4. CO, LPG és füst érzékelése MQ2-es szenzor segítségével

Ebben a projektben füst érzékelést és gázszivárgást fogunk észlelni egy erre alkalmas MQ2-es szenzor segítségével. A projekt célja, a szenzor könyvtárának megfelelő használata és a mért értékek feldolgozása és megjelenítése. Ahogy már az előző projektekben megszokhattuk a mért értékek szintén egy I2C LCD kijelzőre lesznek kiíratva. Az áramkör tartalmaz még egy piezo elemet is mely képes széles frekvenciatartományban hangjelzéseket kibocsátani. Továbbá egy led dióda és villogtatásra kerül, hogy ezzel is szemléltessük azt, hogy bizonyos mért értékek tolerancián túli értékre léptek. A mi esetünkben a tolerancia egy egyszerű feltétel vizsgálattal lesz megszabva.

Szükséges eszközök: Számítógép, Arduino IDE, Arduino Uno, egy db 2x16 karakteres LCD kijelző I2C adapterrel, egy db MQ2-es szenzor, egy db led dióda, egy db piezo elem, USB kábel és vezetékek.

Nézzük is a programunkat, már tudjuk mivel kell kezdenünk. Természetesen a két alapkönyvtárunk marad, mivel még mindig LCD kijelzőt és I2C buszrendszert használunk. Viszont a szenzorunk már nem egy DHT11-es hanem egy MQ2-es szenzor, természetesen ahhoz, hogy működésre bírjuk bekell illesztenünk a neki megfelelő könyvtárat, mely nem más, mint az *MQ2.h.*

```
#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
#include <MQ2.h>  //Az MQ2 szenzor használatához szükséges könyvtár
#include <Wire.h> //A Wire könyvtár beillesztése az I2C busz használatához
const int Analog_Input = A0; //konstans globális integer típusú változó mely az MQ2 szenzor pin-jét tárolja
const int led = 12; //konstans globális integer típusú változó mely a LED dióda pin-jét tárolja
const int piezo = 11; //konstans globális integer típusú változó mely a Piezo elem pin-jét tárolja
int lpg, co, smoke; //globális integer típusú változók az lpg, co, és a fust értékenk a tárolására
MQ2 mq2(Analog_Input); //Az MQ2 szenzor inicializálása és az általa használt bemenet hozzárendelése
void setup()
{
  lcd.init(); //Az LCD kijelző inicializálása
  lcd.backlight(); //Az LCD kijelző háttérvilágításának bekapcsolása
  pinMode(led, OUTPUT); //A LED diódát tartalmazó Pin kimenetté alakítása
  pinMode(piezo, OUTPUT); //A Piezo elemet tartalmazó Pin kimenetté alakítása
  Serial.begin(9600); //A soros porton történő kommunikáció bitrátája
  mq2.begin(); //Az MQ2-es szenzorral való kommunikáció indítása
  lcd.clear(); //Az LCD kijelző tartalmának a törlése
}
void loop() //ciklus
{
  lcd.setCursor(0, 0); //Kurzor pozicionálás ez esetben 0. karakter a 0. sorban
  lcd.print("LPG:"); //Megadott karakterlánc kiíratása
  lcd.setCursor(6, 0); //Kurzor pozicionálás ez esetben 6. karakter a 0. sorban
  lpg = mq2.readLPG(); //Az lpg értékének kiolvasása és változóban tárolása
  lcd.print(lpg); //A változó értékének kiíratása
  lcd.setCursor(10, 0); //Kurzor pozicionálás ez esetben 10. karakter a 0. sorban
  lcd.print("CO:"); //Megadott karakterlánc kiíratása
  co = mq2.readCO(); //A co értékének kiolvasása és változóban tárolása
  lcd.setCursor(15, 0); //Kurzor pozicionálás ez esetben 15. karakter a 0. sorban
  lcd.print(co); //A változó értékének kiíratása
  lcd.setCursor(5, 1); //Kurzor pozicionálás ez esetben 5. karakter a 1. sorban
  lcd.print("FUST:"); //Megadott karakterlánc kiíratása
  smoke = mq2.readSmoke(); //A füst értékének kiolvasása és változóban tárolása
  lcd.setCursor(13, 1); //Kurzor pozicionálás ez esetben 13. karakter a 1. sorban
  lcd.print(smoke); //A változó értékének kiíratása
```

