

#include <LiquidCrystal.h>

#define analogPin 0 // analog pin for measuring capacitor voltage

#define chargePin 7 // pin to charge the capacitor - connected to one end of the charging resistor

#define dischargePin 5 // pin to discharge the capacitor

#define resistorValue 1000.0F // change this to whatever resistor value you are using

// F formatter tells compiler it's a floating point value

LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

unsigned long startTime;

unsigned long elapsedTime;

float microFarads; // floating point variable to preserve precision, make calculations

float nanoFarads;

void setup()

{

pinMode(chargePin, OUTPUT); // set chargePin to output

digitalWrite(chargePin, LOW);

Serial.begin(9600); // initialize serial transmission for debugging

lcd.begin(16, 2);

lcd.print("Go start");

}

void loop()

{

digitalWrite(chargePin, HIGH); // set chargePin HIGH and capacitor charging

startTime = millis();

while (analogRead(analogPin) < 648)

{ // 647 is 63.2% of 1023, which corresponds to full-scale voltage}

elapsedTime = millis() - startTime;

microFarads = ((float)elapsedTime / resistorValue) \* 1000;

lcd.clear();

Serial.print(elapsedTime); // print the value to serial port

lcd.print(elapsedTime);

Serial.print(" mS"); // print units and carriage return

lcd.print(" mS");

lcd.setCursor(0, 1);

if (microFarads > 1)

{

Serial.print((long)microFarads); // print the value to serial port

lcd.print((long)microFarads);

Serial.println(" microFarads"); // print units and carriage return

lcd.print(" microFarads");

}

else

{ nanoFarads = microFarads \* 1000.0; // multiply by 1000 to convert to nanoFarads (10^-9 Farads)

Serial.print((long)nanoFarads); // print the value to serial port

Serial.println(" nanoFarads"); // print units and carriage return

}

digitalWrite(chargePin, LOW); // set charge pin to LOW

pinMode(dischargePin, OUTPUT); // set discharge pin to output

digitalWrite(dischargePin, LOW); // set discharge pin LOW

while (analogRead(analogPin) > 0){}

pinMode(dischargePin, INPUT); // set discharge pin back to input}