**TECHNICAL UNIVERSITY OF MOLDOVA**

**FACULTY OF COMPUTERS, INFORMATICS AND MICROELECTRONICS**

**DEPARTMENT OF SOFTWARE ENGINEERING AND AUTOMATION**

**Laboratory Work Nr. 1.2**

**User interaction. LCD and Keypad**

**Done by: Chihai Nichita**

**st. gr. FAF-213**

**Verified by: Moraru Dumitru**

**univ. lect.**

**Chisinau, 2024**

# 1 THE TASKS OF LABORATORY WORK

Configure an application to work with the STDIO library through the serial interface for text exchange via LCD & Keypad.

Design a MCU-based application to detect a code from a 4x4 keypad, verify the code and display a message on an LCD.

- for a valid code, a green LED should light up, for an invalid code, a red LED.

- use STDIO to scan the keypad and display on the LCD.

# 2 PROGRESS OF THE WORK

For this laboratory work, I utilized the Arduino IDE, a software designed for writing and compiling code for the Arduino development board, along with Proteus, a software used for electronic design simulation.

**2.1 Main function description**

For this project my code performs the task of creating a security entry system using a 4x4 keypad and a 16x2 LCD display. Below I described the main functions that I used in the code in order to complete the laboratory work:

The *setup()* function initializes the LCD display with the specified width and length, and it sets the designated pins (*RED\_PIN* and *GREEN\_PIN*) as output pins for LEDs that indicate correct and incorrect code entries.

In the *loop()* function, the program continuously runs, checking if a key is pressed on the keypad using *keypad.getKey().* Upon pressing a key, the code prints the key on the LCD display at the current cursor position and appends it to the *currentCode* string. The cursor position on the LCD is updated accordingly [[1](#one)].

When a complete code is entered (cursorColumn equals codeLength), the program compares the entered code with the predefined valid code (*validCode*). If the codes match, it lights up the green LED and displays *"CORRECT"* on the LCD. If they do not match, it lights up the red LED and displays *"WRONG"* on the LCD. Subsequently, the program resets cursorColumn, clears the LCD, and resets the *currentCode* string. This allows to run the program multiple times and check on the circuit board how it is manifesting.

