PES UNIVERSITY

END SEMESTER ASSESSMENT (ESA)
B.TECH. (CSE)
IV SEMESTER

UE18CS256 – MICROPROCESSOR AND COMPUTER ARCHITECTURE LABORATORY

PROJECT REPORT

ON

Arduino NiMh Controlled Battery Charger with text display

SUBMITTED BY

NAME	SRN
1) NIKHIL M A	PES2UG19CS257
2) NIKHIL VR	PES2UG19CS258
3) NISHANTH M	PES2UG19CS264
4) NISHANTH JC	PES2UG19CS263

JANUARY – MAY 2021

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ELECTRONIC CITY CAMPUS,

BENGALURU – 560100, KARNATAKA, INDIA

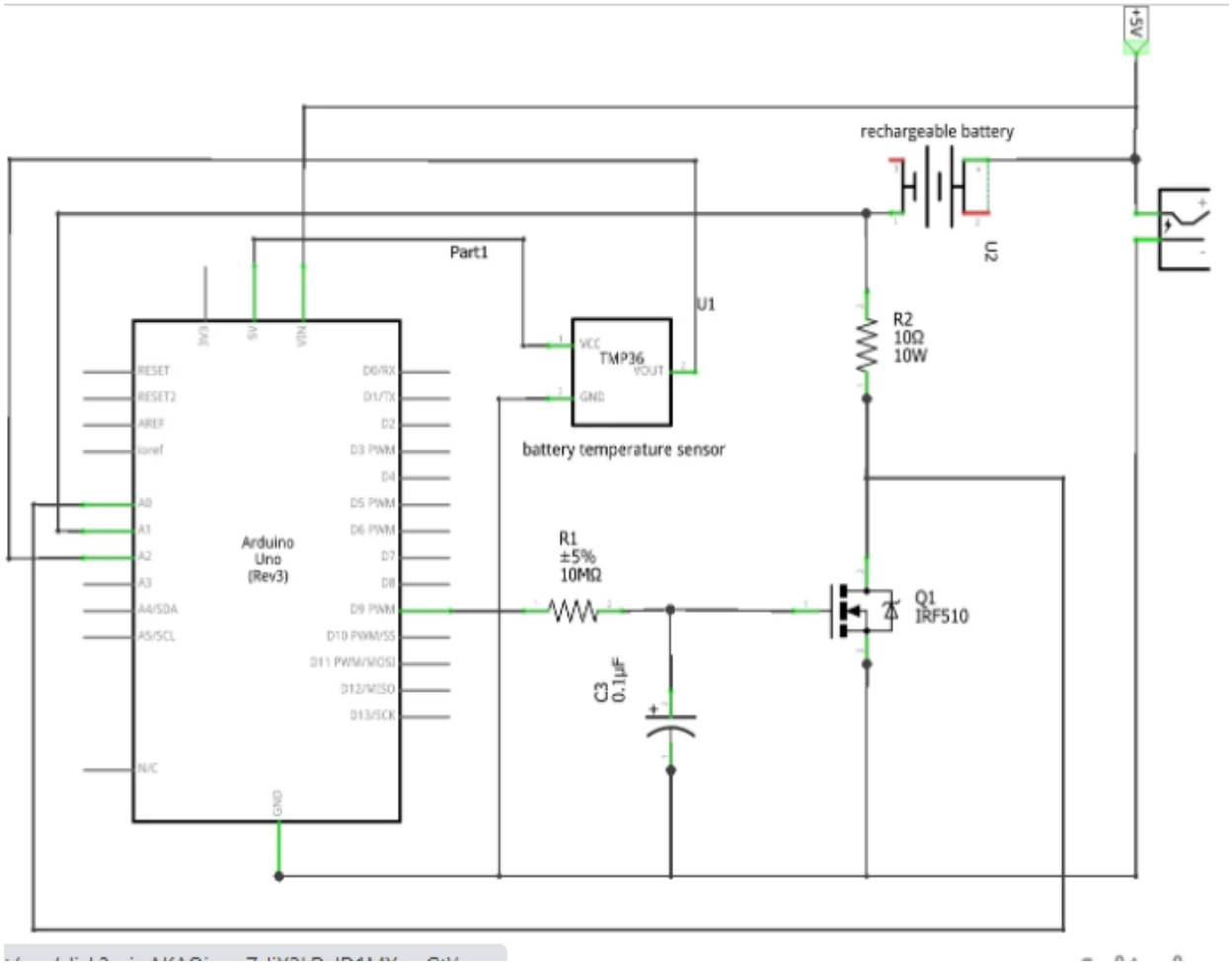
TABLE OF CONTENTS		
Sl.No	TOPIC	PAGE No
1.	ABSTRACT OF THE PROJECT	1
2.	CIRCUIT DIAGRAM	2
3.	ARDUINO CODE	5
4.	SCREEN SHOTS OF THE OUTPUT	11
5.	REFERENCES	14

