Project Presentation

Nixie Tube Clock

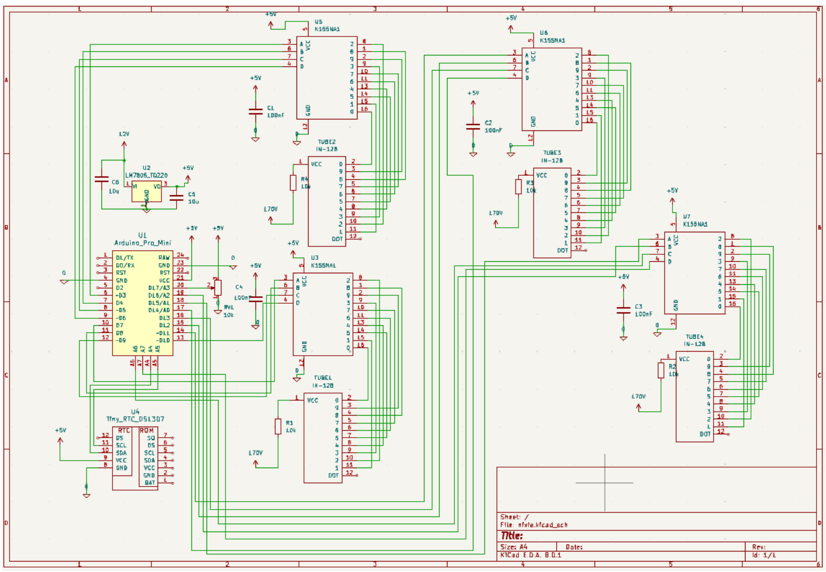
Pavlo Kostushevych

1. Project Assumptions

- The clock will display the current time in hours and minutes format using Nixie tubes.
- Accurate timekeeping will be ensured by using a Real-Time Clock (RTC) module.
- The device will be powered by a direct current (DC) power supply.
- The main control unit will be an Arduino Pro Mini microcontroller, responsible for:
    - reading data from the RTC module,
    - processing information,
    - controlling the Nixie tubes.
- The project will be implemented on a double-sided universal printed circuit board (PCB), allowing flexible component placement and facilitating assembly and soldering.
- Wires and pin-header connectors (male and female) will be used to connect individual circuit elements, enabling easy assembly, disassembly, and modifications during prototyping.
- High-voltage circuits required to power the Nixie tubes will be properly insulated and secured to ensure user safety.

2. Schematic (KiCad)

Below is the schematic diagram of the Nixie clock, developed using KiCad. The project is divided into four main functional blocks: power supply, microcontroller, display system, and high-voltage power supply.



Nixie Clock Schematic Diagram (developed in KiCad)

2.1 Power Supply

The LM7805 voltage regulator converts the 12 V input into a stable 5 V supply for the digital part of the circuit. Filtering capacitors eliminate noise and ensure stable operation.

2.2 Microcontroller

The central control unit is an Arduino Pro Mini. It communicates with the DS1307 RTC module via the I²C interface and sends control signals to the Nixie tube drivers.

2.3 Nixie Displays

Four IN-17 tubes are driven by K155ID1 driver ICs. Each driver controls one tube, and the Arduino selects the active digit via its outputs. Additionally, 10 kΩ resistors are used to limit the current in the signal lines.

2.4 High-Voltage Power Supply

The Nixie tubes require approximately 170 V DC, which is provided by the NCH6100HV module. The high-voltage section is properly isolated at the PCB level to ensure safe operation.

3. Components

Below is a list of components used in the Nixie clock project along with brief descriptions of their functions.

