Cuprins

I. Prefata

Am scris aceasta carte din dorinta de a nota toate proiectele realizate pana acum pe care le puteti face si voi la randul vostru. De asemenea am incercat sa o fac cat mai simplista cu putinta pentru a intelege si procesa cat mai usor informatiile.

Urmatoarele informatii sunt doar niste concepte de baza din cele doua domenii(electronica si programare), asadar nu am intrat foarte mult in detalii amanuntine.

Din punctul meu de vedere daca parcurgeti aceasta carte veti avea cunostintele de baza din cele doua ramuri si puteti alege care din ele vi se potriveste mai bine. Multe persoane incearca sa se lamureasca pe ei insisi ce vor de fapt sa faca in viata, partea fizica sau programare. Ei bine Arduino le poate face pe amandoua. Nici eu nu stiam sigur la inceputul drumului ce imi place mai mult, codarea sau fizica, asa ca am inceput sa observ la ce sunt bun eu de fapt. Am realizat ca partea fizica a acestor proiecte ma intriga mai mult decat a scrie programul in sine.

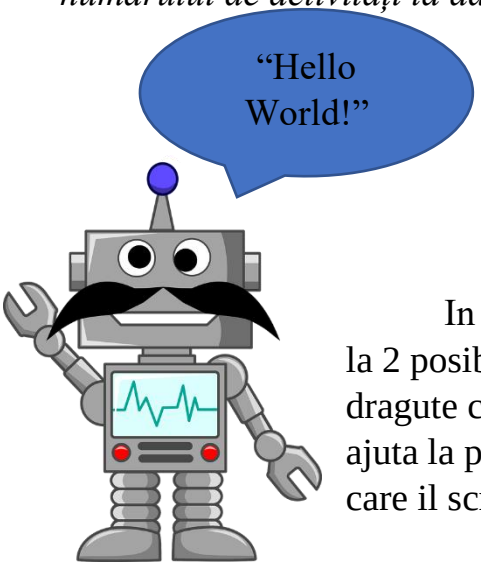
Codurile din acesta carte sunt hibride, in sensul ca sunt luate de pe internet sau carti, dar modificate de mine intr-o oarecare masura. Nu spun ca nu stiu coda, dar nici ca stiu. Eu sunt omul de mijloc. Aparent nu trebuie sa fii excelent la informatica sau matematica pentru a face astfel de proiecte, ci trebuie sa fii cel mai bun la a cauta rezolvarea problemelor pe care le vei intampina in realizarea proiectelor. Daca ai cunostinte minime in electrotehnica si programare poti face mereu ceva ce sa te ajute sa iesi in evidenta.

Chiar daca sunt proiecte simple, pe care le poti gasi oriunde la un magazin bazat pe device-uri, satisfactia pe care ti-o da faptul ca tu ai realizat un dispozitiv este una de neichipuit. Eu unul mereu dupa ce termin un proiect ma simt fericit si satisfacut ca am mai facut un pas inainte in industria tehnologiei si sper ca intr-o buna zi sa fiu unul dintre cei mai buni electronisti din lume si sa gasesc si sa ajut oamenii de rand cat mai mult cu putinta, sa le usurez viata.

Este foarte important ca atunci cand nu intelegi un anumit aspect intr-un domeniu sa nu dai vina pe cel care a scris articolul, pentru ca el si-a dat silinta sa o scrie astfel incat te faca sa procesezi corect informatia nu doar de dragul de a fi scris. De aceea exista in zilele noastre motoare de cautare, poti gasi orice raspuns in mai putin de 5 secunde daca stii sa cauti cum si ce trebuie.

să fie împărtășite deschis, fără scop comercial. Aceste activități nu pot fi folosite în scop comercial sau fără atribuirea sursei.

Ne puteți trimite feedback, sugestii sau recomandări pentru extinderea numărului de activități la adresa robnad2000@gmail.com



II. Introducere

A. Ce este Arduino?

In momentul in care auzim cuvantul “arduino” ne gandim la 2 posibilitati. Una fiind niste placute de dezvoltare foarte dragute care au in componenta lor niste microcontrolare care ajuta la programarea acestora, iar cealalta parte fiind soft-ul pe care il scriem pentru a coda placile sa faca ce dorim noi.

In urmatoarele capitole vom parcurge ambele ramuri in care voi incerca sa va explic cat mai bine cu putinta ce inseamna fiecare domeniu(hardware si software). Aceasta carte o sa va ajute sa intelegeti mai bine termenii de electronica pentru ca vor fi proiecte in care vom folosi si aparate de masura si anumite componente care nu sunt neaparat senzori sau parti componente pe care le vom programa, dar si termeni de programare.

B. De ce sa inveti Arduino?

- In zilele noastre tot ce ne inconjoara sunt dispositive inteligente care ne ajuta in viata de zi cu zi. Nu ar fi rau ca sa intelegem intr-o oarecare masura cum functioneaza acestea.
- Poti sa faci tot felul de proiecte educative care iti vor pune mintea si creativitatea la contributie pentru a te dezvolta pe tine.
- Daca nu stii ce te pasioneaza mai mult, programarea sau electronica, poti incepe prin a face proiecte Arduino pentru a vedea pe ce ramura te incadrezi mai bine.

- Ce e frumos la Arduino este ca nu exista limite… cu cat esti mai creativ cu atat se complica si proiectul si te pune la incercari cat mai multe, asta inseamna sa te dezvolti. Te va ajuta foarte mult sa iesi in evidenta in domeniul tau. Mai pot fi multe de spus de ce sa inveti Arduino, dar nu le voi insira pe toate acum.

C. Instalarea softului

Instalarea acestui program este simpla. In prima faza accesam link-ul urmator: https://www.arduino.cc/en/software. Apoi alegem soft-ul pentru sistemul nostru de operare. Descarcarea urmatoare va fii pentru “Windows 7 and newer” spre exemplu. Apasam click stanga pe “Windows Win 7 and newer”:

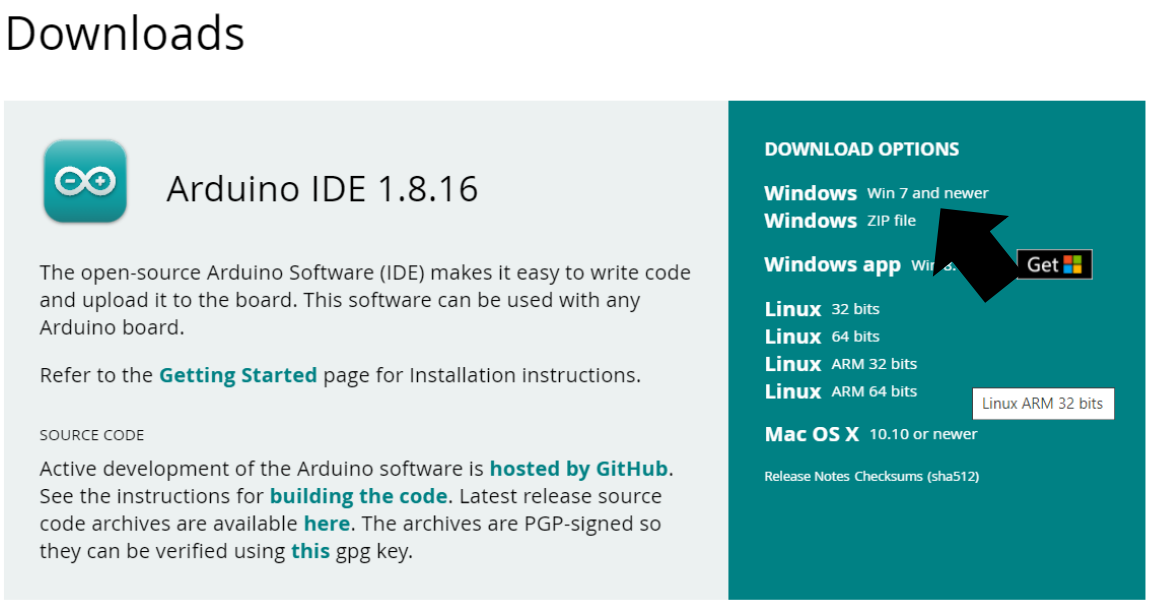
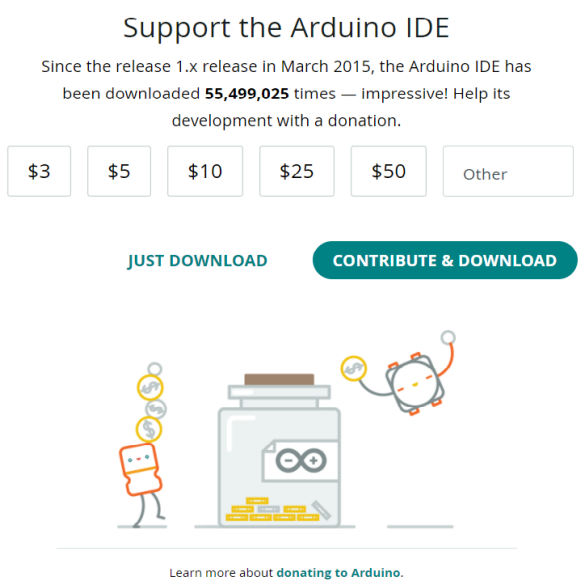


Fig. 1.1

Se va deschide apoi aceasta pagina:

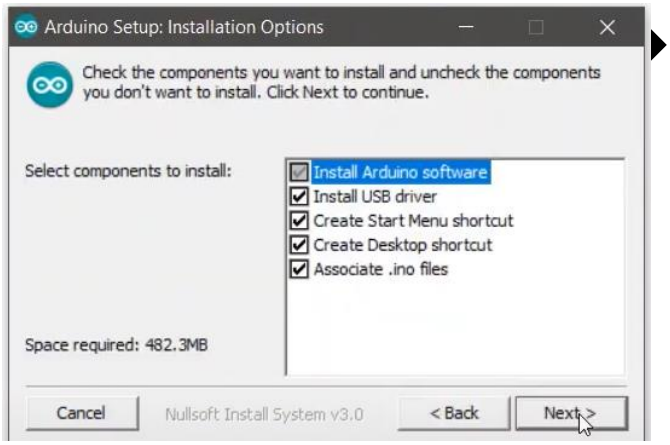


Accesam butonul “Just Download” pentru a incepe descarcarea, sau puteti da click pe “Contribute & Download” pentru a face o mica donatie comunitatii Arduino.

Fig 1.2

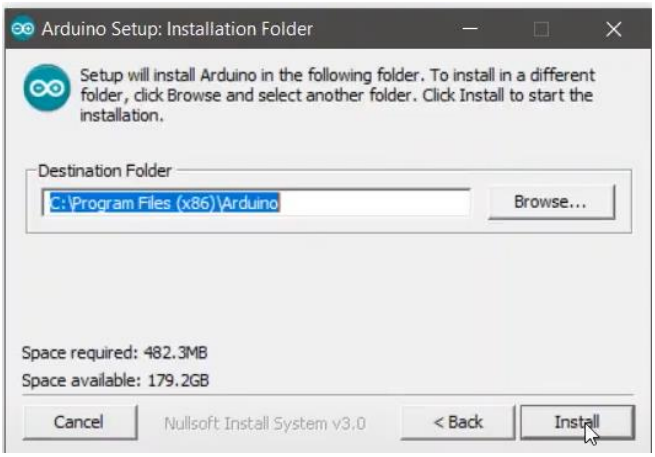
Dupa ce instalarea este completa in browser, deschidem programul si acceptam termenii si conditiile, butonul “I Agree”.

Fig. 1.3



Click stanga pe “Next” dupa ce am selectat toate casutele.

Fig. 1.4



Alegem locatia unde dorim sa instalam soft-ul, iar apoi apasam “Next”.

Fig. 1.5

Dupa finalizarea instalarii deschidem “Arduino” din PC si se va deschide urmatoarea fereastra.

