## 7. Távolságmérés infravörös Sharp 2Y0A21-es szenzor segítségével

Következő projektünk témája a távolságmérés. Ehhez egy Sharp 2Y0A21-es szenzort fogunk használni. A szenzortól kapott jelet pedig egy I2C buszrendszeren kommunikáló LCD kijelzőre és az Arduino által használt soros portra is egyaránt ki fogjuk íratni centiméterekben. Természetesen a mért távolság mindig az aktuális értékét fogja megjeleníteni a kijelzőn és a soros porton is egyaránt ezért a programkódot ennek a feltételnek megfelelően fogjuk létrehozni.

**Szükséges eszközök:** Számítógép, Arduino IDE, Arduino Uno, egy db 2x16 karakteres LCD kijelző I2C adapterrel, egy db Sharp 2Y0A21-es szenzor, USB kábel és vezetékek.

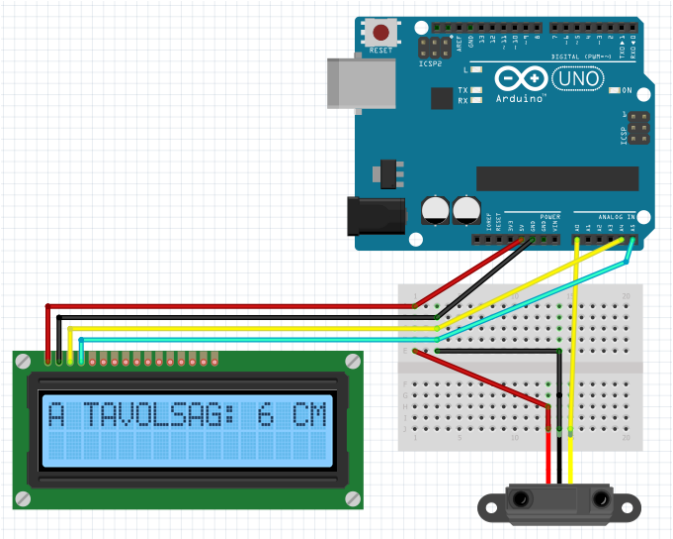
A programot természetesen a könyvtárak meghívásával kell kezdenünk. Mivel az infravörös távolságmérő használatához szükségünk van olyan adatokra és funkciókra melyekhez a szükséges könyvtár nélkül nem tudunk hozzáférni. A szenzorunk a *SharpIr.h* elnevezésű *header* fájlt igényli.

```
#include <SharpIR.h> //A SHARP 2Y0A21 szenzor használatához szükséges könyvtár beillesztése
#include<Wire.h> //Wire könyvtár beillesztése az I2C busz használatához
#include<LiquidCrystal_I2C.h> //Az I2C Folyékony kristályos LCD kijelző kezelő könyvtára
LiquidCrystal_I2C lcd(0x27, 16, 2); //Az általunk használt kijelző karakterkészlete 16 karakter és 2 sor
#define IR A0 //A szenzor adat pinjének a definiálása
#define model 1080 //Pontos típus beállítása
SharpIR SharpIR(IR, model); //A könytárból használt objektum
void setup()
{
  lcd.init(); //Az LCD kijelző inicializálása
  lcd.backlight(); //Az LCD kijelző háttérvilágításának bekapcsolása
  Serial.begin(9600); //A soros porton történő kommunikáció bitrátája
}
void loop() //ciklus
{
  delay(500);
  lcd.clear(); //Az LCD kijelző tartalmának a törlése
  lcd.setCursor(0, 0); //Kurzor pozicionálás ez esetben 0. karakter az 0. sorban
  lcd.print("A TAVOLSAG:"); //Megadott karakterlánc kiíratása
  unsigned long kezdet = millis(); // 32 bites változó deklarálása a kezdettől számított idő tárolására
  int tav = SharpIR.distance(); // this returns the distance to the object you're measuring
  lcd.setCursor(12, 0); //Kurzor pozicionálás ez esetben 12. karakter az 0. sorban
  lcd.print(tav); //A távolság értékenek kiíratása
  lcd.setCursor(14, 0); //Kurzor pozicionálás ez esetben 14. karakter az 0. sorban
  lcd.print("CM"); //Megadott karakterlánc kiíratása
  Serial.print("Tavolsag: "); //Megadott karakterlánc kiíratása a soros portra
  Serial.println(tav); //A távolság értékenek kiíratása a soros portra
  Serial.println(analogRead(A0)); //A szenzorból kapott nem átalakított jel kiíratása a soros portra
  unsigned long veg = millis() - kezdet; //32 bites változó deklarálása a vegző időpont tárolására
  Serial.print("Eltelt ido (ms): "); //Megadott karakterlánc kiíratása a soros portra
  Serial.println(veg); //A távolság értékenek kiíratása a soros portra
}
```