1. ábra Füst érzékelés programkód
(forrás: saját szerkesztés)

Miután a könyvtárak beillesztése megtörtént, folytathatjuk a programozás menetét, mégpedig a változók deklarálásával. Ez esetben hozzunk létre három konstans integer típusú változót melyek a három alkatrészünkre utalnak az MQ2-es szenzorra a piezo elemünkre és végül, de nem utolsó sorban a led diódánkra. Mindhárom változó értéke megfelel az Arduino lapunk egy egy pinjének a sorszámával. Az MQ2-es szenzor egy analóg bemenetre, mégpedig az A0 sorszámúra csatlakozik a led diódánk a 12-es pinre és a piezo elemünk pedig a 11-es re. Ezt követően létrehozunk még három szintén integer típusú változót melyek érétékét az MQ2-es szenzor könyvtárából fogjuk kiolvasni. A következő sorban egy típus definiálás és a szenzor által használt bemenet hozzárendelése történik meg. El is érkeztünk a programunk void setup () szekciójába. Itt a szokásos LCD kijelző inicializálás mellett szintén megkell határoznunk, hogy az Arduino Uno melyik pinjét milyen módban szeretnénk használni. Legyen az ki vagy esetleg bemenet. Mi már tudjuk, hogy a szenzorunk egy analóg bemenetet fog használni, ezért ennek a bemenetnek nem szükséges megadnunk a működési módját, hiszen alapértelmezetten csak

