*Lucrare de Laborator*

**Sistem Inteligent pentru Detectarea Energiei Verzi și Măsurarea Acesteia**

2022-2023

**CUPRINS**

1. Obiectivele lucrării............................................................................ 1

2. Scurtă descriere Arduino.................................................................. 1

3. Descrierea componentelor necesare ............................................... 2

4. Descrierea lucrării............................................................................. 5

5. Montajul propriu-zis ......................................................................... 7

6. Desfășurarea lucrării......................................................................... 10

7. Rezolvarea lucrării............................................................................. 17

1. **Obiectivele lucrării**

Lucrarea de laborator are ca scop crearea unui sistem inteligent pentru detectarea energiei verzi si măsurarea acesteia.

Studenții vor începe familiarizarea cu mediul de dezvoltare Arduino, modul în care se programează mediul de dezvoltarea, cum se va lucra cu datele colectate de la senzori, reproducerea unui anemometru prin intermediul unui senzor de magneți și invățarea unor concepte de energie verde.

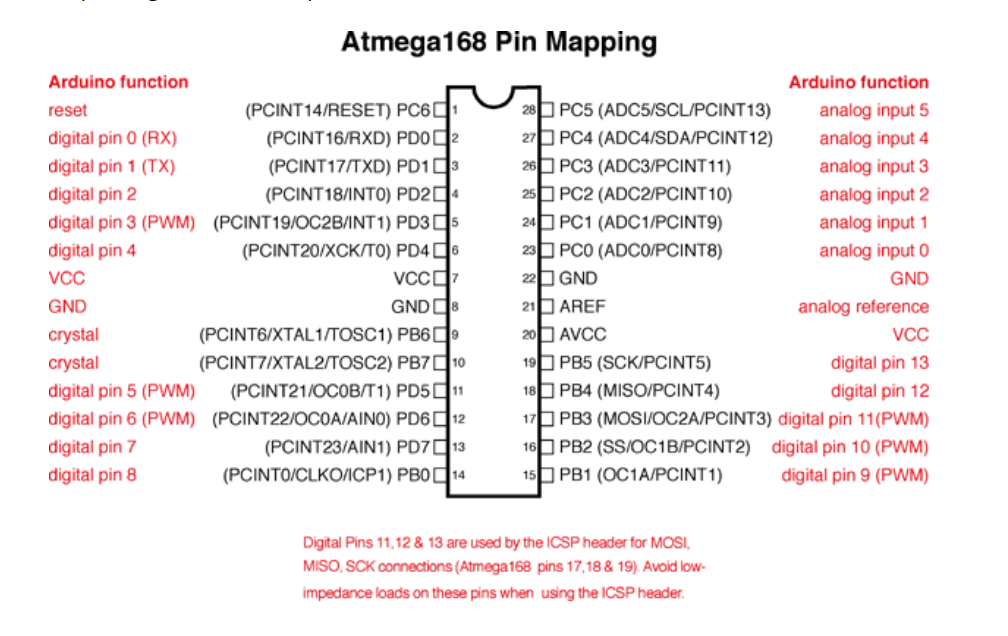
1. **Scurtă descriere a Arduino**

Arduino este o platformă open-source utilizată pentru construirea de proiecte electronice. Arduino constă atât într-o placă de circuite programabile fizic (microcontroller), cât și într-o parte de software sau pe scurt IDE (Integrated Development Environment) care rulează pe computerul dvs. Acesta este folosit pentru a scrie și încărca codul computerului pe un board fizic.

IDE-ul Arduino utilizează o versiune simplificată a C++, facilitând învățarea programării. În cele din urmă, Arduino oferă un factor de formă standard care elimină funcțiile microcontrolerului într-un pachet mai accesibil. UNO este una dintre cele mai populare plăci din familia Arduino și reprezintă o alegere excelentă pentru începători.



Descriere modulului intern de pe placa Arduino:



**Porturile de pe placă:**

• B (pini digital de la 8 la 13);

• C (pini analogici de intrare);

• D (pini digitali de la 0 la 7).

Fiecare port este controlat de 3 regiștrii (aceștia sunt cei definiți în limbajul Arduino): DDR, PORT și PIN.

Registrul DDR determină tipul pinului: intrare/ ieșire.

Registrul PORT controlează starea pinului: high sau low și citește, totodată, starea de input setată pinului (echivalentul din Arduino pentru pinMode() ).

Regiștrii DDR și PORT pot fi amândoi scriși și citiți.

Regiștrii PIN corespund stării de input, pot fi doar citiți.

Fiecare bit din acești regiștrii corespunde unui singur pin.

Există 3 porturi:

• PORTD (DDRD, PORTD, PIND)

• PORTB (DDRB, PORTB, PINB)

• PORTC (DDRC, PORTC, PINC)

**3. Descrierea componentelor necesare**

*-Placă de dezvoltare Arduino UNO + cablu USB de tip A-B;*

**Placă de dezvoltare Arduino UNO**



**Caracteristici tehnice:**

* Tensiune de funcționare: 5V;
* Tensiune de alimentare Jack: 7V - 12V;
* Pini de I/O: 14;
* Pini PWM: 6 (din cei 14 de I/O);
* Pini ADC: 8;
* Memorie flash: 32kB (8 ocupați de bootloader);
* Comunicație TWI, SPI și UART;
* Frecvență de funcționare: 16 MHz.