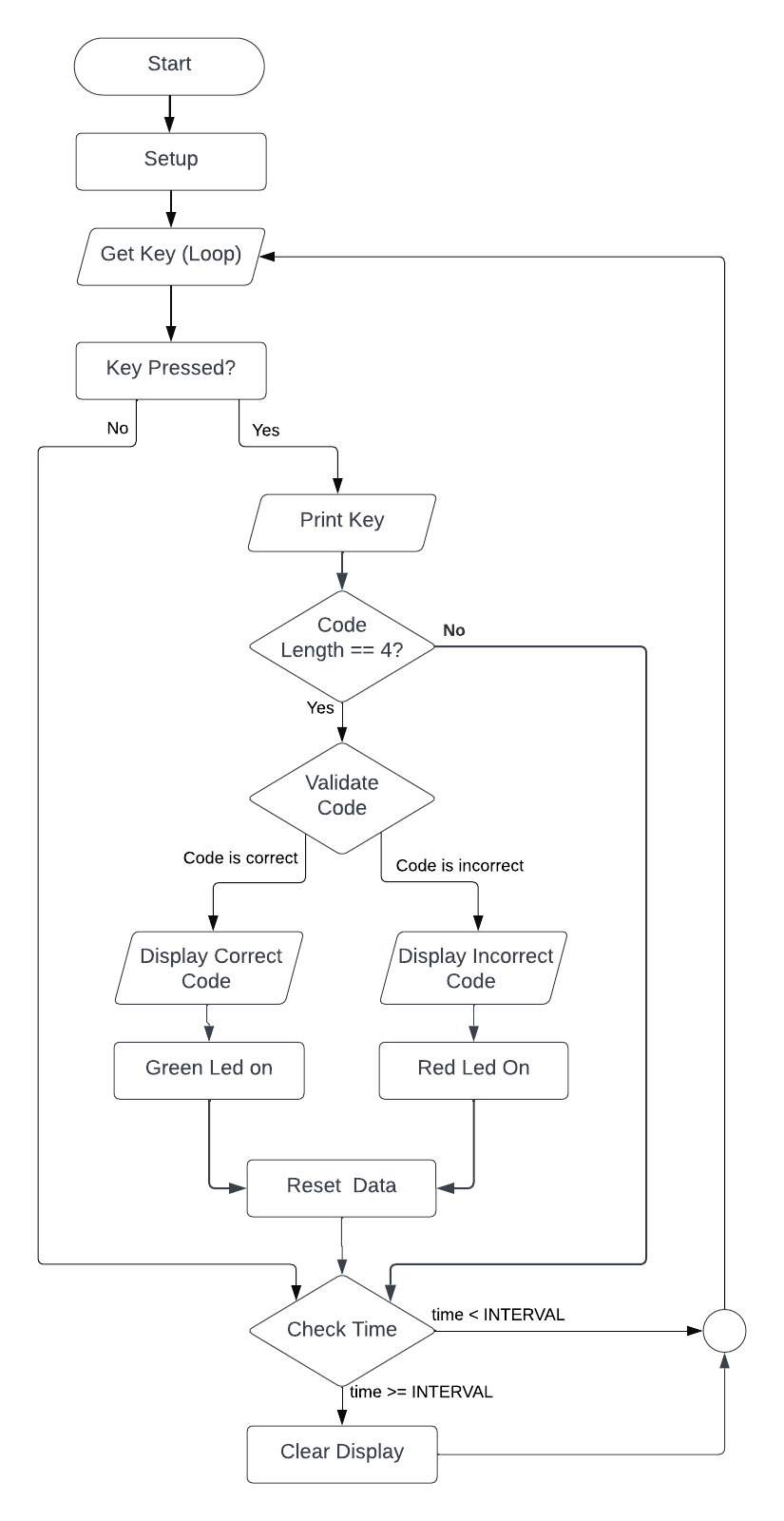
If the entered code was correct, the *previousMillis* variable is updated to the current time to initiate an LED timeout. The program also checks for conditions to turn off the LEDs and clear the LCD after a specific time interval.

The *millis()* function is used to obtain the number of milliseconds since the Arduino started running. The *digitalWrite(pin, state)* function is employed to control the LEDs, turning them on or off based on the specified state.

Moreover, the LCD-related functions such as *lcd.begin(width, length)*, *lcd.setCursor(column, row)*, *lcd.print(value)*, and *lcd.clear()* are utilized to interact with and display information on the LCD. The *makeKeymap(keys)* function configures the key mapping for the keypad, and the Keypad constructor initializes the Keypad object with the specified parameters, including the keymap and pin configurations [[2](#two)].