No.	Component Name	Symbol / Model	Qty	Notes
1	Microcontroller	Arduino Pro Mini	1	5V, 16 MHz
2	Real-time clock module	DS1307 RTC + CR2032	1	With I²C interface
3	Voltage regulator	LM7805L	1	Converts 12 V to 5 V
4	Nixie display tube	IN-17	4	Numeric display, 10 digits
5	Nixie driver IC	K155ID1	4	BCD to 10-line decoder
6	Resistor	10 kΩ	12	Current limiting for display drivers
7	Electrolytic capacitor	100 µF / 25 V	2	Voltage filtering
8	Ceramic capacitor	100 nF	2	Decoupling
9	Male and female headers	—	few	For easy assembly/disassembly
10	Jumper wires	—	few	For connections on the board

No.	Component Name	Symbol / Model	Qty	Notes
11	Double-sided prototyping board	—	1	For mounting all components
12	Power supply	12 V DC	1	Powers the entire circuit
13	HV boost converter	170 V DC boost converter	1	Powers the Nixie tubes

3.1 Nixie Tubes IN-17

Description: IN-17 Nixie tubes are classic display devices that use glow discharge to show digits from 0 to 9. Each digit has its own cathode, which glows when high voltage is applied. These tubes feature a distinctive cylindrical glass enclosure.

3.2 K155ID1 Driver ICs

Description: The K155ID1 integrated circuits are designed for direct control of Nixie tubes. They enable safe switching of high voltage using low-voltage logic signals from the microcontroller.

3.3 LM7805 Voltage Regulator

Description: A linear 5 V voltage regulator powered from a 12 V DC input. It provides a stable supply voltage for the microcontroller and other digital components.

3.4 Arduino Pro Mini (ATmega328P)

Description: A compact microcontroller board based on the ATmega328P chip. It acts as the main control unit—processing time signals and controlling digit display on the Nixie tubes.

3.5 RTC Module DS1307

Description: A real-time clock (RTC) module with an I²C interface that keeps accurate track of date and time (seconds, minutes, hours, day of the week, date, month, year). The module includes battery backup to maintain operation during power loss.

3.6 NCH6100HV High-Voltage Power Supply

Description: A module that converts low DC voltage (e.g., 12 V) to approximately 170 V DC, which is required for proper operation of Nixie tubes. It ensures a stable and safe high-voltage power supply for the displays.

3.7 Universal PCB 90 × 150 mm, Double-Sided

Description: A prototype board that allows through-hole soldering of components and flexible electrical routing on both layers.

3.8 Connection Wires

Description: Signal and power wires used to connect various modules and components during assembly and prototyping.

3.9 Male and Female Pin Headers

Description: Standard connectors that allow easy connection and disconnection of modules. They greatly simplify testing, modifications, and maintenance of the circuit.

3.10 SMD Capacitors (Size 1206)

Description: Filtering components used in the power supply section – they smooth out voltage and suppress impulse noise.

3.11 10 kΩ Resistors

Description: Used as current-limiting resistors in control paths and as pull-up/pull-down elements where needed.

4. Project Cost Estimate

This section presents a detailed cost estimate of the components used in the Nixie clock project. All parts were purchased on the Polish market, reflecting the actual costs of project implementation.
An exception is the IN-17 Nixie tubes, which were obtained free of charge from a friend in Ukraine who works in the electronics industry, where usable components are often discarded. Thanks to this, the overall project cost was significantly reduced.

Component	**Price [PLN]**
Universal double-sided PCB 90×150 mm	9.90
Electrolytic capacitor 10 µF / 50 V, 5×11 mm, THT (2 pcs)	0.20
Ceramic capacitor 100 nF / 50 V, THT (4 pcs)	0.40
Set of 15 cm black wires (100 pcs)	11.50
USB-UART converter FTDI FT232RL miniUSB + USB cable	39.90
Jumper wire set – 20 cm, 3×40 pcs (m-m, f-f, m-f)	17.50
Voltage regulator 5 V L7805ABV – THT TO220	1.90
Arduino Pro Mini 328 – 3.3 V / 8 MHz – SparkFun DEV-11114	63.90