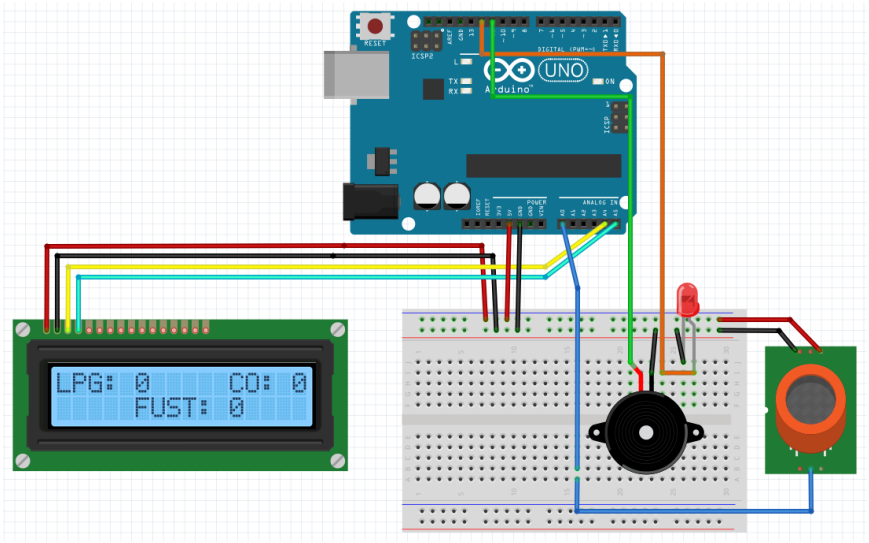
bemenetként funkcionál. Viszont a led dióda és a piezo eleme esetében a pineket már, mint kimeneteket kell használnunk. Ezt a szokásos módon a pinMode paranccsal valósítjuk meg. Ha szeretnénk információt megjeleníteni a soros porton akkor ennek megfelelően járjunk el, vagyis engedélyezzük a kommunikációt az előzőekben is használt baud értékkel. Viszont, szükséges az MQ2-es szenzorral történő kommunikáció elindítása is ezt az *mq2.begin()* funkcióval tehetjük meg. Amennyiben a jelenlegi projektet rögtön az előzőt követően építjük akkor tanácsos az lcd kijelző tartalmának a törlése, mivel nem szeretnénk, hogy az előző projekt által megjelenített karakterek a kijelzőnkön maradjanak. Ami a ciklusunkat illeti kurzorpozicionálással fogunk kezdeni. Helyezzük a mutatót az lcd kijelző bal felső sarkába, vagyis a 0-ik karakterre a 0-ik sorban, majd írjuk ki a következő karakterláncot ”LPG:”. Azt szeretnénk, hogy a három változónk lpg, co, smoke értéke egyszerre jelenjen meg a kijelzőn ne egymást követően. Ezért úgy kell pozícionálnunk a mutatónkat, hogy a megjelenített karakterlánc mellé tudjuk kiíratni a megfelelő változó értékét. Tehát, hogy ha az ”LPG:” karakterláncunk kiíratása a 0-ik karaktertől indul és a 3-ik karakterig tart, akkor a változó értékét tanácsos néhány karakterrel jobbra kiíratni, vegyük figyelembe a változó által igényel karakter mennyiséget. Ezt a már említett módon tudjuk leellenőrizni, a soros porta íratás módszerével. A ciklus 4-ik sorában egy értékátadási műveletet végzünk, ahol az lpg nevezetű globális integer típusú változó értéke megegyezik az MQ2-es szenzortól kapott LPG mennyiségre mutató értékkel. Itt egy hivatkozás segítségével történik az átadás, mégpedig *mq2.readLPG()*. Ezt a folyamatot egy kiíratás követi, ahol az lcd kijelzőre ezúttal már az lpg nevezetű változónk által tárol értéket jelenítjük meg. Az lpg változó értékének kiíratásához hasonlóan járunk el a co és a smoke változók esetében is, apró különbségekkel, mint például a kurzor pozíciójának az elhelyezése.

```
if ((smoke * 2) > 400.00) //Feltétel vizsgálat ha a füst értékének kétszerese nagyobb, mint 400.00 akkor:
{
  lcd.clear(); //Az LCD kijelző tartalmának a törlése
  lcd.setCursor(7, 0); //Kurzor pozicionálás ez esetben 7. karakter a 0. sorban
  lcd.print("FUST"); //Megadott karakterlánc kiíratása
  lcd.setCursor(4, 1); //Kurzor pozicionálás ez esetben 4. karakter a 1. sorban
  lcd.print("ESZLELVE!"); //Megadott karakterlánc kiíratása
  for (int i = 0; i < 10; i++) //for ciklus mely i = 0-tól i < 10-ig fut:
  {
    tone(piezo, 1400); //Piezo elem megszólaltatása 1400 hz frekvencián
    digitalWrite(led, HIGH); //A led dióda világít
    delay(100); //Várakozás 100 mikroszekundum ideig
    tone(piezo, 1200); //Piezo elem megszólaltatása 1200 hz frekvencián
    digitalWrite(led, LOW); //A led dióda nem világít
    delay(100); //Várakozás 100 mikroszekundum ideig
    tone(piezo, 1000); //Piezo elem megszólaltatása 1000 hz frekvencián
    digitalWrite(led, HIGH); //A led dióda világít
    delay(100); //Várakozás 100 mikroszekundum ideig
    tone(piezo, 800); //Piezo elem megszólaltatása 800 hz frekvencián
    digitalWrite(led, LOW); //A led dióda nem világít
    delay(100); //Várakozás 100 mikroszekundum ideig
    tone(piezo, 600); //Piezo elem megszólaltatása 600 hz frekvencián
    digitalWrite(led, HIGH); //A led dióda világít
    delay(100); //Várakozás 100 mikroszekundum ideig
  }
  lcd.clear(); //Az LCD kijelző tartalmának a törlése
}
else //egyébként
{
  digitalWrite(led, LOW); //A led dióda nem világít
  noTone(piezo); //A piezo elem deaktiválása
}
```

2. ábra Füst érzékelés programkód
(forrás: saját szerkesztés)

