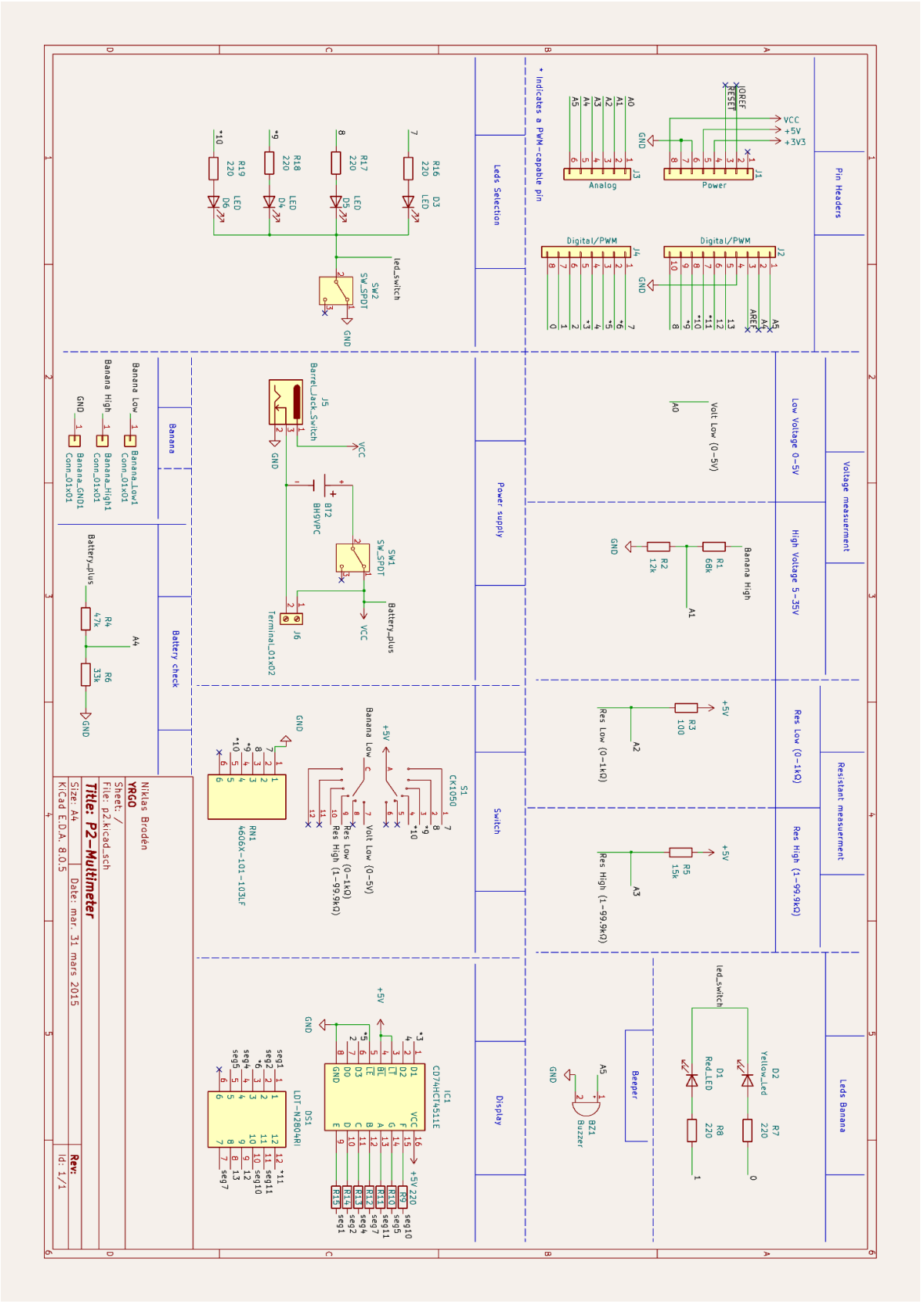


Figure 28 - Full Schematic.

