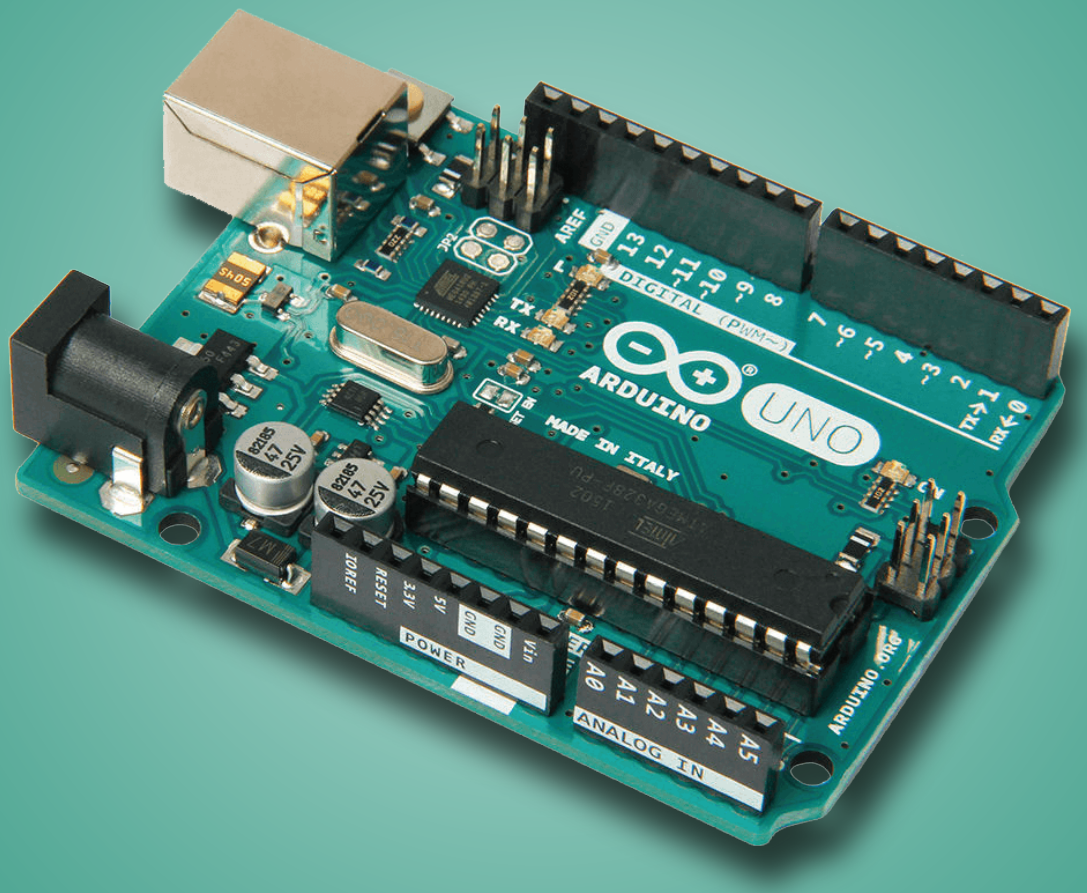
Arduino

Teachers' Book



Level 2

Rana Dajani

Arduino Teachers' Book - Level 2

Published by **LKD Educational Resources 2023**

Amman - Jordan Tel: +962 6 5374141

Fax: +962 6 5516404 P.O.Box: 851346

Email: info@lkd.com.jo / info@ lkdebooks.com

Websites: www.lkd.com.jo / www.lkdebooks.com

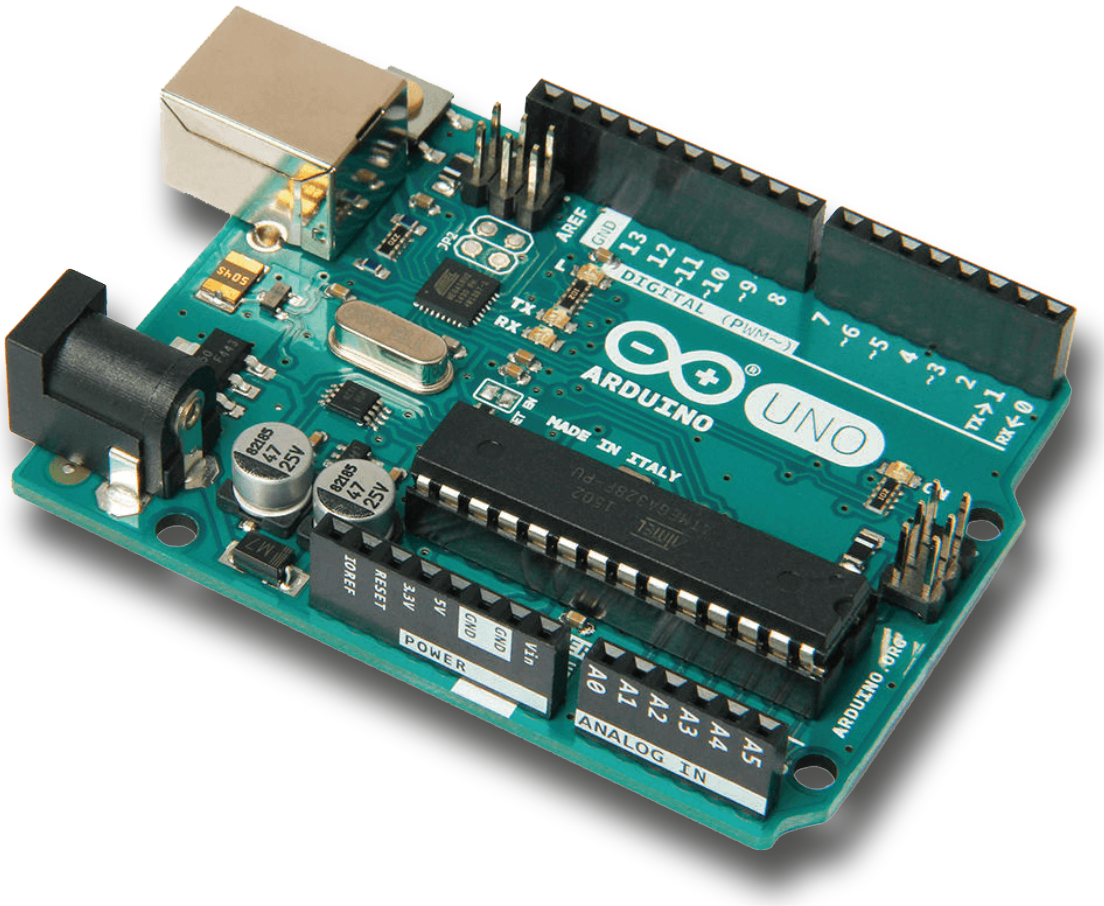
Author

Rana Dajani

ISBN: **2023/9/4940**

Arduino

Teachers' Book



Level 2

Rana Dajani

Table of Contents

Introduction

This is the level two teachers' guide for Arduino projects' book, the continuation of our level one book that builds upon the basics of electronics and programming using the Arduino C++ language. This guide is designed to provide you with the necessary resources to support your students as they delve deeper into the world of Arduino and expand their skills and knowledge.

In level one, students were introduced to the main components of the Arduino board, including the Arduino microprocessor, breadboards, jump wires, and various input and output components such as LEDs, buzzers, buttons, switches, sensors, potentiometers, and LCD screens. They learned how to use digital and analog pins, as well as PWM (pulse width modulation) to control and interact with these components.

Building on this foundation, level two focuses on two comprehensive themes: smart car and smart house, with ten lessons for each theme. The projects in this book cover a wide range of applications, allowing students to further explore the capabilities of Arduino and apply their knowledge in real-world scenarios.

The table of contents in this book has been color-coded to differentiate between different types of lessons:

- The first five projects for each theme can be completed using an online virtual simulator.
- The next four projects require physical handling of the components in the kit, allowing students to build and test their projects in a tangible way.
- The tenth and final project is open-ended, challenging students to apply the skills they have learned to create a fully developed project of their own design.

As a teacher, you play a vital role in guiding your students as they progress through the level two Arduino projects' book. This guide provides tips, strategies, and resources to help you effectively teach Arduino to your students, and we encourage you to adapt the materials to your own teaching style and the unique needs of your students. By continuing to explore the possibilities of Arduino, your students will further develop their creativity, critical thinking, and problem-solving skills, preparing them for success in the world of electronics, coding, and innovation.

