

## Anschluss eines LCD Displays an das Tino Radiation Measurement Shield

Bernd Laquai, 22.6.14

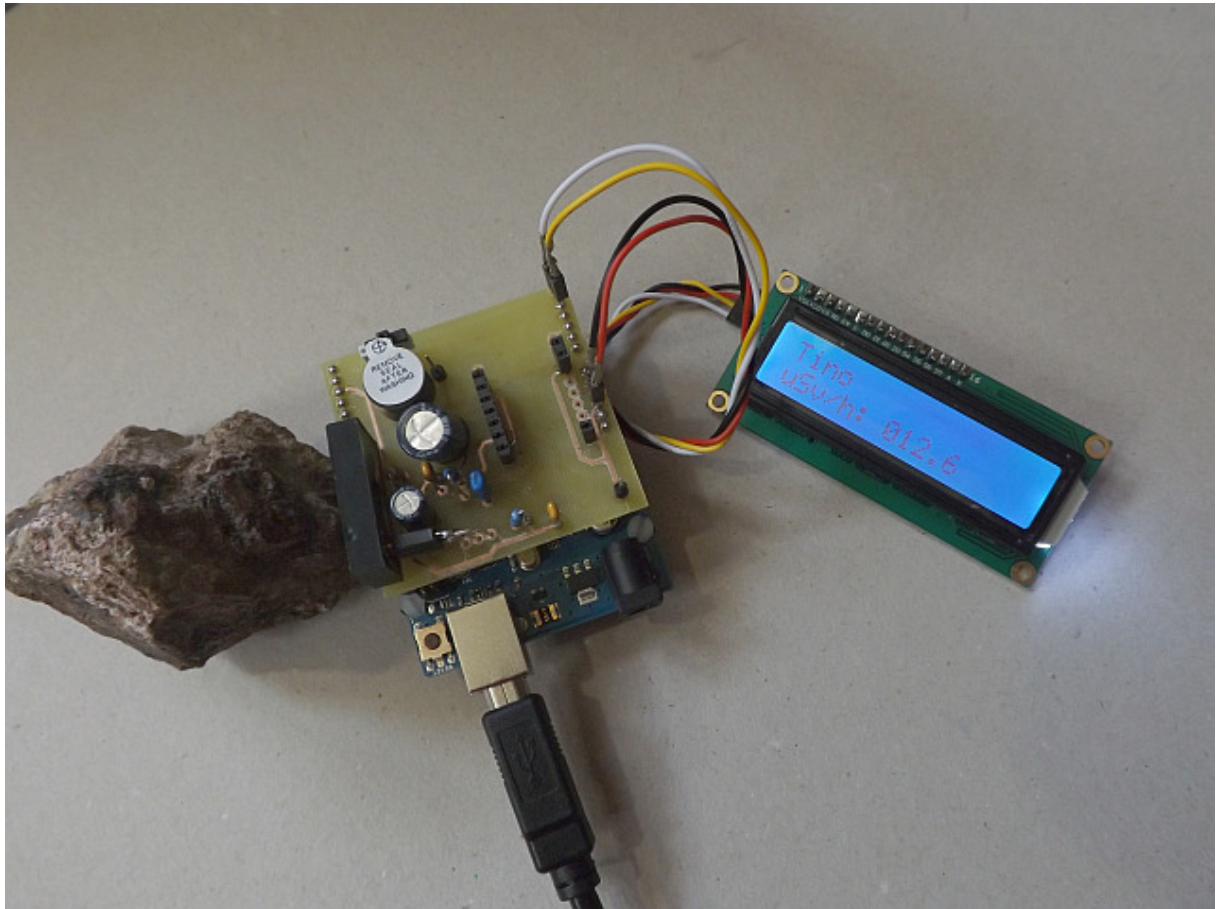


Abb. 1: Messung eines Granits mit Uran-Vererzungen mit dem Tino Shield mit LCD-Display

Es mag Situationen geben, da ist ein 7-Segment LED Display mit 4-Ziffern zu eingeschränkt in der Zahl der Zeichen oder es ist von der eigenen Helligkeit (z.B. bei Sonnenlicht) nicht ausreichend. Dann empfiehlt sich der Einsatz eines LCD Displays. Unbeleuchtete LCD Displays haben bei starkem Auflicht einen deutlich besseren Kontrast, bei Dunkelheit dagegen, sind sie ohne Beleuchtung nicht ablesbar. Es gibt allerdings auch LCD Displays mit einschaltbarer LED Hintergrundbeleuchtung, was in gewissen Situationen auch eine günstige Lösung darstellen kann. Ein weiterer Vorteil normaler LCD Displays ist oftmals auch der geringere Stromverbrauch.

