EEPROM Overview

Electrically erasable programmable read-only memory (EEPROM). This is a read-mostly memory that can be written into at any time without erasing prior contents. EEPROM can update byte level, that mean it can read/write byte or bytes. The write operation takes considerably longer than the read operation, on the order of several hundred microseconds per byte. The EEPROM combines the advantage of nonvolatility with the flexibility of being updatable in place, using ordinary bus control, address, and data lines. EEPROM is more expensive than EPROM and also is less dense, supporting fewer bits per chip.

The microcontroller on the Arduino and Genuino AVR based board has EEPROM: memory whose values are kept when the board is turned off (like a tiny hard drive). This library enables you to read and write those bytes.

The supported micro-controllers on the various Arduino and Genuino boards have different amounts of EEPROM: 1024 bytes on the ATmega328P, 512 bytes on the ATmega168 and ATmega8, 4 KB (4096 bytes) on the ATmega1280 and ATmega2560. The Arduino and Genuino 101 boards have an emulated EEPROM space of 1024 bytes.

EEPROM Library

To use this library

#include <EEPROM.h>

EEPROM.read(address) Reads a byte from the EEPROM. Locations that have never been written to have the value of 255. address is the location to read from, starting from 0 (int)

EEPROM.write(address, value) Write a byte to the EEPROM.

EEPROM.update(address, value) Write a byte to the EEPROM. The value is written only if differs from the one already saved at the same address. the EEPROM has also a limit of 100,000 write cycles per single location, therefore avoiding rewriting the same value in any location will increase the EEPROM overall life.

EEPROM.get(address, data) Read any data type or object from the EEPROM.

EEPROM.put(address, data) Write any data type or object to the EEPROM.

EEPROM[address] This operator allows using the identifier ‘EEPROM` like an array. EEPROM cells can be read and written directly using this method.

Project overview

Some stores have variable prices of items that can be changed depending on discounts or promotions. It's difficult to update the prices of all the products in the store or even multiple branches in many different places at the same time.

To solve that problem we will build a dynamic price tag with Arduino and a LCD1602 to be able to show and update the price of the product. Product pricing information will be saved in the EEPROM and reloaded when the device is restarted.



Figure 1: Dynamic price tag

Hardware Required

* Arduino or Genuino Board
* LCD Screen (compatible with Hitachi HD44780 driver)
* pin headers to solder to the LCD display pins
* 10k ohm potentiometer
* 220 ohm resistor
* hook-up wires
* breadboard