**2.2 Block diagram**

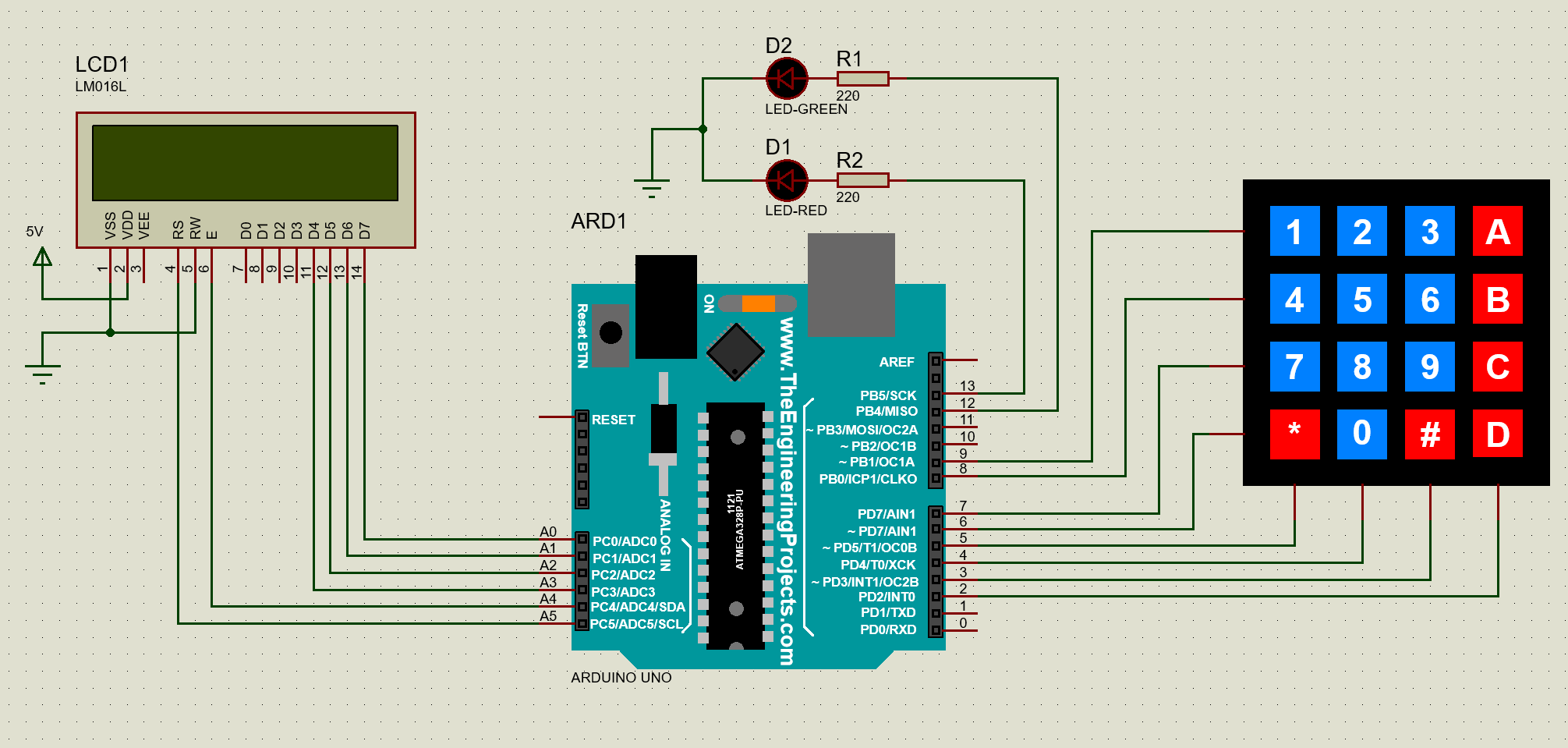
This block diagram provides a visual representation of the main components and their interactions for my Arduino program.



**Figure 1 Block diagram**

**2.3 Simulated schematic**

This is the simulated Arduino Uno scheme assembled in Proteus, here are represented 7 elements: Arduino Uno with its microcontroller, a red and green LED, two resistors a 4x4 Keypad and a LCD [[3](#three)].