Component	Price [PLN]
RTC module DS1307 + 32 kb EEPROM 24C32 I²C	5.95
Male pin header 1×40, 2.54 mm pitch, black (10 pcs)	3.90
Resistor justPi THT CF carbon, 1/4 W, 10 kΩ (30 pcs)	1.90
High voltage power supply 170 V DC NCH6100HV	41.59
Nixie tube drivers K155ID1 (4 pcs)	80.00
Nixie tubes IN-17 (4 pcs)	0.00*

| Total | 279.34 PLN |

5. Code

Below is the **full source code** of the Nixie Clock project, written in Arduino C++, with comments explaining its functionality:

```
#include <MD_DS1307.h>
#include <Wire.h>

// Pin definitions for controlling the Nixie tubes
#define A1 3
#define B1 4
#define C1 5
#define D1 6
#define A2 7
#define B2 8
#define C2 9
#define D2 10
#define A3 11
#define B3 12
#define C3 13
#define D3 14
#define A4 15
#define B4 16
#define C4 2
#define D4 1

#define pot A3 // Potentiometer pin (not used in the code)

// Arrays of pins for convenient control
char A[4] = {A4, A3, A2, A1};
char B[4] = {B4, B3, B2, B1};
char C[4] = {C4, C3, C2, C1};
char D[4] = {D4, D3, D2, D1};
```

```
// Variables for storing digits and time
int zero;
int one;
int two;
int three;
int hour;
int minute;

void setup() {

// Set control pins as outputs
  pinMode(A1, OUTPUT);
  pinMode(B1, OUTPUT);
  pinMode(C1, OUTPUT);
  pinMode(D1, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(B2, OUTPUT);
  pinMode(C2, OUTPUT);
  pinMode(D2, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(B3, OUTPUT);
  pinMode(C3, OUTPUT);
  pinMode(D3, OUTPUT);
  pinMode(A4, OUTPUT);
  pinMode(B4, OUTPUT);
  pinMode(C4, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(pot, INPUT); // Potentiometer as input (unused)

  // Initially turn off all digits on the tubes (set pins HIGH
  for (char i = 0; i < 4; i++) {
    digitalWrite(A[i], HIGH);
    digitalWrite(B[i], HIGH);
    digitalWrite(C[i], HIGH);
    digitalWrite(D[i], HIGH);
  }

    // Initialize RTC module
  if (!RTC.isRunning())
    RTC.control(DS1307_CLOCK_HALT, DS1307_OFF);
Serial.begin(9600);
}

void loop() {
  RTC.readTime(); // Read current time from RTC
  hour = RTC.h;
  minute = RTC.m;
```

```
  // Calculate digits to be displayed
  zero = (hour / 10) % 10;
  one = hour % 10;
  two =  (minute / 10) % 10;
  three = minute % 10;

  // Debug output to serial port
Serial.println(hour);
Serial.println(minute);
Serial.println(zero);
Serial.println(one);
Serial.println(two);
Serial.println(three);

offAll(); // Turn off all digits

    // Display digits on respective tubes
  writenumber(0, zero);
  writenumber(1, one);
  writenumber(2, two);
  writenumber(3, three);

  delay(1000); // Refresh every second

}

// Function to display digit b on tube with index a
void writenumber(int a, int b) {
  switch (b) {
    case 0:
      digitalWrite(A[a], LOW);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], LOW);
      break;
    case 1:
      digitalWrite(A[a], HIGH);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], LOW);
      break;
    case 2:
      digitalWrite(A[a], LOW);
      digitalWrite(B[a], HIGH);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], LOW);
      break;
    case 3:
      digitalWrite(A[a], HIGH);
```

```
      digitalWrite(B[a], HIGH);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], LOW);
      break;
    case 4:
      digitalWrite(A[a], LOW);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], HIGH);
      digitalWrite(D[a], LOW);
      break;
    case 5:
      digitalWrite(A[a], HIGH);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], HIGH);
      digitalWrite(D[a], LOW);
      break;
    case 6:
      digitalWrite(A[a], LOW);
      digitalWrite(B[a], HIGH);
      digitalWrite(C[a], HIGH);
      digitalWrite(D[a], LOW);
      break;
    case 7:
      digitalWrite(A[a], HIGH);
      digitalWrite(B[a], HIGH);
      digitalWrite(C[a], HIGH);
      digitalWrite(D[a], LOW);
      break;
    case 8:
      digitalWrite(A[a], LOW);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], HIGH);
      break;
    case 9:
      digitalWrite(A[a], HIGH);
      digitalWrite(B[a], LOW);
      digitalWrite(C[a], LOW);
      digitalWrite(D[a], HIGH);
      break;
  }
}

// Turns off the digit on tube with index a (sets all pins HIGH)
void off(int a) {
  digitalWrite(A[a], HIGH);
  digitalWrite(B[a], HIGH);
  digitalWrite(C[a], HIGH);
  digitalWrite(D[a], HIGH);
```

```
}

// Turns off all digits on all tubes//
void offAll() {
  for (int i = 0; i < 4; i++) {
    digitalWrite(A[i], HIGH);
    digitalWrite(B[i], HIGH);
    digitalWrite(C[i], HIGH);
    digitalWrite(D[i], HIGH);
  }
}
```