*1. ábra Infravörös távolságmérés programkód*
*(forrás: saját szerkesztés)*

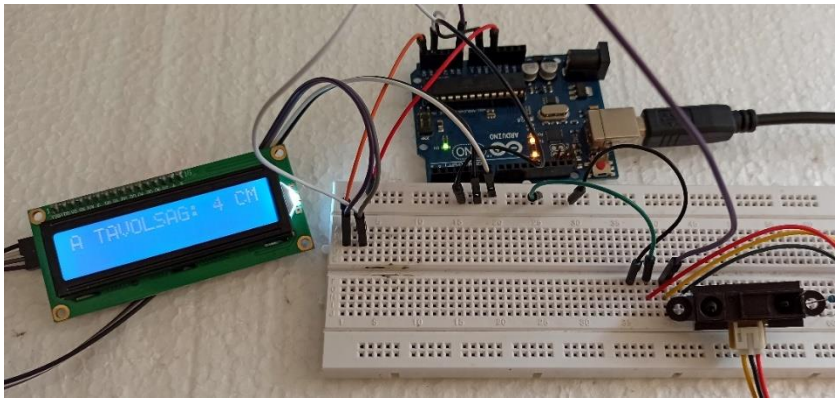
Ha ezt és a többi szükséges könyvtárat is sikeresen importáltuk akkor definiáljuk a szenzor adat pinjét a következő módon *#define IR A0*, ahol az *IR*-rel a szenzorunkra tudunk hivatkozni az *A0*-val pedig a megfelelő bemenetet tudjuk hozzárendelni, vagyis az *IR* az Arduino A0 sorszámú analóg bemenetére csatlakozik. Ezt követi egy típus definiálás, ahol a szenzorunk pontos modelljét állítjuk be, így a könyvtár számára is világos, hogy a többféle ilyen célra használt Sharp gyártmányú szenzor közül melyiket is fogjuk használni. Majd ezután az előbb definiált paramétereket, hozzárendeljük a SharpIR.h könyvtár SharpIR objektumához. Következhet is a void setup() rész, ahol szokásosan elvégezzük a szükséges lcd kijelző műveleteket, valamint, a soros port kommunikáció bitrátája értékének a megadását mely ez esetben is 9600 baud. Rátérhetünk tehát a void loop() ciklusunkra, itt mindjárt egy 500 milliszekundumos késleltetéssel indulunk, ez az idő elég ahhoz, hogy a szenzor értéke stabilizálódjon. Egy esetleges lcd.clear() funkció után egy megadott karakterlánc kiíratása következik, ahol szükséges a kurzort is pozícionálni, ez a pozíció a 0-ik karakter a 0-ik sorban. Jöhet a megjelenítésre szánt üzenet, mely a következő ”A TÁVOLSÁG:”. A következő sorban egy nagyobb nem negatív szám tárolására alkalmas változó deklarálása történik, a változó értéke pedig egyenlő a millis() funkció visszaadott értékével. A millis() funkció arra szolgál, hogy az Arduino elindítását követően ez a funkció akár egy óra méri az időt és milliszekundumokban tudjuk belőle kiolvasni. Vagyis ebben az esetben a *kezdet* nevezetű változónk a program futásától mért idő milliszekundumokban kifejezett értékét tárolja. Erre azért van szükségünk, hogy valós időben tudjunk dolgozni, mint tudjuk a delay() funkció itt nem válna hasznunkra mivel ez a funkció szünetelteti a programunk futását az általunk megadott időre. Tovább haladva a szekvenciában egy újabb deklarációt végzünk ez esetben egy integer típusú változó melynek értéke egyenlő a Sharp szenzorunk által használt könyvtár egy funkciójával mely a *SharpIR.distance()*. Ez a funkció adja vissza az infravörös távolságmérőnkből kapott értéket. Jelenítsük meg az lcd kijelzőn ezt az értéket, elsőként helyezzük az lcd mutatóját a 12-ik karakterre a 0-ik sorban, majd az lcd.print() funkció segítségével írassuk ki a távolságot tartalmazó változónkat. Sajnos az *lcd.print(Sharp.IR.distance())* funkció egyenes kiíratása nem lehetséges ezért mindig az ezen funkció által visszakapott értéket tartalmazó változót kell kiíratnunk. Amint megtörtént a kiíratás ne felejtsük a megfelelő mértékegység hozzárendelését és karakterek formájában való kiíratását, mi esetünkben a ”CM”-t. Ennek végeztével kiíratjuk a szenzortól kapott nyers jelet a soros portra majd egy újabb, nagyobb nem negatív szám tárolására alkalmas változó segítségével és egy gyors matematikai kivonás műveletével megkapjuk a

távolság mérés időtartamát, ezt egy üzenet formájában a *Serial.print()* funkció segítségével megjelenítjük a soros porton. Fontos megjegyezni, hogy ha Serial.println() funkció helyett a *Serial.print()* funkciót választjuk akkor tanácsos az idézőjelek közé beszúrni egy \n-t a sortörés elvégzéséhez. Így átláthatóbbá válik a megjelenítés.




*2. ábra Infravörös távolságmérés bekötési rajz*
*(forrás: saját szerkesztés)*

A fenti 2. ábra mutatja a szenzor helyes bekötésének módját, miszerint az Arduino 5 voltot biztosító pinjét a szenzor vörös vezetékéhez a GND-t pedig a szenzor fekete vezetékéhez kell csatlakoztatnunk, végül a sárga vezeték az Arduino A0 sorszámú analóg bemenetére csatlakozik. A program feltöltését és az áramkör elindítását követően a működés így néz ki (3.ábra).




*3. ábra Infravörös távolságmérés működés közben*
*(forrás: saját szerkesztés)*