Arduino UNO vezérlésű digitális óra fejlesztői dokumentációja

Tartalom

Specifikációk, követelmények:

Arduino vagy Raspberry Pi alapú mikroelektronikai rendszer létrehozása.

Értékelési szempontok:

- Elégtelen: nem felel meg a minimális kritériumoknak, akár tartalmilag vagy minőségben
- Elégséges: viszonylag egyszerűbb a feladat, és készül hozzá fejlesztői dokumentáció és használati útmutató
- Közepes: bonyolultabb a feladat, amely kiegészül még valamilyen mérő/adatgyűjtő rendszerrel (adatbázis) és egyaránt készül hozzá fejlesztői dokumentáció és használati útmutató
- Jó: a közepes szint elvárásait még ki kell egészíteni valamilyen adatvizualizálási rendszerrel

- Jeles: a jó osztályzat elvárásait még ki kell egészíteni valamilyen plusz kommunikációs csatornával. Ez lehet pl.: tweet, facebook post, email notification, push üzenet a telefonon stb. (csak a képzelet szabhat határokat)

Tervezési fázis:

Eleinte sokat gondolkoztam, hogy mit is tudnék megvalósítani és nagyon nem jutott eszembe semmi. Rájöttem, hogy körül kell néznem a saját környezetemben, ahhoz, hogy legyen bármi elképzelésem arról, hogy mit is akarok megvalósítani. Itthon van egy digitális órám, ami ugyan ezen az elven működik. Bedugjuk a tápot, az óra kiírja az időt szép világosan. Ezzel csak az a baj, hogy ez az óra napokkal később már több mint nyolc perces sietéssel jár. Így esett a választásom erre a projektre, hiszen szeretnék egy használhatót a szobámba. Sikerült a szükséges modulokat is egy oldalról beszereznem, így jóval megkönnyítve és lerövidítve a projekt menetét.

Futási környezet

Arduino IDE 2.0.1 futtatására alkalmas operációs rendszer:

- Windows:
  - Windows 10, vagy újabb
- Linux:
  - AppImage 64 bits (X86-64)
- MacOS:
  - 10.14: “Mojave” or newer, 64 bits

Periféria követelmények:

- Billentyűzet
- Egér
- Monitor

A fejlesztői környezet

Az Arduino Uno felprogramozásához az Arduino IDE 2.0.1 nevű programozási környezetet használtam, mely C++ alapokon nyugszik. Én Windows 10-et használtam a projekt készítése folyamán, hiszen ez volt a minumum specifikáció ehhez a programhoz.

Felhasznált modulok

- Arduino UNO REV3 fejlesztői panel
- DS3231 + AT24C32 I2C RTC valós idejű memória modul
- CR2032 Gombelem
- KC-1602-BB-I2C LCD kijelző
- HS-005 próbapanel

- RC-40-10/FF jumper kábel
- RC-40-10/MF jumper kábel

A fontosabb modulok leírása

DS3231 RTC

A modul egy valós idejű óra (Real time clock vagy másképp RTC modul), amely képes kezelni órát, percet és másodpercet rendkívül nagy pontossággal. Ugyancsak képes kezelni a napokat hónapokat éveket. A pontosságért a DS3231 típusú chip felel, míg az adattárolásért AT24C32, amely 32kBit nagyságú.

Az óra pontossága +/- 2ppm (0-50°C). Ez annyit jelent, hogy szélsőséges esetekben 1.2 másodperc lehet a késés havonta. Normál kvarckristályos órák esetén ez 30 másodperc is lehet.

A modul 3,3 vagy 5 V-os feszültséggel is képes működni, ami sok fejlesztői platformra, mikrokontrollerre alkalmassá teszi a használatát. Az akkumulátor bemenete 3V-os (CR2032 típusú akkumulátor), amely elősegíti, hogy a modul több mint egy évig megtarthatja az információkat.

A modul az I2C kommunikációs protokollt használja, amely megkönnyíti az összeköttetést az Arduino-val.

