# Table of Contents

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Contents</td>
<td>i</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>HARDWARE</td>
<td>1</td>
</tr>
<tr>
<td>CODE</td>
<td>4</td>
</tr>
<tr>
<td>Pseudo Code:</td>
<td>4</td>
</tr>
<tr>
<td>Coding Tools:</td>
<td>4</td>
</tr>
<tr>
<td>1 – JavaScript Block Editor:</td>
<td>4</td>
</tr>
<tr>
<td>2 – Python Editor [ADVANCED]</td>
<td>5</td>
</tr>
<tr>
<td>3 – Microsoft Block Editor [OLD]</td>
<td>6</td>
</tr>
<tr>
<td>Running Programs:</td>
<td>7</td>
</tr>
<tr>
<td>Error Codes:</td>
<td>7</td>
</tr>
<tr>
<td>Projects</td>
<td>8</td>
</tr>
<tr>
<td>1- Coin Toss</td>
<td>8</td>
</tr>
<tr>
<td>2- Dice Roll</td>
<td>9</td>
</tr>
<tr>
<td>3- LED Control</td>
<td>10</td>
</tr>
<tr>
<td>4- Light Sensing</td>
<td>11</td>
</tr>
<tr>
<td>5- Tone Control</td>
<td>13</td>
</tr>
<tr>
<td>6- RGB LED</td>
<td>14</td>
</tr>
<tr>
<td>7- Temperature Sensing</td>
<td>16</td>
</tr>
<tr>
<td>8- Servo Control</td>
<td>17</td>
</tr>
<tr>
<td>9- Motor Control</td>
<td>18</td>
</tr>
<tr>
<td>10- Accelerometer Speed Control</td>
<td>19</td>
</tr>
</tbody>
</table>
INTRODUCTION

Welcome to the wonderful world of micro:bit!
The micro:bit v2 is an easy to use yet surprisingly powerful piece of hardware that will allow you to create many impressive gadgets. The board was designed in the UK by the BBC and measures only 4 x 5 cm.
The system is an ARM-based embedded system microcontroller (ARM Cortex M0).
The module is best suited for beginners or younger users as an educational introduction to computing, coding and hardware.
*Before getting started, review the basic safety tips here: https://microbit.org/get-started/user-guide/safety/

HARDWARE

Figure 1 - BBC micro:bit v2

Figure 2 - Technical Specifications
Hardware block diagram:

Figure 3 – Hardware Block Diagram

Hardware built-in micro:bit v2:

- 16MHz 32 bit ARM Cortex M4 Nordic nRF52833
- 512 kB Flash
- 128 kB RAM
- 2.4 GHz Bluetooth BLE
- USB 2.0 OTG (On-The-Go)
- 3.3V regulator (for USB only)
- 3 axis accelerometer and magnetometer (I2C) – NXP/Freescale LSM303AGR
- 5x5 LED array
- 2 programmable tactile push buttons – 1 reset button
- Ring connectors (3 x I/O, Power, Ground)
- 19 assignable GPIO pins
- PCB mounted magnetic speaker 80dB @ 5V, 10cm (2700Hz)
- Microphone with sensitivity -38dB ±3dB @ 94dB SPL

Reference: https://tech.microbit.org/hardware/
## Hardware included in this kit:

<table>
<thead>
<tr>
<th>Description</th>
<th>Abra Part NO.</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro:bit v2</td>
<td>MICROBIT</td>
<td>All</td>
</tr>
<tr>
<td>400 Tie solderless bread-board</td>
<td>ABRA-6</td>
<td>3,4,5,6,7,8,9</td>
</tr>
<tr>
<td>Breakout Connector with Headers</td>
<td>BOB-13989</td>
<td>3,4,5,6,7,8,9</td>
</tr>
<tr>
<td>Alligator Clips</td>
<td>TL-155-1/2</td>
<td>3,4,5,6,7,8,9</td>
</tr>
<tr>
<td>Male/Male Jumper Wires</td>
<td>759-ADA</td>
<td>3,4,5,6,7,8,9</td>
</tr>
<tr>
<td>AAA Batteries</td>
<td>30-AAA-4</td>
<td>All</td>
</tr>
<tr>
<td>Battery Holder</td>
<td>BAT-H-2AAA</td>
<td>All</td>
</tr>
<tr>
<td>USB Cable</td>
<td>CAB-600-R</td>
<td>All</td>
</tr>
<tr>
<td>LED - Green</td>
<td>LED-5G</td>
<td>3</td>
</tr>
<tr>
<td>RGB LED</td>
<td>LED-5RGB-4</td>
<td>6</td>
</tr>
<tr>
<td>Pushbutton - N.O</td>
<td>PBS-155</td>
<td>3</td>
</tr>
<tr>
<td>Potentiometer - 1kΩ</td>
<td>P1K-MIN-PC</td>
<td>3</td>
</tr>
<tr>
<td>Photocell 5kΩ (Light, Detecting Resistor)</td>
<td>PHOTO-300</td>
<td>4</td>
</tr>
<tr>
<td>Piezo Buzzer</td>
<td>BUZ-120</td>
<td>5</td>
</tr>
<tr>
<td>Temperature Sensor</td>
<td>TMP36</td>
<td>7</td>
</tr>
<tr>
<td>Servo Motor</td>
<td>FS90</td>
<td>8</td>
</tr>
<tr>
<td>DC Motor</td>
<td>MOT-500</td>
<td>9,10</td>
</tr>
<tr>
<td>Transistor</td>
<td>BC337</td>
<td>9,10</td>
</tr>
<tr>
<td>Resistor - 150Ω (Brown-Green-Brown)</td>
<td>R1/4-150</td>
<td>3,4,6,9,10</td>
</tr>
<tr>
<td>10kΩ (Brown-Black-Orange)</td>
<td>R1/4-10K</td>
<td></td>
</tr>
<tr>
<td>2.2kΩ (Red-Red-Red)</td>
<td>R1/4-2.2K</td>
<td></td>
</tr>
</tbody>
</table>
CODE

We’ve seen how powerful micro:bit v2 hardware is. Now let’s learn how to unleash it with code.
Reference: http://microbit.org/code/

Pseudo Code:

Every program begins in Pseudo Code. Pseudo code is the list of instructions written in plain English that is worked out before the program is coded. This helps the programmer logically decide on code structure and operations. (See PROJECTS section for examples)

Coding Tools:

The micro:bit v2 is unique in that it offers the possibility of programming in many different languages with many different environments. Beginner programmers can get the basics using a “block editor” by simply dragging and dropping segments of already written code. Experienced users can write bare-bone scripts.
Reference: https://www.microbit.co.uk/create-code#

1 – JavaScript Block Editor: [RECOMMENDED]
https://makecode.microbit.org/#
Sample projects and User Guide can be found here:
https://makecode.microbit.org/reference
2 – Python Editor [ADVANCED]

http://python.microbit.org/editor.html#
1. Script sample code
2. File name
3. Save .py file
4. Download hex file for uploading onto board

---

**3 – Microsoft Block Editor [OLD]**

[Link](https://www.microbit.co.uk/app/#)

---

**Figure 5 - Python**

**Figure 6 - Microsoft**
Running Programs:

Now that our script has been written and tested on the simulator, we are ready to download it and install it onto the micro:bit v2.

Reference: https://microbit.org/get-started/user-guide/web-usb/

Requirements: USB cable, Windows 7 (or later), MAC OS X 10.6 (or later), Internet

Procedure:

1. Connect the micro:bit v2 into your computer USB. Your computer should recognize the device and create a MICROBIT drive
2. Compile the script (varies with each program editor). This creates a .hex file
3. Save file locally on your computer (ex: My Documents)
4. Locate the downloaded .hex file and folder and copy it (drag and drop) into the MICROBIT drive folder. This will compile the file onto the hardware. (You cannot do multiple files at a time)
5. The micro:bit v2 LED will flash for a few seconds. The software has been incorporated into the flash memory. This means that even after unplugging your device your program will remain. It will execute next time the board is powered.

