Update: since this blog was written, the Project was transferred to a more professional looking project box and is now called the Beeruino, please search to see that blog – also code has been posted to Git.

parts used:

1. empty cigar box (smaller plastic project boxes also are ideal, but cigar box was free)
2. Arduino UNO R3
3. Arduino UNO R3 compatible ethernet shield + SD card that plugs into the SD slot
4. two Dallas 1-wire temperature sensors (1 meter long, internal cable length was extended)
5. 4×20 blue LCD Screen – works over I2C
6. an RTC (real-time-clock), also works over I2C
7. miscellaneous: wires, shrink wraps, hot glue, plastic wire ties and some light soldering

If you are one of those people – who is reading this and inside your head you are saying "why the hell should I do any of this shit, I just buy!!" – guess what, you are not a Maker and you will learn nothing from buying things that others have created. Once you learn, you have full control over your creation and any future ideas/goals, you are not tied to a product that someone else has created.

In this quick blog I wanted to share a quick story about how I converted an empty cigar box into a data logger. It uses the Arduino Uno and two Dallas 1-wire sensor to capture and record both the internal temperature inside the fermentor and the external temperature (outside the fermentor), so that we have a base to compare against. You should see a higher temperature inside, because when yeast ferments, that is considered an exothermic process – https://en.wikipedia.org/wiki/Exothermic_process

The primary goal was to create a small, portable system (small size and weight wise) and also for it to be independent, meaning be able to do everything on its own without external dependencies like the internet, or some network, at this stage we don't want to send live

data to the internet or log to a database // but those things can certainly be done and in the future can be nice to have.

To have this system somewhat nice, I have added a 4×20 blue LCD screen – it uses the I2C interface to make the hook up easy. Also an RTC (real-time-clock) was installed to work on the same I2C bus, this adds date+time.

In addition I have used 4-pin aviation plugs to make the sensors connection modular, so that they can be easily unplugged (cleaning or swapping) without messing with the internal electronics or wires…

Hot glue was used to stick things in place inside the cigar box, along with some limited soldering, and shrink-wrap, and wire ties to keep things organized and properly connected for solid connections.

The programming code is not super complex, and it being shared below:

It assumes that the external temp. sensor is being pulled from index(0) and internal from index(1). Having the temp sensors assigned to a static index will allow you to switch the aviation plugs and still have them assigned correctly and display from the right sensor without having to worry about which plug which should be.

A quick video and Arduino C++ code below…

The data is recorded on an SD card (inside the ethernet shield) which is plugged-in on top of the Arduino UNO R3.

Arduino dallas-1 wire code

C++

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

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

LiquidCrystal_I2C lcd(0x27,20,4);  // set the LCD address to 0x27 for a 16 chars and 2 line display

#include <OneWire.h>
#include <DallasTemperature.h>
#include <SPI.h>
#include <SD.h>
File myFile;

const int chipSelect = 4;
unsigned short int counter = ; // value range between 0 to 65,536 // we have no need to store negative value ranges
// Data wire is plugged into pin 2 on the Arduino
#define ONE_WIRE_BUS 2
// Setup a oneWire instance to communicate with any OneWire devices
// (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);
// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

void setup(void)
{
  lcd.init();                      // initialize the lcd
  lcd.init();

// start serial port
Serial.begin(9600);
// Start up the library
sensors.begin();
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
Serial.print("Initializing SD card...");
// see if the card is present and can be initialized:
if (!SD.begin(chipSelect)) {
Serial.println("Card failed, or not present");
// don't do anything more:
return;
}

Serial.println("card initialized.");

// use this code if you want the file removed upon device reboot or reset ???
//Serial.println("Removing file.txt...");
//SD.remove("file1.txt");
if (SD.exists("brewday2.txt")) {
Serial.println("file1 exists.");
} else {
Serial.println("file1 doesn't exist.");
}
}
void loop(void)
{
delay(5000);

  lcd.backlight();
  lcd.setCursor(,);
//  lcd.print("kodiakbrewing.com ");
  lcd.print("Log Counter: ");

  lcd.setCursor(13,);
  lcd.print(counter);

  lcd.setCursor(,1);
  lcd.print("log to: brewday2.txt");
  lcd.setCursor(12,1);


  lcd.setCursor(,2);
  lcd.print("Int. Temp: ");
  lcd.setCursor(11,2);
  lcd.print(sensors.getTempCByIndex(1) * 1.8 + 32.0); // Convert to F

  lcd.setCursor(,3);
  lcd.print("Ext. Temp: ");
  lcd.setCursor(11,3);
  lcd.print(sensors.getTempCByIndex() * 1.8 + 32.0); // Convert to F

sensors.requestTemperatures();
Serial.print("External Temperature is: ");
Serial.print(sensors.getTempCByIndex() * 1.8 + 32.0); // Convert to F
Serial.print('\n');
Serial.print("Internal Temperature is: ");
Serial.print(sensors.getTempCByIndex(1) * 1.8 + 32.0);
Serial.print('\n');
Serial.println(counter);
// make a string for assembling the data to log:
String external_temp = "";
String internal_temp = "";
// read the two sensors and append to the string:
external_temp = String(sensors.getTempCByIndex() * 1.8 + 32.0);
internal_temp = String(sensors.getTempCByIndex(1) * 1.8 + 32.0);

File dataFile = SD.open("brewday2.txt", FILE_WRITE);
// counter = counter + 1;
counter++;
// if the file is available, write to it:
if (dataFile) {
dataFile.println(String(counter) + "," + external_temp + "," + internal_temp);
dataFile.close();
}
// if the file isn't open, pop up an error:
else {
Serial.println("error opening file.txt");
}
}
```

Share this:

- Twitter
- Google
- Facebook
- Print

Like this:

Like Loading...