KC-1602-BB-I2C LCD kijelző

Ez az LCD képernyő 16 karaktert képes megjeleníteni két sorban (1602). Kék színű háttérvilágítással rendelkezik, az I2C kommunikációnak köszönhetően könnyedén (két adatvezeték segítségével) használható Arduino projekthez.

Az algoritmusok és a kódok:

Könyvtárak:

```
#include <Wire.h>                 // for I2C communication
#include <LiquidCrystal_I2C.h>     // for LCD
#include <RTClib.h>                // for RTC
LiquidCrystal_I2C lcd(0x27, 16, 2);
RTC_DS3231 rtc;              // create rtc for the DS3231 RTC module, address is fixed at 0x68
```

Egyedi funkciók:

updateRTC():

```
/*
  function to update RTC time using user input
*/
void updateRTC()
{
  lcd.clear();  // clear LCD display
```

```
  lcd.setCursor(0, 0);
  lcd.print("Szerkesztes...");
  // ask user to enter new date and time
  const char txt[6][15] = { "evet [szam]", "honapot [1~12]", "napot [1~31]",
                      "orat [0~23]", "percet [0~59]", "mp-t [0~59]"};
  String str = "";
  long newDate[6];
  while (Serial.available()) {
   Serial.read();  // clear serial buffer
  }
  for (int i = 0; i < 6; i++) {
   Serial.print("Kerek egy ");
   Serial.print(txt[i]);
   Serial.print(": ");
   while (!Serial.available()) {
    ; // wait for user input
   }
   str = Serial.readString();  // read user input
   newDate[i] = str.toInt();   // convert user input to number and save to array
   Serial.println(newDate[i]); // show user input
  }
  // update RTC
  rtc.adjust(DateTime(newDate[0], newDate[1], newDate[2], newDate[3], newDate[4],
newDate[5]));
  Serial.println("RTC Frissitve!");
}
```

Ez a funkció felelős azért, hogy a felhasználótól bekérje a dátumot és az időt, és frissítse az RTC belső óráját a felhasználó bemeneti adataival.

updateLCD()

```
void updateLCD()
{
  /*
    create array to convert digit days to words:
    0 = Sunday   |   4 = Thursday
    1 = Monday    |   5 = Friday
    2 = Tuesday   |   6 = Saturday
    3 = Wednesday |
  */
  const char dayInWords[7][4] = {"SUN", "MON", "TUE", "WED", "THU", "FRI", "SAT"};
  /*
    create array to convert digit months to words:
    0 = [no use]  |
    1 = January   |   6 = June
    2 = February  |   7 = July
    3 = March     |   8 = August
    4 = April     |   9 = September
```

```
    5 = May       |   10 = October
    6 = June      |   11 = November
    7 = July      |   12 = December
  */
  const char monthInWords[13][4] = {" ", "JAN", "FEB", "MAR", "APR", "MAY", "JUN",
                                    "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};
  // get time and date from RTC and save in variables
  DateTime rtcTime = rtc.now();
  int ss = rtcTime.second();
  int mm = rtcTime.minute();
  int hh = rtcTime.twelveHour();
  int DD = rtcTime.dayOfTheWeek();
  int dd = rtcTime.day();
  int MM = rtcTime.month();
  int yyyy = rtcTime.year();
  // move LCD cursor to upper-left position
  lcd.setCursor(0, 0);
  // print date in dd-MMM-yyyy format and day of week
  if (dd < 10) lcd.print("0");  // add preceeding '0' if number is less than 10
  lcd.print(dd);
  lcd.print("-");
  lcd.print(monthInWords[MM]);
  lcd.print("-");
  lcd.print(yyyy);
  lcd.print("  ");
  lcd.print(dayInWords[DD]);
  // move LCD cursor to lower-left position
  lcd.setCursor(0, 1);
  // print time in 12H format
  if (hh < 10) lcd.print("0");
  lcd.print(hh);
  lcd.print(':');
  if (mm < 10) lcd.print("0");
  lcd.print(mm);
  lcd.print(':');
  if (ss < 10) lcd.print("0");
  lcd.print(ss);
  if (rtcTime.isPM()) lcd.print(" PM"); // print AM/PM indication
  else lcd.print(" AM");
}
```

Ez a funkció frissíti az LCD-n megjelenő szöveget.