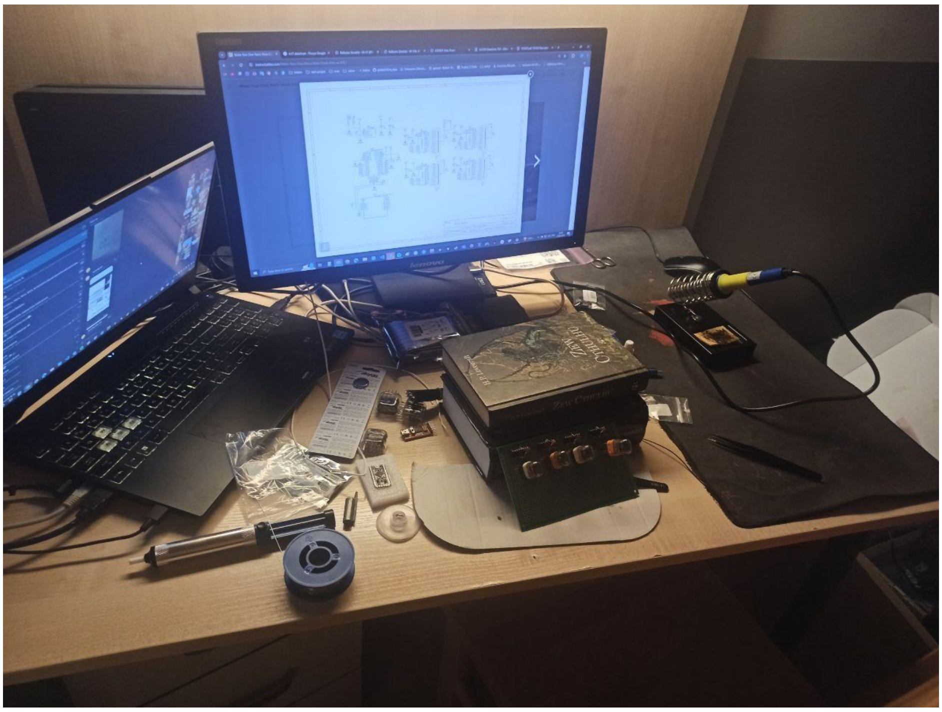
Code Explanation

The presented code is responsible for controlling four **IN-17 Nixie tubes**, displaying the current time retrieved from a **DS1307 RTC module**.