Troubleshooting Tips

"It is not working!" is a common exclamation made when working with electronics. But, have no fear. There are numerous troubleshooting tips and possible solutions to problems that frequently come up that make diagnosing problems much simpler than it may seem.

Hardware Checks

Check Your Connections

It is best to double-check wired connections just in case a wire was connected incorrectly or is loose. Certain components have a certain polarity and will only function when it is connected in the correct position. If everything appears to be wired correctly, you might want to try going back to the beginning and looking at the basic hookup. Also, disconnecting the parts and rewiring each circuit can help troubleshoot. This will assist in narrowing down the problem.

- □ Is there continuity between the pins, and is the connection secure?
- □ Is the wire or cable bad?
- □ Was the component connected using the correct polarity?
- □ Did you use the basic hookup?
- □ Is there a problem with the USB Hub or USB port that you are using?

Check Your Logic Levels

- □ Can you connect a 3.3V sensor to a 5V board?

Check Your Power Supply

- □ Is your power supply sufficient?

Software Checks

Board Selection

- □ Did you select the correct board definition?

COM Port Selection

- □ Did you select the correct COM port?
- □ Remove any connections to the 0(RX) and 1(TX) pins when uploading code.
- □ COM port number changes when connecting board on different USB ports.

Installed Arduino Libraries

- □ Does your example code require libraries to be installed?

Program Checks

If you get a compilation error when trying to compile or upload code to your board, check for errors in syntax, typos and more. When compilation fails, the IDE (Integrated Development Enviroment) will present you with the errors on its bottom part. However, the error messages generated by the Arduino IDE are limited in their description and therefore not always very helpful, in which case it is a good idea to use Google to search for solutions to the problem.

Best Practices

- Write code in small chunks and test each of them.
- Provide meaningful names for variables and functions.
- Write comments to explain coding choices for future reference.
- Make sure your code has a proper indentation and remains readable at all times. you can use Alt-t to auto-indent the whole sketch.

Common syntax errors are:

- missing semicolon ; at the end of each line
- misplaced or missing parenthesis { }
- typo of misspelled words or case sensitivity
- undefined variables and functions

Serial Monitor

Use the serial monitor to print an overview of the current state of the program. As such:

- variables
- inputs – sensor readings
- prints that indicate the programs flow, like inside an 'if' statement to see whether the condition was met
- outputs – e.g. PWM values before writing them to the pin
- anything else you find important to print to screen

Check, Test and Recheck

Online Research for Resources

Try using an online search engine, chances are, there is someone in the world who has done the project (or something similar) already, and they have provided some documentation online.

Suggested Time
45-60 minutes

Lesson 1: RGB LED

Overview

In this lesson, students will review and expand their knowledge of programming and controlling components with Arduino by working with a new component, the RGB LED. They will learn about multidimensional arrays in C++ and how to use the analogWrite() function to control the intensity of red, green, and blue LEDs in the RGB LED, creating different colors. They will also learn how to use the PWM pins on the Arduino board and incorporate delay time to create a rainbow pattern with the RGB LED.

Vocabulary

- **RGB LED**: a type of LED that can display different colors by combining red, green, and blue LEDs.
- **Multidimensional arrays**: arrays with more than one dimension, such as a table with rows and columns, used to store and manipulate data.

Before the Lesson

- Review prior knowledge of coding and electric circuits concepts from Arduino Projects Book – Level 1
- It is important to ensure that each student has the necessary tools and components.

The Lesson Plan

Objectives:

- To review and expand students' knowledge of programming and controlling components with Arduino
- To teach students how to work with RGB LED and create a rainbow pattern using multidimensional arrays and the analogWrite() function

Materials:

- Arduino board and Arduino IDE software
- Breadboard, jumper wires and USB cable
- RGB LED
- 220Ω resistors

Procedure:

Introduction (*10 mins*)

- Review the concept of programming and its importance in automating, managing, and analyzing data and information accurately.
- Introduce the RGB LED and explain how it can display different colors by combining red, green, and blue LEDs.
- Explain the concept of multidimensional arrays in C++ and how they can be used to store and manipulate data in a table-like format.
- Review the use of PWM pins on the Arduino board and the analogWrite() function to control the intensity of an output signal.