Alapértelmezett funkciók:

setup()

```
void setup()
{
  Serial.begin(9600); // initialize serial
```

```
  lcd.init();      // initialize lcd
  lcd.backlight();  // switch-on lcd backlight
  rtc.begin();      // initialize rtc
}
```

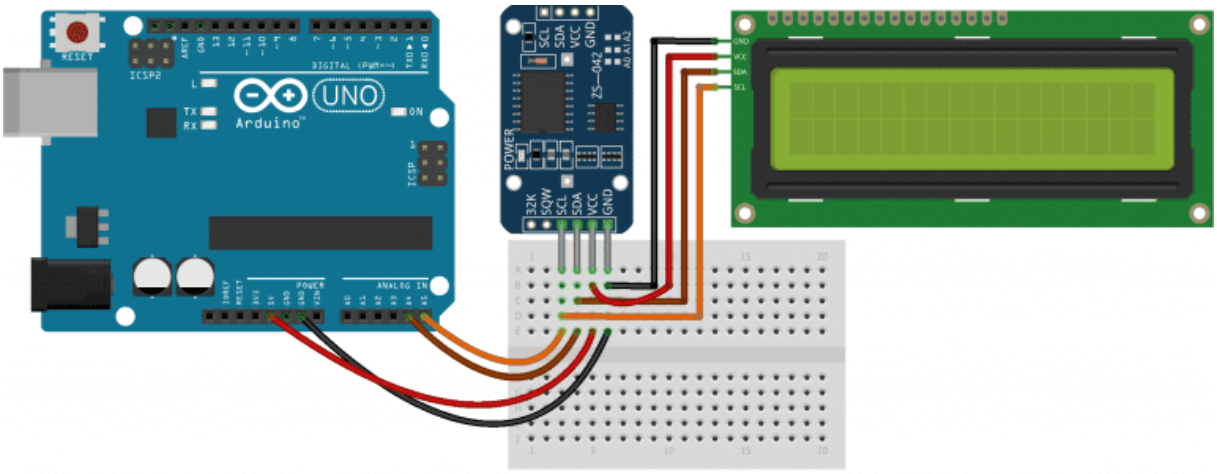
Az LCD-nél inicializálnunk kell az LCD objektumot, és be kell kapcsolnunk a kijelző háttérvilágítását.

loop()

```
void loop()
{
  updateLCD();  // update LCD text
  if (Serial.available()) {
    char input = Serial.read();
    if (input == 'u') updateRTC();  // update RTC time
  }
}
```

Ha a felhasználó elküldi az „u” karaktert a soros monitoron keresztül, az azt jelenti, hogy módosítani akarja az rtc beállított időpontját és dátumát.