## *-Senzor de lumină Adafruit ALS-PT19;*

**ALS-PT19**



**Caracteristici tehnice:**

* Tensiune de alimentare: 2,5 V până la 5,5 V;
* Iesire analogică;
* Răspuns rapid pentru spectrul ochiului uman;
* Raport ridicat de suprimare a radiației infraroșii;
* Performanta la temperatura garantata, -40C pănă la 85C;
* Variație mică a sensitivității pentru diverse surse de lumină;
* Liniaritate bună a iesirii pe o gamă largă de iluminare;
* Dimensiuni: 7,8 x 10,6 x 2,4 mm;
* Greutate: 0,2 g.

## *-Senzor magnetic;*

## HALL

## 

**Caracteristici tehnice:**

* Măsurare precisă de câmp magnetic;
* Consum mic de energie;
* Măsurare 3D, liniară, unghiulară și de rotație;
* Gamă de temperatură versatilă.

*-LCD;*

# **LCD 1602**



**Caracteristici tehnice:**

* Tensiune de alimentare: 5V;
* Curent: 1.1mA;
* Tensiune de alimentare backlight: 4.2V;
* Curent backlight: 100mA;
* Dimensiuni: 80mm x 36mm x 12mm.

**4. Descrierea lucrării**

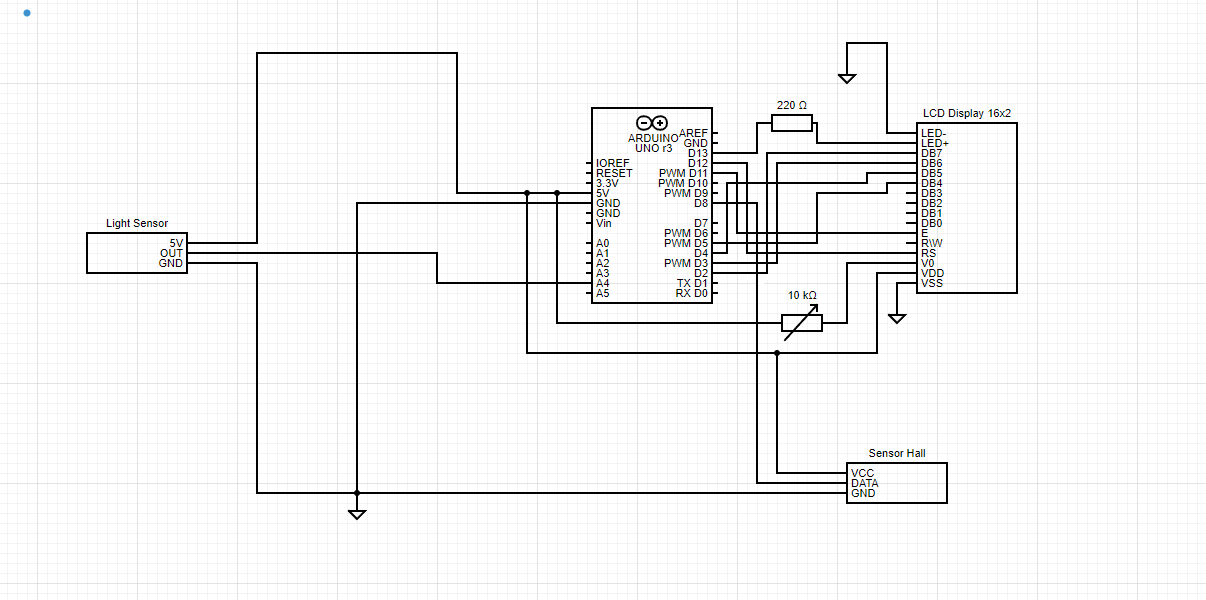
Se va crea un sistem având ca unitate de calcul un modul Arduino UNO , la care se vor atașa, folosind o placă de experiment (breadboard), un senzor de lumină, un magnet lipit pe o “aripă” a unui fidget spinner , un senzor de magnet, și un display LCD pentru afișare datelor preluate de la senzori și a diferitelor mesaje adiționale.

Sistemul o să stea intr-o stare de așteptare, în care display-ul LCD este oprit pentru a economisi energie. Dacă senzorul de lumină și anemometrul nu sunt activi, LCD-ul va afișa un mesaj care va preciza că nu există energie verde disponibilă. LCD-ul se mai poate activa dacă senzorii transmit date. În acest caz pe LCD se va afișa un mesaj care precizează că energia verde este existentă și intensitatea acesteia ( intensitatea luminii).