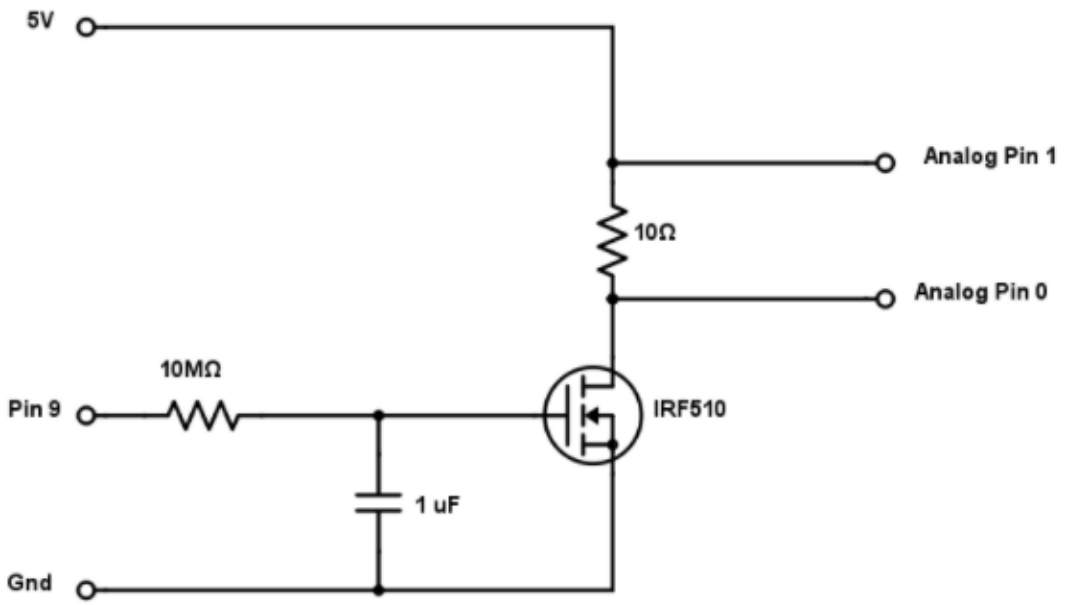
ABSTRACT OF THE PROJECT:

The circuit design for this charger is a basic Arduino controlled power supply. The circuit is powered by a 5-volt regulated voltage source such as an AC adapter or an ATX computer power supply. Most USB ports would not appropriate for this project because of the current limitations. The 5V source charges the battery through a 10 ohm power resistor and a power MOSFET. The MOSFET sets how much current is allowed to flow into the battery. The resistor is included as an easy way to monitor the current. This is done by connecting each terminal to analog input pins on the Arduino and measuring the voltage on each side. The MOSFET is controlled by a PWM Output pin on the Arduino. The pulses of the pulse width modulation signal are smoothed out into a steady voltage signal by a 1M resistor and a 1 μF capacitor. This circuit allows the Arduino to monitor and control the current flowing into the battery.

