8 Izpis podatkov

Prikazovanje podatkov je ena od ključnih nalog vseh merilnih sistemov. To lahko naredimo na različne načine:

1. v obliki tabele na zaslon računalnika,
2. v grafični obliki,
3. na zaslon samostojne merilne naprave,
4. shranimo podatke na SD spominsko kartico …

8.1 Serijski izpis

Spodnji primer programske kode prikazuje kako lahko z Arduino krmilniki izpisujemo osnovne podatke neke meritve. Izpisovali bomo dva podatka v ločena stolpca. Stolpca bosta ločena s tabulatorjem, kar predstavlja označba **\t**. V prvem stolpcu bomo izpisovali trenutni čas (**millis();**), v drugega pa vrednost analogne pretvorbe (**analogRead(0);**).

Program 1: Serijski izpis podatkov.

```
1   void setup() {
2     Serial.begin(9600);
3     Serial.println("t[ms]\tADC");
4   }
5
6   void loop() {
7     Serial.print(millis());
8     Serial.print("\t");
9     Serial.println(analogRead(0));
10    delay(100);
11  }
```

8.2 Grafični izris

Zelo nazorno je spremljati izris podatkov v grafičnem načinu. Do neke mere nam to funkcijo ponuja programsko okolje Arduino IDE, vendar v zelo okenjeni obliki. Preskusite naslednji program:

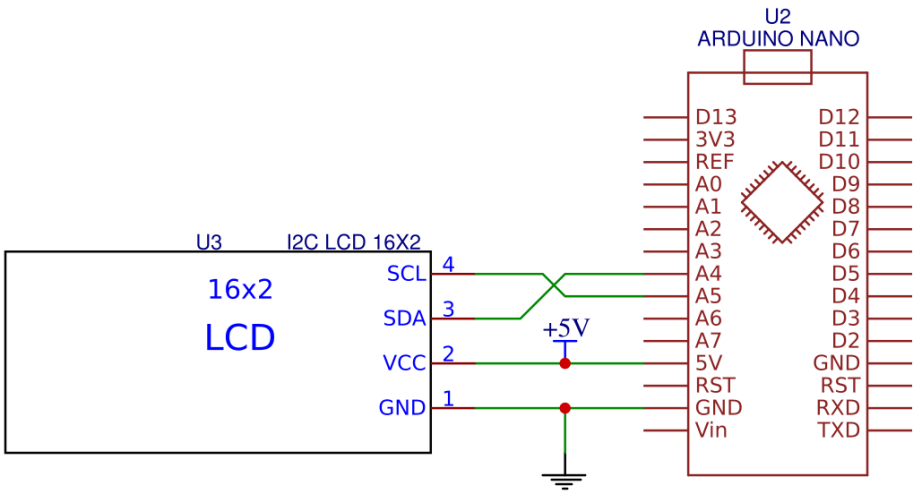
Program 2: Izrisovanje vrednosti s serijskim grafikonom.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.print("0,1023,");
  Serial.println(analogRead(A0));
  delay(10);
}
```

Kot lahko opazimo računalniku pošiljamo 3 različne števila: 0, 1023 in ADC vrednost. Prvi dve nam slušita kot območje, saj bi v nasprotnem primeru program sam prilagajal skalo na y osi (ang.: autofit).

8.3 Izpis na LCD

Če želimo narediti merilno napravo, ki bo delovala samostojno, bomo verjetno potrebovali dodaten ekran za prikaz izmerjenih vrednosti. Zato ga moramo priključiti na krmilnik Arduino, kot kaže slika sl. 1.



Slika 1: Vezava I²C LCD-ja na krmilnik Arduino.

Ena naj enostavnejših rešitev je, da krmilniku Arduino UNO dodamo LCD ekranček z I²C vodilom. Ker je I²C vodilo zasnovano tako, da nanj lahko priključimo več naprav, moramo za vsako napravo poznati njen naslov. Le tega lahko pridobimo od proizvajalca naprave, ga nastavimo sami ali pa z ustreznim programom pregledamo vse naslove priključenih naprav (programska koda(Arduino 2019)). Za uporabo LCDja na podatkovnem vodilu I²C bomo potrebovali knjižnico **LiquidCrystal_I2C**, ki si jo

lahko presnamete s spletne strani johnrickman. Knjižnico lahko presnamete tako, da kliknete na gumb `Code \bigtriangledown` in nato še `Download zip`. V Arduino IDE pa `Tools` -> `Library Manager` -> `Add zip Library` in dodate pravkar presneto zip datoteko.

Program 3: Izpis podatkov na LCD prikazovalnik.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C DaqLcd(0x27, 16, 2);

void setup() {
  DaqLcd.init();
  DaqLcd.backlight();
  DaqLcd.clear();
}

void loop() {
  int i = 0;
  long ch = 0;
  for (i=0;i<64;i++){
    ch += analogRead(7);
  }

  float val = ch/64.0;
  DaqLcd.clear();
  DaqLcd.print(val);
}
```

8.3.1 NALOGA: Izpis podatkov

Prikažite podatke na različne načine:
1. v obliki tabele,
2. v grafični obliki in
3. na LCD zaslon merilne naprave.

8.4 Shranjevanje podatkov na SD spominsko kartico

Za krmilnike Arduino Uno (in druge) lahko dokupimo module z režami za SD spominsko kartico. En takih je modul za vzorčenje in beleženje meritev je (angl.:) Data Logger Shield. Modul vsebuje vtičnico za SD kartico, kot tudi bolj točen časovni oscilator (angl.: RTC), s katerim lahko merimo čas na nekaj sekund točno na mesec.

Pri vnašanju podatkov v datoteko je smiselno, da izberemo način vnosa podatko, kjer so podatki ločeni

z vejico. Tako oblikovane podatke lahko shranimo v datoteko s končnico `csv` (angl.: comma separated values). Primer programske za vzorčenje in shranjevanje podatkov s pomočjo programske knjižnice `SD.h` je relativno enostaven.

```
#include <SD.h>
#include <SPI.h>
// define constants for the pin numbers
#define chipSelect 10
#define dataPin 11
void setup() {
 // Initialize serial port for communication with PC
 Serial.begin(9600);
 // Initialize SD card
 Serial.print("Initializing SD card...");
 if (!SD.begin(chipSelect)) {
  Serial.println("initialization failed!");
  return;
 }
 Serial.println("initialization done.");
 // create a new file
 File dataFile = SD.open("datalog.csv", FILE_WRITE);
 // write to the file
 if (dataFile) {
 dataFile.println(\"Data Logging Start!");
 dataFile.close();
 }
}

void loop() {
 // log data
 int data = analogRead(dataPin);
 Serial.println(data);
 // open the file. note that only one file can be open at a time,
 // so you have to close this one before opening another.
 File dataFile = SD.open("datalog.txt", FILE_WRITE);
 // if the file is available, write to it:
 if (dataFile) {
  dataFile.print(millis());dataFile.print(",");dataFile.println(data);
  dataFile.close();
 }
 delay(1000);
}
```

Arduino. 2019. “Arduino Playground - I2cScanner.” 2019. https://playground.arduino.cc/Main/I2cScanner/.