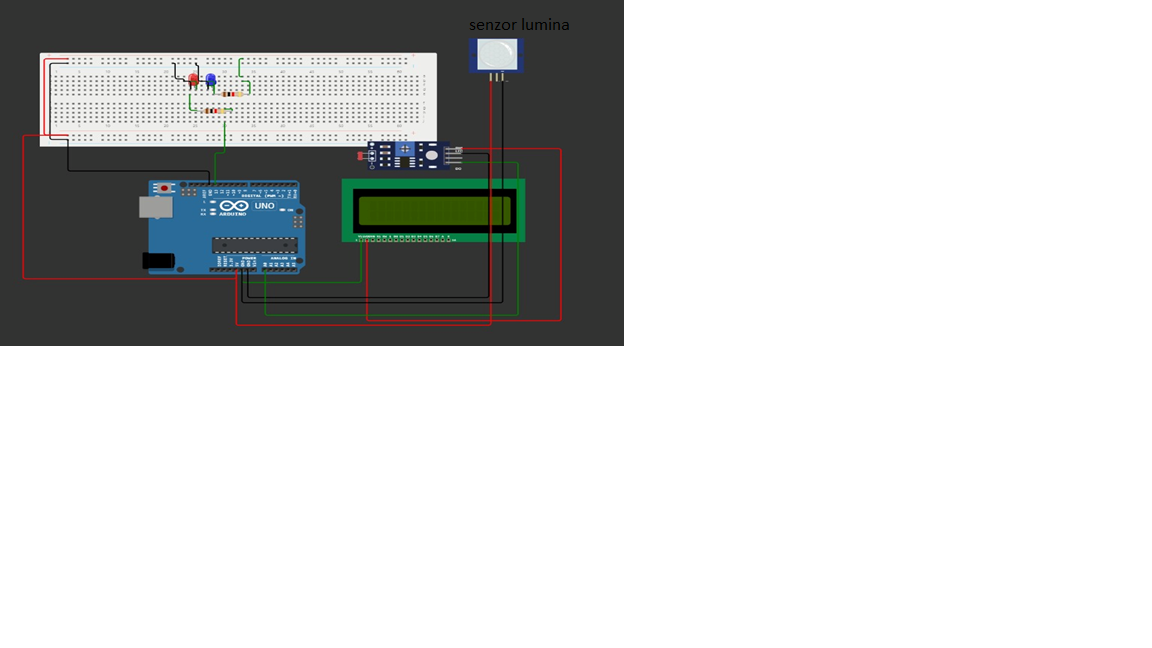
Anemometrul va funcționa pe baza de magnet lipit pe un fidget spinner și senzor de magnet. Se va considera într-o stare inactivă dacă senzorul se află in aceeași stare de mai mult de 10 secunde, oricare ar fi aceasta.

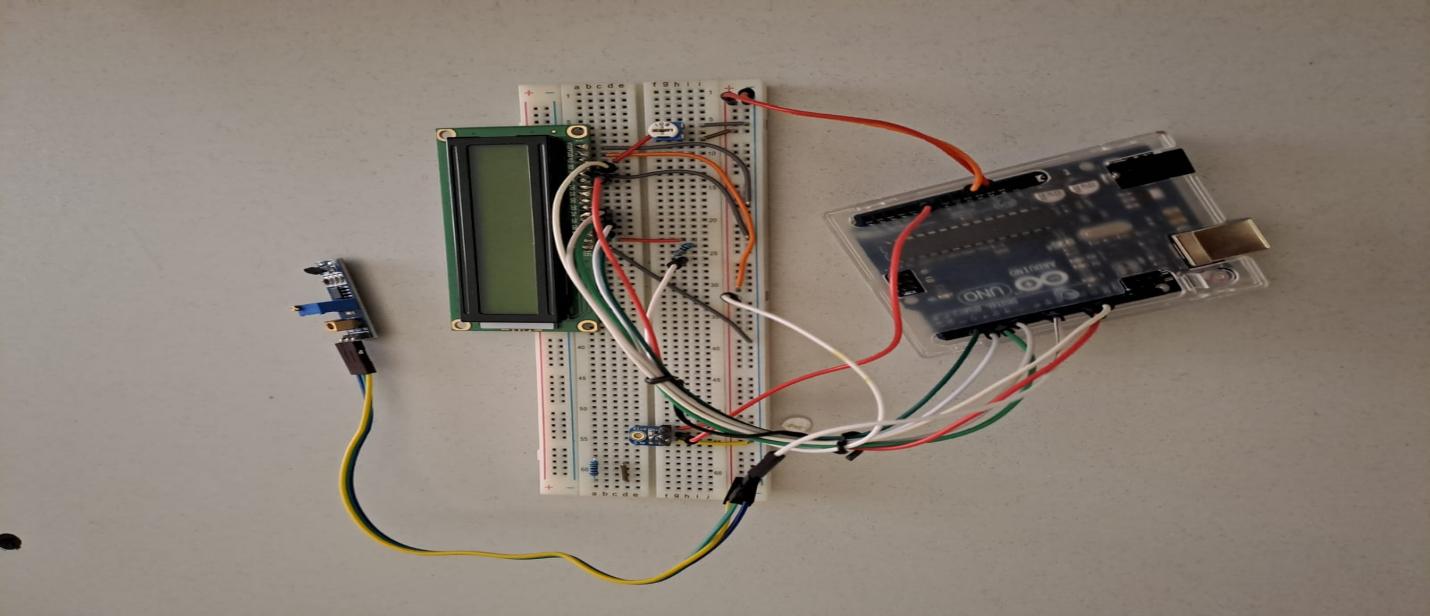
Fiind un sistem care se bazează doar pe doi senzori si un afișaj LCD, complexitatea pentru a realiza circuitul este mică, ceea ce reprezinta un plus la capitolul de familiarizare cu sistemul de dezvoltare Arduino fiind important ca orice student care nu a mai folosit Arduino să înțeleagă atât ușurința de dezvoltare cât și capacitățile acestuia.