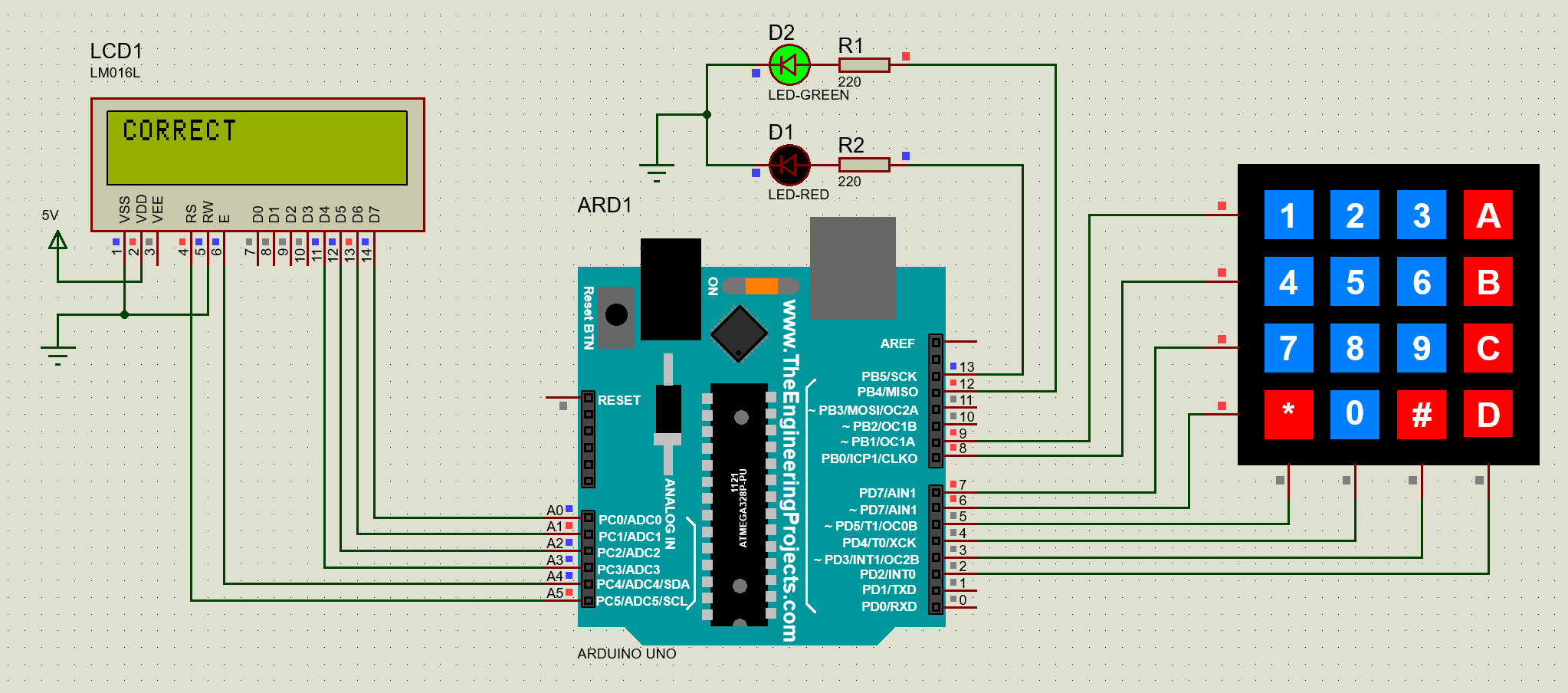


**Figure 2. Electrical schematic**

This circuit shows the pins that I used to connect the elements with the Arduino UNO. For the connection of LEDs, I used pin 12 for green LED and pin 13 fore red LED. For the connection with LCD I used pins A0, A1, A2, A3, A4, A5 and mapped them accordingly to the LCD component, so that when I start the simulation it will run correctly. I also powered the LCD with a 5V source and grounded RS and VSS pin.

