

GlucoMeter

Introduction

This project aims to build a **functional, low-cost, and extensible blood glucose monitor** using the **Arduino Uno** board, based on the **ATmega328P microcontroller**.

The goal is to create a **simplified version of commercial glucometers** that can be easily reproduced for *educational purposes* or used in *low-resource environments*. It focuses on making **glucose level measurement** more **accessible** for people.

The initial idea came from exploring how **test strips work electrochemically** and attempting to replicate the measuring device using **analog** and **digital components**.

I believe this project is useful for understanding the integration of **bio-sensing, analog signal amplification**, and **embedded systems**, offering both **educational value** and the potential to evolve into a **practical, low-cost health monitoring tool**.

General Description

The *electrochemical test strips* generate a **small electrical current** through a redox reaction. This current is **proportional** to the concentration of glucose in the sample. However, the generated current is extremely small, typically in the range of **nanoamperes** to **microamperes** and cannot be measured directly by the ATmega328P's **ADC**. To overcome this, the project uses an **operational amplifier** (op-amp) that converts the current signal from the test strip into a **measurable voltage**, using a feedback resistor to set the gain of the amplifier. The amplified voltage is then digitized using the built-in ADC, the value is then processed to calculate the **glucose concentration**, based on calibration parameters corresponding to the specific test strips used.

For development and testing purposes, a potentiometer is used to simulate the small and variable current generated by an electrochemical test strip. The potentiometer provides a controllable analog voltage input which, after amplification, is read by the ADC to mimic real sensor behavior.

Modules used:

- **ADC** – used to read the analog voltage signal from the potentiometer simulating the test strip output.
- **I2C** – used to communicate with the LCD 1602 display module for showing glucose measurement results.
- **UART** – for serial communication to log data or debug via PC.
- **GPIO** – to control LEDs which indicate glucose level categories by turning on different colored LEDs.
- **PWM** – used to control LED brightness.

- **Interrupts** – employed to detect button presses immediately, allowing the program to respond instantly to user input.



Hardware Design

Description

This project integrates a set of analog and digital components to create a **functional blood glucose monitor**, capable of measuring glucose concentration in the blood by analyzing the **electrochemical current generated** from a test strip (simulated using a potentiometer).

Main Hardware Components:

- **Arduino Uno (ATmega328P)** – The central microcontroller that processes signals, reads analog values through its ADC, and controls output peripherals like LEDs and the LCD.
- **OP07 Operational Amplifier** – Used in a **transimpedance amplifier configuration** to convert the small current from the test strip into a measurable voltage. A feedback resistor determines the gain and ensures compatibility with the 0-5V ADC range of the Arduino.
- **Potentiometer (50kΩ)** – Used to simulate the small and variable electrical current generated by an electrochemical test strip. By adjusting the potentiometer, we can vary the input voltage, mimicking different glucose concentrations for testing and calibration purposes.
- **Feedback Resistor** – Connected between the output and inverting input of the OP07, typically selected in the **kΩ-MΩ** range to produce appropriate voltage output for microampere-level currents.
- **LEDs (x6)** – Provide visual alerts for glucose thresholds: hypoglycemia, normal, and hyperglycemia. Connected to Arduino digital pins (D3, D5, D6, D9, D10, D11). Some LEDs may be controlled via PWM to adjust brightness or indicate different alert levels.
- **1602 LCD with I2C Interface** – Displays the measured glucose value and classification. Connected via I2C on pins SDA and A5 SCL.
- **Push Button** – Used to manually initiate a measurement or reset the display. Connected to a digital input pin with pull-up resistor (pin D2) to allow interrupt-based input detection.
- **Breadboard and Jumper Wires** – For rapid prototyping and modular hardware assembly.

Pin Assignment and Justification

Component	Arduino Pin(s)	Reason for Selection
Sensor Output (from OP07)	A2	Analog input pin used to read the amplified voltage proportional to glucose level via ADC.
LEDs	D3, D5, D6, D9, D10, D11	PWM-capable digital pins allow brightness modulation of LEDs to indicate different alert levels clearly.