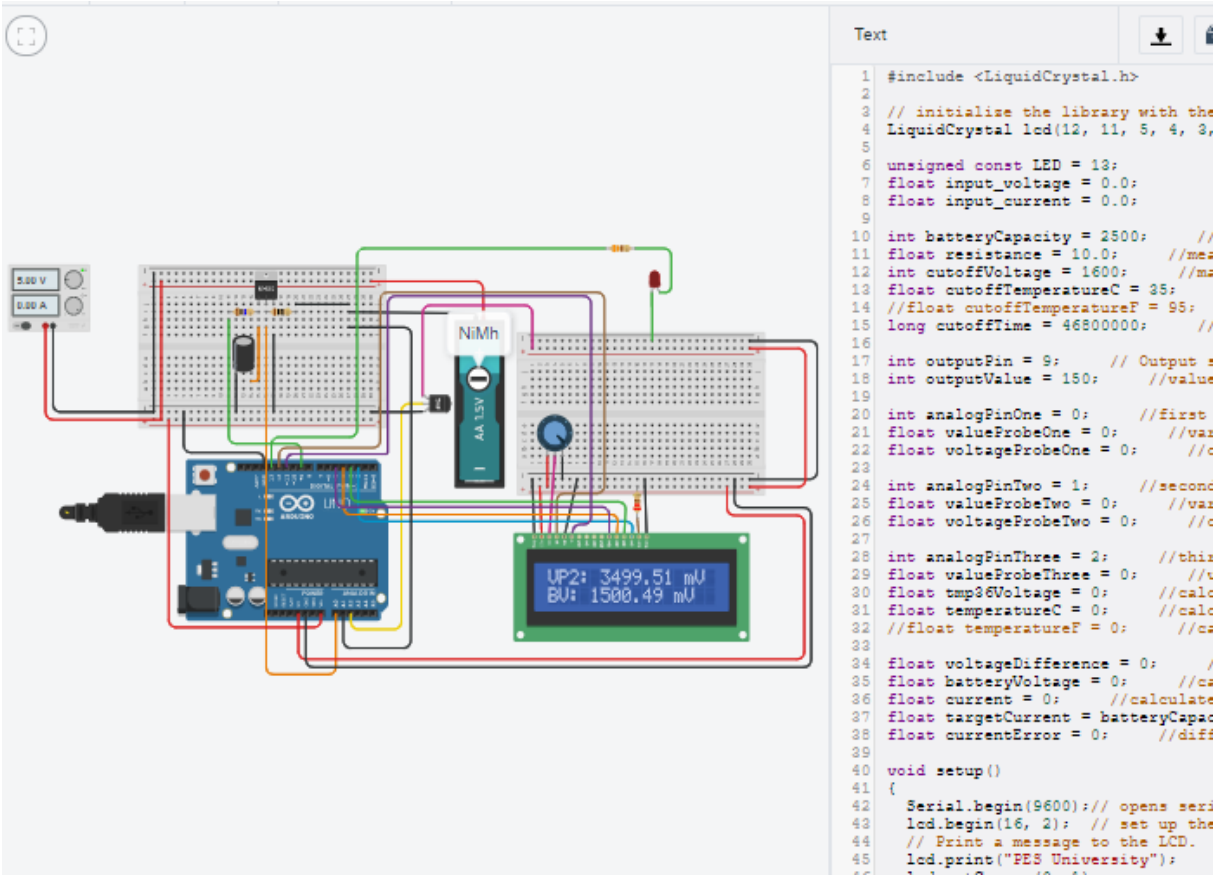
As an extra precaution, we included a TMP36 temperature sensor to monitor the temperature of a battery. This sensor outputs a signal voltage that directly corresponds to the temperature. So it doesn't require calibration or balancing like a thermistor does. The sensor is placed as close to the battery as possible to get the correct temperature reading. The pins of the sensor are then connected 5V, GND, and an analog input pin on the Arduino.

CIRCUIT DIAGRAM:





ARDUINO CODE:



```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

unsigned const LED = 13;
float input_voltage = 0.0;
float input_current = 0.0;

int batteryCapacity = 2500;       //capacity rating of battery in mAh
float resistance = 10.0;        //measured resistance of the power resistor
int cutoffVoltage = 1600;       //maximum battery voltage (in mV) that should not be exceeded
float cutoffTemperatureC = 35;          //maximum battery temperature that should not be exceeded (in degrees C)
//float cutoffTemperatureF = 95;          //maximum battery temperature that should not be exceeded (in degrees F)
long cutoffTime = 46800000;         //maximum charge time of 13 hours that should not be exceeded

int outputPin = 9;        // Output signal wire connected to digital pin 9
int outputValue = 150;       //value of PWM output signal

int analogPinOne = 0;       //first voltage probe connected to analog pin 1
float valueProbeOne = 0;       //variable to store the value of analogPinOne
float voltageProbeOne = 0;       //calculated voltage at analogPinOne

int analogPinTwo = 1;       //second voltage probe connected to analog pin 2
float valueProbeTwo = 0;       //variable to store the value of analogPinTwo
float voltageProbeTwo = 0;       //calculated voltage at analogPinTwo

int analogPinThree = 2;       //third voltage probe connected to analog pin 2
float valueProbeThree = 0;       //variable to store the value of analogPinThree
float tmp36Voltage = 0;       //calculated voltage at analogPinThree
float temperatureC = 0;       //calculated temperature of probe in degrees C
//float temperatureF = 0;       //calculated temperature of probe in degrees F

float voltageDifference = 0;          //difference in voltage between analogPinOne and analogPinTwo
float batteryVoltage = 0;       //calculated voltage of battery
float current = 0;       //calculated current through the load (in mA)
float targetCurrent = batteryCapacity / 10;       //target output current (in mA) set at C/10 or 1/10 of the battery capacity per hour
float currentError = 0;       //difference between target current and actual current (in mA)


void setup()
{
  Serial.begin(9600);// opens serial port, sets data rate to 9600 bps
  lcd.begin(16, 2);  // set up the LCD's number of columns and rows
  // Print a message to the LCD.
  lcd.print("PES University");
  lcd.setCursor(0 ,1);
  lcd.print("MPCA LAB PROJECT");
  delay(5000);
  lcd.clear();
```

```
  lcd.print("Nikhil M A");
  lcd.setCursor(0 ,1);
  lcd.print("Nikhil V R");
  delay(5000);
  lcd.clear();
  lcd.print("Nishanth M");
  lcd.setCursor(0 ,1);
  lcd.print("Nishanth J C");
  delay(5000);
  lcd.clear();
  lcd.print("Battery Charger");
  delay(5000);
  lcd.clear();
  Serial.begin(9600);      //   setup serial
  pinMode(LED, OUTPUT);
  pinMode(outputPin, OUTPUT);        // sets the pin as output



oid loop()


  analogWrite(outputPin, outputValue);   //Write output value to output pin

  Serial.print("Output: ");        //display output values for monitoring with a computer
  Serial.println(outputValue);

  valueProbeOne = analogRead(analogPinOne);       // read the input value at probe one
  voltageProbeOne = (valueProbeOne*5000)/1023;           //calculate voltage at probe one in
nilliVolts
  Serial.print("Voltage Probe One (mV): ");        //display voltage at probe one
  Serial.println(voltageProbeOne);

  lcd.print("Output: ");
  lcd.print(outputValue);
  lcd.setCursor(0, 1);
  lcd.print("VP1: ");
  lcd.print(voltageProbeOne);
  lcd.print(" mV");
  delay(5000);
  lcd.clear();

  valueProbeTwo = analogRead(analogPinTwo);       // read the input value at probe two
  voltageProbeTwo = (valueProbeTwo*5000)/1023;           //calculate voltage at probe two in
nilliVolts
  Serial.print("Voltage Probe Two (mV): ");        //display voltage at probe two
  Serial.println(voltageProbeTwo);

  batteryVoltage = 5000 - voltageProbeTwo;        //calculate battery voltage
  Serial.print("Battery Voltage (mV): ");        //display battery voltage
  Serial.println(batteryVoltage);

  lcd.print("VP2: ");
  lcd.print(voltageProbeTwo);
  lcd.print(" mV");
```

```
  lcd.setCursor(0, 1);
  lcd.print("BV: ");
  lcd.print(batteryVoltage);
  lcd.print(" mV");
  delay(5000);
  lcd.clear();

  current = (voltageProbeTwo - voltageProbeOne) / resistance;        //calculate charge current
  Serial.print("Target Current (mA): ");        //display target current
  Serial.println(targetCurrent);
  Serial.print("Battery Current (mA): ");        //display actual current
  Serial.println(current);

  lcd.print("TC: ");
  lcd.print(targetCurrent);
  lcd.print(" mA");
  lcd.setCursor(0, 1);
  lcd.print("BC: ");
  lcd.print(current);
  lcd.print(" mA");
  delay(5000);
  lcd.clear();

  currentError = targetCurrent - current;          //difference between target current and measured
current
  Serial.print("Current Error    (mA): ");        //display current error
  Serial.println(currentError);

  valueProbeThree = analogRead(analogPinThree);       // read the input value at probe three
  tmp36Voltage = valueProbeThree * 5.0;         // converting that reading to voltage
  tmp36Voltage /= 1024.0;

  temperatureC = (tmp36Voltage - 0.5) * 100 ;          //converting from 10 mv per degree wit 500
mV offset to degrees ((voltage - 500mV) times 100)
  Serial.print("Temperature (degrees C) ");         //display the temperature in degrees C
  Serial.println(temperatureC);

  lcd.print("CE: ");
  lcd.print(currentError);
  lcd.print(" mA");
  lcd.setCursor(0, 1);
  lcd.print("T: ");
  lcd.print(temperatureC);
  lcd.print(" deg C");
  delay(5000);
  lcd.clear();

 /*
  temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;         //convert to Fahrenheit
  Serial.print("Temperature (degrees F) ");
  Serial.println(temperatureF);
 */

  Serial.println();         //extra spaces to make debugging data easier to read
  Serial.println();
```