*Schema electrică:*



*Schema montajului:*





**5. Montajul propriu-zis**

Pașii necesari pentru crearea montajului:

a)Placuța Arduino la breadboard;



* Pinul de +5V și GND se leagă la breadboard;

b) Senzor de lumina;

* Pinul de +5V și GND se leagă la breadboard;

c)Senzor magnetic;

e) *LCD*



* Pinul VDD se leagă la +5V la breadboard;
* Pinul VO se leagă la pinul din mijloc al potențiometrului;
* Pinul RS se leagă la pinul 8 al placuței Arduino;
* Pinul RW se leagă la pinul GND la breadboard;
* Pinul E se leagă la pinul 9 al plăcuței Arduino;
* Pinul D4 se leagă la pinul 10 al plăcuței Arduino;
* Pinul D5 se leagă la pinul 11 al plăcuței Arduino;
* Pinul D6 se leagă la pinul 12 al plăcuței Arduino;
* Pinul D7 se leagă la pinul 13 al plăcuței Arduino;
* Pinul A se leagă la +5V la breadboard;

f) *Montajul realizat:*

**6. Desfășurarea lucrării**

Pentru învățarea corectă și pentru a înțelege fiecare componentă în parte, am hotarât să împărțim această lucrare în 4 laboratoare:

1. Senzorul de lumina;

2. Senzor magnetic;

3. LCD;

4. Conectarea tuturor componentelor.

1. **Senzorul de lumină:**

Componente necesare:

-Placuta Arduino;

-Breadboard;

-Senzor de lumină (ALS PI 19);

-Fire;

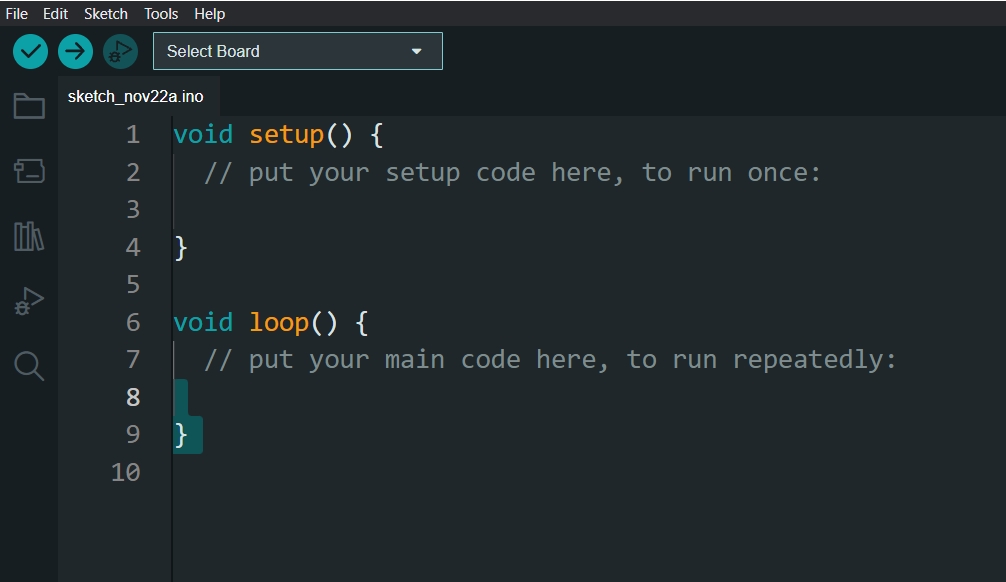
Schema electrică

**Introduce codul de mai jos:**

|  |  |
| --- | --- |
| void setup() { |  |
|  |  |
|  |  | Serial.begin(9600); |
|  |  | } |
|  |  |  |
|  |  | void loop() { |
|  |  | // reads the input on analog pin A0 (value between 0 and 1023) |
|  |  | int analogValue = analogRead(A0); //trimitem date prin pinu AO |
|  |  | **//verificați valorile afisate în terminal** |
|  |  | Serial.print("Analog reading = "); |
|  |  | Serial.print(analogValue); // the raw analog reading |
|  |  |  |
|  |  | // We'll have a few threshholds, qualitatively determined |
|  |  | if (analogValue < **/\*valoarea dark\*/)** { |
|  |  | Serial.println(" - Dark"); |
|  |  | } |
|  |  | else if (analogValue < **/\*valoare dim\*/** ) { |
|  |  | Serial.println(" - Dim"); |
|  |  | } |
|  |  | else if (analogValue < **/\*valaore light\*/** ) { |
|  |  | Serial.println(" - Light"); |
|  |  | } |
|  |  | else if (analogValue < **/\*valoare bright\*/** ) { |
|  |  | Serial.println(" - Bright"); |
|  |  | } |
|  |  | else { |
|  |  | Serial.println(" - Very Bright"); |
|  |  | } |
|  |  |  |
|  |  | delay(500); |
|  |  |  |

Arduino IDE:

Vedem în partea de sus “Select Board”, selectăm placută Arduino UNO:

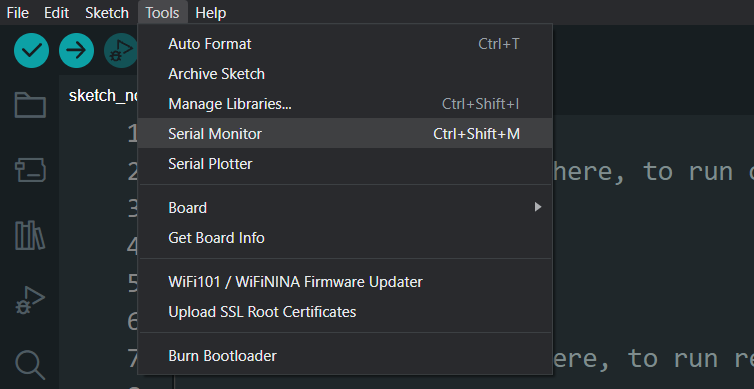


Apăsăm pe upload;



Dacă nu ne dă erori , putem continua.