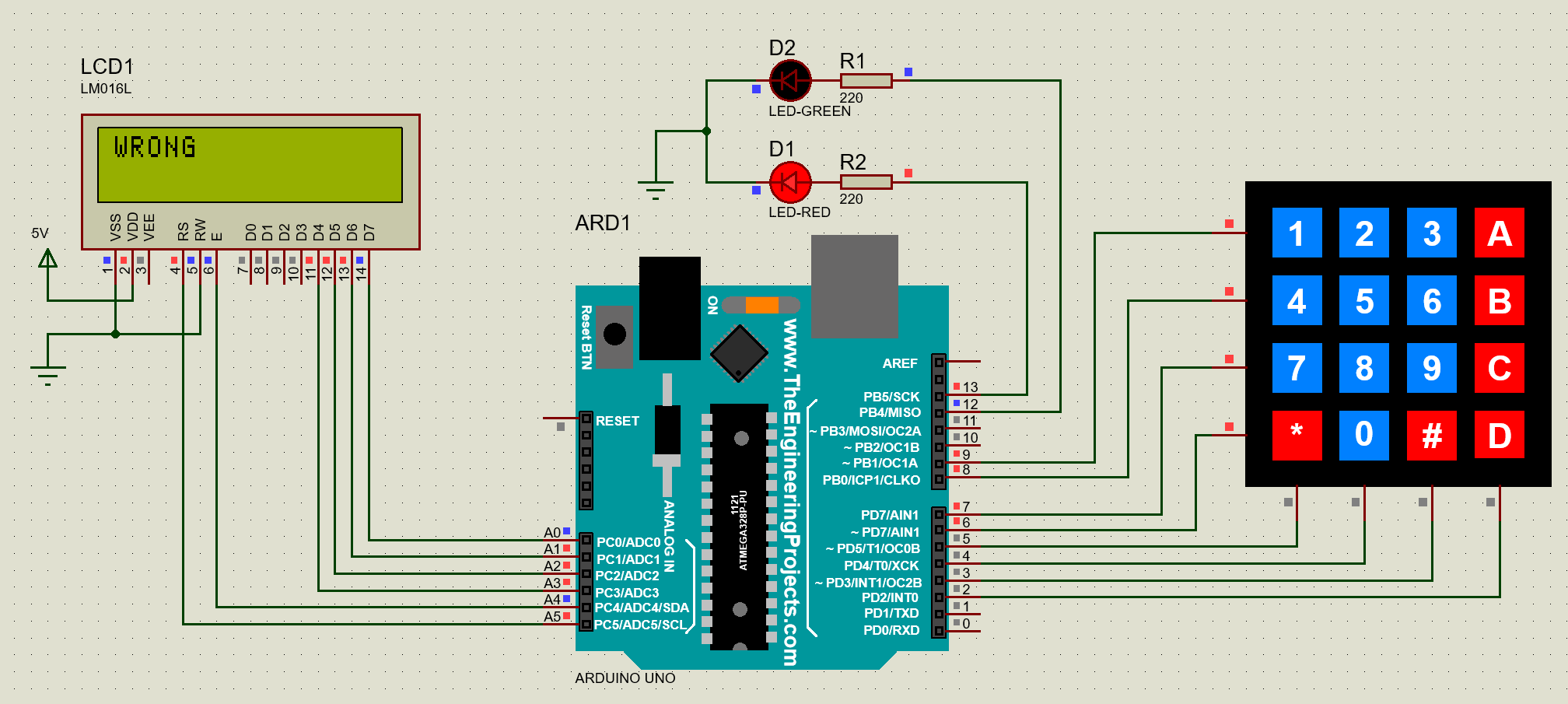
**2.4 Photos of running simulation**

This example shows the simulation of the valid code typed ‘1A2B’, when it happens the LCD will print a certain message and the green LED will turn on.