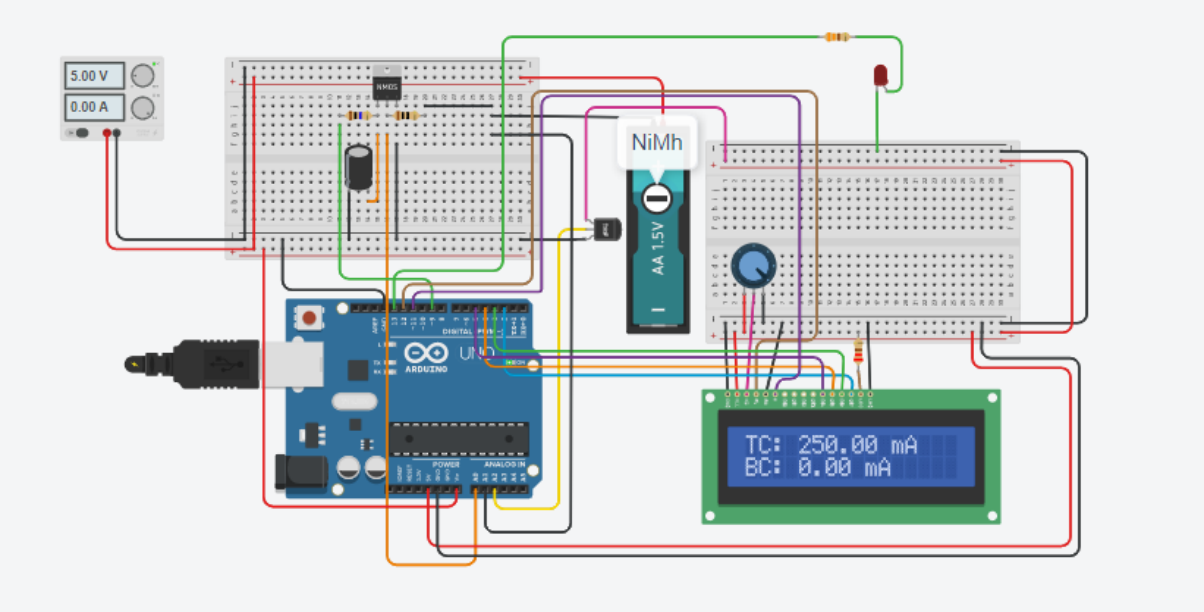
```
  if(abs(currentError) > 10)        //if output error is large enough, adjust output
   {
    outputValue = outputValue + currentError / 10;

    if(outputValue < 1)       //output can never go below 0
     {
      outputValue = 0;
     }

    if(outputValue > 254)        //output can never go above 255
     {
      outputValue = 255;
     }

    analogWrite(outputPin, outputValue);        //write the new output value
   }


  if(temperatureC  >  cutoffTemperatureC)             //stop  charging  if  the  battery  temperature
exceeds the safety threshold
   {
    outputValue = 0;
    Serial.print("Max Temperature Exceeded");
    int i=1;
    while(i>0)
    {
          lcd.print("MaxTempEx");
          lcd.setCursor(0, 1);
          lcd.print("Restart    Program");
          digitalWrite(13, HIGH);
          delay(500);
          digitalWrite(13, LOW);
          delay(500);
          i=i+1;
    }
    delay(5000);
    lcd.clear();
    exit(-1);
   }

  /*
  if(temperatureF > cutoffTemperatureF)        //stop charging if the battery temperature exceeds
the safety threshold
   {
    outputValue = 0;
   }
   */

    if(batteryVoltage > cutoffVoltage)          //stop  charging  if  the  battery  voltage  exceeds  the
safety threshold
   {
    outputValue = 0;
    Serial.print("Max Voltage Exceeded");
    int i=1;
```