Sehr häufig werden zweizeilige Displays mit je 16 Zeichen pro Zeile eingesetzt, die nach dem Hitachi-Standard über einen parallelen 8-bit Bus angesteuert werden. Für diese Displays gibt es aber auch serielle Interfaces, die meist nach dem I2C-Standard arbeiten, so dass man außer Power und Ground nur zwei weitere Leitungen benötigt. Beim I2C-Standard ist das die Clock-Leitung SCK und die Daten-Leitung SDA. Ein häufig anzutreffendes Interface, das den seriellen I2C Bus auf den Display internen Hitachi 8-bit Bus umsetzt, stammt von der Firma YwRobot trägt die Bezeichnung Arduino LCM1602 IIC und ist meist auf der Rückseite als kleine Platine mit einem Poti zum Einstellen des Kontrasts auf dem parallelen LCD Display aufgesetzt.

Die LCD Displays, ob parallel oder seriell, sind aber meist so groß, dass sie nicht direkt auf ein Shield in der Größe des Arduino Mikrocontroller-Boards aufgesetzt werden können ohne es zu überragen. Sie sind daher eher für den Einbau in ein Gehäuse gedacht.

Da diese Displays sehr populär sind gibt es auch hier bereits sehr gute Bibliotheksunterstützung. Die passende Bibliothek zu den I2C-Bus Displays mit dem Namen „LiquidCrystal\_I2C“ bekommt man von Fransisco Malpartida unter:

<https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads>

Die Bibliothek muss ausgepackt, ggf. umbenannt und in den Arduino Library Ordner verschoben werden um sie benutzen zu können.

Um den Anschluss eines solchen Displays an das Tino Shield zu demonstrieren wurde hier ein blaues LCD Display mit Backlight verwendet, das über die Firma EXP-Tech (exp-tech.de) zu beziehen ist (I2C LCD 1602 Module, Best. Nr. EXP-T04-020). Das Display wurde an den Tino Prototyp über eine Stiftleiste so angeschlossen, so dass es auch wieder abgenommen werden kann.

Außer der LiquidCrystal\_I2C wird für den Betrieb in dem Tino-Beispielprogramm noch die wire.h Standard-Arduino-Bibliothek für den I2C Bus eingesetzt. Sie implementiert beim Arduino UNO (für andere Boards möglicherweise anders!) auf dem Analog Pin A4 die Datenleitung SDA und auf A5 den Clock SCL. Um das Display an das Tino Shield anzuschließen, muss also das mit dem Display mitgelieferte 4 adrige Kabel entweder auf den Arduino Konnektor-Leisten über oben aufgelötete Stifte abnehmbar oder direkt und fest angelötet werden. Da jeweils SDA und SCL sowie Gnd und VDD nebeneinander liegen kann man paarweise angeordnete Verbinder benutzen.

Der Rest ist dann nur noch eine Frage der Software-Anpassung. Ein einfaches Beispielprogramm findet sich in folgendem Listing.

```
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#define MAXCNT 10
#define CalFactor 3.4

volatile int counter = 0;
unsigned long oldTime = 0;
float rate = 0.0;
#define I2C_ADDR 0x27 // Define I2C Address where the PCF8574A is
#define BACKLIGHT_PIN 3
#define En_pin 2
#define Rw_pin 1
#define Rs_pin 0
#define D4_pin 4
#define D5_pin 5
#define D6_pin 6
#define D7_pin 7

int speaker = 5;
//Initialise the LCD
LiquidCrystal_I2C lcd(I2C_ADDR, En_pin, Rw_pin, Rs_pin, D4_pin, D5_pin, D6_pin, D7_pin);
```

```

void setup()
{
  pinMode(speaker, OUTPUT);
  Serial.begin(9600);

  // Define LCD as 16 column x 2 rows
  lcd.begin (16,2);

  // Switch on the backlight
  lcd.setBacklightPin(BACKLIGHT_PIN,POSITIVE);
  lcd.setBacklight(HIGH);

  // Goto first column (0 not 1!), first line (0 not 1!),
  lcd.setCursor ( 0, 0 );

  // Print at cursor location
  lcd.print("Tino");

  // Go to first column (0 not 1!), second line (which is 1 not 2!)
  lcd.setCursor ( 0, 1 );

  // Print at cursor location
  lcd.print("uSv/h: ");

  lcd.print(rate,1);
  attachInterrupt(0, count, RISING);
}

void loop() {
  unsigned long time;
  unsigned long dt;

  time = millis();
  if (counter >= MAXCNT) {
    dt = time-oldTime;
    oldTime = time;
    counter = 0;
    rate = (float)MAXCNT*60.0*1000.0/(float)dt/CalFactor;

    lcd.setCursor ( 8, 1 );
    lcd.print(" ");
    lcd.setCursor ( 8, 1 );
    lcd.print(rate,1);
  }
}

void count()
{
  counter++;
  digitalWrite(speaker, HIGH);
  delayMicroseconds(50000);
  digitalWrite(speaker, LOW);
}

```

**Listing1: Betrieb eines LCD-Displays am Tino Shield**