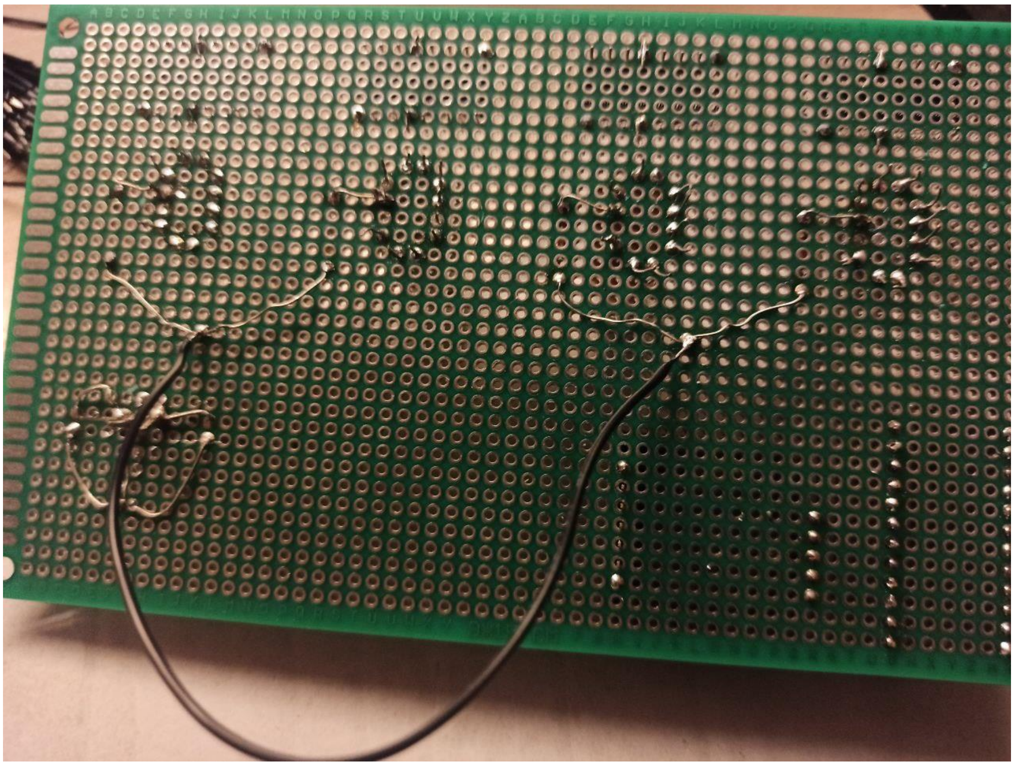
- In the `setup()` function, the control pins and the clock module are initialized.
- The `loop()` function reads the current time every second and extracts each digit (HH:MM), then passes the digits to the `writenumber()` function.
- `writenumber()` sets the appropriate BCD pin states to light the correct digit on the corresponding tube.
- The helper functions `off()` and `offAll()` ensure that no more than one digit is active on any tube at a time, preventing multiple digits from overlapping or glowing simultaneously (a common problem in Nixie tube circuits).

6. Assembly and Soldering Process

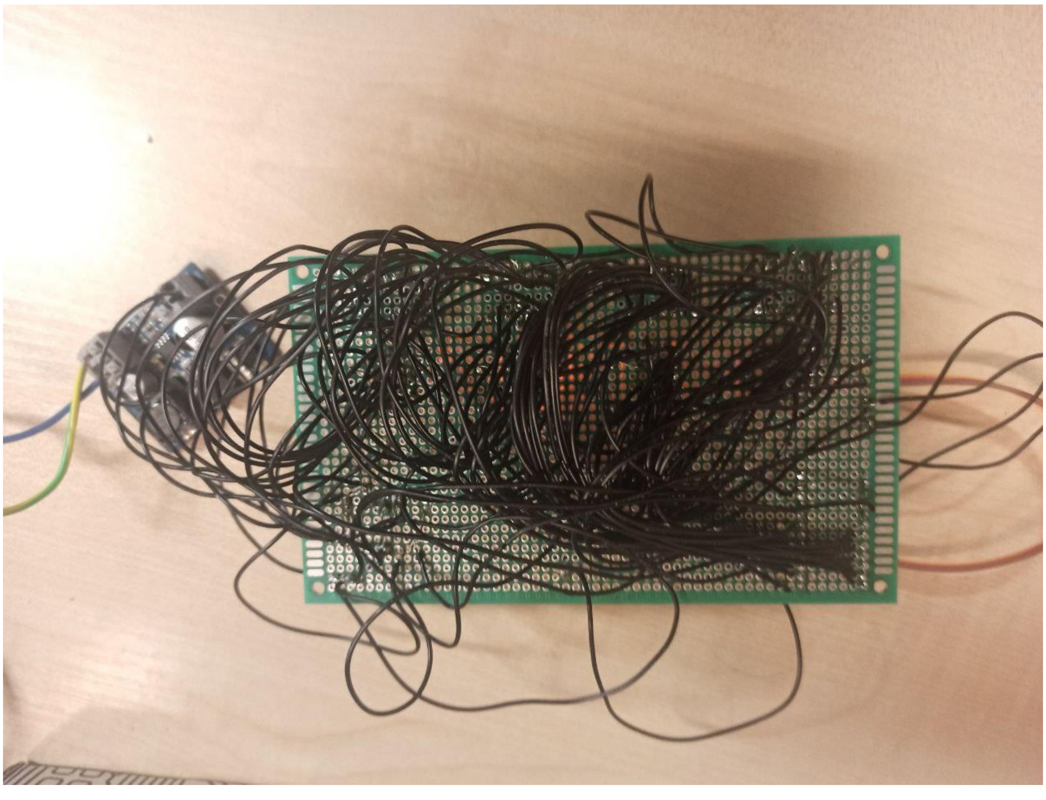
The following series of photos illustrates the step-by-step process of building the Nixie tube clock on a double-sided prototype board.