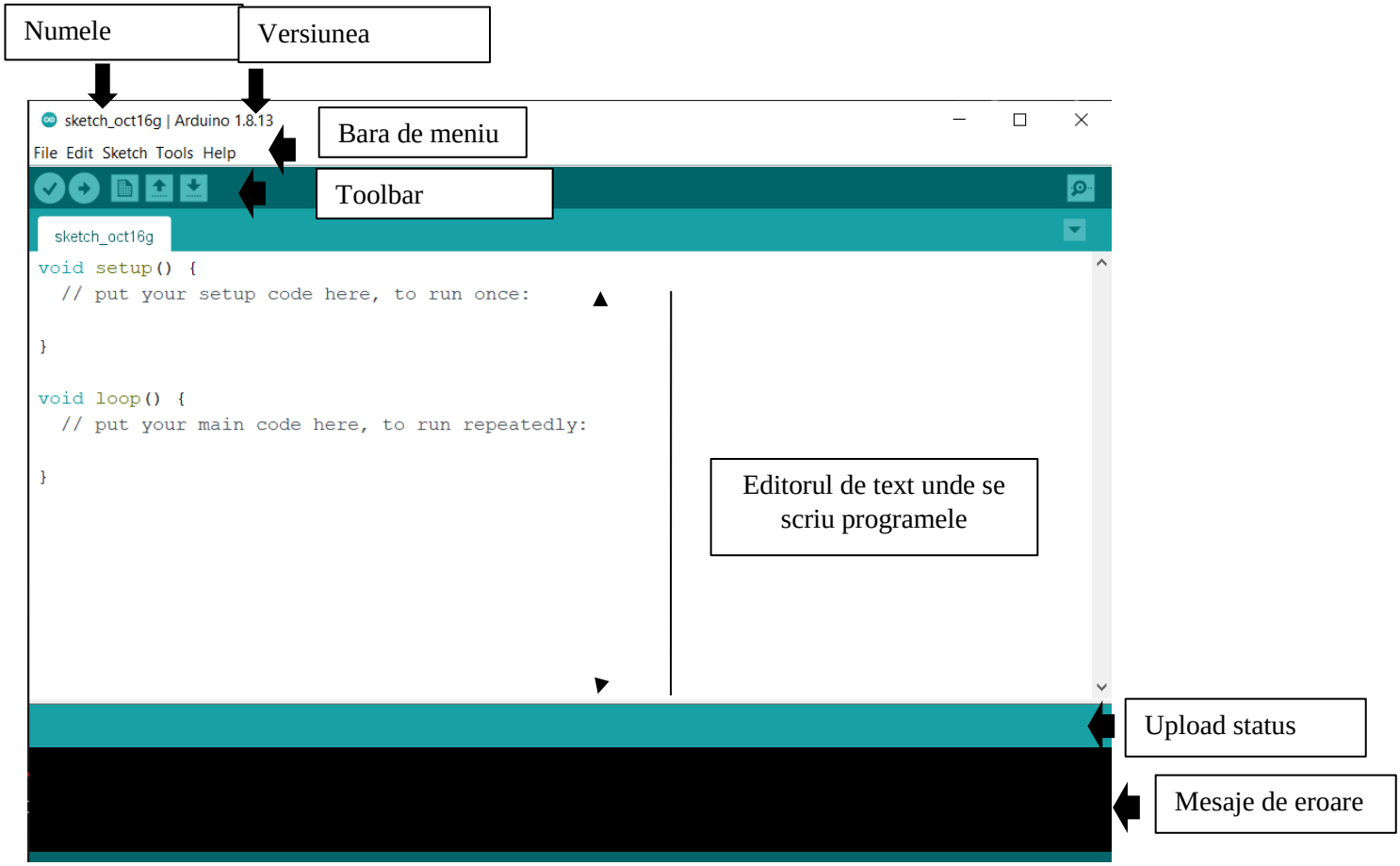


Fig. 1.6

D. Structura unui program Arduino

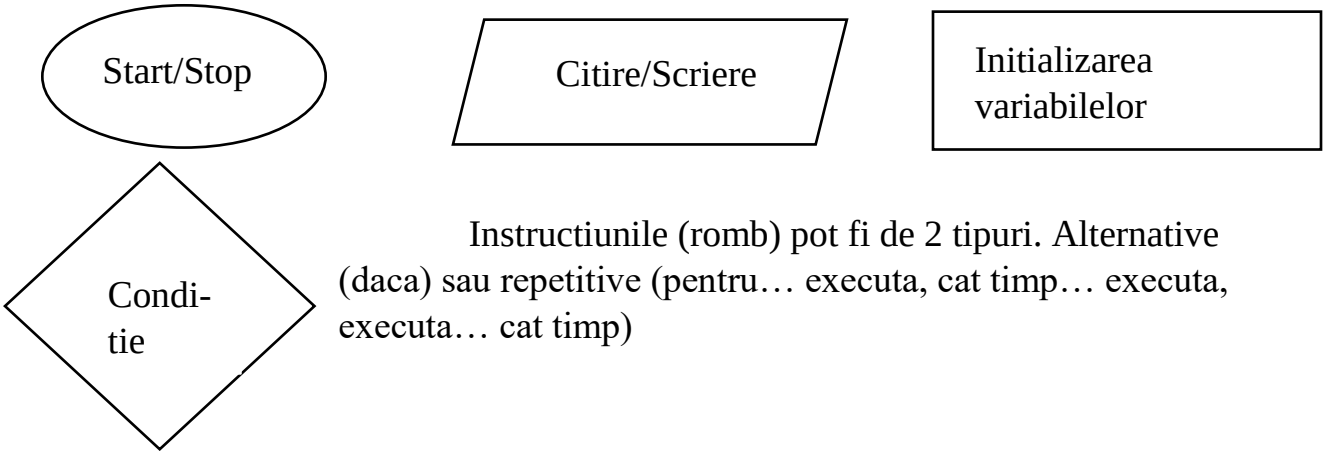
Mereu cand vom deschide programul vor fi afisate in editorul de text doua secvente, “void setup()” si “void loop()”. Sectiunea de “setup” se va executa doar o singura data atunci cand placa este alimentata sau se apasa butonul de reset al placii specific, iar cea de “loop” se va executa incontinuu pana cand placii ii va fi scoasa alimentarea sau i se va descarca bateria.

III. Scheme logice si pesudocod

Inainte de a incepe sa lucram in Arduino, as prefera sa facem mici exercitii de codare in pseudocod. Pseudocodul este o scriere naturala de sintaxe, ce ajuta la

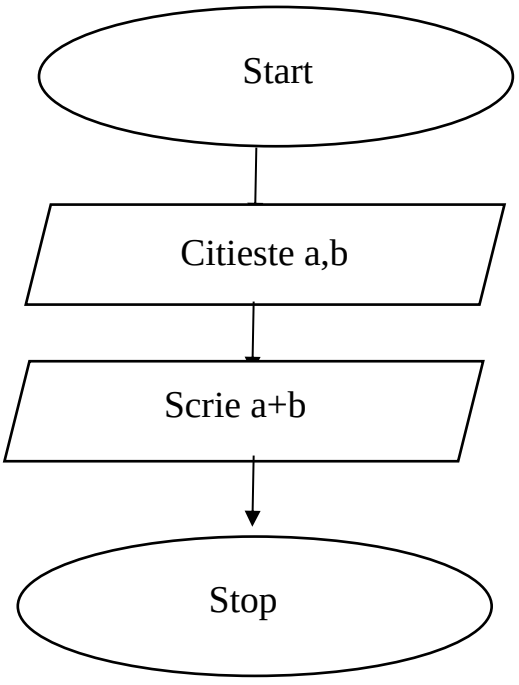
intelegerea, simplificarea si realizarea claritatii scrierii algorimtilor intr-un timp scurt. Schemele logice ajuta si mai mult la simplificarea si vizualizarea problemelor. Ele sunt reprezentate prin figuri geometrice.

Sintaxele utilizate:



Instructiunile (romb) pot fi de 2 tipuri. Alternative (daca) sau repetitive (pentru… executa, cat timp… executa, executa… cat timp)

Schema logica:



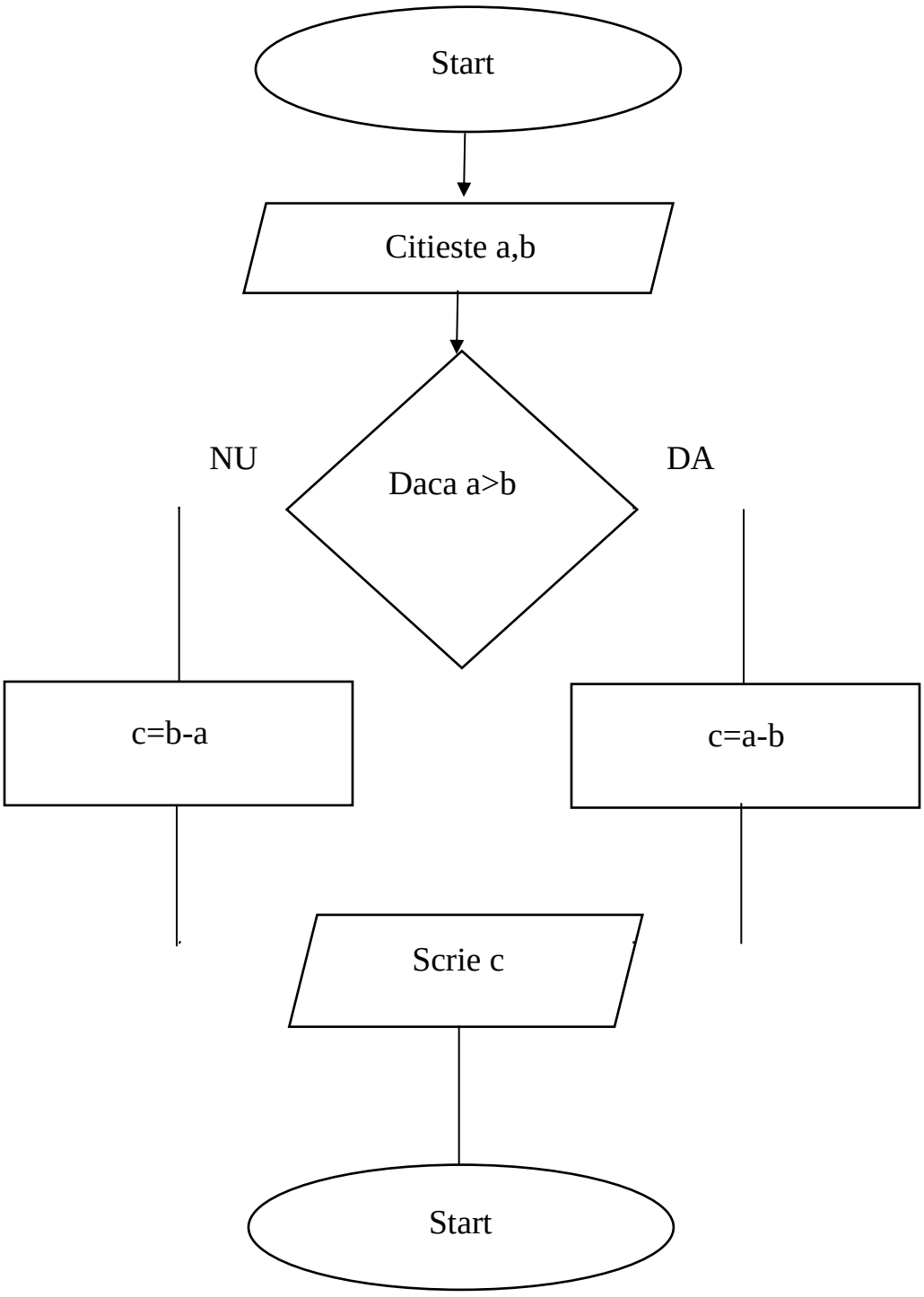
Pseudocod:

```
start
    citeste a,b
scrie a+b
stop
```

Practic codul nostru face suma a 2 numere citite de la tastatura si o afiseaza pe ecran.

Vom complica putin lucrurile utilizand o instructiune de tip alternativ.

Schema logica:



Pseudocod:

start

citeste a,b //nr. de la tastatura

daca (a>b) atunci c = a-b // daca conditia e adevarata face diferenta a-b

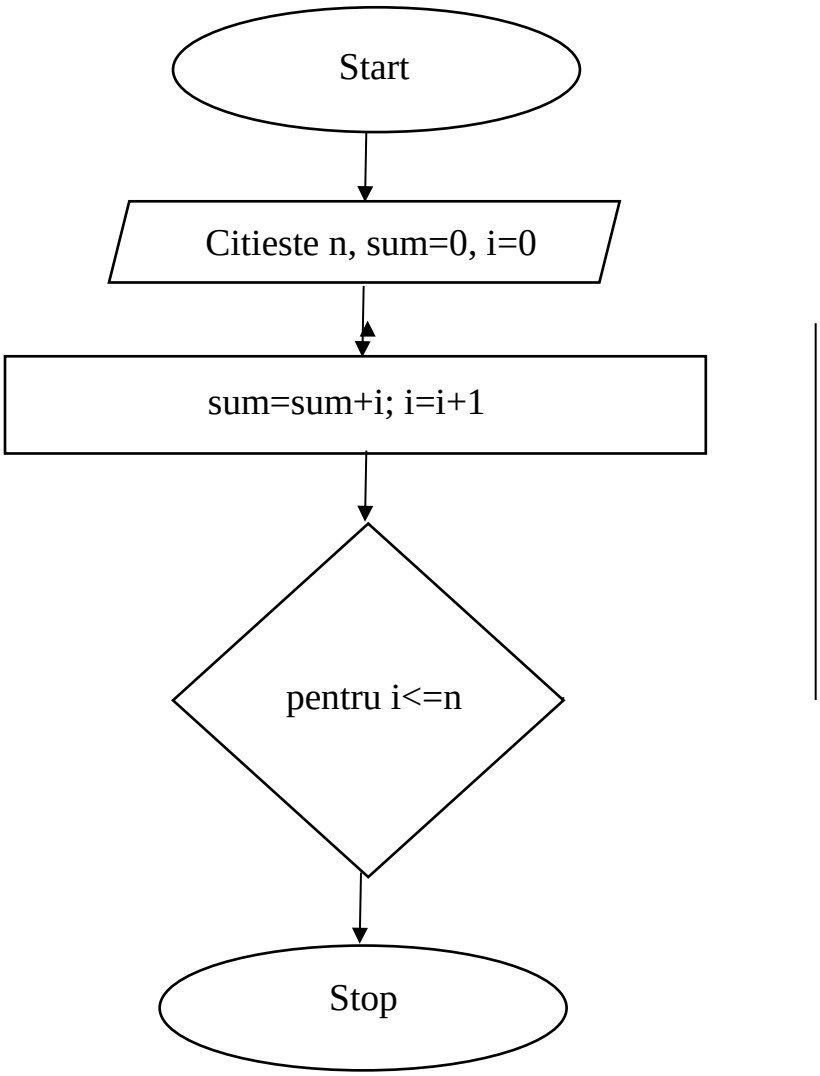
altfel c = b-a // daca conditia este falsa se face diferenta b-a

scrie c //scriem c

stop

Programul urmator este de a citi un nr. de la tastatura si apoi se fac suma primilor n numere.