YRGO

Source code

```
// Pin assignment - Common cathode for the 3 individual digits on
3*seven-segment display
const int DIGIT1 = 11;
const int DIGIT2 = 12;
const int DIGIT3 = 13;

// Pin assignment - decimal point
const int DM_POINT = 6;

// Pin assignment - binary values for digital output to BCD-encoder
const int BCD_A = 2;
const int BCD_B = 3;
const int BCD_C = 4;
const int BCD_D = 5;

// Pin assignment - function selection
const int FUNCTION_1 = 7;
const int FUNCTION_2 = 8;
const int FUNCTION_3 = 9;
const int FUNCTION_4 = 10;

// Pin assignment - LEDs
const int YELLOW_LED = 0; // Pin connected to Yellow LED
const int RED_LED = 1;    // Pin connected to Red LED

const int WAIT_TIME_MS = 2; // Fine-tuned delay for display multiplexing

const float U_REF = 5.0; // Reference voltage for ADC

// Real resistor values for calculations
const int R1_LOW = 100;
const int R1_HIGH = 15000;

// High voltage scaling resistors
const float R_HIGH = 67600.0; // 67.6k ohms
const float R_LOW = 11760.0;  // 11.76k ohms
const float SCALING_FACTOR = (R_HIGH + R_LOW) / R_LOW;

// Battery check resistor values
```

YRGO

```
const float R4_BATT = 46450.0; // 46.45k ohms
const float R6_BATT = 32600.0; // 32.60k ohms

// Battery thresholds
const float BATTERY_LOW_THRESHOLD = 6.5;
const float HYSTERESIS_OFFSET = 0.5;
const float NO_BATTERY_THRESHOLD = 1.0;

// Global variables
bool batteryCheckActive = true;
bool batteryLow = false;
int hundratal = 0, tiotal = 0, ental = 0;
unsigned int counter = 0;
int dp1 = 0, dp2 = 0, dp3 = 0;
float U_in = 0;
long int r2 = 0;
bool continuityDetected = false;

// Function prototypes
void displayDigits();
float readBatteryVoltage();

void setup() {
    // Configure pins
    pinMode(DIGIT1, OUTPUT);
    pinMode(DIGIT2, OUTPUT);
    pinMode(DIGIT3, OUTPUT);
    pinMode(DM_POINT, OUTPUT);
    pinMode(A5, OUTPUT); // Buzzer

    pinMode(BCD_A, OUTPUT);
    pinMode(BCD_B, OUTPUT);
    pinMode(BCD_C, OUTPUT);
    pinMode(BCD_D, OUTPUT);

    pinMode(FUNCTION_1, INPUT);
    pinMode(FUNCTION_2, INPUT);
    pinMode(FUNCTION_3, INPUT);
    pinMode(FUNCTION_4, INPUT);

    pinMode(YELLOW_LED, OUTPUT);
```

```
    pinMode(RED_LED, OUTPUT);

    // Turn off all indicators at startup
    digitalWrite(DIGIT1, HIGH);
    digitalWrite(DIGIT2, HIGH);
    digitalWrite(DIGIT3, HIGH);
    digitalWrite(DM_POINT, HIGH);
    digitalWrite(A5, LOW); // Beeper off
    digitalWrite(YELLOW_LED, LOW);
    digitalWrite(RED_LED, LOW);
}

void loop() {
    // Continuously update the display
    displayDigits();

    // Turn off LEDs by default
    digitalWrite(RED_LED, LOW);
    digitalWrite(YELLOW_LED, LOW);

    // Check battery status
    if (batteryCheckActive) {
        float batteryVoltage = readBatteryVoltage();
        if (batteryVoltage < NO_BATTERY_THRESHOLD) {
            // No battery detected
            batteryCheckActive = false;
            digitalWrite(A5, LOW); // Buzzer off
        } else if (batteryVoltage < BATTERY_LOW_THRESHOLD) {
            // Battery low
            if (!batteryLow) {
                batteryLow = true;
                digitalWrite(A5, HIGH); // Buzzer on
            }
        } else if (batteryVoltage > BATTERY_LOW_THRESHOLD + HYSTERESIS_OFFSET)
{
            // Battery level normal
            batteryLow = false;
            digitalWrite(A5, LOW); // Buzzer off