În partea de sus , în secțiunea „tools”, găsim “Serial monitor”;



O să se deschidă jos o parte de terminal unde o să ne afiseze valorile citite (=intensitatea luminii). Cerința este de a nota valori pentru diferite stări (dark, roomlight, light,bright și very bright).

(Sfat, pentru dark: acoperim cu degetul senzorul de lumină si notam valoarea citita de la senzorul de lumina; la fel, light este valoarea dintr-o camera luminată moderat si pentru valorea very bright putem aprinde lanterna de la telefon și citim valoarea. Pentru roomlight si bright, ne luam intervale aflate între dark și light, respective light si very bright)

1. **Senzor magnetic**

Componente necesare:

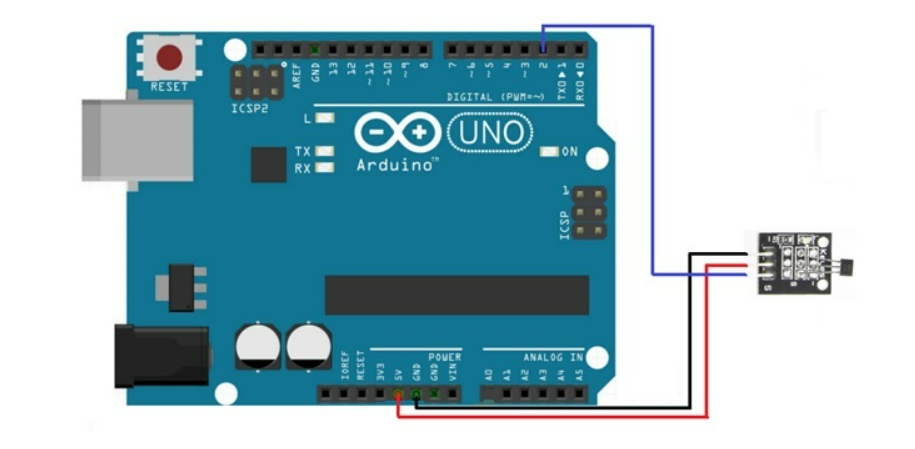
-Placuta arduino

-Breadboard

-Senzor magnetic

-Fire

Urmărim pasii de la laboratorul 1 și realizăm circuitul.



**După care realizăm cerințele evidențiate in cod:**

const int hallPin = A1 ; // initializing a pin for the sensor output

const int ledPin = 13 ; // initializing a pin for the led. Arduino has built in led attached to pin 13

// variables will change :

int hallState = 0 ; // initializing a variable for storing the status of the hall sensor.

void setup ( )

**// Setarea pinului la care este conectat ledul ca pin de iesire**

**// Inițiere pin de senzor ca pin de intrare în Arduino :**

Serial.begin( 9600 ) ;

}

void loop ( )

**// Citire senzorului și stocarea stării efectului hall**

**// Verificarea dacă starea modulului este ridicată sau scăzută (high or low)**

**// Aprinde LED-ul dacă starea modulului este ridicată (high)**

}

else {

**// stingere led**

}

}

1. **LCD**

Componente necesare:

-Placuta arduino

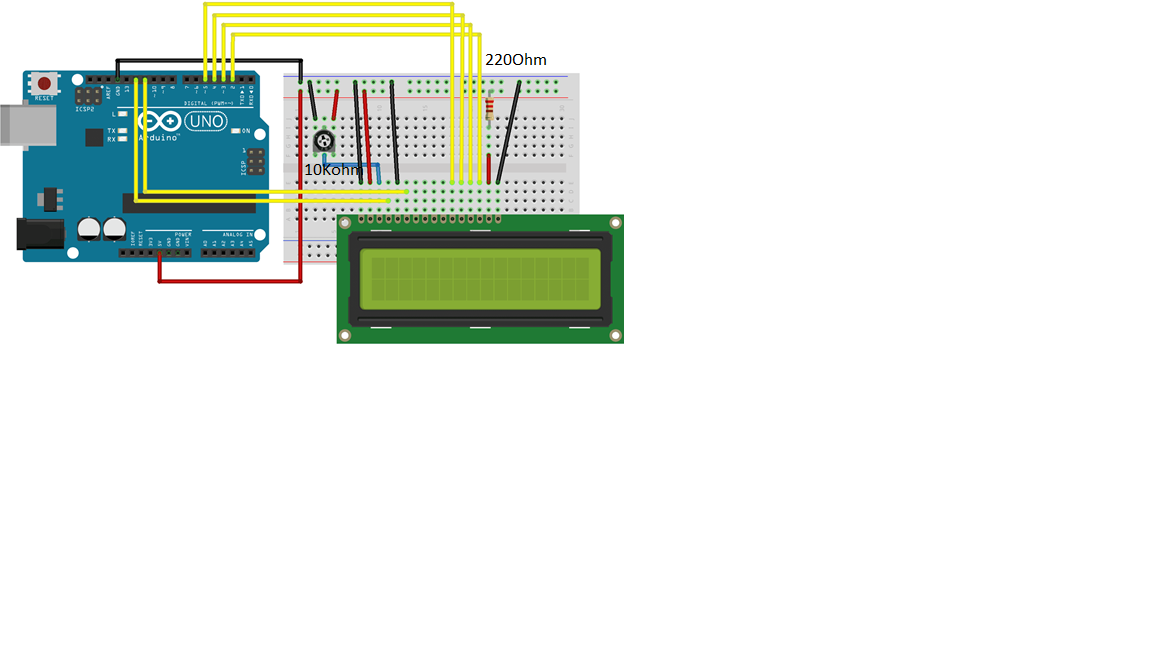
-Breadboard

-LCD

-Rezitență

-Fire

Urmărim pasii de la laboratorul 1 și realizăm circuitul.



După care am realizat cerințele evidențiate în cod

// include the library code:

#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin

// with the arduino pin number it is connected to

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {

**// setați numărul de coloane și rânduri al ecranului LCD**

**// Imprimați un mesaj pe LCD.**

}

void loop() {

**// setați cursorul pe coloana 0, linia 1**

**// (notă: linia 1 este al doilea rând, deoarece numărarea începe cu 0): // print the number of seconds since reset**

lcd.print(millis() / 1000);

}

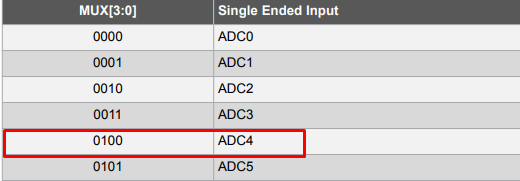
1. **Descrierea registrilor folositi**

**Registrul ADMUX** = 0x44 - 0b01000100

Bitii 7:6 – seteaza referinta de tensiune pentru convertorul analog-digital

cu valoarea maxima la care poate ajunge semnalul analogic citit.

Bitii 3:0 – valoarea acestor biti seteaza intrarile de tip analog care sunt conectate la convertorul analog-digital



Pentru configuratia 0100 – se foloseste canalul A4

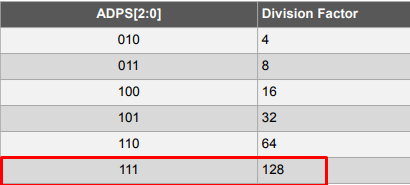
**Registrul ADCSRA** = 0xFA – 0b10101111

Bitul 7 - Valoarea folosita 1 activeaza convertorul analog-digital

Bitul 5 – Valoarea folosita 1 activeaza Auto Triggering al convertorului analog-digital

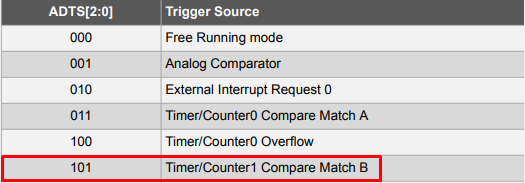
Bitul 3 – Valoarea folosita 1 activeaza Conversion Complete Interrupt

Bitii 2:0 – se folosesc pentru a seta frecventa de ceas intre 50kHz si 200kHz pentru a obtine o precizie mai mare



**Registrul ADCSRB** = 0x05 – 0b0101

Configuratia de biti folosita implica citirea semnalului analogic sa se realizeze atunci cand numaratorul 1 trece prin valoarea comparatorului B



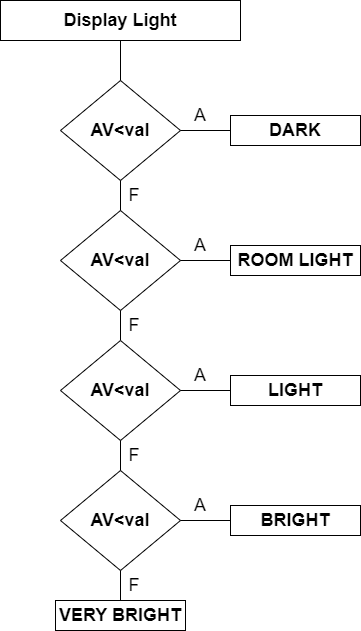
**Registrul TCCR1B** = 0x03 – 0b0011

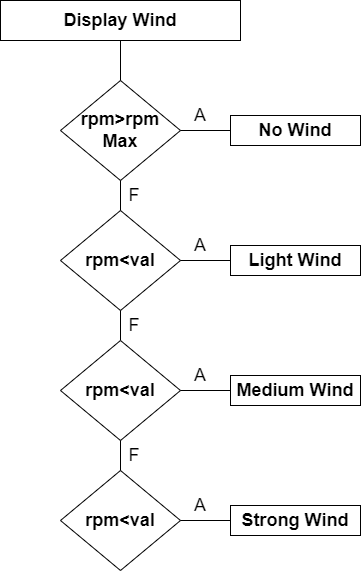
Valoarea folosita indica faptul ca numaratorul functioneaza in mod normal (prescaler de 256)