Activity (*30 mins*)

- Instruct students to build their circuits.
- Instruct students to open the Arduino IDE and write the given program following the pseudo code instructions that will blink the LED on and off using the digitalWrite() and delay() functions.
- Have students upload their program to the Arduino board and observe the blinking LED.

Conclusion (*5 mins*)

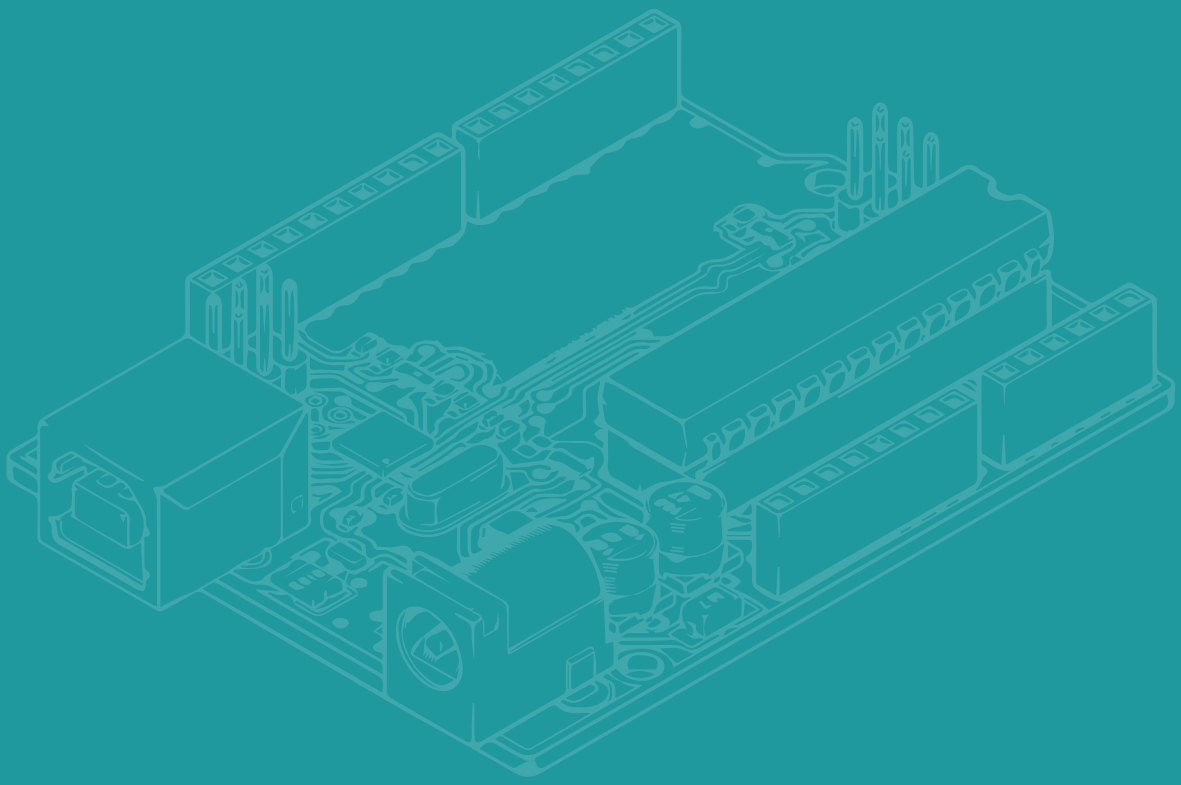
- Have students share their programs with the class and explain how they created the rainbow pattern using multidimensional arrays and the analogWrite() function.

Challenge Assessment:

Encourage students to experiment and try to solve the challenge extension of the lesson.

Suggested Challenge Solution:

```
int R=11;
int B=10;
int G=9;
int r=6;
int b=5;
int g=3;
int rainbow[7][3]={ {255,0,0}, //red
                    {255,153,153}, //orange
                    {148,0,211}, //yellow
                    {0,0,255}, //green
                    {0,191,255}, //blue
                    {20,255,100}, //navy
                    {255,255,0}, //voilet
                  };
void setup(){
  pinMode(R, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(G, OUTPUT);
}
void loop(){
  analogWrite(R, random(255));
  analogWrite(r, random(255));
  analogWrite(B, random(255));
  analogWrite(b, random(255));
  analogWrite(G, random(255));
  analogWrite(g, random(255));
  delay(200);
}
```



ISBN: 2023/9/4940

ARDUINO