Schema logica:



Pseudocod:

start

citeste n, sum=0, i=1 //se citesc variabilele

pentru (i=1, i<=n) executa {sum=sum+i; //cat timp conditia este adevarata se

bucla va continua sa se execute

i=i+1}// pentru variabila i=1, i<=n se executa cele 2 calcule

stop

Pentru… executa, cum putem observa in codul de mai sus conditia este ca “i” sa fie egal cu “1” si tot i sa fie “<=n”. In prima faza “i” este “1” si este mai mic decat numarul introdus de noi(n), asa ca se va executa “sum=sum+i”, astfel “sum” devine “1”, iar apoi se face “i=i+1”, iar “i” devine “2”. Apoi din nou se va face verificarea daca “i<=n”, iar daca este adevarat se vor executa din nou cele 2 calcule. Bucla va continua astfel pana cand conditia “i<=n” nu mai este adevarata.

Pentru instructiunile repetitive de tip “executa…cat timp” si “cat timp…executa” schema logica se face identic ca si “pentru…executa”, si are aproximativ acelasi principiu de functionare.

Acum te invit si pe tine sa faci niste scheme logice, iar apoi sa scrii pseudocodul lor.

1. O schema logica + pseudocod in care sa ai 3 instructiuni “daca…executa”;
2. O schema logica + pseudocod in care sa citesti 2 numere a si b de la tastatura si sa le inversezi.(a ia valoarea b si b ia valoarea a).
3. O schema logica + pseudocod in care sa faci suma a primilor 100 de numere folosind o instructiune repetitiva “cat timp… executa”;
4. O schema logica + pseudocod in care sa citesti un nr. n de la tastatura, iar apoi sa faci suma primilor n termeni folosind instructiunea repetitiva “executa… cat timp”;
5. O schema logica + pseudocod in care sa calculez delta unei functii de gradul al 2-lea.
6. O schema logica + pseudocod in care sa imi afisezi pe ecran primii 50 de termeni naturali nenuli;
7. O schema logica + pseudocod in care sa vezi daca un numar este divizibil cu 2, iar daca este adevarat sa se afiseze “Numarul este par”, iar daca conditia este falsa sa se afiseze “Numarul este impar”.
8. O schema logica + pseudocod in care sa calculati media aritmetica a 3 numere citite de la tastatura.

9. O schema logica + pseudocod in care sa cititi doua numere de la tastatura, a si b, iar apoi “daca (a > b)” atunci se va face diferenta si impartirea lor, iar daca conditia este falsa se va face adunarea si inultirea acestora.
10. Te-as ruga sa te gandesti acum tu singur la o schema logica, iar mai apoi sa ii scrii pseudocodul. Pot fi de exemplu instructiuni si din viata de zi cu zi nu neaparat sa apartina de matematica. Mult success!

IV. Algoritmi

A. Ce este un algoritm?

Un algoritm este o succesiune logica de instructiuni necesare pentru rezolvarea unei probleme oarecare. Desi algoritmul este un termen matematic, in zilele noastre este folosit in informatica.

B. Stiati ca?

Primul programator a fost de fapt o femeie numita Ada Lovelace, în secolul al XIX-lea.

Cu ajutorul ei astazi algoritmii sunt baza principala a fiecarui program. Practic tot ce ne inconjoara in zilele de astazi sunt de fapt dispozitive “high tech” care functioneaza cu ajutorul algorimilor.

C. Variabile in Arduino

Ca absolut orice alt limbaj de programare si limbajul Arduino foloseste de asemenea variabile asemanatoare cu cele din C/C++.

Structura de baza: tipul_variabilei numele = valoare;

Tipuri de variabile

- **int** provine de la integer(intreg), si poate lua valori in multimea numerelor intregi cuprinse intre -2^{31} pana la 2^{31};
- **float/double** iau valori in multimea numerelor reale;
- **char/string** pastreaza caracter/e din tabelul ASCII;

- **bool/boolean** iau valori logice true(1) sau false(0);
- **void** este utilizat doar la declaratiile unei functii care nu returneaza nici o informatie la functia de la care a fost apelata;
- **long** este o variabila folosita la stocarea numerelor de dimensiuni extinse si stocheaza 32 biti(4 octeti), de la -2.147.483.648 până la 2.147.483.647.
- **byte** un bit stocheaza un nr. nesemnat pe 8 biti, de la 0 la 255.

Mai sunt si altele desigur, dar acestea sunt cele mai des intalnite in programare.

D. Instructiuni

a. Instructiunea alternativa (“if”)

Sintaxa:

```
if(conditie){ executie 1 ;}
        else if{ executie 2 ; }
                else{ executie 3 ;} …
```

- Se verifica conditia, daca este “True” se va executa “executie 1”;
- Daca este fals merge pe ramura pe “else if” , daca este “True” se executa “executie 2” (daca exista);
- Daca si ramura de “else if” este False, iar daca este “True” se va executa “executie 3”(daca exista), etc;

b.Instructiunea de decizie multipla (“switch”)

Sintaxa:

```
switch(variabila){
        case val_1: expresie1; break;
        case val_2: expresie2; break;
        case val_3: expresie3; break;
```

…

case val_n: expresien; break;

default: expresie n+1; break;}

- Se compara valoarea variabilei “variabila” cu val_1, daca sunt egale se executa expresie 1 si break;
- Se compara valoarea variabilei “variabila” cu val_2, daca sunt egale se executa expresie 2 si break;
- Se compara valoarea variabilei “variabila” cu val_3, daca sunt egale se executa expresie 3 si break;
- Daca nici o valoare “variabila” nu exte gasita in instructiunea switch atunci se executa cazul “default”.

c. Instructiuni repetitive

❖ While(cat timp… executa)

Sintaxa:

while(conditie){

instructiuni;}

- cat timp conditia este adevarata se vor executa instructiunile.

❖ Do…while(executa…cat timp)

Sintaxa:

do{

instructiuni;

}while(conditie);

- Se va executa “instructiuni”, iar mai apoi se verifica “conditie”;
- Daca “conditie” este “True” atunci se va repeta “instructiuni”, daca este “False” atunci se iese din bucla.

- ❖ For(pentru…executa)

Sintaxa:

```
for(int i=val_1; i<=val_2; i= i + val_3){
        instructiuni;}
```

- o Initializam i cu “val_1”;
- o Comparam i cu “val_2”,daca este “False” se sfarseste bucla, daca este “True” se reia executia “instructiuni”.

E. Declararea vectorilor

Vectorul este o multime compusa de forma:

```
tip vector[x] = {a, b, c, …};
```

Unde:

“tip” – tipul variabilei;

“vector” – numele vectorului;

“x” – numarul de elemente;

“a, b, c…” – elementele.

F. Declararea matricelor

O matrice este o multime compusa din mai multe mutimi de forma:

```
tip matrice[x][y] = {{a,b}, {c,d}…};
```

Unde:

“tip” – tipul matricei;

“matrice” – numele matricei;

“x” – nr. de multimi din matrice;

“y” – nr. de elemente din fiecare multime;

“{a,b},{c,d}” – reprezinta multimile din matrice.

G. Declararea String-urilor

Un string este o multime formata strict din caractere de forma:

String string = “”;

Unde:

“String” – tipul string;

“string” – numele string-ului.

V. Semnale de pe placa Arduino

Sunt 3 tipuri de semnale pe care placa Arduino le poate utiliza.

A. Semnal digital

Un astfel de semnal este usor de utilizat datorita faptului ca el poate avea doar 2 valori, HIGH(1) si LOW(0). Pentru a scrie sau citi valori in semnale digitale in Arduino se scriu urmatoarele secvente “digitalWrite(pin, valoare)” pentru scriere si “digitalRead(pin, valoare)” pentru citire. Pe placa Arduino toti pinii digitali vor avea initiala “D” (ex. D1, D2, D3, etc).

Semnalul digital arata astfel:

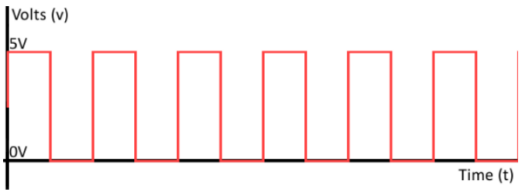


Fig. 1.7

B. Semnal analogic

In schimb un semnal analogic atunci cand dorim sa scriem in el sintaxa este “analogWrite(pin, valoare)” si valorile sunt cuprinse intre LOW(0) si HIGH(255), iar daca dorim sa citim valori se va scrie sintaxa “analogRead(pin, valoare)”, iar intervalul este cuprins intre 0 si 1023. Pinii analogici au initiala “A” (A0, A1, etc).

Semnalul analogica poate arata astfel:

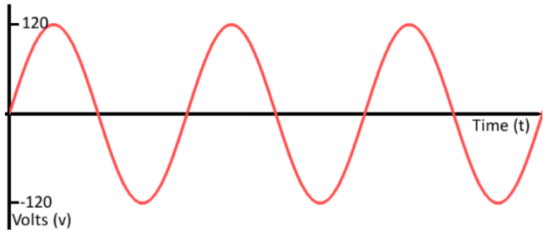


Fig. 1.8

C. Semnal PWM(“Pulse Width Modulation”)

Frecventa semnalului PWM pe majoritatea pinilor este de 490Hz. Pe placile Arduino Uno si similar, pinii 5 si 6 au o frecventa de aproximativ 980Hz. Functia PWM este disponibila pe pinii 3, 5, 6, 9, 10, 11.

Semnalul PWM arata astfel:

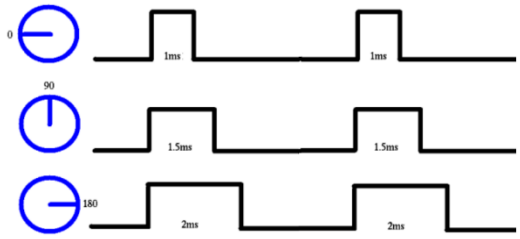


Fig. 1.9

Placa de dezvoltare are incorporat un convertor analogic-digital, care imi transforma tensiunea analogica in valoare digitala.

Acest tip de dispozitiv nu are incorporat un convertor digital-analogic, insa poate modula latimea pulsului(PWM) un semnal digital pentru a realiza unele dintre functiile unei iesiri analogice. Functia utilizata pentru a emite un semnal PWM este “analogWrite(pin, valoare)”.

Pentru a mapa o valoare analogica, care variaza intre 0 si 1023 la un semnal de iesire PWM, care variaza de la 0 la 255, putem utiliza functia map(val, L, H ,L H). Aceasta functie are 5 parametrii, unul este variabila in care este stocata valoarea analogica in timp ce celelalte sunt 0, 1023 si 0, 255.