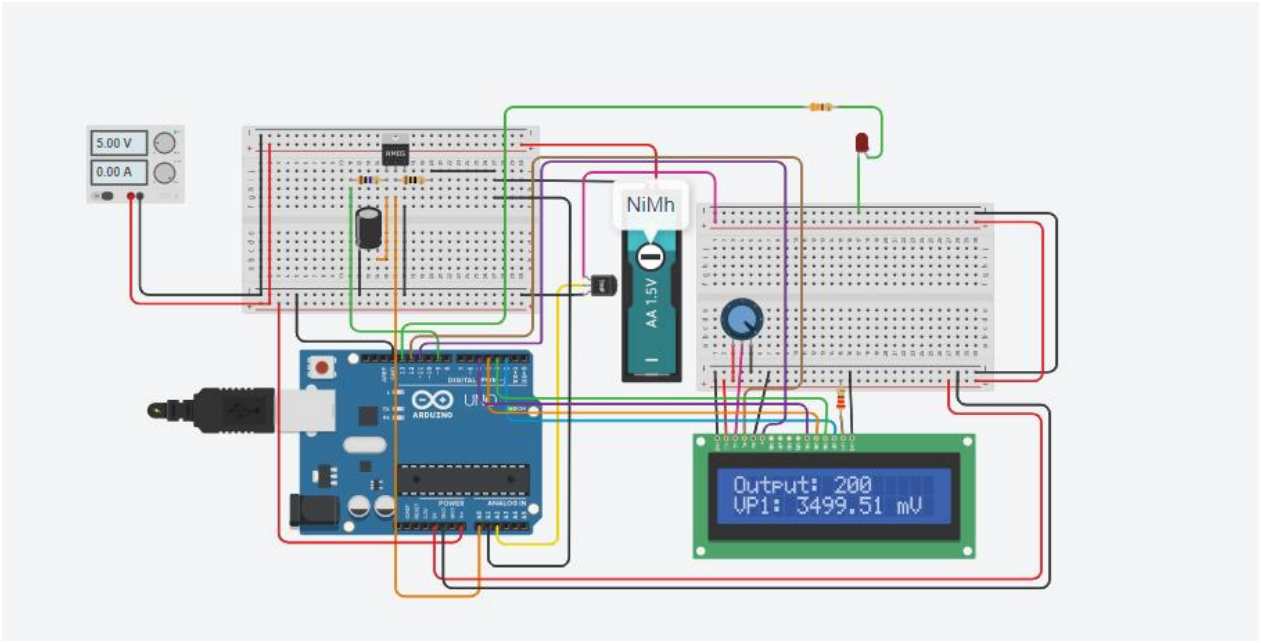
```
 while(i>0)
 {
       lcd.print("MaxVoltEx");
       lcd.setCursor(0, 1);
       lcd.print("Restart    Program");
       digitalWrite(13, HIGH);
       delay(500);
       digitalWrite(13, LOW);
       delay(500);
       i=i+1;
 }
 delay(5000);
 lcd.clear();
   exit(-1);
}

if(millis() > cutoffTime)        //stop charging if the charge time threshold
{
 outputValue = 0;
 Serial.print("Max Charge Time Exceeded");
 int i=1;
 while(i>0)
 {
       lcd.print("MaxChEx");
       lcd.setCursor(0, 1);
       lcd.print("Restart    Program");
       digitalWrite(13, HIGH);
       delay(500);
       digitalWrite(13, LOW);
       delay(500);
       i=i+1;
 }
 delay(5000);
 lcd.clear();
   exit(0);
}

lcd.setCursor(0, 1);
lcd.print("          ...Loading");
delay(5000);//delay 5 seconds before next iteration
lcd.clear();
```