LCD 1602 I2C Module	SDA, SCL	Dedicated I2C pins on Arduino Uno, used for communication with the LCD to display glucose values and status.
Push Button	D2	Digital pin with hardware interrupt capability used to detect button presses instantly and trigger actions.
Power Supply for Op-Amp	5V and GND	Provides the necessary power for the OP07 amplifier with a common ground reference to the Arduino board.

Interfaces and Communication:

- **Analog Input (A2):** Reads the output voltage from the OP07 amplifier via the ADC.
- **Digital Outputs (D3, D5, D6, D9, D10, D11):** PWM-capable pins used to drive LEDs for visual alerts based on glucose classification, allowing brightness modulation.
- **I2C (SDA, SCL):** For communication with the 1602 LCD module displaying measurement data.
- **UART:** Used for serial communication with a PC for debugging or data logging.

Power and Consumption:

- The system is powered through the Arduino Uno's 5V rail (typically via USB).
- The total current draw is modest (~70–100 mA), suitable for **portable applications** powered by a USB power bank.
- Power optimization strategies may include turning off the display or entering low-power sleep modes between measurements.

This modular and extensible design allows for future upgrades such as **Bluetooth integration**, **OLED displays**, or **buzzer alarms**. It demonstrates the synergy between **bio-sensing**, **analog electronics**, and **embedded systems** in an accessible, low-cost platform.

Bill of Materials:

Componentă	Cantitate	Link sursă
Arduino Uno (ATmega328P cu CH340)	1	[www.optimusdigital.ro/ro/compatibile-cu-arduino-uno/1678-placa-de-dezvoltare-compatibila-cu-arduino-uno-atmega328p-i-ch340.html?search_query=0104110000016997&results=1]
LCD 1602 with I2C Interface and Yellow-Green Backlight	1	[https://www.optimusdigital.ro/ro/optoelectronice-lcd-uri/62-lcd-1602-cu-interfata-i2c-si-backlight-galben-verde.html?search_query=0104110000026989&results=1]
Amplificator Operațional OP07	1	[https://www.optimusdigital.ro/ro/componente-electronice-altele/2948-amplificator-operaional-op07.html?search_query=0104210000025866&results=1]
LED-uri (culori diferite)	6	[https://www.optimusdigital.ro/ro/optoelectronice-led-uri/696-led-rou-de-3-mm-cu-lentile-difuze.html?search_query=0104210000006186&results=1]
Rezistori 220Ω	7	[https://www.optimusdigital.ro/ro/componente-electronice-rezistoare/33-rezistoare-set.html?search_query=0104110000001481&results=1]
Buton 6x6x6	1	[https://www.optimusdigital.ro/ro/butoane-i-comutatoare/1119-buton-6x6x6.html?search_query=0104210000010862&results=1]
Breadboard HQ (830 points)	1	[https://www.optimusdigital.ro/ro/prototipare-breadboard-uri/13244-breadboard-175-x-67-x-9-mm.html?search_query=breadboard&results=125]
Breadboard Jumper Wires Set	20+	[https://www.optimusdigital.ro/ro/fire-fire-mufate/12-set-de-cabluri-pentru-breadboard.html?search_query=0104210000001532&results=1]
Potențiometru 50kΩ	1	[https://www.optimusdigital.ro/ro/componente-electronice-potențiometre/1885-potențiometru-mono-50k.html?search_query=potențiometru&results=172]
Fire Colorate Mamă-Tată (10p, 15 cm)	1	[https://www.optimusdigital.ro/ro/toate-produsele/876-set-fire-mama-tata-10p-15-cm.html?search_query=0104210000008982&results=1]