VI. Placa Arduino

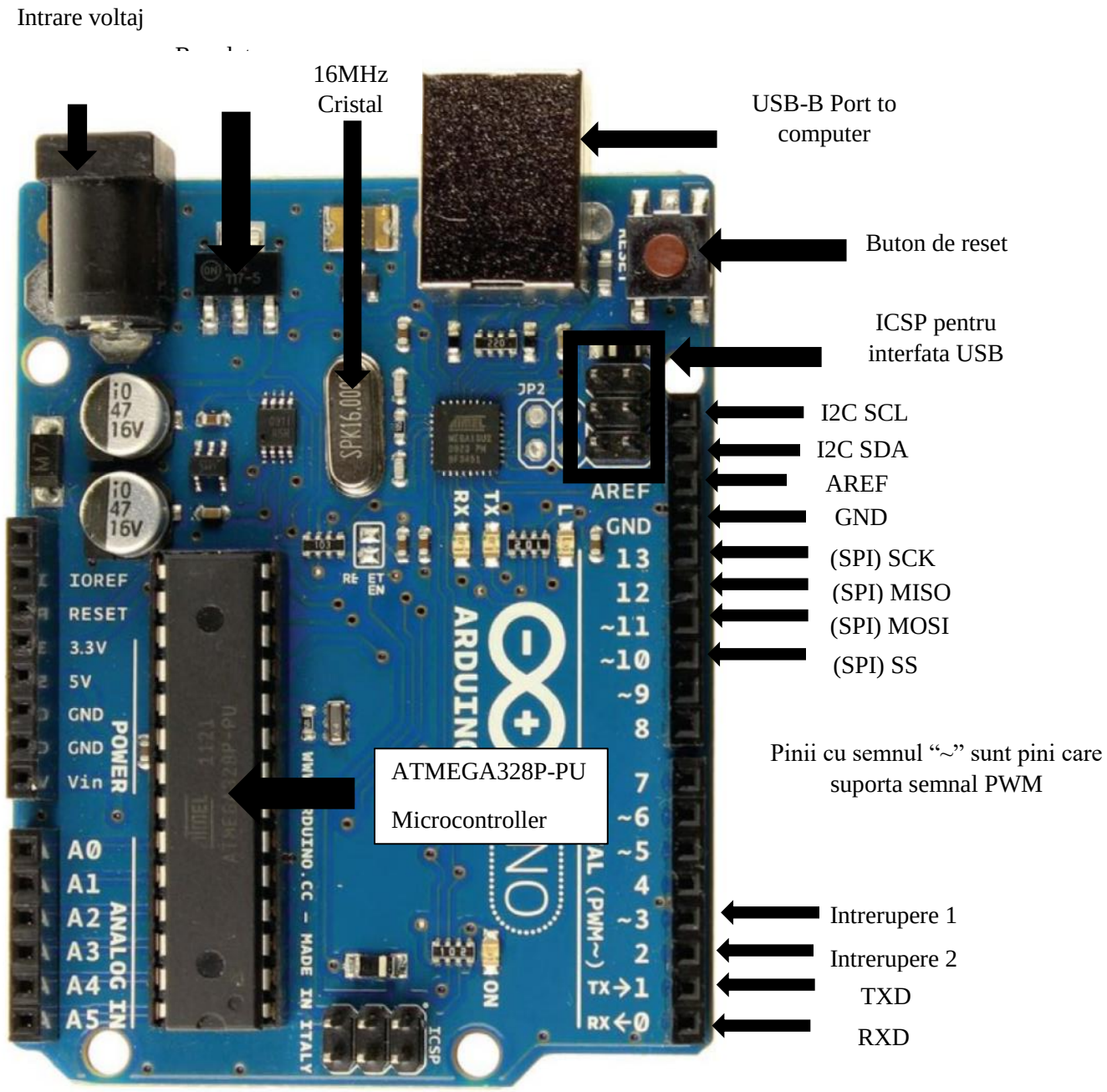


Fig. 1.10 Arduino UNO(1)

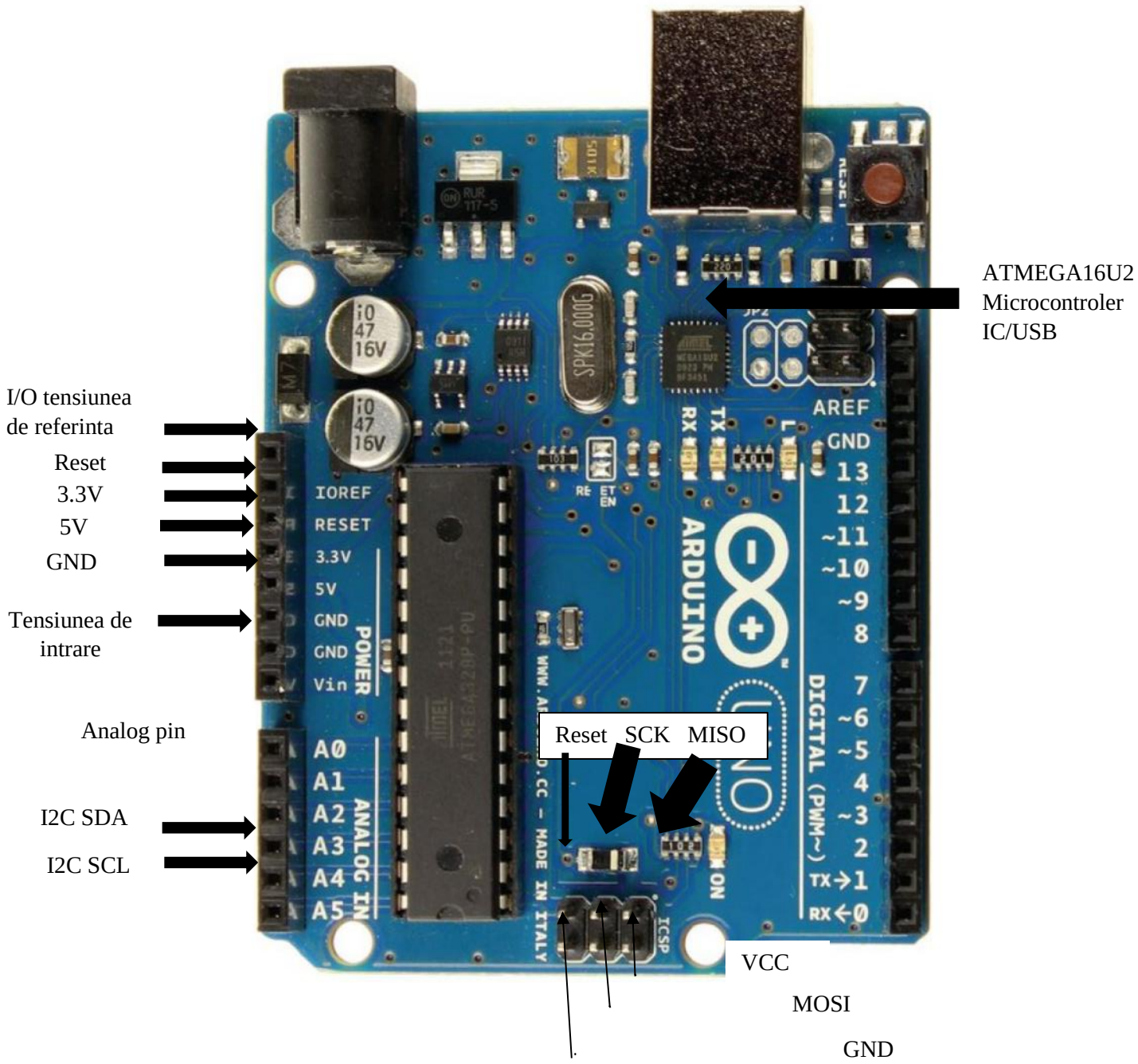


Fig. 1.11 Arduino UNO (2)

Semnificatia pinilor:

- Vin(voltage input) - Acest pin este utilizat pentru alimentarea placii Arduino Uno folosind o sursa de alimentare externa. Tensiunea trebuie să se incadreze între 7-12V.
- 3.3V si 5V - Acestea furnizeaza regulat 5 și 3.3v pentru alimentarea componentelor externe in conformitate cu specificatiile producatorului.
- GND(ground) - Pinii GND sunt utilizati pentru a inchide circuitul electric si pentru a oferi un nivel comun de referinta logica in intregul circuit.
- RESET – Reseteaza Arduino.
- IOREF - Acest pin este referinta de intrare/ieșire. Ofera referinta de tensiune cu care funcționeaza microcontrolerul.
- A0-A5 – pini analogici.
- D0-D13 – pini digitali de intrare si iesire. Cele cu semnul "~" au capacitate de semnal PWM.
- RX si TX – 0(RX) si 1 (TX). Folosit pentru a receptiona (RX) și a transmite (TT) date seriale TTL. Acest pini sunt conectati la pinii corespunzatori ai cipului serial ATmega8U2 USB-to-TTL.
- D2-D3 – sunt intreruperi externe. Acesti pini pot fi configurati pentru a declansa o intrerupere.
- SPI - Serial Peripheral Interface (SPI) este un protocol de date seriale utilizat de microcontrolere pentru a comunica cu unul sau mai multe dispozitive externe intr-o conexiune de tip magistrala.
- SS (Slave Select) determina cu ce dispozitiv comunică Masterul în prezent.
- MISO (Master In Slave Out) - O linie pentru trimiterea datelor catre dispozitivul Master.
- MOSI (Master Out Slave In) - Linia Master pentru trimiterea datelor catre dispozitivele periferice.
- SCK (Serial Clock) - Un semnal de ceas generat de dispozitivul Master pentru sincronizarea transmisiei de date.

- I2C - Pinii SCL / SDA sunt pinii dedicati pentru comunicația I2C (I2C este un protocol de comunicație denumit in mod obisnuit „magistrala I2C”). Pe Arduino Uno se gasesc pe pinii analogici A4 și A5.

- SCL este linia de ceas proiectata pentru a sincroniza transferurile de date.
- SDA este linia utilizata pentru transmiterea datelor.
- ICSP (In-Circuit Serial Programming). Numele provine din anteturile de programare in sistem (ISP). Producatori precum Atmel care lucreaza cu Arduino si au dezvoltat propriile headere de programare seriala in circuit. Acesti pini permit utilizatorului sa programeze firmware-ul placilor Arduino.

VII. Breadboard

Inainte de a incepe partea de cablare si de construire de circuite as vrea sa va explic cum se utilizeaza un breadboard.

Acest instrument este perfect pentru crearea unor circuite prototip sau pentru testarea rapida a unui circuit. Sunt 2 tipuri de panouri pe aceasta placa de test, orizontale si vertical. Pe verticala se va conecta sursa placii 0(GND)/3.3/5V. Panourile verticale sunt reprezentate prin ROSU(+) si NEGRU(- /GND). In schimb panourile verticale sunt pentru a atribui mai multor componente posibilitatea de a face legatura una cu cealalta. De exemplu alegem un pin de pe placa, D4, si il plasam pe breadboard, atunci tot randul din fata sau din spatele pinului ales pe breadboard pe plan orizontal, vor avea acces la pinul D4.

Dupa cum se poate observa in imagine pe mijloc exista o anumita distanta. Aceasta separa instrumentul in 2 parti simetrice. Partea din mijloc este utilizata in mare parte pentru a face conexiunea intre microcontrolare, deoarece separa pinii din partea stanga cu pinii din partea dreapta. Astfel fiecare parte va avea contributie proprie.

Veti vedea ca dupa cateva proiecte simple o sa intelegeti principiul de baza a acestei placi.

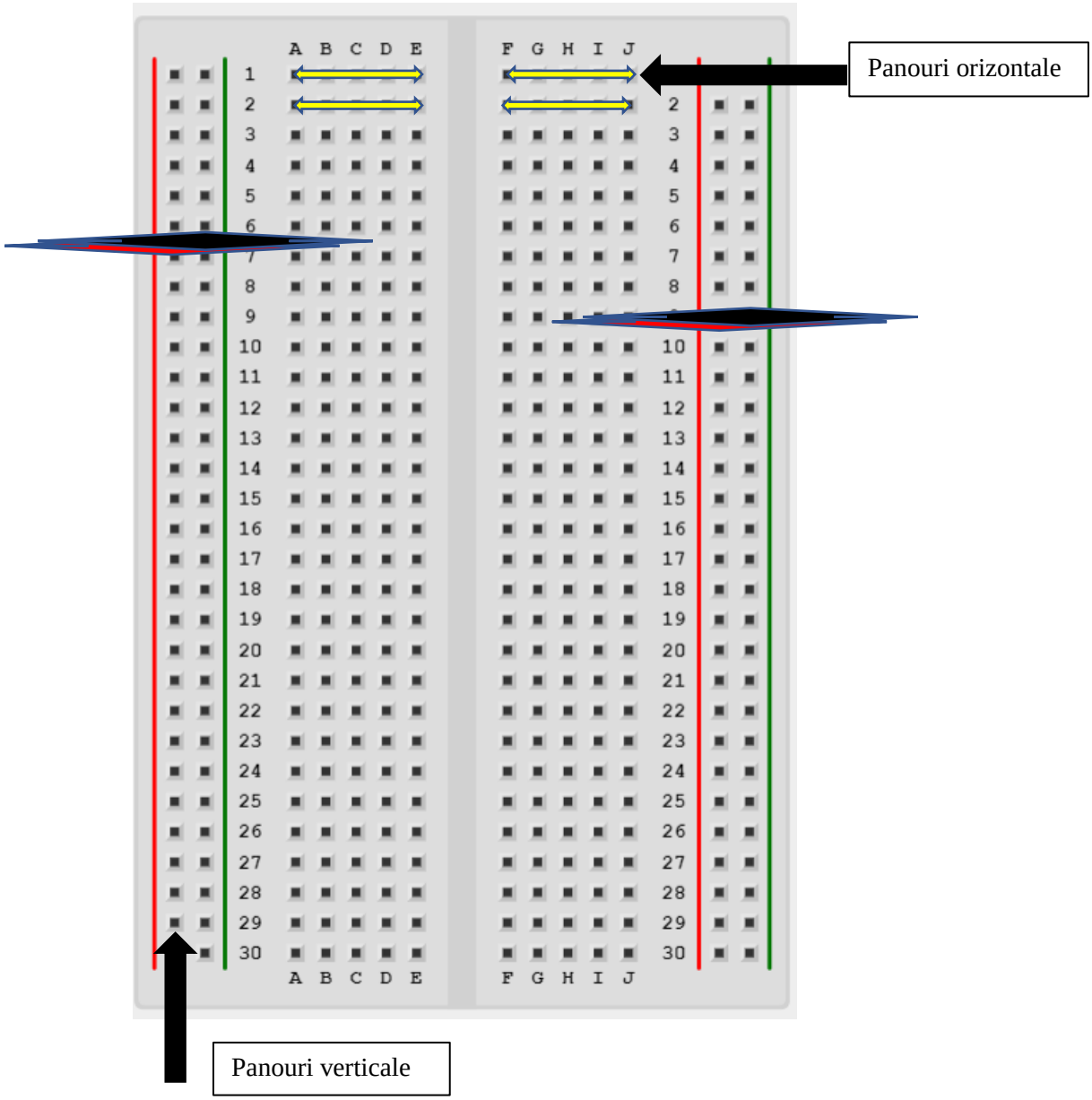


Fig. 1.12

VIII. Legea lui Ohm

Legea lui Ohm spune ca intensitatea curentului electric(I) ce strabate un conductor este direct proportionala cu tensiunea electrica(U) si invers proportionala cu rezistenta(R) acesteia.

