Code Documentation:

Project Title: EcoCrank - Exercise Hand Crank Power Bank

Project Start Date	11/16/24
Project End Date	01/18/25
Project Manager	Denille Rylie C. Galas
Team Members	Josiah R. Bucas, Jade D. Penales, Anthony Lee Aleister Ortiz

Below is the code inputted in the Arduino Uno R3, along with the explanation of each command and their functions. The Code is backed up Via GitHub which can be located at the bottom.

---

1. Including Required Libraries

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

- <Wire.h>: Enables communication via the I2C protocol.
- <LiquidCrystal_I2C.h>: Simplifies working with I2C-enabled LCDs.

---

2. Setting Up the LCD

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

- Creates an LCD object.
- 0x27: The I2C address of the LCD (specific to your module).
- 16, 2: The LCD has 16 columns and 2 rows.

---

3. Pin and Voltage Configuration

```
const int batteryPin = A0;
const float referenceVoltage = 5.0;
const float voltageDividerRatio = 2.0;
float batteryVoltage = 0.0;
int batteryPercentage = 0;
```

- batteryPin: The analog pin (A0) connected to the voltage divider.
- referenceVoltage: Reference voltage of the Arduino (commonly 5V for Arduino Uno).
- voltageDividerRatio: Accounts for the resistor divider ratio (e.g., 10k/10k reduces battery voltage by half).
- batteryVoltage: Stores the actual voltage calculated from the analog pin.
- batteryPercentage: Represents the battery level as a percentage.

4. Timing Variables

```
unsigned long previousMillis = 0;
const unsigned long interval = 6000;
bool showVoltage = true;
```

- previousMillis: Tracks the last time a display update occurred.
- interval: Sets the duration (6000 ms = 6 seconds) between display toggles.
- showVoltage: Toggles between displaying voltage and a progress bar.

---

5. setup() Function

```
lcd.init();
lcd.backlight();
```

- lcd.init(): Initializes the LCD.
- lcd.backlight(): Turns on the LCD backlight for visibility.

```
lcd.setCursor(0, 0);
lcd.print("Battery Voltage");
lcd.setCursor(0, 1);
lcd.print("Monitor System");
delay(5000);
lcd.clear();
```

- Displays an initialization message ("Battery Voltage Monitor System") for 5 seconds.
- lcd.clear(): Clears the LCD after the initialization.

---

6. loop() Function

This is the main program logic, repeated continuously.

I. Read Battery Voltage

```
int rawValue = analogRead(batteryPin);
batteryVoltage = (rawValue / 1023.0) * referenceVoltage * voltageDividerRatio;
```

- analogRead(): Reads the analog voltage at A0 (returns a value between 0 and 1023).
- Converts the raw reading to actual voltage using:
  - rawValue / 1023.0: Normalizes to a fraction of referenceVoltage.
  - * referenceVoltage: Converts to the actual input voltage.
  - * voltageDividerRatio: Compensates for the voltage divider.

II. Calculate Battery Percentage

```
batteryPercentage = map(batteryVoltage * 100, 300, 420, 0, 100);
if (batteryPercentage > 100) batteryPercentage = 100;
if (batteryPercentage < 0) batteryPercentage = 0;
```

- Assumes 3.0V is 0% and 4.2V is 100%.
- map(): Scales voltage to a percentage:
  - Converts batteryVoltage (e.g., 3.7V = 370) into a percentage (scaled between 0 and 100).
- Limits batteryPercentage to valid bounds (0–100%).

III. Toggle Display

```
unsigned long currentMillis = millis();
if (currentMillis - previousMillis >= interval) {
  previousMillis = currentMillis;
  showVoltage = !showVoltage;
}
```

- millis(): Returns the elapsed time since the Arduino started.
- Checks if interval (6 seconds) has passed since the last toggle.
- showVoltage = !showVoltage: Alternates the display mode.

IV. Display Voltage or Progress Bar

```
if (showVoltage) {
  displayVoltage();
  delay(2000);
} else {
  displayProgressBar();
  delay(2000);
}
```

- Depending on showVoltage, calls displayVoltage() or displayProgressBar() and delays 2 seconds before updating.