****

**Figure 3 Simulation when the code is valid**

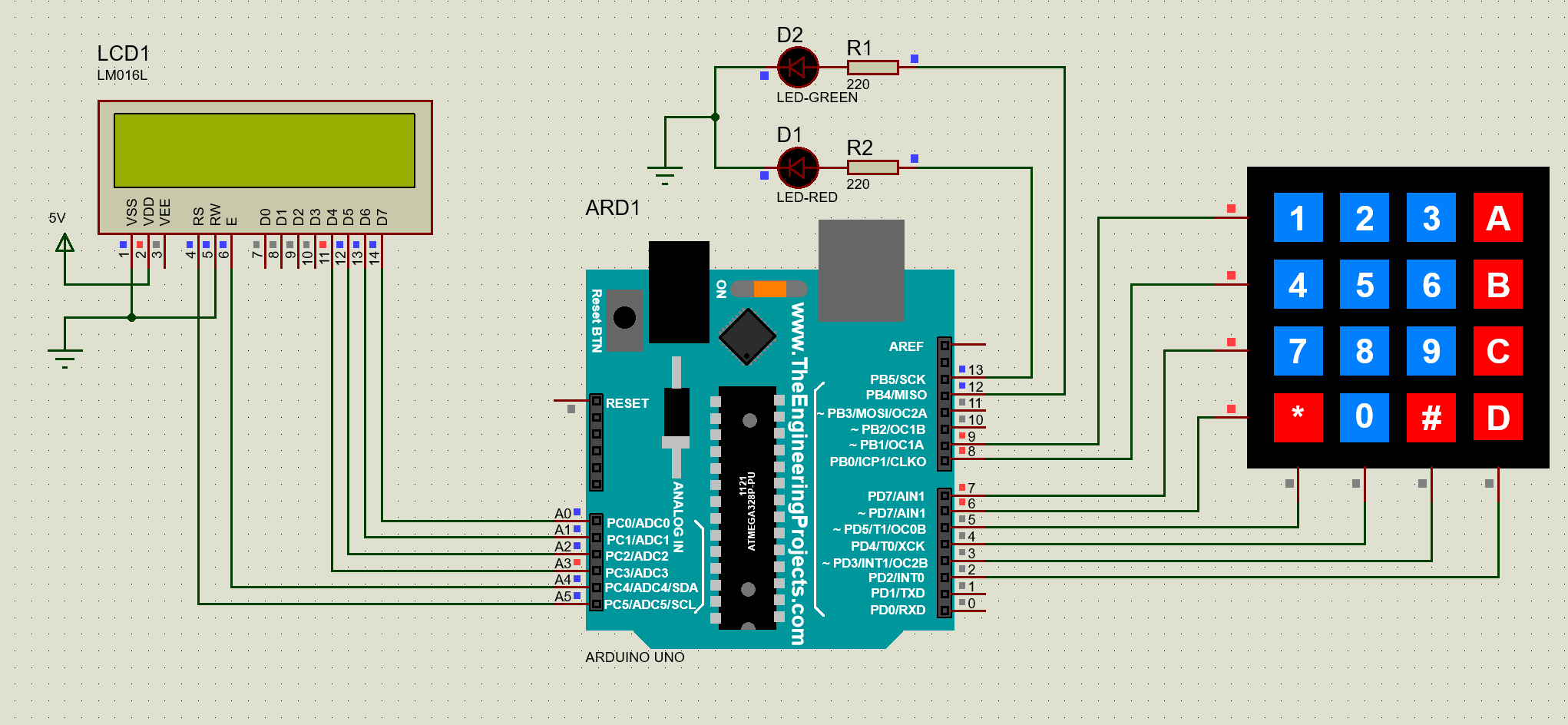
This example shows the simulation of an invalid code, when user types an incorrect code the LCD will print a certain message. In this case ‘WRONG’, and the red LED will turn on accordingly. This LED will turn red and type the message ‘WRONG’ as long as the user will come up with the correct code. So, the simulation has just one valid case, the rest of them will trigger the invalid ones, and show the certain result.



**Figure 4 Simulation when the code is invalid**

This image shows how the circuits looks when the simulation has started. To see the code printed on the LCD, the user has to type on the Keypad which is on the right of the image. After the user types in some numbers, letters, symbols, the LCD will print whether the code typed is correct or wrong.

Also, the number of characters that can be displayed on the LCD is 4, after the user inserted the 4th character, the LCD will print whether the code that was inserted is valid or invalid.



**Figure 5 Start simulation**

# CONCLUSION

This laboratory work was a great experience for me. I learned a lot of small concepts, rules that helped me realize this project.

First of all, I used a 4x4 keypad and a 16x2 LCD display that offered a hands-on way to implement access control. As users enter their codes, the LCD display provides real-time feedback, making the entire process intuitive and engaging.

Second of all, I was able to create a simple scheme that has helped me simulate my project in real-time. I managed to correctly connect the elements used in my project, such as Arduino Uno, two resistors, green and red LED, 16x2 LCD and a 4x4 keypad.

To sum up, this laboratory work has deepened my understanding of electronic circuits and Arduino development. The project’s success will contribute to my future projects in the field of embedded systems.

# BIBLIOGRAPHY

**1** Arduino LCD: LCD documentation for arduino. [online], [accesed 15.02.2024]. Available at: [Arduino - LCD | Arduino Tutorial (arduinogetstarted.com)](https://arduinogetstarted.com/tutorials/arduino-lcd)

**2** Arduino LCD Keypad: LCD and Keypad documentation for arduino. [online], [accesed at 15.02.2024]. Available at: [Arduino - Keypad - LCD | Arduino Tutorial (arduinogetstarted.com)](https://arduinogetstarted.com/tutorials/arduino-keypad-lcd)

**3** Proteus Documentation: Information for building circuits in proteus. [online], [accesed at 16.02.2024]. Available at: [How to Simulate Arduino Projects Using Proteus | Arduino | Maker Pro](https://maker.pro/arduino/projects/how-to-simulate-arduino-projects-using-proteus)