$I = \frac{U}{R}$ Unde: I – intensitatea curentului masurata in ampere;

U – tensiunea electrica masurata in volti;

R – rezistenta masurata in Ohm(Ω).

Este usor de utilizat aceasta formula, sunt doar 3 variabile din care daca cunoastem 2 trebuiesc schimbate in formula mentionata mai sus.

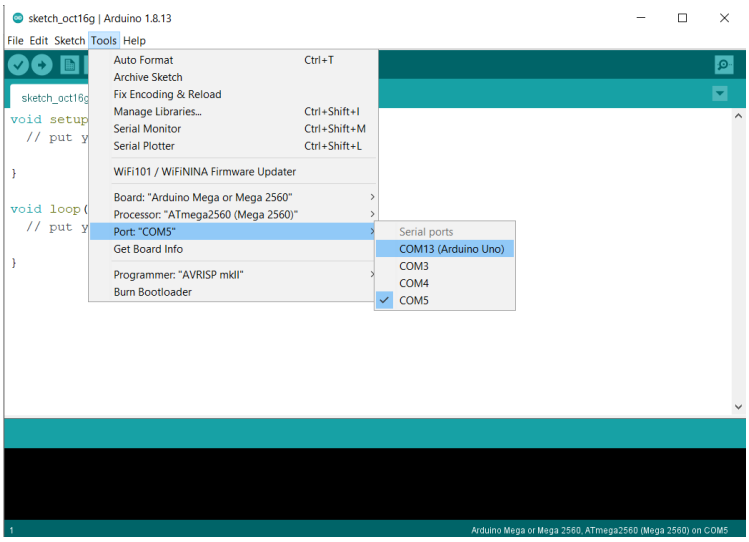
IX. Proiecte Arduino

Toate proiectele viitoare sunt realizate cu placa Arduino Uno, daca aveti alta placa de dezvoltare(nano, mega, Leonardo etc.) nu este nici o problema. Se pot realiza aceleasi proiecte si cu ele, deoarece toate placile de dezvoltare Arduino au aproximativ aceiasi structura(pini analogici, digitali, GND, 3.3V,5V,etc.), trebuie doar setat tipul placii in soft.

In urmatoarele proiecte vom discuta despre anumiti senzori cu care vine echipat un kit Arduino(senzor ultrasonic, de umiditate si temperatura, prezenta umana etc.). De asemenea vom realiza anumite dispozitive interesante precum un termometru, semafor, pian, o masinuta care evita obstacolele, dar vom vorbi si despre rezistente, capacitori, bobine etc. Fiecare proiect va fii alcatuit din cateva subcapitole: ce reprezinta acest proiect, componentele necesare realizarii proiectului, conexiuni, diagrama, conexiunea se face in felul urmator, codul sursa, informatii.

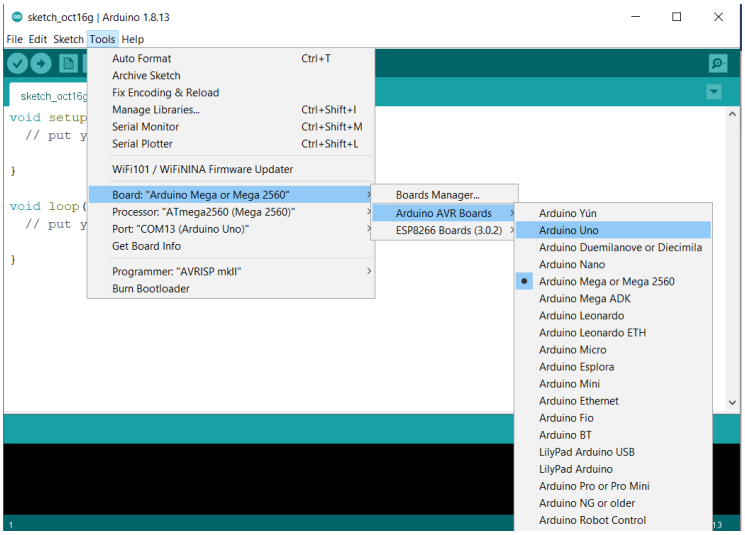
Iar acum sa incepem… deschidem soft-ul pentru a configura partea hardware(placa). Si urmam urmatoarele instructiuni:

Conectam cablul USB tip B la PC si la placa de dezvoltare. Intram in aplicatie si selectam din bara de meniu optiunea “Tools”.



Selectam “Port”, iar apoi alegem portul care este asociat cu dispozitivul nostru(ex. COM13(Arduino Uno))

Fig. 1.13



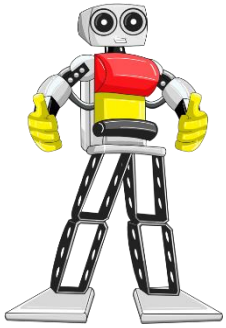
Tot la “Tools” selectam “Board” □ “Arduino AVR Boards” □ placa pe care o utilizam(ex. Arduino Uno).

Fig. 1.14

Toolbar-ul contine:

1. Verificarea codului ca sa nu existe erori;
2. Incarcarea codului pe placa de dezvoltare;
3. Deschide o noua fereastra;
4. Deschide un proiect salvat anterior;
5. Salveaza codul actual.

Fig. 1.15



Acum ca am clarificat anumite aspecte putem trece la treaba. Vom incepe cu cel mai simplu cod si totodata cea mai simpla diagrama a partilor electronice.

1. Blink LED

A. Ce reprezinta acest proiect?

Vom aprinde si stinge un simplu LED(Blink LED).

B.Componentele necesare realizarii proiectului:

- Arduino Uno;
- Rezistor(220Ω);
- Led;
- Fire.

C.Diagrama de conectare

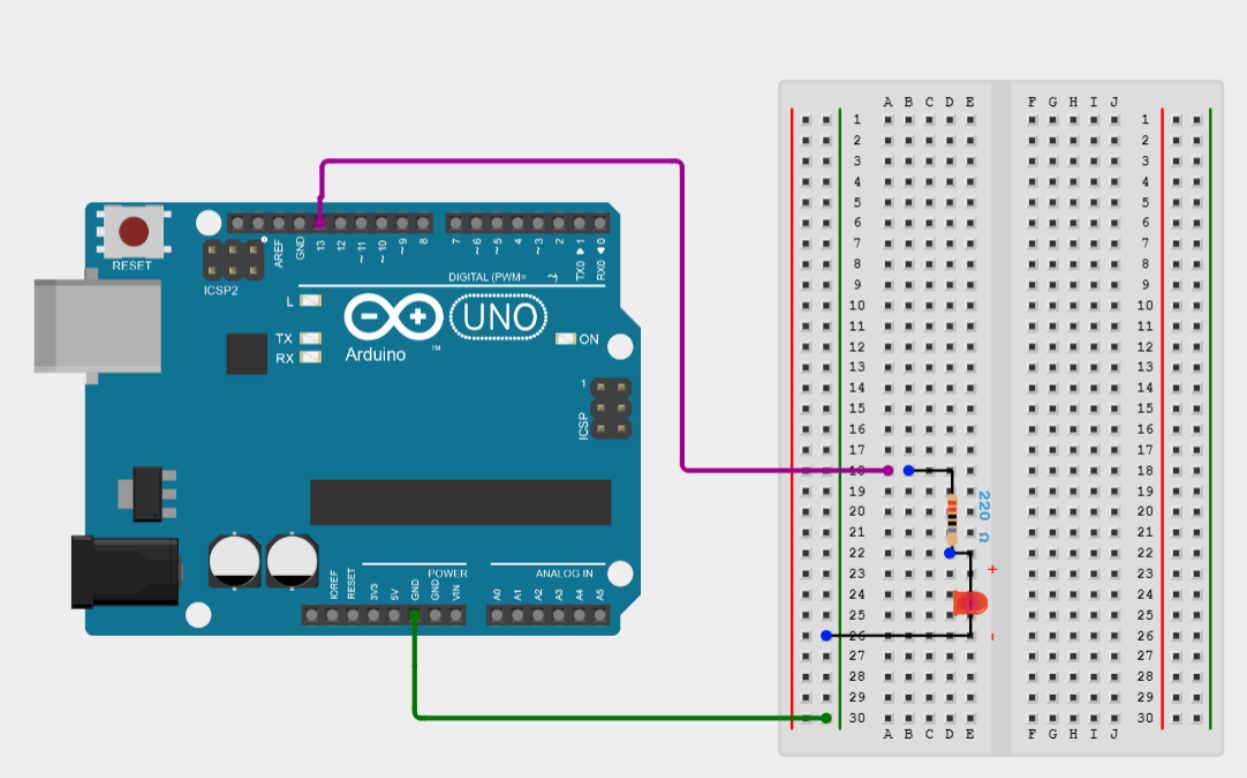


Fig. 1.16 Diagrama pentru un LED blink

D. Conexiunea se face in felul urmator:

- Anodul(+) se conecteaza la rezistenta de 220Ω, iar apoi la D13;
- Catodul se conecteaza la GND.

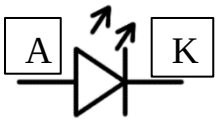
E. Codul sursa

```
void setup() {
  pinMode(13,OUTPUT); // Initializam led-ul pe pinul 13 al placii
} //LED-ul este de tip OUTPUT
void loop() {
  digitalWrite(13,LOW); //Led oprit
```

```
  delay(1000);    //Delay 1 sec.
  digitalWrite(13,HIGH); //Led pornit
  delay(1000);    //Delay 1 sec
}
```

F. Informatii

Simbol



LED(in engleza: Light Emitting Diode, adica dioda emitatoare de lumina) este de fapt o dioda semiconductoare ce emite lumina. Ea este alcatuida dintr-un anod si un catod.

Rolul unei diode este de a lasa curentul sa treaca printr-o singura directie(ex. protectie de polaritate inversa).

Simbol

Scopul principal al unei rezistente este de a rezista curentului electric. Ea se masoara in Ohm(Ω). Orice si oricine are rezistenta electrica, deci exista o anumita toleranta la componente ca urmare nu va arata intotdeauna 100% valoarea exacta(ex. masuram o rezistenta de 220 Ω cu ajutorul aparatului de masura, pe afisaj sunt sanse sa arate de exemplu 219,3 Ω). Inclusiv noi oamenii avem in corp o anumita rezistenta. Va invit si pe voi sa verificati. Setati multimetrul pe sectiunea de Ohm(Ω) iar apoi prindeti de cele 2 terminale ale firelor iar pe afisaj va aparea valoarea.

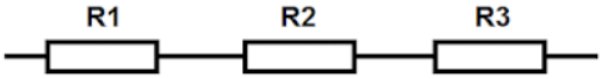
Daca nu ai nici un aparat de masura ce imi poate masura valoarea unei rezistente nu este capat de lume. Poti de asemenea sa cauti pe un motor de cautare codul culorilor la rezistente si vei afla cum sa o calculezi.

Mereu cand aveti un led folositi o rezistenta de 220 Ω pentru a proteja led-ul. Desigur functioneaza si fara rezistenta, dar asa ii cresti rata de viata a diodei semiconductoare, pentru ca in timp se poate arde.

Chiar daca nu iti iese din prima, la orice proiect ar fi, mai intai verifici sa vezi daca nu sunt erori in cod, dupa care treci la schema electrica sa analizezi cablajul, daca si acelea sunt in regula verificam componentele daca sunt functionale, deoarece pot sa

fie defecte chiar daca vin din fabrica. Nu te da batut niciodata si mereu intreba-te “De ce?”, “Care ar putea fi problema?”.