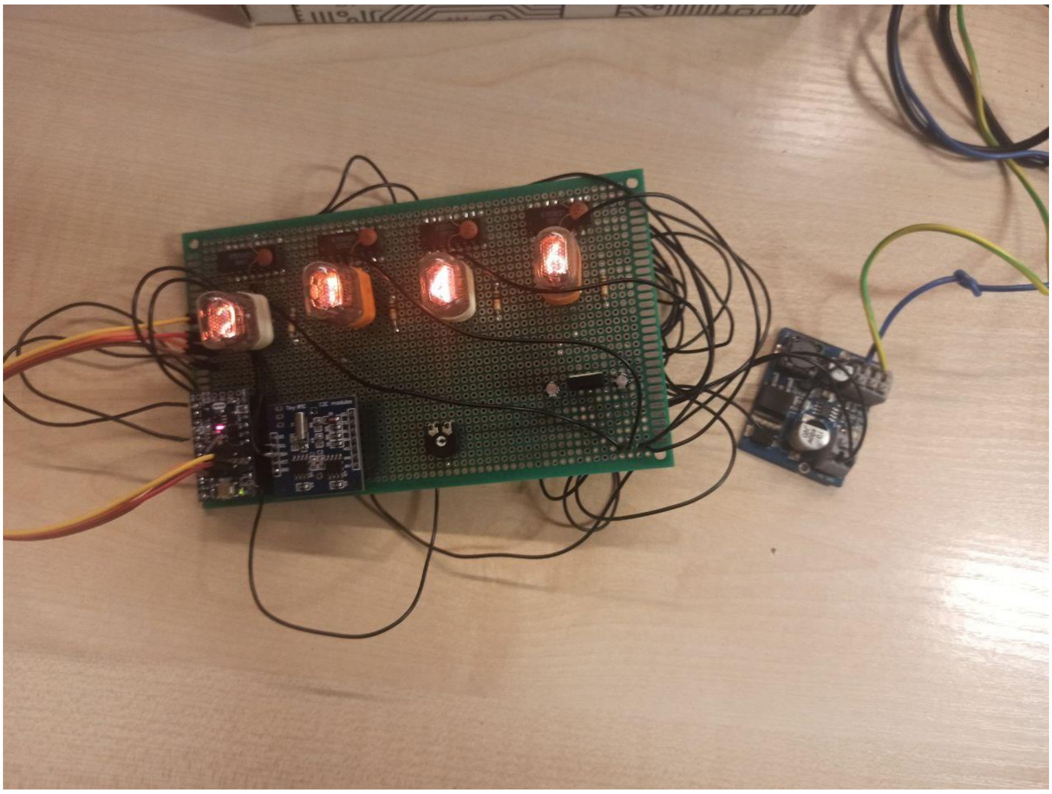
The first photo shows the workplace setup during the assembly and soldering of the Nixie-based clock.