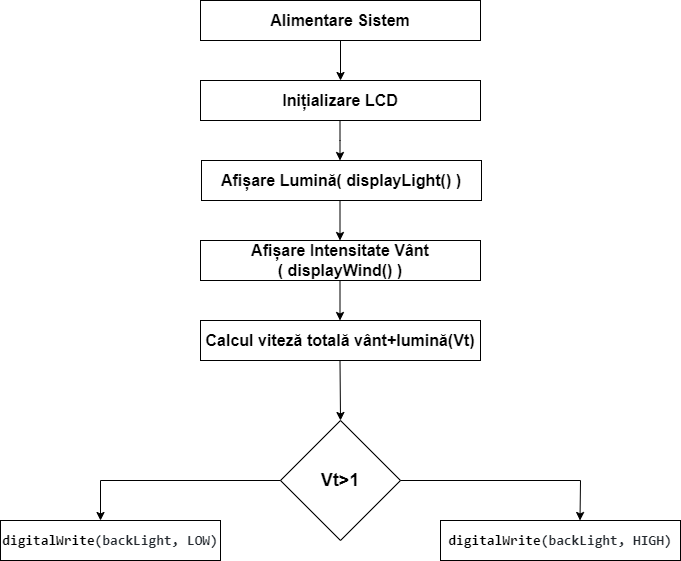
**Registrul TIMSK1** = 0x04 – 0b0100

Configuratia folosita intrerupe comparatorul B al timerului 1

1. **Diagrame software**







1. **Pașii lucrarii**

Îmbinăm codurile de la LCD, senzorul de lumină și senzorul magnetic, pentru a obține codul final.

1. **Rezolvarea lucrării**

**Codul final:**

// include the library code:

#include <LiquidCrystal.h>

/////////////////////////////////////////////////////////////////////////////////////////////////////

// initialize the library by associating any needed LCD interface pin

// with the arduino pin number it is connected to

const int

backLight = 13,

rs = 12,

en = 11,

d4 = 5,

d5 = 4,

d6 = 3,

d7 = 2,

hallPin = A1 ; // initializing a pin for the sensor output

/////////////////////////////////////////////////////////////////////////////////////////////////////

int V1 = 0, V2= 0, Vt= 0;

/////////////////////////////////////////////////////////////////////////////////////////////////////

// const for RPM and magent

const int hallSensorPin = 8; // connect the hall effect sensor on pin 2

const unsigned long sampleTime = 1000;

const int maxRPM = 1260; // maximum RPM for LCD Bar

int rpmMaximum = 0;

/////////////////////////////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////////////////////////////

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

/////////////////////////////////////////////////////////////////////////////////////////////////////

void setup() {

// This will initialize the hall effect sensor pin as an input pin to the Arduino

pinMode(hallSensorPin,INPUT);

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

lcd.clear();

//Economy mode

pinMode(backLight, OUTPUT);

//Start

digitalWrite(backLight, HIGH);

lcd.print("Initializing");

delay(1000);

lcd.clear();

//Serial

Serial.begin(9600);

}

void loop() {

delay(500);

// int rpm = getRPM();

// if (rpm > rpmMaximum) rpmMaximum = rpm;

// if (rpm > 100) {

// digitalWrite(backLight, HIGH);

// }

// else

// {

// digitalWrite(backLight, LOW);

// }

//lcd.clear();

//displayBar(rpm);

// lcd.setCursor(0, 0);

//displayRPM(rpm);

displayWind();

displayLight();

lcd.setCursor(12, 0);

Vt= V1+V2;

if (Vt < 1)

{

digitalWrite(backLight, LOW);

Serial.println(Vt);

lcd.print("P:");

lcd.print(Vt);

}

else

{

digitalWrite(backLight, HIGH);

Serial.println(Vt);

lcd.print("P:");

lcd.print(Vt);

}

}

void displayLight()

{

// lightcode.c

// reads the input on analog pin A0 (value between 0 and 1023)

int analogValue = analogRead(A0);

Serial.print("Analog reading = ");

Serial.print(analogValue); // the raw analog reading

lcd.setCursor(0, 0);

// We'll have a few threshholds, qualitatively determined

if (analogValue < 50) {

Serial.println(" Dark");

lcd.print("Dark");

V2=0;

}

else if (analogValue < 150) {

Serial.println(" Room light");

lcd.print("Room light");

V2=1;

}

else if (analogValue < 250) {

Serial.println(" Light");

lcd.print("Light");

V2=1;

}

else if (analogValue < 450) {

Serial.println(" Bright");

lcd.print("Bright");

V2=2;

}

else {

Serial.println(" Very Bright");

lcd.print("Very Bright");

V2=3;

}

}

int getRPM()

{

int count = 0;

boolean countFlag = LOW;

unsigned long currentTime = 0;

unsigned long startTime = millis();

while (currentTime <= sampleTime)

{

if (digitalRead(hallSensorPin) == HIGH)

{

countFlag = HIGH;

}

if (digitalRead(hallSensorPin) == LOW && countFlag == HIGH)

{

count++;

countFlag=LOW;

}

currentTime = millis() - startTime;

}

int countRpm = int(60000/float(sampleTime))\*count;

return countRpm;

}

void displayRPM(int rpm)

{

// lcd.clear();

lcd.setCursor(0, 1);

lcd.print(rpm,DEC);

lcd.setCursor(7,1);

lcd.print(rpmMaximum, DEC);

lcd.setCursor(13,1);

lcd.print("MAX");