Exemplu de a calcula 3 rezistente legate in serie:

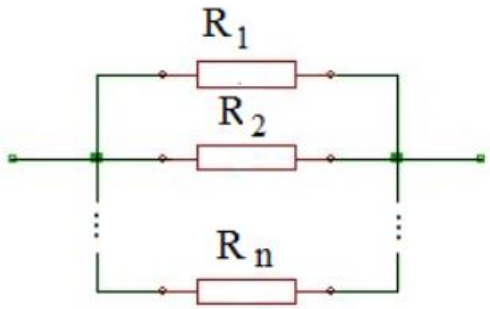


$R_T = R_1 + R_2 + R_3$

Daca avem n rezistente formula de calcul este: $R_T = R_1 + R_2 + \dots + R_n$

Exemplu de a calcula mai multe rezistente legate in paralel:

Aceasta formula este exact inversul formulii a rezistoarelor legate in serie, adica:



$1/R_T = 1/R_1 + 1/R_2 + \dots 1/R_n$

2. Semafor

A. Ce reprezinta acest proiect?

Este vorba de 3 LED-uri, verde, galben si rosu si le vom porni ca si un semafor.

B. Componentele necesare realizarii proiectului:

- Arduino Uno;
- 3 LED-uri, de preferat rosu, galben si verde;
- 3 rezistori de 220Ω;
- Fire.

C. Diagrama

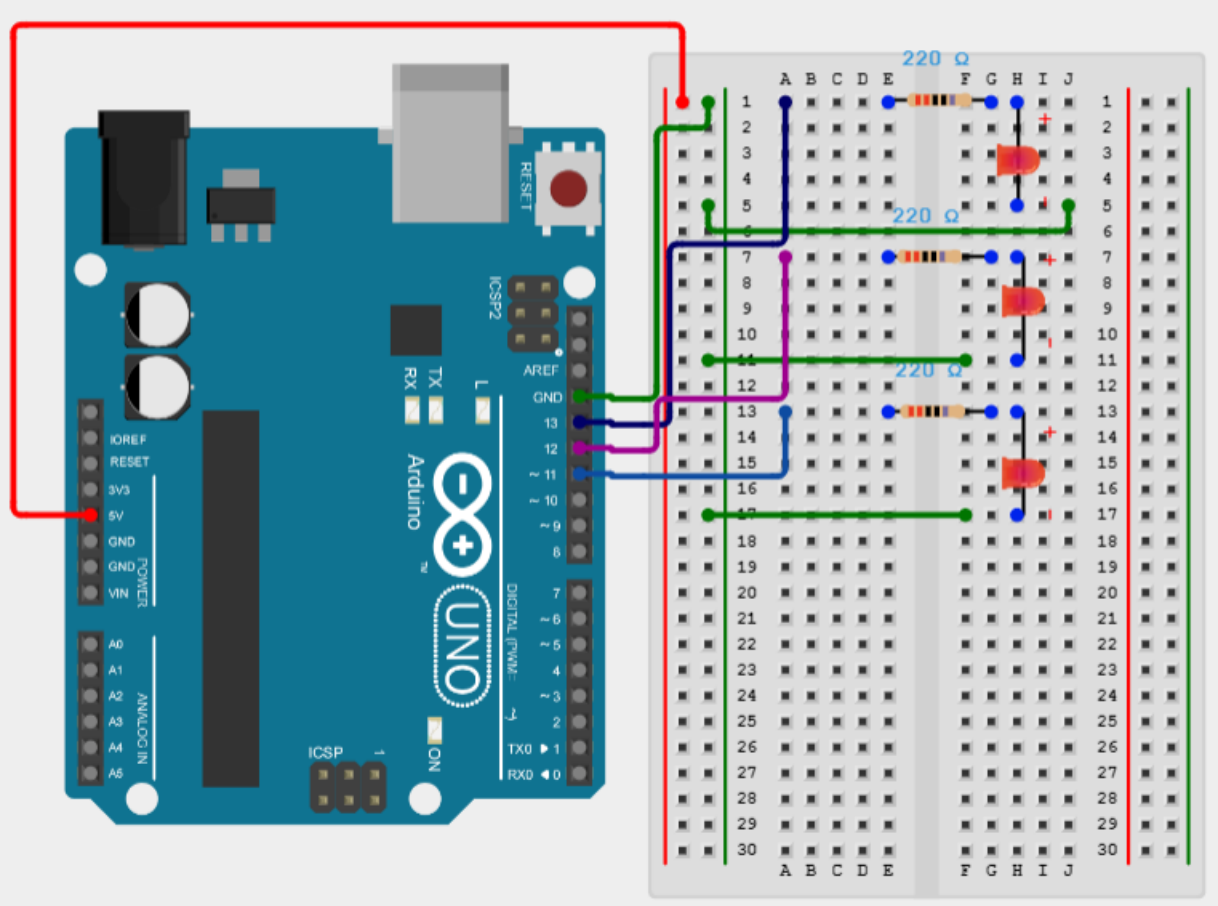


Fig. 1.17 Semafor

D. Conexiunea se face in felul urmator:

 Anodul(+) se conecteaza la rezistenta de 220Ω, iar apoi la D13, D12 si D11;

Catodul se conecteaza la GND.

E. Codul sursa

```
void setup() {
  pinMode(11,OUTPUT);
  pinMode(12,OUTPUT);
  pinMode(13,OUTPUT);
}
void loop() {
  digitalWrite(13, HIGH);
```

```
delay(3000);
digitalWrite(12,HIGH);
delay(1000);
digitalWrite(13, LOW);
digitalWrite(12, LOW);
digitalWrite(11, HIGH);
delay(3000);
digitalWrite(12,HIGH);
delay(1000);
digitalWrite(11, LOW);
digitalWrite(12, LOW);
digitalWrite(13, HIGH);
delay(3000);}
```

3. Tranzistor + LED

A. Ce reprezinta acest proiect?

Vom aprinde si stinge un led utilizand un tranzistor de tip NPN.

B. Componente necesare realizarii proiectului;

- Arduino Uno;
- LED;
- Tranzistor (NPN, tip 2N2222);
- 2 rezistente de 220Ω;
- Fire.

C.Diagrama

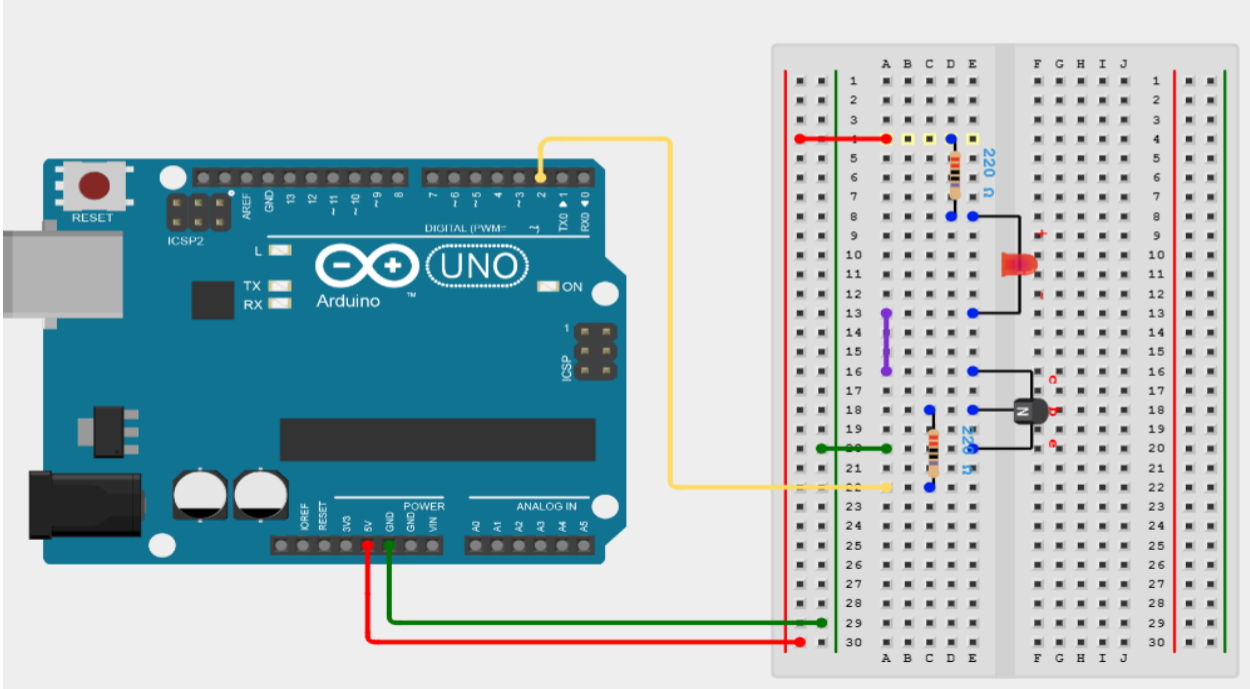


Fig. 1.18 LED + Tranzistor

D. Conexiunea se face in felul urmator:

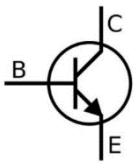
- Anodul(+) se conecteaza la rezistenta de 220Ω, iar apoi la 5V;
- Catodul se conecteaza la pinul C(colector) al tranzistorului;
- La baza tranzistorului se leaga o rezistenta de 220Ω, iar apoi conectam la placa Arduino la pinul D2;
- Emitorul tranzistorului se conecteaza la GND.

E.Codul sursa

```
const int transistor = 2; //tranzistorul va prelua pinul 2 al placii
void setup(){
  pinMode(transistor, OUTPUT);
}
void loop()
{
```

```
  digitalWrite(transistor,LOW); //Tranzistorul nu va primi semnal(OFF)
  delay(1000); // Delay 1 sec.
  digitalWrite(transistor,HIGH);//Tranzistorul primeste semnal(ON)
  delay(1000); //Delay 1 sec.
}
```

F.Informatii



Tranzistorul este o componenta electrica care are cel putin 3 terminale(pini: baza, emitor, colector). Aceasta componenta poate fi de mai multe feluri(NPN sau PNP), dar te las pe tine sa descoperi diferenta intre ele cautand pe internet. Ne putem gandi la un tranzistor ca fiind un buton automat. In momentul in care tranzistorul nostru primeste un anumit semnal la baza, atunci el imi va inchide circuitul, iar cand nu primeste semnal circuitul meu va ramane deschis.La baza tranzistorului se pune o rezistenta pentru a proteja componenta.

!!! Daca nu stii cum se foloseste un anumit dispozitiv oricat de usor ar fi… cauta mereu pe internet informatii despre ea(datasheet). Acolo vei gasi absolut tot ce iti trebuie sa stii despre componenta ta electronica. Chit ca este doar un LED nu este o rusine sa cauti pe internet sa vezi care este anodul sau catodul ei spre exemplu.

4. Un simplu buton

A. Ce reprezinta acest proiect?

Depinzand de starea butonului LED-ul se va aprinde sau stinge. Practic butonul este folosit pe post de intrerupator.

B. Componente necesare realizarii proiectului:

- Arduino Uno;
- LED;
- Rezistor (220Ω);
- Rezistor(10kΩ);
- Buton;
- Fire.

C.Diagrama

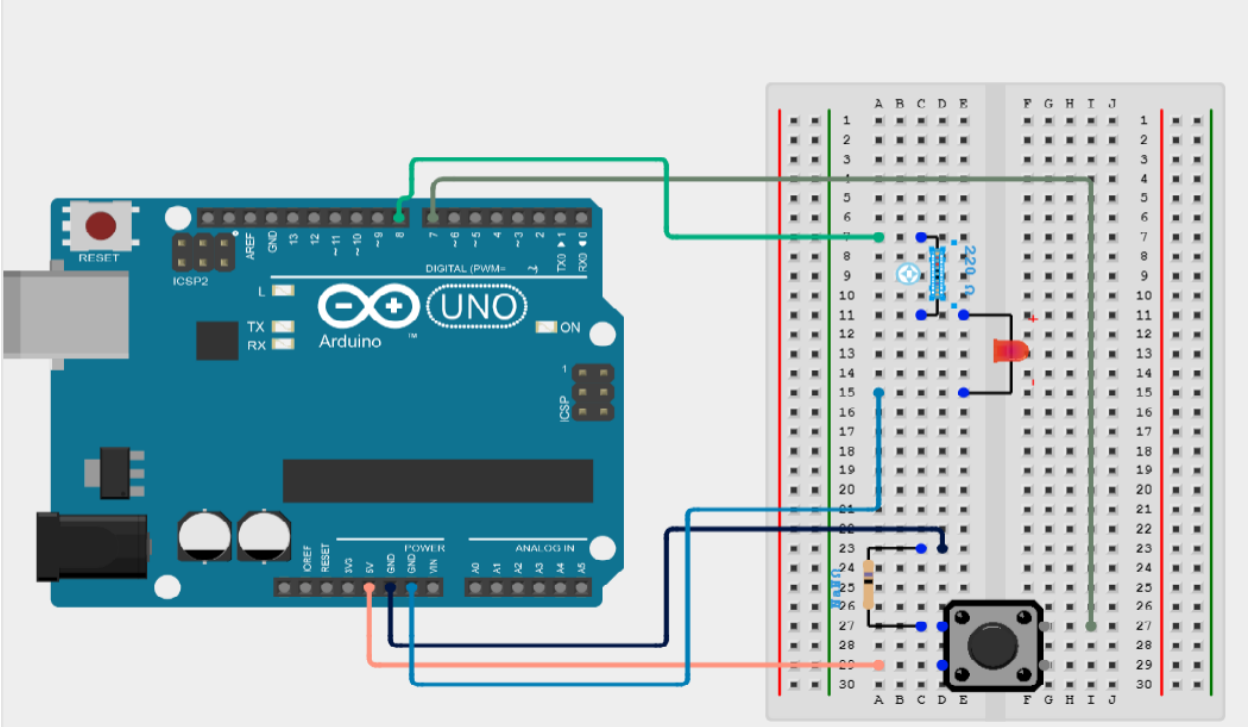


Fig. 1.19 LED + buton

D. Conexiunea se face in felul urmator:

 Anodul(+) se conecteaza la rezistenta de 220Ω, iar apoi la pinul D8;

 Catodul (-) se conecteaza la GND;

 Unul din terminalele butonului se leaga la 5V, celalalt la GND, iar al 3 lea la D7.