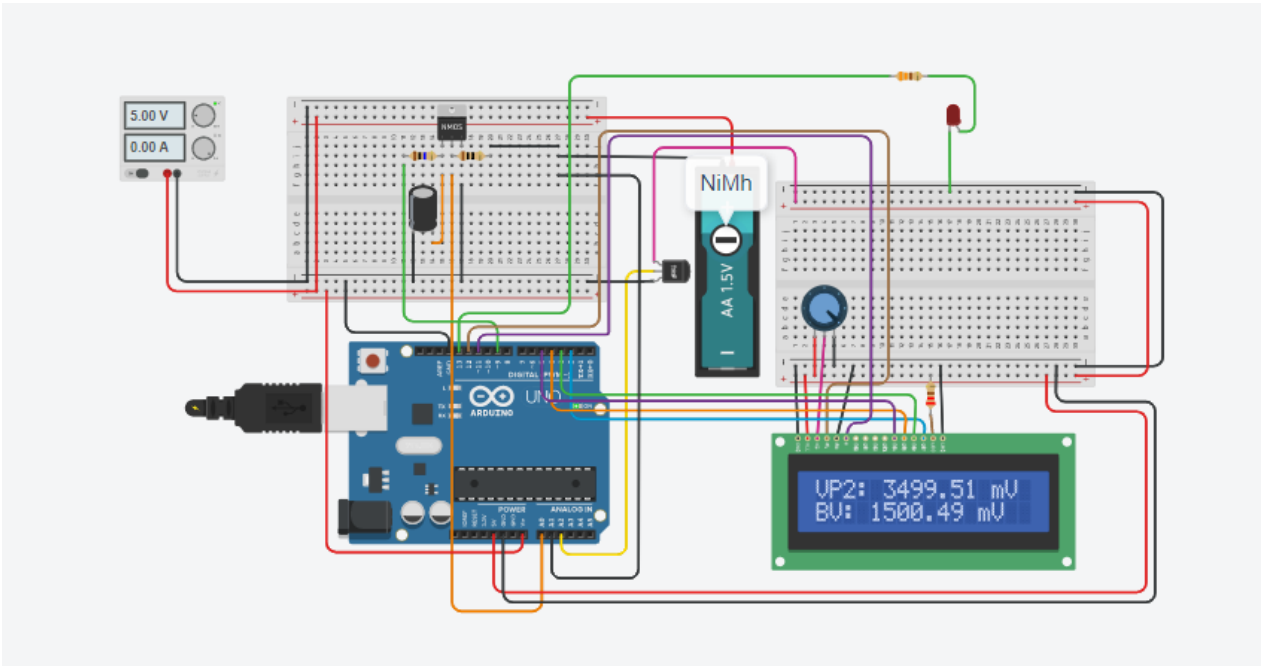
SCREEN SHOTS OF THE OUTPUT:

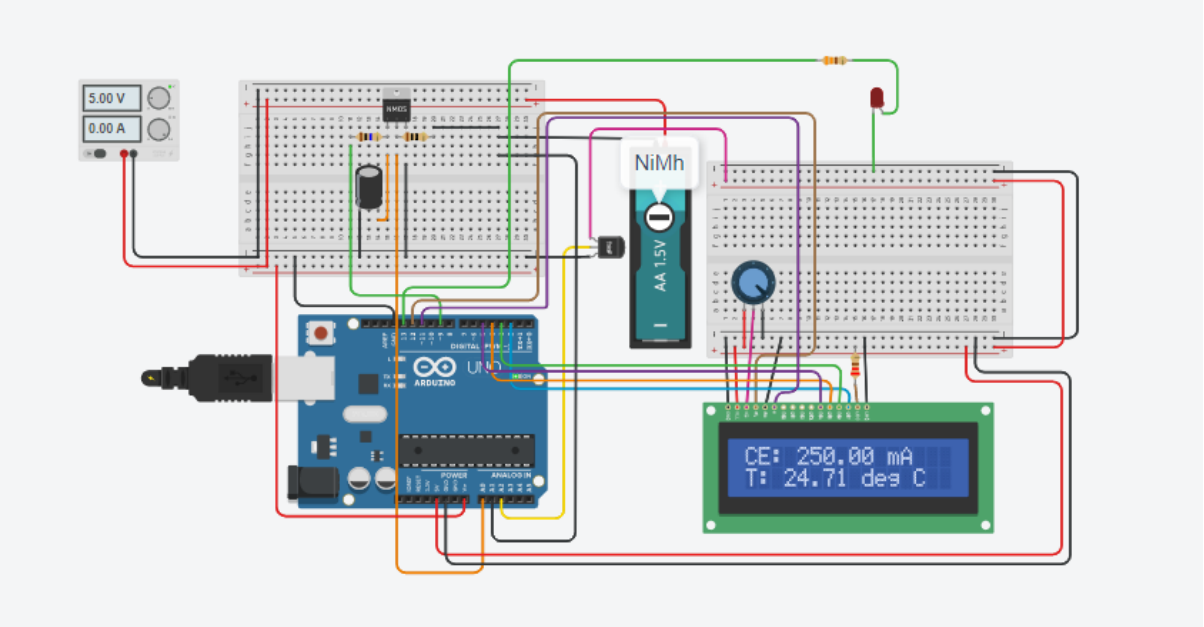
Battery characteristics:

1)Output
2)VP 1 – Voltage Probe 1
3)VP2 – Voltage Probe 2
4)BV – Battery Voltage
5)TC – Target Current
6)BC – Battery Current
7)CE – Current Error
8) T – Temperature

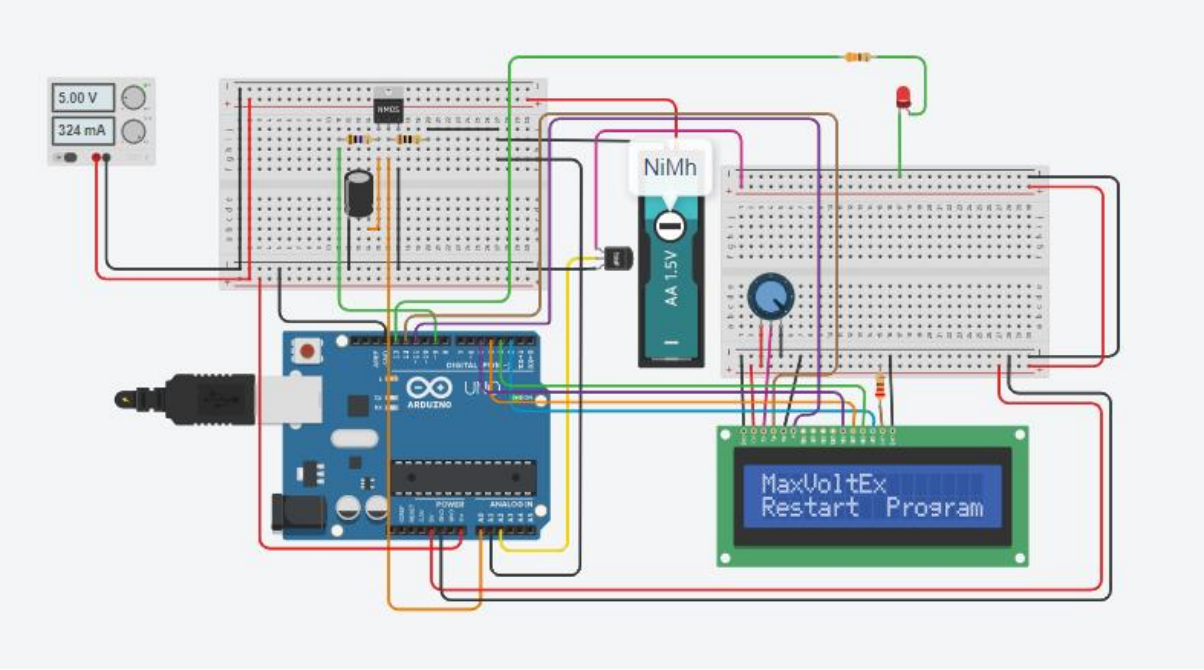



5.00 V
0.00 A
NiMh
AA 1.5V
Output: 200
VP1: 3499.51 mV


5.00 V
0.00 A
NiMh
AA 1.5V
VP2: 3499.51 mV
BV: 1500.49 mV



LED is blinking since maximum voltage is exceeded. The LCD screen displays the message prompting user to restart the program.



REFERENCES

Our Project is made public on tinkercad and this is the link where one can access it.

Link->
https://www.tinkercad.com/things/imSwDhu70jK-mpca-project-nimh-controlled-battery-charger-with-text-display

If link does not work, under public circuits, enter the keywords-> MPCA Nimh and the first project is our project