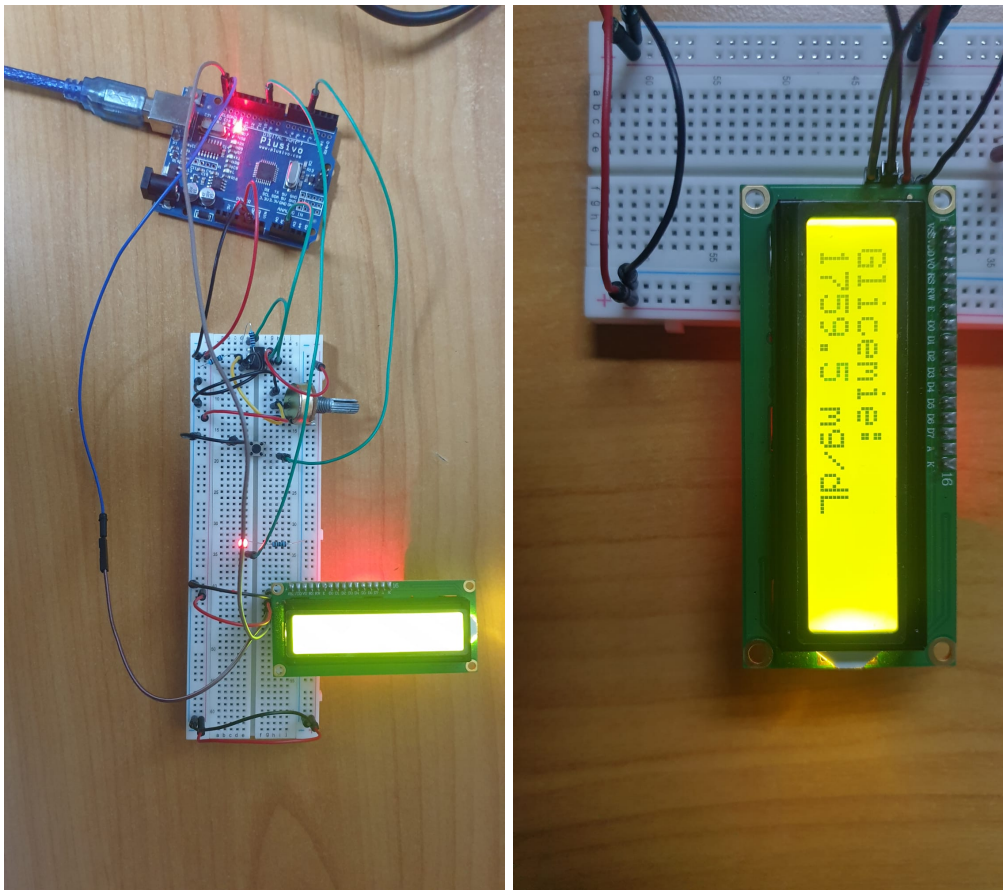
Circuit diagram:



Signal diagram:



Physical Circuit:



Software Design

Development Environment

The firmware was developed using the **Arduino IDE**, targeting the **ATmega328P** microcontroller on

the Arduino Uno platform. Code is written in **C/C++** using only core Arduino libraries:

- 'Wire.h' – for I2C communication,
- 'LiquidCrystal_I2C.h' – for interfacing with the LCD display.

No third-party libraries beyond the standard Arduino ecosystem were required.

Algorithm and Logic

The application firmware performs the following main steps:

1. System Initialization

- Sets pin modes for analog input, LEDs, LCD, and the interrupt pin for the button.
- Initializes the I2C communication and LCD display.

1. Measurement Trigger

- A measurement is initiated by pressing a push button connected to digital pin `D2`.
- This pin is configured to trigger an **external interrupt**.
- The interrupt sets a flag that is handled in the `loop()` function.

1. Analog Signal Acquisition

- The analog signal (a voltage) from the output of the OP07 amplifier is read using `analogRead()` on pin `A2`.
- The ADC returns a value between 0 and 1023, corresponding to a voltage between 0 and 5V.
- This voltage is proportional to the current generated by the simulated test strip (using potentiometer).

1. Current and Glucose Calculation

- The current in microamperes is calculated using Ohm's law: $I = V_{out} / R_f$
- The glucose level (mg/dL) is then estimated using a fixed proportionality factor derived during calibration: $Glucose = (Current_in_uA) * Sensitivity_Coefficient$
- The sensitivity coefficient was selected manually during testing using reference glucose values.