**Windows**

![Windows Devices and Drives](image)

**MAC**

![MAC Devices and Drives](image)

**Figure 7 – Upload**

Error Codes:

Programming can be tricky and often doesn’t work on the first try. When you see the micro:bit v2 frowny face, something is wrong. Here are a few error codes to look out for:

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>MICROBIT_I2C_LOCKUP</td>
<td>I2C bus is not working</td>
</tr>
<tr>
<td>20</td>
<td>MICROBIT_OOM</td>
<td>No free memory available</td>
</tr>
<tr>
<td>30</td>
<td>--</td>
<td>Heap space corruption</td>
</tr>
<tr>
<td>40</td>
<td>--</td>
<td>Uninitialized object type</td>
</tr>
<tr>
<td>41</td>
<td>--</td>
<td>Out of bounds</td>
</tr>
<tr>
<td>42</td>
<td>--</td>
<td>Cannot execute script</td>
</tr>
</tbody>
</table>

Reference: https://makecode.microbit.org/device/error-codes
Projects

*Other PROJECTS ideas can be found at: https://www.microbit.co.uk/blocks/lessons

1- Coin Toss

“This project helps you decide with a simple coin toss”

LEVEL: Beginner

MATERIALS: microbit, Battery, USB cable

PSEUDO CODE:

1. When the device is shook (or button pressed), do coin toss
2. Coin toss can have 1 of 2 random results
3. Set heads to 0 and tails to 1
4. If the randomly generated toss is 0, print heads (H) on LED screen
5. Else, the toss must be 1, print tails (T) on LED screen

SAMPLE CODE: ALTERNATE CODE:
2- **Dice Roll**

“This project creates a useful gadget for rolling dice and playing games”

**LEVEL:** Beginner

**MATERIALS:** microbit, Battery, USB cable

**PSEUDO CODE:**

1. When the device is **shook** (or button is pressed) roll dice,
2. Coin toss can have 1 of 6 random outcomes
3. **Set** result to random outcome
4. If result is 1, **print** 1 on LED screen
5. Else if result is 2, **print** 2 on LED screen
6. Repeat for all possible cases

**SAMPLE CODE:**

![Figure 10 - Dice Roll](image)
3- **LED Control**

“This project shows you how to turn an LED ON and OFF with pushbuttons and control its brightness with a potentiometer.”

**LEVEL:** Intermediate

**MATERIALS:** LED, Resistor(150), Potentiometer, Pushbutton Jumper wire, Breadboard, microbit Breakout, Battery pack, USB cable

**HARDWARE:**

![micro:bit breakout](image)

*LED:*

Longer leg is +

Shorter leg is −

(connect to black wire)

**Figure 11 - LED Circuit**

**PSEUDO CODE:**

1. Assign **button** to a **pin** (P0)
2. Check and update **light state** when button pin is **pressed**
3. If light state is **OFF** (0), change it to **ON** (1)
4. Else, change light state to **OFF** (0)
5. Check light state **forever**
6. If light state has been set to **ON**(1), control LED brightness
7. LED brightness is controlled by reading the **analog** value of **potentiometer pin** (P1) and **analog writing** it on to **LED pin** (P2)
8. Else, light state is **OFF**, LED must be turned off (**digital write** P2 to 0)
SAMPLE CODE:

![LED Control Code]

4- **Light Sensing**

“This project demonstrates how to measure light intensity with a sensor and use it for control (TIP: Try controlling an LED brightness with this technique from Project 3)”

**LEVEL:** Intermediate

**MATERIALS:** Photocell, Resistor (10k), Microbit breakout, Breadboard, Battery pack, USB cable

**HARDWARE:**

![LDR Circuit]
**PSEUDO CODE:**

1. Check sensor reading **forever**
2. Set sensor value to variable **Light**
3. **Light** is read from analog pin **P0**
4. **Light** can have values 0 to 1024 (analog pin)
5. If **Light** is greater than **512**, show SUN on **LED**
6. Else, show MOON on **LED**

**SAMPLE CODE:**

![Diagram](Image)
5- Tone Control

“This project plays with sound and varies a tone using a piezo buzzer”

LEVEL: Intermediate

MATERIALS: Piezo buzzer, Microbit breakout, Breadboard, Battery pack, USB cable

HARDWARE:

![Figure 15 - Piezo Buzzer Circuit](image)

PSEUDO CODE:

1. When Button A is pressed,
2. Play tone (note) for (beat)

SAMPLE CODE:

![Figure 16 - Tone Control](image)
6- **RGB LED**

“This project lets you manipulate light, change colors and patterns”

**LEVEL**: Intermediate

**MATERIALS**: RGB LED, 150Ω resistor, Microbit breakout, Breadboard, Battery pack, USB cable

**HARDWARE:**

![Micro:bit breakout circuit](image-url)

**PSEUDO CODE:**

1. Keep LED running forever
2. Assign analog write variables Red to P0
3. Assign analog write variables Green to P1
4. Assign analog write variables Blue to P2
5. If button A, increase the level of Green by 10
6. If Green surpasses 1020, reset to 0
7. If button B, increase the level of Red by 10
8. If Red surpasses 1020, reset to 0
9. If button A+B increase the level of Blue by 10
10. If Blue surpasses 1020, reset to 0
SAMPLE CODE:

```plaintext
forever
  analog write pin P0 to Red
  analog write pin P1 to Green
  analog write pin P2 to Blue
  if button A is pressed and Green < 1020 then set Green to Green + 10
  else if button A is pressed and Green = 1020 then set Green to 0
  if button B is pressed and Red < 1020 then set Red to Red + 10
  else if button B is pressed and Red = 1020 then set Red to 0
  if button A+B is pressed and Blue < 1020 then set Blue to Blue + 10
  else if button A+B is pressed and Blue = 1020 then set Blue to 0
```

*Figure 18 - LED RGB Control*
7- **Temperature Sensing**  
“This project creates a very accurate thermometer”

**LEVEL:** Intermediate  
**MATERIALS:** TMP36, Microbit breakout, Breadboard, Battery pack, USB cable

**HARDWARE:**

![Micro:bit breakout](image)

**Figure 19 - TMP36 Temperature Circuit**

**PSUEDO CODE:**

1- Check temperature sensor *forever*  
2- Create variable to hold sensor value from *analog read pin P0*  
3- Create variable to map temperature from lowest signed sensor input(124) and highest signed sensor input (496) to lowest temperature (-20) and highest temperature (100)  
4- Show number corresponding to temperature

**SAMPLE CODE:**

![Sample code](image)

**Figure 20 - Temperature Control**
8- Servo Control
“This project gets the gears rotating by controlling a servo motor”

LEVEL: Advanced

MATERIALS: Servo motor, Microbit breakout, Breadboard, Battery pack, USB cable

HARDWARE:
**Note: In order to provide enough power to the servo motor, a separate breakout board/power supply can be used while the Microbit board is used to provide the control signal.**

![Servo Motor Circuit](image1)

**Figure 21 - Servo Motor Circuit**

PSEUDO CODE:
1- When Button A is pressed
2- Command servo on P0 to 180 degrees
3- Pause 1 second
4- When Button B is pressed
5- Command servo back to 0 degrees
6- Pause 1 second

SAMPLE CODE:

![Servo Control Code](image2)

**Figure 22 - Servo Control**
9- **Motor Control**

“This project gets the wheels turning by controlling a motor”

**LEVEL:** Advanced

**MATERIALS:** DC motor, BC337 Transistor, Microbit breakout, Breadboard, Battery pack, USB cable

**HARDWARE:**

![Motor Circuit](image1.png)

**PSEUDO CODE:**

1- On start, create variable called **duty** and set it to **0**
2- Run motor **forever**
3- Ramp up motor speed by **analog write** the value of **duty** to **P0** until it reaches max value of **1023**
4- Ramp down motor speed until it reaches min value of **0**

**SAMPLE CODE:**

![Motor Control](image2.png)
10- Accelerometer Speed Control

“This project will get you accelerating motors with the built-in accelerometer”

LEVEL: Advanced

MATERIALS: DC motor, BC337 Transistor, Microbit breakout, Breadboard, Battery pack, USB cable

HARDWARE:

![Accelerometer Motor Circuit](image)

PSEUDO CODE:

1- Run motor **forever**
2- **Analog write** the absolute value (positive values only) of the built-in accelerometer to **P0**

SAMPLE CODE:

![Accelerometer Motor Control](image)