        }
    }
```

```
    // Perform measurements and other logic
    int fkn1 = digitalRead(FUNCTION_1);
    int fkn2 = digitalRead(FUNCTION_2);
    int fkn3 = digitalRead(FUNCTION_3);
    int fkn4 = digitalRead(FUNCTION_4);

    int function_select = ((fkn4 << 3) | (fkn3 << 2) | (fkn2 << 1) | (fkn1 <<
0));

    switch (function_select) {
        case 0b0001: // Low Voltage
            U_in = analogRead(A0) * (U_REF / 1023.0);
            if (U_in > U_REF) U_in = U_REF;
            counter = (unsigned int)(U_in * 10); // Convert to 3-digit format
            dp1 = LOW; dp2 = HIGH; dp3 = LOW; // Decimal after tens place
            digitalWrite(YELLOW_LED, HIGH);
            break;

        case 0b0010: // High Voltage
            U_in = analogRead(A1) * (U_REF / 1023.0) * SCALING_FACTOR;
            counter = (unsigned int)(U_in * 10);
            dp1 = LOW; dp2 = HIGH; dp3 = LOW;
            digitalWrite(RED_LED, HIGH);
            break;

        case 0b0100: // Low Resistance
            U_in = analogRead(A2) * (U_REF / 1023.0);
            if (U_in < 0.01) {
                r2 = -1;
            } else {
                r2 = (long int)(U_in * R1_LOW) / (U_REF - U_in);
            }
            if (r2 >= 0 && r2 < 10) digitalWrite(A5, HIGH);
            else digitalWrite(A5, LOW);
            counter = (r2 >= 0) ? r2 : 0;
            dp1 = LOW; dp2 = LOW; dp3 = LOW;
            digitalWrite(YELLOW_LED, HIGH);
            break;

        case 0b1000: // High Resistance
            U_in = analogRead(A3) * (U_REF / 1023.0);
```

```
            r2 = (long int)(U_in * R1_HIGH) / (U_REF - U_in);
            if (r2 < 10000) {
                counter = r2 / 100;
                dp1 = LOW; dp2 = HIGH; dp3 = LOW;
            } else {
                counter = r2 / 1000;
                dp1 = LOW; dp2 = LOW; dp3 = HIGH;
            }
            digitalWrite(YELLOW_LED, HIGH);
            break;

        default:
            counter = 0;
            dp1 = LOW; dp2 = LOW; dp3 = LOW;
            digitalWrite(YELLOW_LED, HIGH);
            break;
    }
}


float readBatteryVoltage() {
    float rawReading = analogRead(A4);
    return rawReading * (U_REF / 1023.0) * ((R4_BATT + R6_BATT) / R6_BATT);
}

void displayDigits() {
    // Hundreds digit
    hundratal = (counter / 100) % 10;
    digitalWrite(BCD_A, hundratal & 1);
    digitalWrite(BCD_B, (hundratal >> 1) & 1);
    digitalWrite(BCD_C, (hundratal >> 2) & 1);
    digitalWrite(BCD_D, (hundratal >> 3) & 1);
    digitalWrite(DM_POINT, dp1);
    digitalWrite(DIGIT1, LOW);
    delay(WAIT_TIME_MS);
    digitalWrite(DIGIT1, HIGH);

    // Tens digit
    tiotal = (counter / 10) % 10;
    digitalWrite(BCD_A, tiotal & 1);
    digitalWrite(BCD_B, (tiotal >> 1) & 1);
```

```
    digitalWrite(BCD_C, (tiotal >> 2) & 1);
    digitalWrite(BCD_D, (tiotal >> 3) & 1);
    digitalWrite(DM_POINT, dp2);
    digitalWrite(DIGIT2, LOW);
    delay(WAIT_TIME_MS);
    digitalWrite(DIGIT2, HIGH);

    // Units digit
    ental = counter % 10;
    digitalWrite(BCD_A, ental & 1);
    digitalWrite(BCD_B, (ental >> 1) & 1);
    digitalWrite(BCD_C, (ental >> 2) & 1);
    digitalWrite(BCD_D, (ental >> 3) & 1);
    digitalWrite(DM_POINT, dp3);
    digitalWrite(DIGIT3, LOW);
    delay(WAIT_TIME_MS);
    digitalWrite(DIGIT3, HIGH);
}
```