---

7. displayVoltage()

```
void displayVoltage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Voltage:");
  lcd.setCursor(0, 1);
  lcd.print(batteryVoltage, 2);
  lcd.print("V");
}
```

- Clears the LCD.
- Displays "Voltage:" on the top row.
- Displays the calculated batteryVoltage (2 decimal places) followed by "V".

---

8. displayProgressBar()

```
void displayProgressBar() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Battery Level:");
```

- Clears the LCD and displays "Battery Level:" on the first row.

a. Draw Progress Bar

```
int numBars = map(batteryPercentage, 0, 100, 0, 10);
lcd.setCursor(0, 1);
for (int i = 0; i < numBars; i++) {
  lcd.print((char)255);
}
for (int i = numBars; i < 10; i++) {
  lcd.print(" ");
```

}

- numBars: Maps batteryPercentage to a value between 0 and 10 (corresponds to 10 segments).
- Uses the ASCII character 255 (a solid block) to represent filled segments.
- Pads the remaining 10 segments with spaces.

b. Display Percentage

```
lcd.setCursor(11, 1);
lcd.print(batteryPercentage);
lcd.print("%");
```

- Displays the batteryPercentage at the right of the progress bar.

ARDUINO CODE FOR VOLTAGE MONITORING

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h> // Library for I2C LCD

LiquidCrystal_I2C lcd(0x27, 16, 2); // Set the LCD address to 0x27 for a 16x2 display

const int batteryPin = A0;          // Pin connected to the voltage divider
const float referenceVoltage = 5.0; // Arduino reference voltage (5V for Uno)
const float voltageDividerRatio = 2.0; // Ratio of the voltage divider (e.g., 10k/10k)
float batteryVoltage = 0.0;         // Variable to store battery voltage
int batteryPercentage = 0;          // Variable to store battery percentage

unsigned long previousMillis = 0;   // For timing the alternation
const unsigned long interval = 6000; // Switch every 6 seconds (8000 ms)
bool showVoltage = true;            // Toggle between voltage and progress bar

void setup() {
  lcd.init();       // Initialize the LCD
  lcd.backlight();  // Turn on the backlight

  // Display "Battery Voltage Monitor System" for initialization
  lcd.setCursor(0, 0);
  lcd.print("Battery Voltage");
  lcd.setCursor(0, 1);
  lcd.print("Monitor System");
  delay(5000);  // Show initialization message for 2 seconds

  // Clear the display after initialization
  lcd.clear();
```

```
}

void loop() {
  // Read battery voltage
  int rawValue = analogRead(batteryPin);
  batteryVoltage = (rawValue / 1023.0) * referenceVoltage *
voltageDividerRatio;

  // Calculate battery percentage (assuming 3.0V = 0% and 4.2V = 100%)
  batteryPercentage = map(batteryVoltage * 100, 300, 420, 0, 100);
  if (batteryPercentage > 100) batteryPercentage = 100;
  if (batteryPercentage < 0) batteryPercentage = 0;

  // Alternate display every interval
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    showVoltage = !showVoltage; // Toggle between voltage and progress bar
  }

  // Display voltage or progress bar
  if (showVoltage) {
    displayVoltage();
    delay(2000); // Delay for 2 seconds before updating again
  } else {
    displayProgressBar();
    delay(2000); // Delay for 2 seconds before updating again
  }
}

void displayVoltage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Voltage:");
  lcd.setCursor(0, 1);
  lcd.print(batteryVoltage, 2); // Display voltage with 2 decimal places
  lcd.print("V");
}
```

```
void displayProgressBar() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Battery Level:");

  // Create a progress bar
  int numBars = map(batteryPercentage, 0, 100, 0, 10); // Map percentage to 10
blocks / 10 spaces
  lcd.setCursor(0, 1);
  for (int i = 0; i < numBars; i++) {
    lcd.print((char)255);
  }
  for (int i = numBars; i < 10; i++) {
    lcd.print(" ");
  }

  lcd.setCursor(11, 1);
  lcd.print(batteryPercentage);
  lcd.print("%");
}
```

GITHUB LINK:
https://github.com/eliyrden/BatteryMonitorLCD.git