E. Codul sursa

```
#define LED_PIN 8 //Definim LED-ul pe pinul 8 al placii
#define BUTTON_PIN 5 //Definim butonul pe pinul 8 al placii
void setup() {
  pinMode(LED_PIN, OUTPUT);  //LED-ul este de tip OUTPUT
  pinMode(BUTTON_PIN, INPUT); //Butonul este de tip INPUT
}
```

```
void loop() {
  if (digitalRead(BUTTON_PIN) == HIGH) {
    digitalWrite(LED_PIN, LOW);
  }else {digitalWrite(LED_PIN, HIGH);}}
```

F. Informatii

Un buton este un tip de comutator utilizat pentru controlul unor aparate. Exista tot felul de butoane de dimensiuni si forme diferite. Poate indeplini diferite functii, depinde de noi cumdorim sa il programam.

5. LED (RGB)

A. Ce reprezinta acest proiect?

Vom programa un LED de tip RGB(red, green, blue) sa isi schimbe culoarea automat in diferite culori.

B. Componente necesare:

- Arduino Uno;
- LED RGB;
- 3 rezistente de 220Ω;
- Fire.

C. Diagrama

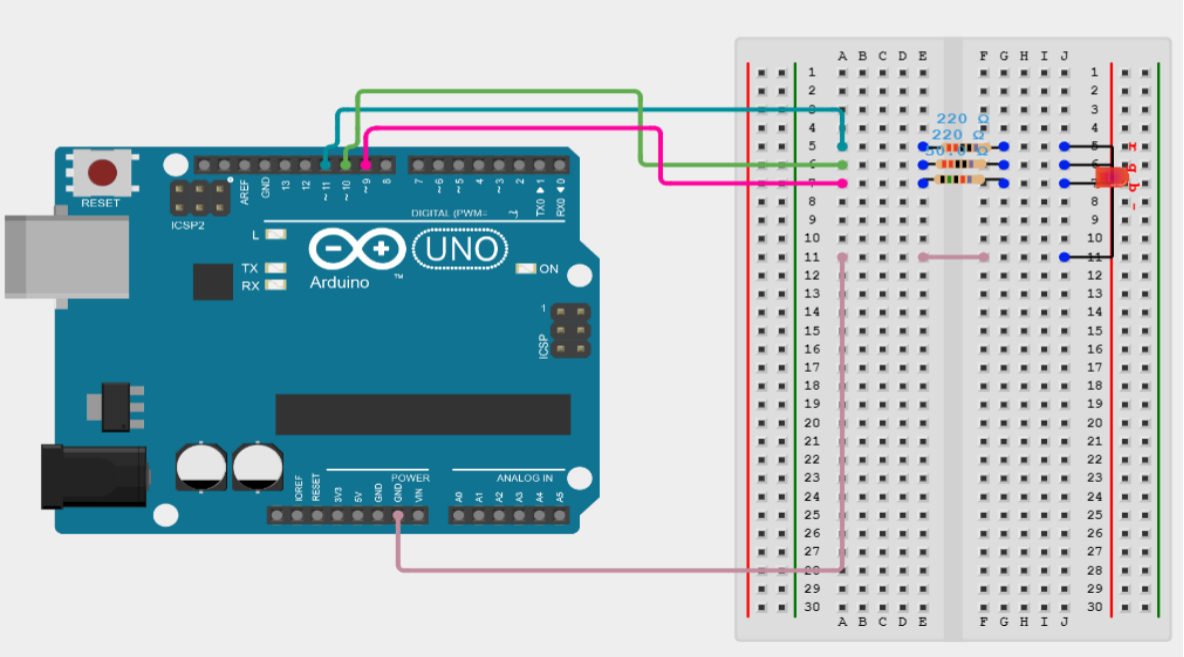


Fig. 1.20 RGB Led

D. Conexiunea se face in felul urmator:

 Catodul (-) la GND;

 Cele 3 pinuri conectate fiecare la cate o rezistenta de 220Ω, iar apoi la D9(albastru), D10(verde) si D11(rosu).

E.Codul sursa

```
int red_light_pin= 11;  //Rosu e pe pinul 11
int green_light_pin = 10;  //Verde e pe pinul 10
int blue_light_pin = 9;  //Albastru e pe pinul 9
void setup() {
  pinMode(red_light_pin, OUTPUT);
  pinMode(green_light_pin, OUTPUT);
  pinMode(blue_light_pin, OUTPUT);
}
void loop() {
```

```
  RGB_color(255, 0, 0); // Rosu
  delay(1000);
  RGB_color(0, 255, 0); // Verde
  delay(1000);
  RGB_color(0, 0, 255); // Albastru
  delay(1000);
  RGB_color(255, 255, 125); // Zmeura
  delay(1000);
  RGB_color(0, 255, 255); // Cian
  delay(1000);
  RGB_color(255, 0, 255); // Magenta
  delay(1000);
  RGB_color(255, 255, 0); // Galben
  delay(1000);
  RGB_color(255, 255, 255); // Alb
  delay(1000);
}
void RGB_color(int red_light_value, int green_light_value, int blue_light_value)
 {
  analogWrite(red_light_pin, red_light_value);
  analogWrite(green_light_pin, green_light_value);
  analogWrite(blue_light_pin, blue_light_value);}
```

F. Informatii

Chiar daca sunt doar 3 culori ele pot crea si altele, aproape orice culoare. Imagineaza-ti trei LED-uri diferite rosu, verde si albastru care functioneaza

simultan intr-un loc foarte restrans la diferite intensitati. Acest efect creaza diferite nuante de culori.

6. Intensitatea unui LED

A. Ce reprezinta acest proiect?

Aici vom discuta despre 2 posibile proiecte. Vom utiliza un potentiometru o data pentru a regla intensitatea luminii, iar celalalt cod este despre a opri si porni LED-ul in functie de valoarea potentiometrului.

B. Componente necesare:

- Arduino Uno;
- Potentiometru(10kΩ);
- LED;
- Rezistenta (220Ω);
- Fire.

C. Diagrama

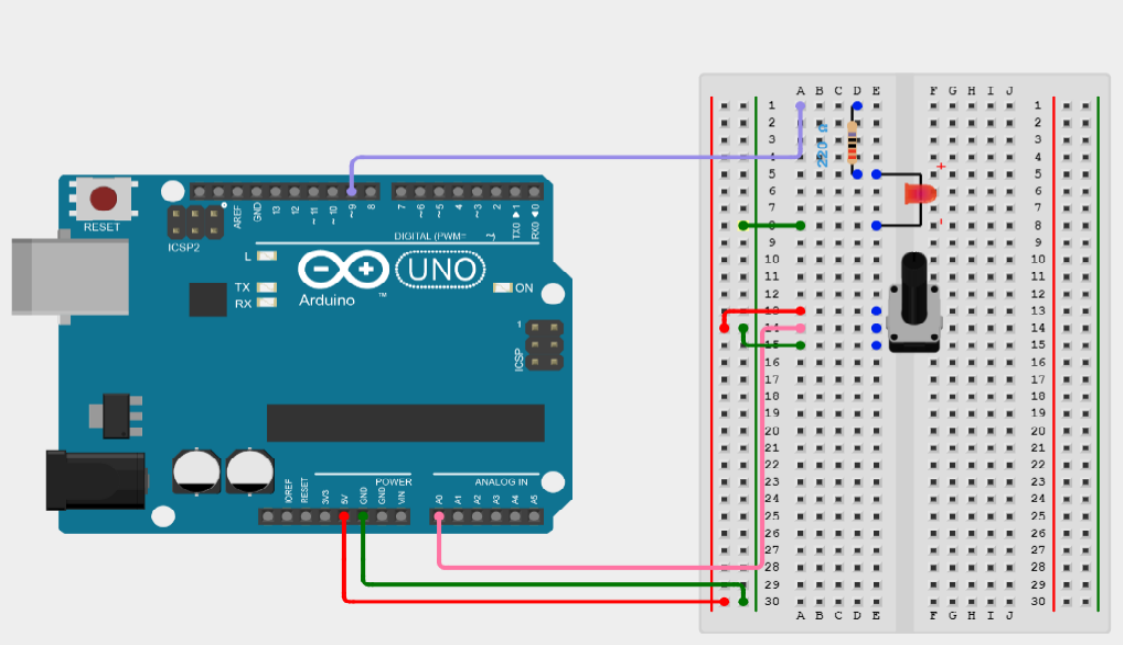


Fig. 1.21 LED + potentiometru

D. Conexiunea se face in felul urmator:

 Anodul(+) se va conecta la un rezistor de 220Ω, iar apoi la D9;

 Catodul(-) este conectat la GND;

 Primul si ultimul pin se conecteaza la 5V respectiv GND, iar pinul din mijloc la pinul A0.

E.Codul sursa

a.Intensitatea luminii LED-ului cu ajutorul potentiometrului

```
const int analogPin = A0;//Pinul analog al potentiometrului
const int ledPin = 9;//Pinul LED-ului
int inputValue = 0;//variabila pentru stocarea valorii de la potentiometru
int outputValue = 0;//variabila pentru stocarea valorii LED-ului
void setup(){}
void loop(){
inputValue = analogRead(analogPin);//Citim valoarea potentiometrului
outputValue = map(inputValue,0,1023,0,255);//Convertim de la 0-1023 proportional cu o valoare din intervalul 0-255
analogWrite(ledPin,outputValue);//Intensitatea LED-ului depinde de valoare}
```

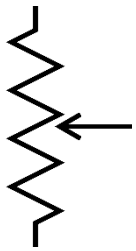
b.LED blink folosind potentiometrul

```
int potPin = A0;    // Am selectat pinul A0(INPUT) pentru potentiometru
int ledPin = 9;   // Selectam pinul pentru LED
int val = 0;       // Variabila pentru a stoca valoarea provenita de la senzor
void setup() {
  pinMode(ledPin, OUTPUT);  // Declaram ledPin ca si OUTPUT
}
```

```
void loop() {
  val = analogRead(potPin);    // Citim valoarea potentiometrului
  digitalWrite(ledPin, HIGH);  // Aprindem LED-ul
  delay(val);                  // Oprim programul pentru o perioada de timp
  digitalWrite(ledPin, LOW);   //Oprim LED-ul
  delay(val);
}
```

F. Informatii

Simbol



Un potentiometru(vine de la “potential difference and metering”) este un instrument pentru variatia potentialului electric.

Practic ea este o rezistenta reglabila care isi schimba valoarea in functie de orientarea cursorului. Are 3 terminale si se actioneaza mecanic.

7. Doua proiecte interesante utilizand o fotorezistenta

A. Ce reprezinta acest proiect?

Vom face 2 proiecte interesante folosind o fotorezistenta . Unul dintre ele este un fel de lampa de veghe, in sensul ca in momentul in care fotorezistenta noastra ia sub o anumita valoare atunci LED-ul se va aprinde, iar celalalt este despre faptul ca LED-ul va lumina in functie de valoarea fotorezistentei.

B. Componente necesare:

- Arduino Uno;
- LED;
- Fotorezistor;
- Rezistente(220Ω si 100 Ω);
- Fire.