Serial.print("RPM = ");

Serial.print(rpm);

Serial.print(" MAX RPM = ");

Serial.println(rpmMaximum);

}

void displayBar(int rpm)

{

int numOfBars=map(rpm,0,maxRPM,0,15);

lcd.setCursor(0,1);

if (rpm!=0)

{

for (int i=0; i<=numOfBars; i++)

{

lcd.setCursor(i,1);

lcd.write(1023);

}

}

}

void displayWind()

{

int rpm = getRPM();

lcd.setCursor(0, 1);

if (rpm > rpmMaximum) rpmMaximum = rpm;

if (rpm < 100) {

lcd.clear();

lcd.setCursor(0, 1);

lcd.print("No Wind");

// lcd.setCursor(13,1);

// lcd.print(rpm,DEC);

V1=0;

}

else if (rpm < 200) {

lcd.clear();

lcd.setCursor(0, 1);

digitalWrite(backLight, HIGH);

lcd.print("Light Wind");

// lcd.setCursor(13,1);

// lcd.print(rpm,DEC);

V1=1;

}

else if (rpm < 300) {

lcd.clear();

lcd.setCursor(0, 1);

digitalWrite(backLight, HIGH);

lcd.print("Medium Wind");

// lcd.setCursor(13,1);

// lcd.print(rpm,DEC);

V1=2;

}

else if (rpm < 400) {

lcd.clear();

lcd.setCursor(0, 1);

digitalWrite(backLight, HIGH);

lcd.print("Strong Wind");

// lcd.setCursor(13,1);

// lcd.print(rpm,DEC);

V1=2;

}

else {

lcd.clear();

lcd.setCursor(0, 1);

digitalWrite(backLight, HIGH);

lcd.print("V.Strong Wind");

// lcd.setCursor(13,1);

// lcd.print(rpm,DEC);

V1=3;

}

}

**Soluții la LCD:**

// include the library code:

#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin

// with the arduino pin number it is connected to

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

// Print a message to the LCD.

lcd.print("hello, world!");

}

void loop() {

// set the cursor to column 0, line 1

// (note: line 1 is the second row, since counting begins with 0):

lcd.setCursor(0, 1);

// print the number of seconds since reset:

lcd.print(millis() / 1000);

}

**Soluții la Lightcode:**

void setup() {

Serial.begin(9600);

}

void loop() {

// reads the input on analog pin A0 (value between 0 and 1023)

int analogValue = analogRead(A0);

Serial.print("Analog reading = ");

Serial.print(analogValue); // the raw analog reading

// We'll have a few threshholds, qualitatively determined

if (analogValue < 10) {

Serial.println(" - Dark");

}

else if (analogValue < 40) {

Serial.println(" - Dim");

}

else if (analogValue < 150) {

Serial.println(" - Light");

}

else if (analogValue < 450) {

Serial.println(" - Bright");

}

else {

Serial.println(" - Very Bright");

}

delay(500);

}

**Soluții la Magnetcode:**

#include <LiquidCrystal.h>

const int hallSensorPin = 8; // connect the hall effect sensor on pin 2

const unsigned long sampleTime = 1000;

const int maxRPM = 1260; // maximum RPM for LCD Bar

int rpmMaximum = 0;

const int

backLight = 13,

rs = 12,

en = 11,

d4 = 5,

d5 = 4,

d6 = 3,

d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {

pinMode(hallSensorPin,INPUT);

lcd.begin(16, 2);

lcd.print("Initializing");

delay(1000);

lcd.clear();

}

void loop()

{

delay(100);

int rpm = getRPM();

if (rpm > rpmMaximum) rpmMaximum = rpm;

lcd.clear();

displayRPM(rpm);

displayBar(rpm);

}

int getRPM()

{

int count = 0;

boolean countFlag = LOW;

unsigned long currentTime = 0;

unsigned long startTime = millis();

while (currentTime <= sampleTime)

{

if (digitalRead(hallSensorPin) == HIGH)

{

countFlag = HIGH;

}

if (digitalRead(hallSensorPin) == LOW && countFlag == HIGH)

{

count++;

countFlag=LOW;

}

currentTime = millis() - startTime;

}

int countRpm = int(60000/float(sampleTime))\*count;

return countRpm;

}

void displayRPM(int rpm)

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(rpm,DEC);

lcd.setCursor(7,0);

lcd.print(rpmMaximum, DEC);

lcd.setCursor(13,0);

lcd.print("MAX");

Serial.print("RPM = ");

Serial.print(rpm);

Serial.print(" MAX RPM = ");

Serial.println(rpmMaximum);

}

void displayBar(int rpm)

{

int numOfBars=map(rpm,0,maxRPM,0,15);

lcd.setCursor(0,1);

if (rpm!=0)

{

for (int i=0; i<=numOfBars; i++)

{

lcd.setCursor(i,1);

lcd.write(1023);

}

}

}

**Proiect realizat de studenții următori, de la Facultatea de Inginerie Electrică și Știința Calculatoarelor, Profilul Calculatoare, anul IV, an universitare 2022-2023:**

* **Eduard Olteanu**
* **Teleki Ferenc-Tibor**
* **Momoi Alexandru**
* **Morel Irina**
* **Tudorache Robert Cosmin**
* **Niculae Florin**
* **Gheorghe Luca-Teodor**