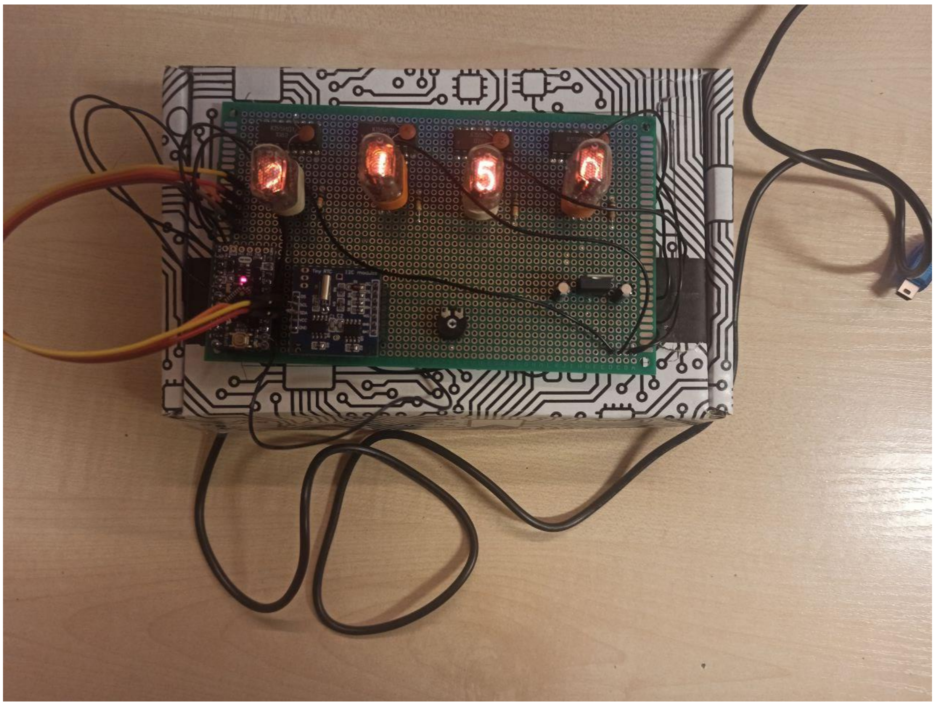
The second image displays the bottom side of the universal PCB, showing the early stage of assembly.



View of the board after most of the wiring has been completed.



The fourth image presents the powered-on clock circuit in operation.



The final stage of the assembly is depicted with the board placed in a temporary enclosure.

A short video demonstrating the Nixie tube digits test is available on YouTube:
https://youtube.com/shorts/KTlzbwLbgn8?feature=share

6. Summary:

The main challenge during the project development was suboptimal time management, which led to completing the project near the end of the semester and giving up on designing a dedicated printed circuit board (PCB). Instead, a universal prototype board and jumper wires were used, which made precise soldering more difficult.

After the assembly phase, no new hardware issues occurred. Any further difficulties appeared during testing of the Arduino firmware and the RTC module.

Several significant errors were identified during testing:

- The second tube did not display even digits due to the lack of a low signal on Arduino pin 11. This issue was resolved by using a potentiometer.
- The first tube did not work initially because pins A6 and A7 were used — these are analog input-only and not suitable for signal output. After switching to proper digital pins, the tube functioned correctly.
- RTC configuration problems stemmed from an incorrect serial baud rate. Although the code specified 9600, the serial monitor required 4800. This issue was resolved through trial and error, which led to proper configuration and stable operation of the RTC module.

After adding a backup battery to the RTC and performing tests, the clock retains the correct time even after being powered off and back on.

All photos and videos documenting the project — including the assembly process, soldering stages, and final testing — can be found in the respective folders of the GitHub repository: https://github.com/pavlokostushevych/Nixie-Tube-Clock.git