1. Classification and Output

- Based on calculated glucose value, the result is categorized into:
 1. Hypoglycemia,
 2. Normal,
 3. Hyperglycemia.
- A corresponding LED is lit using PWM pins (D3, D5, D6, D9, D10, D13) for visual alert.
- The result is also displayed on the I2C LCD, showing both the glucose level and category.

Current Implementation Status

- Analog signal reading via ADC (A2) – fully functional.
- Glucose calculation and classification – implemented and verified.
- PWM output on LEDs (pins D3, D5, D6, D9, D10, D13).
- LCD display via I2C – correctly displays readings and thresholds.
- External interrupt on D2 – used for manual measurement triggering.

Implemented functions

- **handleButtonInterrupt()** – Interrupt service routine (ISR) called when the user presses the button. It toggles the state of the glucometer (*glucometerON*) and sets a flag (*buttonPressed*) for state management.
- **turnOffAllLEDs()** – Sets PWM output of all six LEDs to zero using *analogWrite*. This ensures a clean reset when the system is turned off or reinitialized.
- **adjustRedPWM(diff)** – Called when hyperglycemia is detected. It calculates the intensity of red LEDs proportionally to how far the glucose level exceeds 150 mg/dL.
- **adjustYellowPWM(diff)** – Called in hypoglycemia cases. It increases the brightness of yellow LEDs proportionally to how far below 100 mg/dL the glucose level is.
- **adjustGreenPWM(glucose)** – Called when glucose levels are within normal range. It calculates a ratio within the 100–150 mg/dL band to smoothly vary the intensity between the two green LEDs.

Results Obtained

During the operation of the project, the current value generated by the potentiometer, which simulates the current from an electrochemical test strip, is read through the Arduino's analog input pin. Based on this current, the code calculates the estimated glucose concentration and displays it on the 1602 LCD screen.

According to the calculated glucose level, RGB LEDs are controlled via PWM signals to indicate different glycemetic states:

- For values below 100 mg/dL, yellow LEDs light up, indicating hypoglycemia.
- For values between 100 and 150 mg/dL, green LEDs are active, indicating a normal level.
- For values above 150 mg/dL, red LEDs turn on, signaling hyperglycemia.

These results were validated through manual testing by adjusting the potentiometer and observing the corresponding changes on the display and LED intensity, confirming the correct operation of the system in simulating glucose measurement.

The system responds quickly to changes in current and provides a clear user interface through the LCD display and visual LED indicators.

Conclusion

This project successfully demonstrates a basic but functional glucometer built with Arduino. By simulating the sensor signal with a potentiometer and using an operational amplifier, we were able to measure small currents and convert them into glucose levels. The system provides clear feedback through an LCD display and colored LEDs, offering both visual and numerical information to the user.

Download

Source code:

<https://github.com/AdrianIulian26/GlucoMeter>

Jurnal

- **03.05.2025** - Created the wiki page.
- **05.05.2025** - Conducted more in-depth research on the project.
- **10.05.2025** - Prepared the Introduction, General Description and Block Diagram.
- **15.05.2025** - Order the operational amplifier and potentiometer to complete the project..
- **18.05.2025** - Completed the Hardware Design milestone.

Bibliografie/Resurse

Hardware

- OP07 Operational Amplifier Datasheet

[<https://www.ti.com/lit/ds/symlink/op07.pdf> TI OP07 Datasheet]

- How Electrochemical Glucose Sensors Work

[<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6790891/> PMC Article on Electrochemical Glucose Biosensors]

- Principles of Glucose Meter Operation (Explanation)

[<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3904486/> NIH PMC Article on Glucose Meter Principles]

Software

- PWM Control with Arduino (Explanation and Examples)

[<https://www.arduino.cc/en/Tutorial/PWM> Arduino PWM Tutorial]

- Arduino Serial Communication Basics

[<https://www.arduino.cc/en/Serial/Begin> Arduino Serial]

[Export to PDF](#)

From:

<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:

http://ocw.cs.pub.ro/courses/pm/prj2025/eradu/iulian_adrian.andrei



Last update: **2025/05/26 16:15**