Viszont mi azt szeretnénk, hogy ha a szenzorunk füstöt észlel akkor jelezzen nekünk, hang és fény jelzéssel is egyaránt. Itt jön képbe a két további hardver melyet használunk. A füst mértékét mi szabjuk meg, tanácsos ezt az értéket letesztelni, akár egy füstölőpálcika segítségével és a szenzortól kapott jelet, pedig a soros porton megjeleníteni. Itt eldönthetjük, hogy mekkora tolerancia értéket szeretnénk megadni. A mi esetünkben ez legyen a füst mért értékének a kétszerese és ha ez az érték nagyobb, mint 400.00 egység akkor történjen a riasztás. Ezt egy egyszerű feltétel vizsgálattal tudjuk megoldani, a feltétel pedig a következőképpen néz ki *if ((smoke * 2) > 400.00)*. Ha a feltételünk igaz akkor töröljük az lcd kijelzőnk jelenlegi tartalmát és egy vészjelző üzenetet ”FÜST ÉSZLELVE!” fogunk megjeleníteni. Ezt az üzenetet középre és két sorban szeretnénk kiíratni. Ezért pozícionáljuk a kurzort a 7-ik karakterre a 0-ik sorban és ki írjuk a ”FUST” karakterláncot majd ezt követően, a kurzort az 1-es sor 4-ik karakterére helyezzük és az ”ÉSZLELVE!” üzenetet jelenítjük meg. Ezután a hang és fény jelzést hozzuk létre. Egy for ciklus segítségével megadjuk, hogy a led diódánk és a piezo elemünk jelezzen. Ez esetben ezt egy szekvencia tízszeri lefuttatásával végezzünk el. A szekvencia a piezo eleme megszólaltatásával

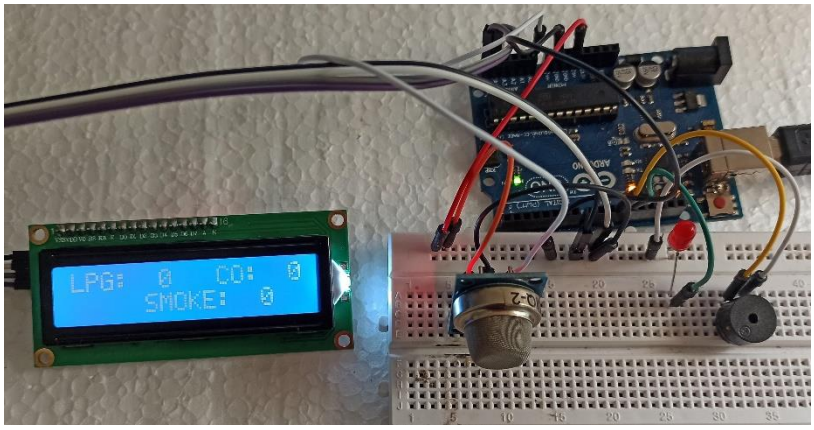
kezdődik. A frekvencia 1400hz majd ezt követi a led dióda bekapcsolása, a hang és fény jelzések közti intervallumot egy már használt *delay()* funkció segítségével mindössze, 100 milliszekundumra állítjuk ez az érték viszonylag gyors figyelmeztető hang és fény jelzést eredményez. A frekvencia értékét csökkentsük egészen 600hz-ig és minden csökkenést a led dióda ki és bekapcsolása kövesse. Ezzel vége is a ciklus magjának. Innen kilépve töröljük az lcd tartalmát. Végül a feltétel vizsgálatunk egyébként ágában meghatározzuk, hogy mi történjen akkor, ha az általunk meghatározz feltétel nem teljesül. Itt egyszerűen a led diódát és a piezo elemünket is kikapcsolt állapotba helyezzük. A led dióda esetében ezt egy LOW a piezo elem esetében pedig a noTone(piezo) parancs segítségével tudjuk megoldani.



3. ábra Füst érzékelés bekötési rajz
(forrás: saját szerkesztés)

A fenti 2.ábra lesz segítségünkre az áramkör megépítésében. Az lcd kijelző marad a megszokott pinekhez csatlakoztatva (SDA, SCL). Az MQ2-es szenzor A0-val jelzett pinje az Arduino szintén A0-val jelzett analóg bemenetére a VCC csatlakozó az Arduino állandó 5 voltot biztosító pinjére a GND pedig a közös GND ponthoz csatlakozik. A led dióda anódja és a piezo elem pozitív csatlakozója a 12-es és a 11-es pinekhez csatlakoznak. A led dióda katódja és a piezo elem negatív csatlakozója pedig a közös földhöz, vagyis a GND pinhez csatlakozik.

Az áramkör megépítését és a programkód feltöltését követően a következő látványban részesülhetünk (3.ábra).



4. ábra Füst érzékelés működés közben
(forrás: saját szerkesztés)