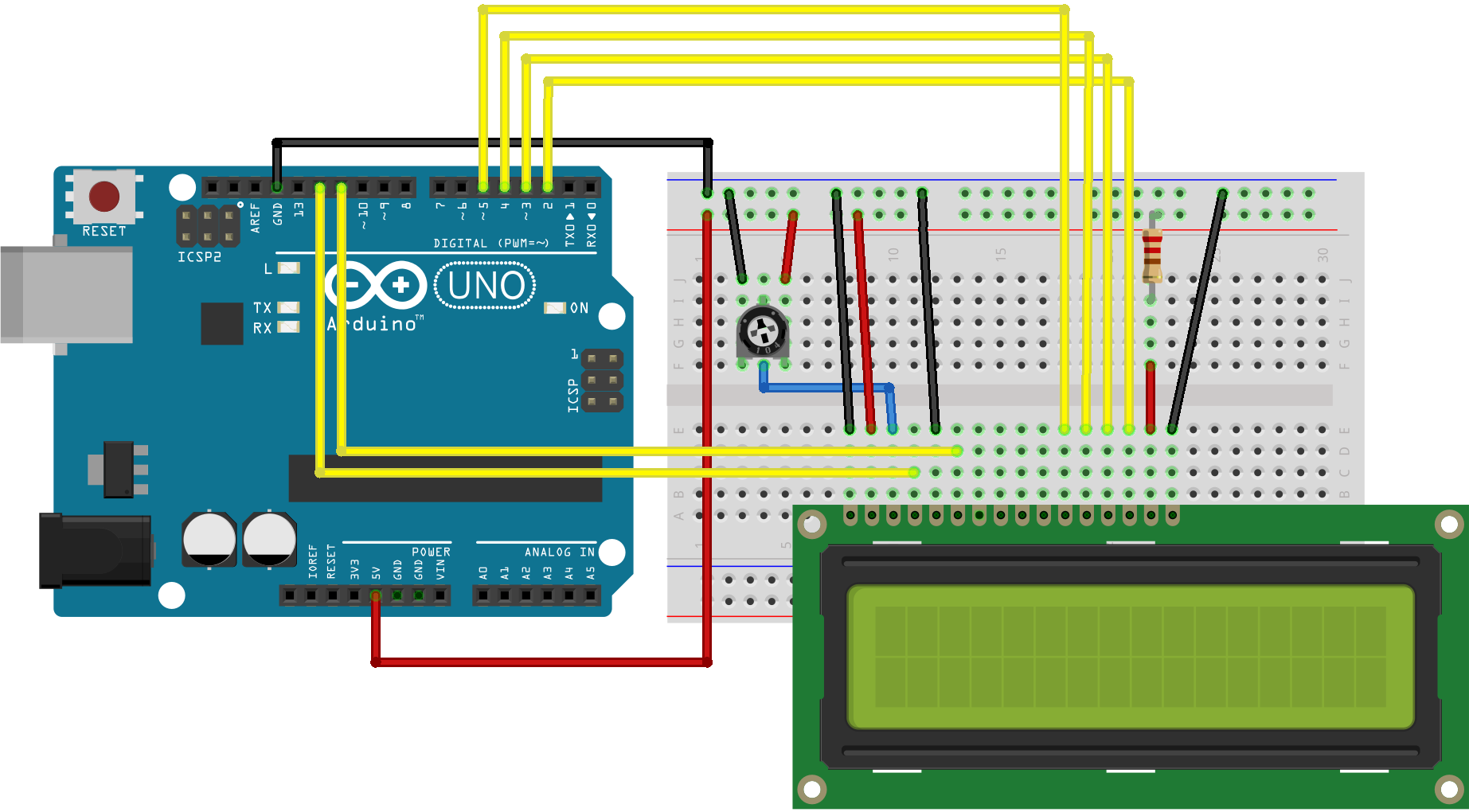
Circuit

Before wiring the LCD screen to your Arduino or Genuino board we suggest to solder a pin header strip to the 14 (or 16) pin count connector of the LCD screen, as you can see in the image above.

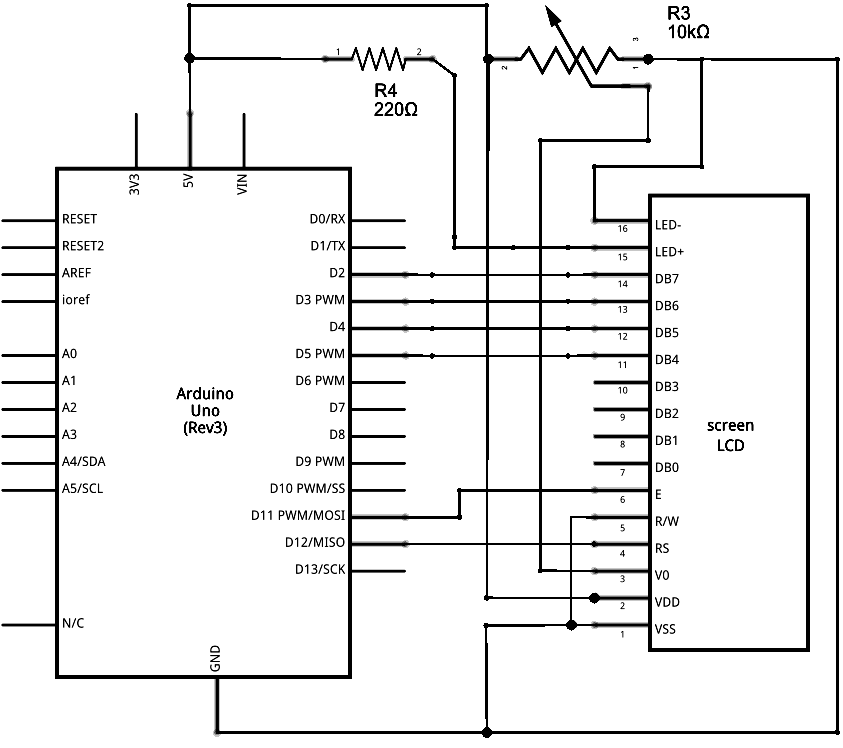
To wire your LCD screen to your board, connect the following pins:

* LCD RS pin to digital pin 12
* LCD Enable pin to digital pin 11
* LCD D4 pin to digital pin 5
* LCD D5 pin to digital pin 4
* LCD D6 pin to digital pin 3
* LCD D7 pin to digital pin 2

Additionally, wire a 10k pot to +5V and GND, with it's wiper (output) to LCD screens VO pin (pin3). A 220 ohm resistor is used to power the backlight of the display, usually on pin 15 and 16 of the LCD connector



Schematic



Code

#include <**LiquidCrystal**.h>

#include <**EEPROM**.h>

void writeString(char add,String data);

String readString(char add);

String inputString = "";         // a String to hold incoming data

bool isReadString = false;  // whether the string is complete

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

**LiquidCrystal** lcd(rs, en, d4, d5, d6, d7);

void setup() {

 // put your setup code here, to run once:

 lcd.begin(16, 2); // init LCD

 // reserve 200 bytes for the inputString:

 inputString.reserve(200);

**Serial**.begin(9600);

 String data = "No data is written in EEPROM";

 if(**EEPROM**.read(0) != 0xFF){ // read data if available in EEPROM

   data = readString(0);

 }

 lcd.print(data);

**Serial**.println(data);

 delay(10);

}

void loop() {

 if(isReadString){

   writeString(0, inputString);

   lcd.print(inputString);

**Serial**.println(inputString);

   inputString = "";

   isReadString = false;

 }

}

void writeString(char address, String data){

 int \_size = data.length();

 int i;

 for(i=0; i<\_size; i++){

**EEPROM**.write(address+i, data[i]);

 }

**EEPROM**.write(address + \_size, '\0');   //Add termination null character for String Data

}

String readString(char address){

 char data[100]; //Max 100 Bytes

 int length=0;

 unsigned char tempChar;

 tempChar = **EEPROM**.read(address);

 while(tempChar != '\0' && length<500){       //Read until null character

   tempChar = **EEPROM**.read(address+length);

   data[length] = tempChar;

   length++;

 }

 data[length]='\0';

 return String(data);

}

void serialEvent() {

 while (**Serial**.available()) {

   // get the new byte:

   char inChar = (char)**Serial**.read();

   // add it to the inputString:

   inputString += inChar;

   // if the incoming character is a newline, set a flag so the main loop can

   // do something about it:

   if (inChar == '\n') {

     isReadString = true;

   }

 }

}

Homework

Modify code to update and show price information in two lines of LCD1602 as figure 1. The 1st line will show product name and the 2nd show the price of product.