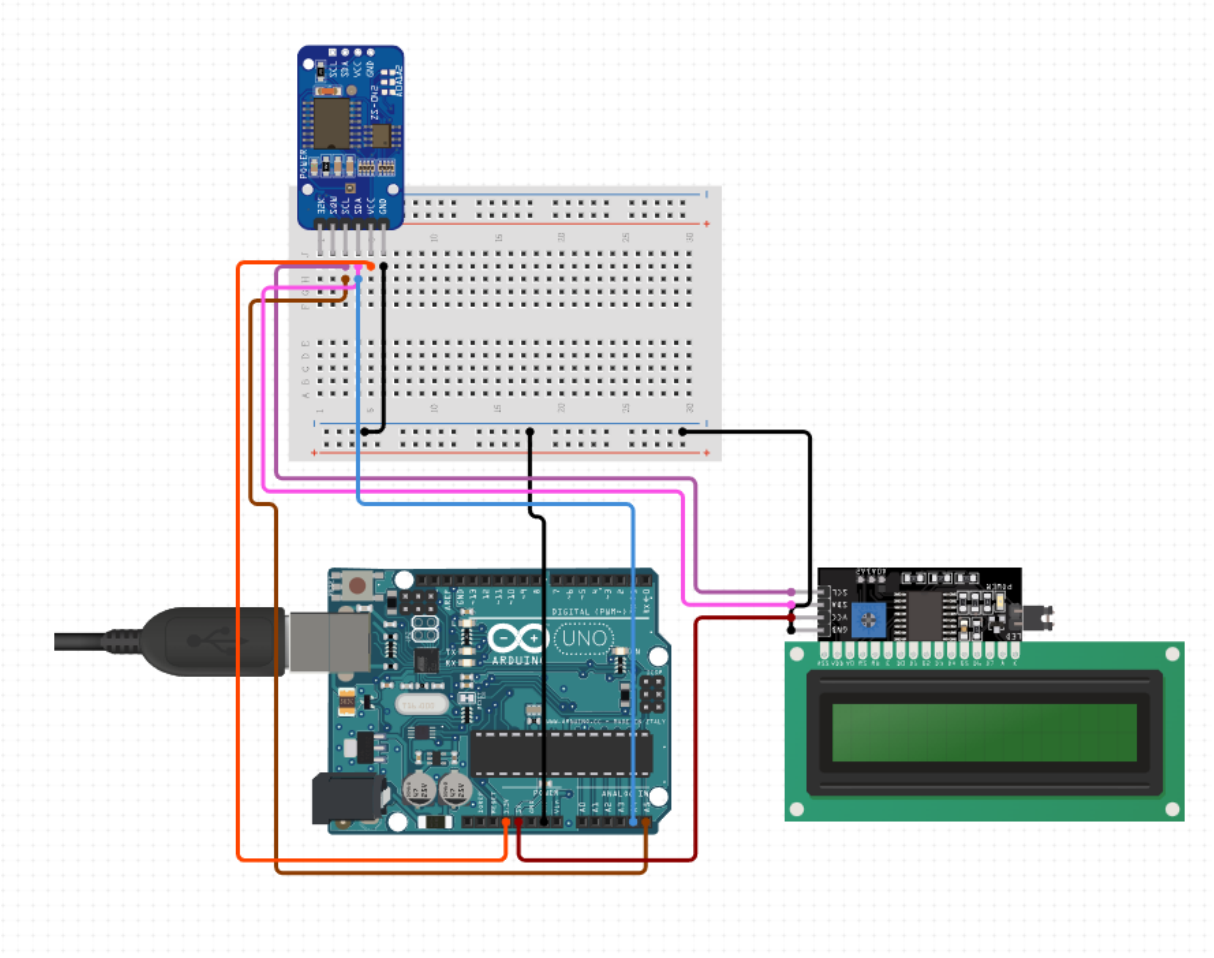
Kapcsolási rajz



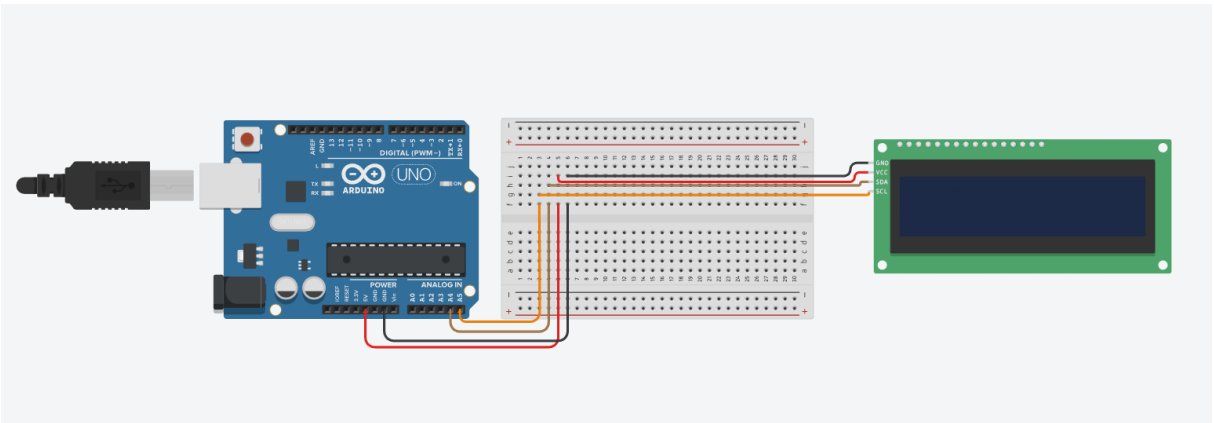
Fellépő hibák a tervezés, tesztelés során:

A tervezés közben egy olyan -számomra- hibát találtam, ami mellett nem tudtam dönteni. Mivel ezelőtt nem tanultam semmi ilyesmit, ezért persze internetes oldalakon, próbáltam kapcsolási rajzot összerakni. Sikerült is, kisebb nagyobb sikerrel. Csak 2 oldalon próbáltam és mindegyiken más-más sikerrel sikerült megcsinálni.

Az első példa azt mutatja be, hogy itt a kapcsolási rajz teljesen kész állapotban van, viszont nem tudom lefuttatni szimuláció céljából.



A második példa, pedig azt mutatja be, hogy az RTC modul híján teljes az ábra. Sajnos ezen az oldalon nem volt megtalálható az RTC modul, pedig ezt az oldalt találtam csak, ahol le is lehet szimulálni a projekt működését.



Egy másik probléma, hogy laza a csatlakozás, könnyen szétcsúsznak az érintkezések, így nehéz megtalálni azt a pozíciót, amikor jól érintkezik.

Működési vázlat, szimuláció online

A működési vázlatok nagy részét online raktam össze a TinkerCad és a circuito.io nevű weboldalakon. A legtöbb alkatrész megtalálható volt ezen a két oldalon. RTC modul híján sajnos nem tudtam online szimulálni.

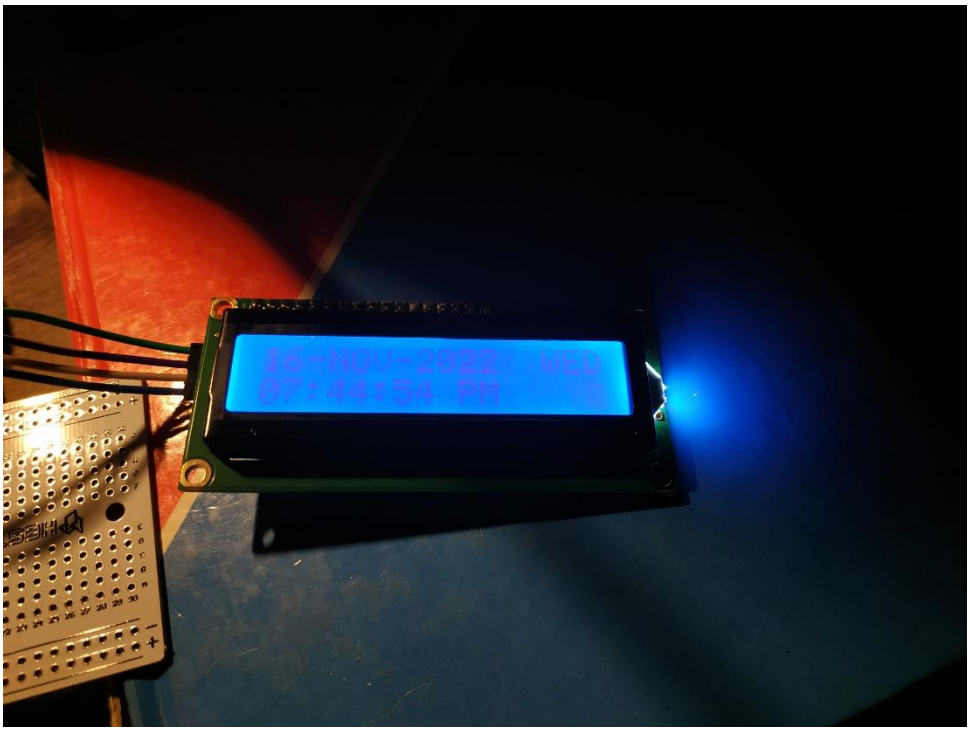
Fejlesztési lehetőségek:

Mivel ez nem egy végleges verzió, így rengeteg tovább fejlesztési lehtőség rejlik ebben a projektben. Ezek közül pár:

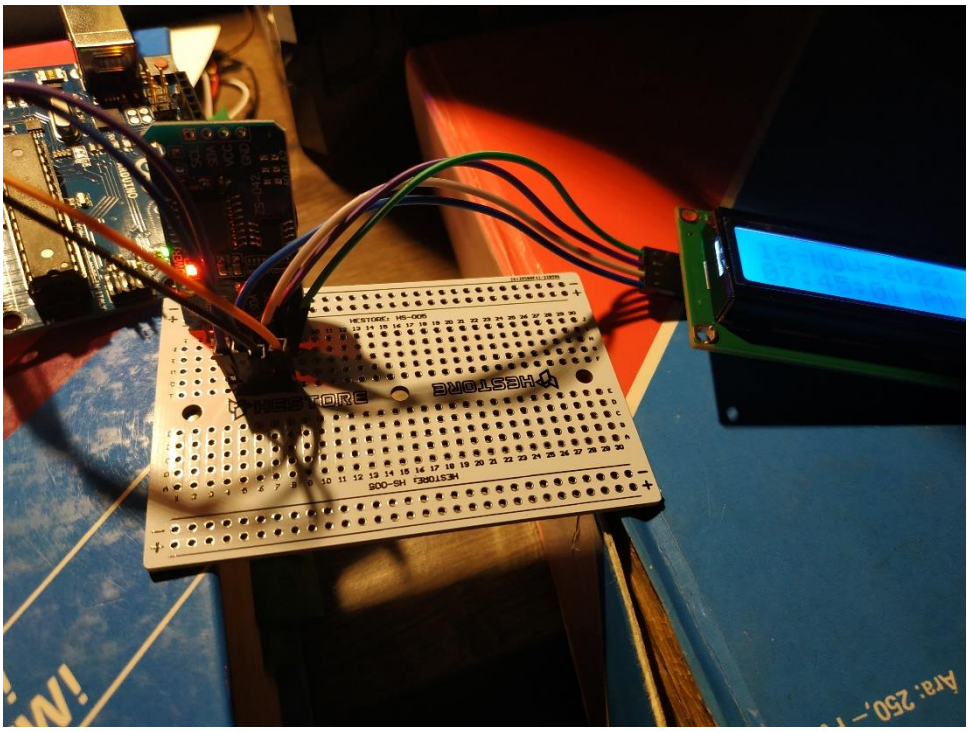
- Konnektorból, esetleg akkumulátorról való működtetés
- Óra vázának kialakítása
- Óra/dátum átállítása gombokkal

Képek az óráról:

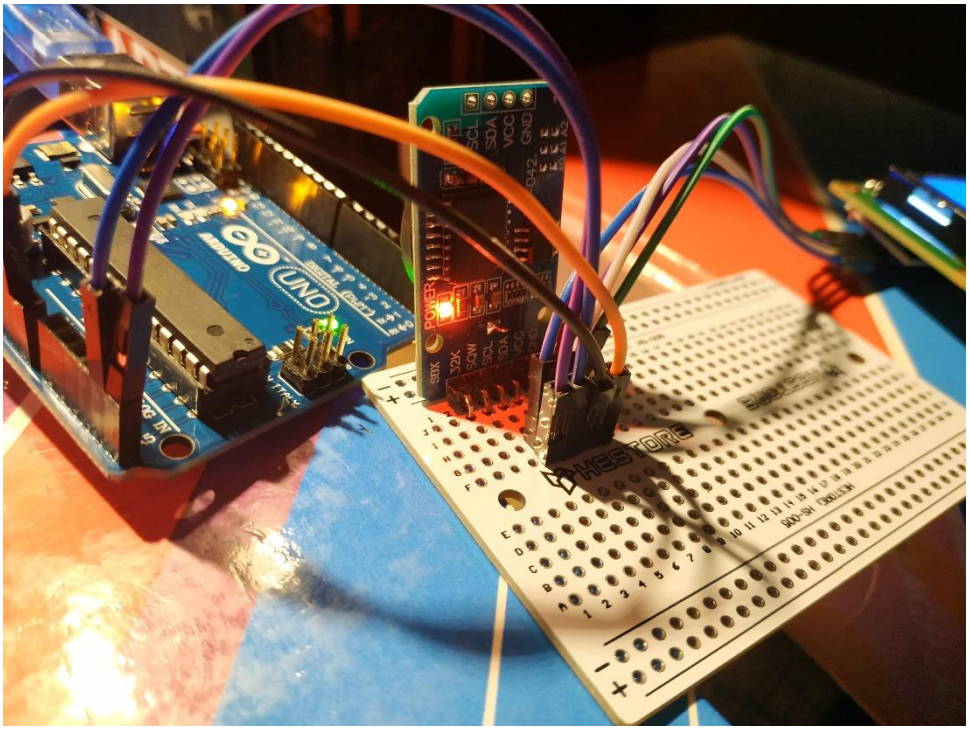
LCD kijelző kész állapotban:



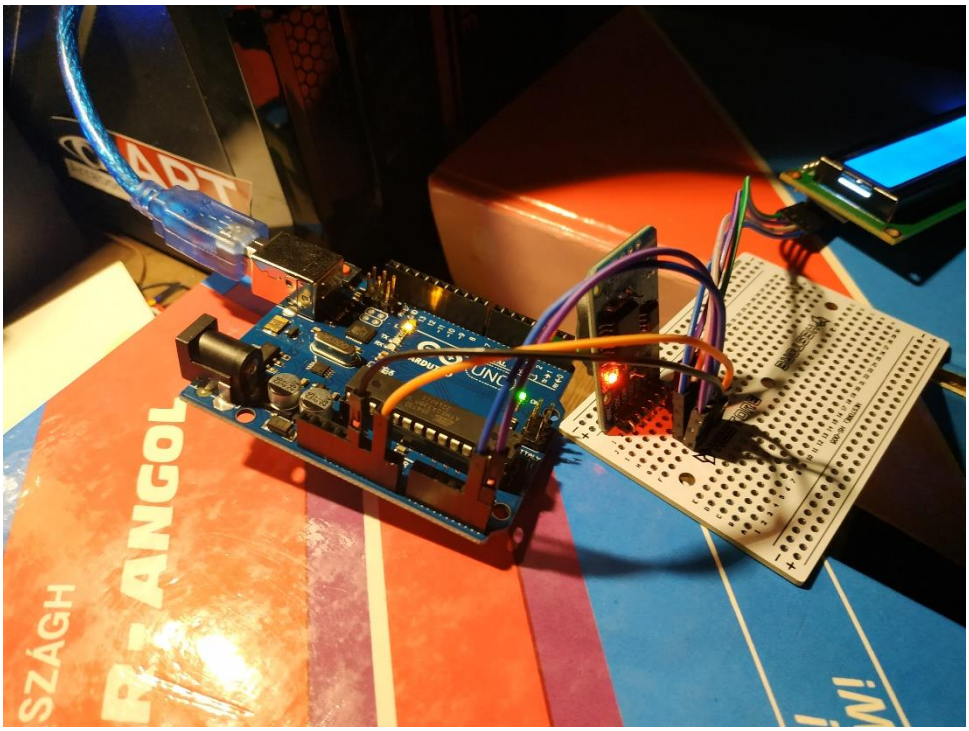
A kapcsolás:



RTC modul:



Arduino UNO:



Szerző:

- Név: Játékos Ádám Csaba
- Neptun-kód: DLHKIJ
- Email: jatekadi@gmail.com

- Tantárgy kódja: GKNB_INTM020
- Neve: Mikroelektromechanikai rendszerek