C. Diagrama

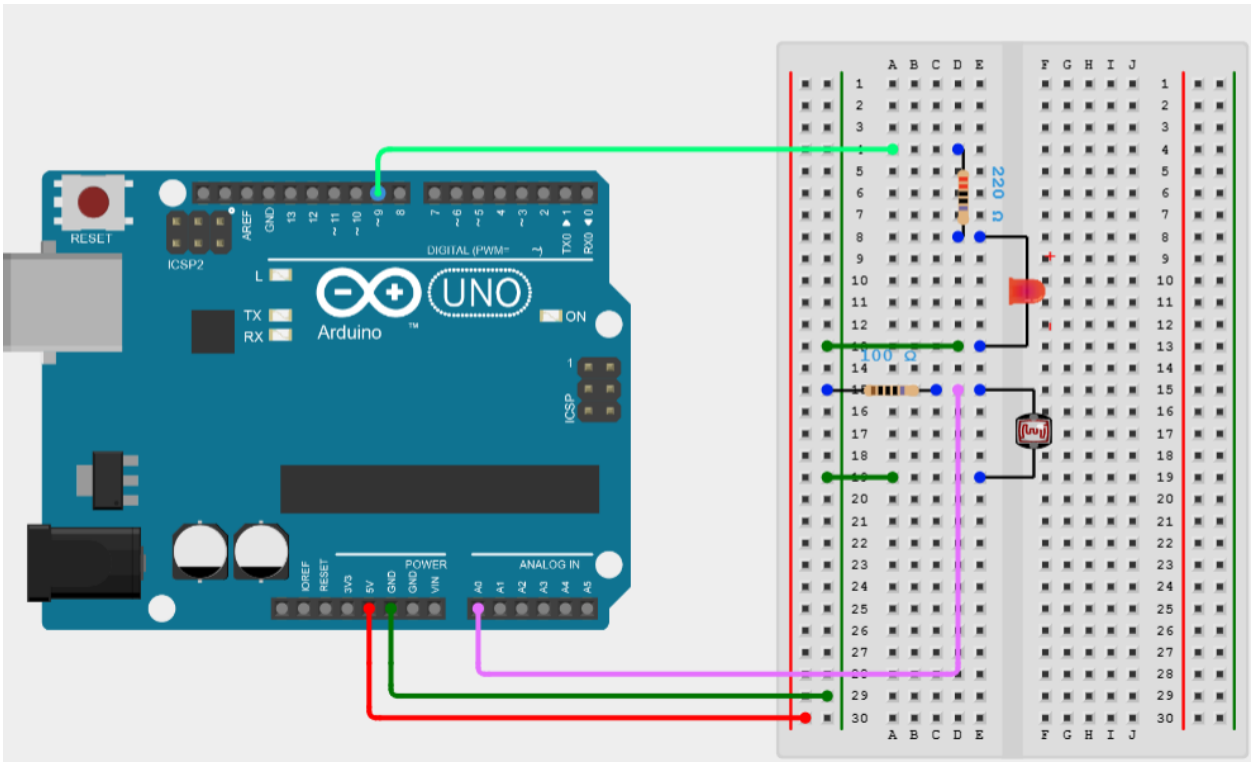


Fig. 1.22 LED + fotorezistor

D. Conexiunea se face in felul urmator:

 Anodul(+) se conecteaza la o rezistenta de 220Ω, iar apoi la pinul D9;

 Catodul(-) se conecteaza la GND;

 Un terminal al fotorezistentei este conectat la o rezistenta de 100Ω, la pinul A0 si la 5V;

 Celalt terminal este setat pe GND.

E. Codul sursa

a.Intensitarea LED-ului folosind un fotorezistor

```
int sensorValue = 0; //Initializam val. fotorez. ca fiind 0
void setup() {
  pinMode(A0, INPUT);// Fotorezistorul este de tip INPUT si este pe pinul A0
  pinMode(9, OUTPUT);
```

```
  Serial.begin(9600);
}
void loop() {
  sensorValue = analogRead(A0);//Citim valoarea fotorezistorului
  Serial.println(sensorValue); // Afisam valoarea fotorezistorului in Serial Monitor
  analogWrite(9, map(sensorValue, 0, 1023, 0, 255));
  delay(100); // Delay 100 millisecunde
}
```

b. Blink LED folosind un fotorezistor

```
int sensorValue = 0;
void setup() {
  pinMode(A0, INPUT);
  pinMode(9, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  if(sensorValue < 100){digitalWrite(9,HIGH);
  }else digitalWrite(9,LOW);
  delay(100); // Delay 100 millisecunde}
```

F.Informatii

Cu linia de cod “Serial.println(sensorValue)” afisam in Serial Monitor(Fig. 1.21) valorile fotorezistorului.

“Serial.begin()” seteaza rata de date in biti pe secunda pentru transmisia de date in serie. Atentie! Pentru a comunica corect cu “Serial monitor” trebuie sa ne asiguram ca folosim una din ratele de transmisie enumerate in meniu(coltul din dreapta jos, Fig 1.23).

Fig. 1.21 Serial Monitor

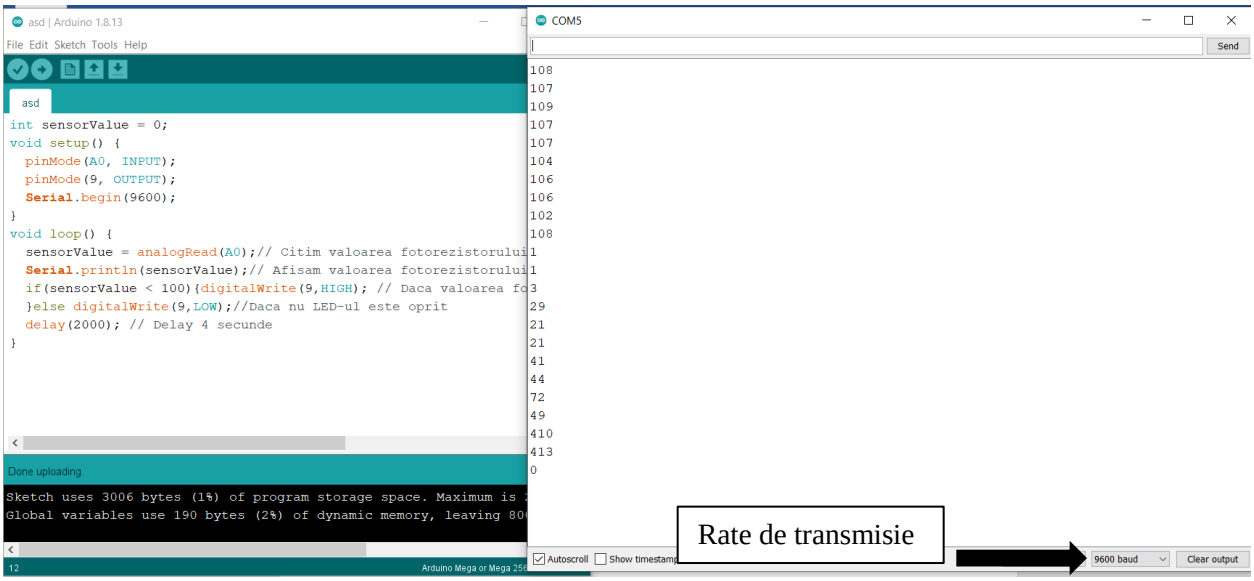
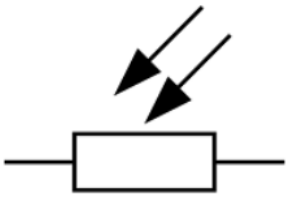


Fig. 1.23 Rata de transmisie

Simbol



Fotorezistorul se poate imparti astfel: “foton”(particula de lumina) si resistor. Un fotorezistor cum ii spune si numele este o rezistenta dependenta de intensitatea luminii. Sunt cunoscute drept senzori de lumina.

8. Utilizarea unui termistor

A. Ce reprezinta acest proiect?

Vom masura temperatura si o vom afisa in “serial monitor” cu ajutorul unui termistor.

B. Componente necesare:

- Arduino Uno;
- Termistor;
- Rezistor 10kΩ;
- Fire.

C. Diagrama

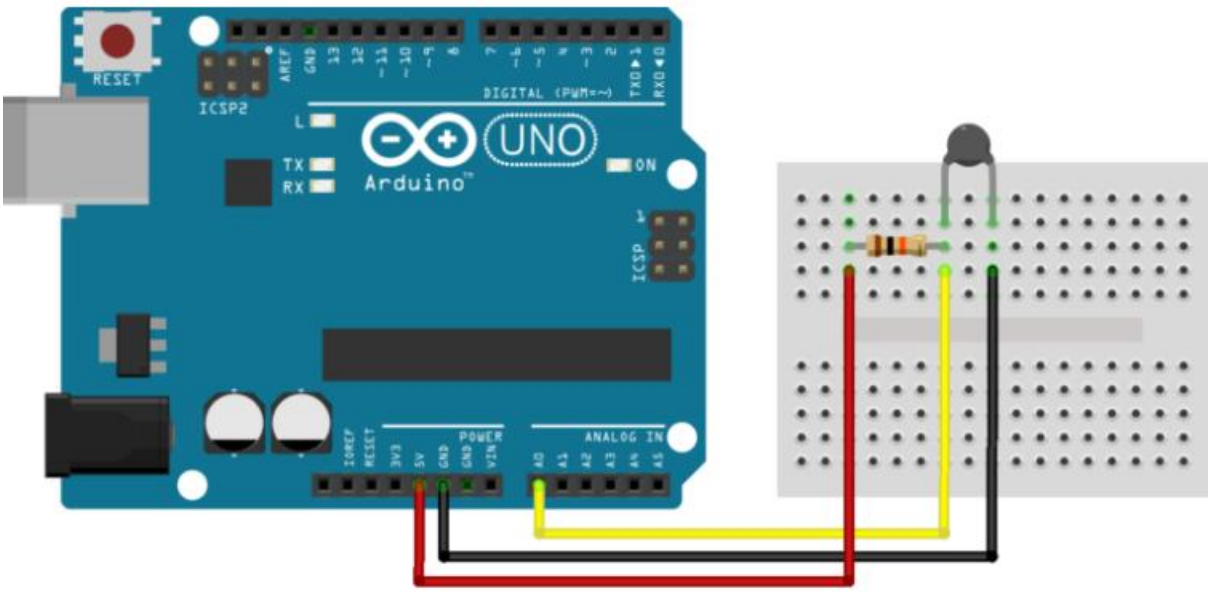


Fig. 1.22 Termistor + Arduino

D. Conexiunea se face in felul urmator:

 Un pin al termistorului se conecteaza la o rezistenta de 10kΩ, iar apoi se conecteaza la A0;

 Terminalul ramas al rezistorului se conecteaza la 5V;

 Ultimul pin al termistorului se conecteaza la GND.

E. Codul sursa

```
#define analogPin A0 //Termistorul initializat pe pinul A0
#define beta 3950 //sectiunea Beta de la termistor
#define resistance 10 //Valoarea rezistentei
void setup()
{
Serial.begin(9600);
}
void loop()
{
  long a =analogRead(analogPin); //Citim val. termistorului
  //Formula de calcul pentru temperatura
  float tempC =beta /(log((1025.0 * 10 / a - 10) / 10) + beta / 298.0) - 273.0;
  float tempF= 1.8*tempC + 32.0;//convertim in F
  Serial.print("TempC: ")
  Serial.print(tempC);
  Serial.println("  C");
  Serial.print("TempF: ");
  Serial.print(tempF);
  Serial.println(" F");
  Serial.println();
  delay(2000);
}
```

F. Informatii

Termistorul este un un fel de rezistor care este dependent de temperatura mai mult decat in rezistentele standard. Termistoarele in general sunt folosite in mare parte pentru a limita curentul de intrare intr-un circuit, senzori de temperatura(ce am realizat si noi in acest proiect), protectori de supracurent etc.

Aceste dispozitive sunt de 2 tipuri:

- NTC, rezistenta scade odata cu cresterea temperaturii;
- PTC, rezistenta creste pe masura ce temperatura creste.

9. Telecomanda

A. Ce reprezinta acest proiect?

Vom porni si opri un LED folosind o telecomanda de masina.

B. Componente necesare:

- Arduino Uno;
- LED;
- Rezistor (220Ω);
- Senzor IR;
- Telecomanda de masina;
- Fire.

C. Diagrama

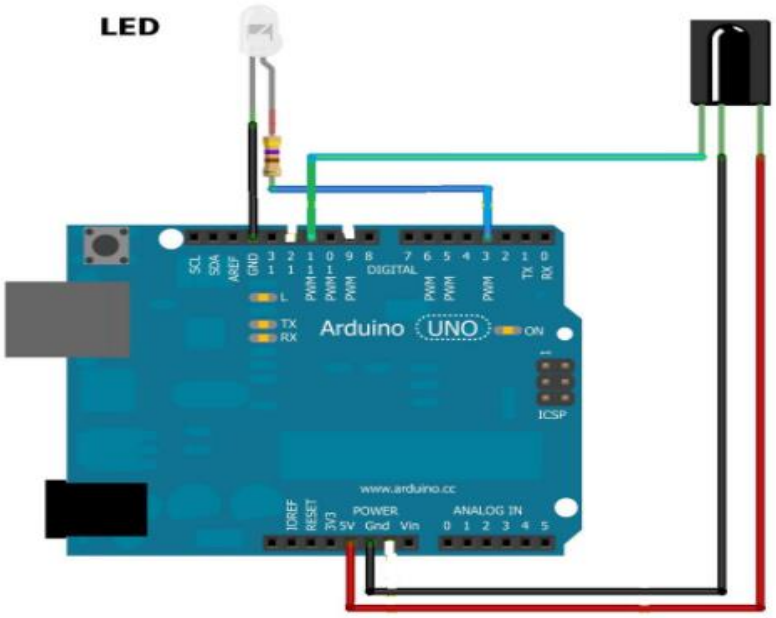
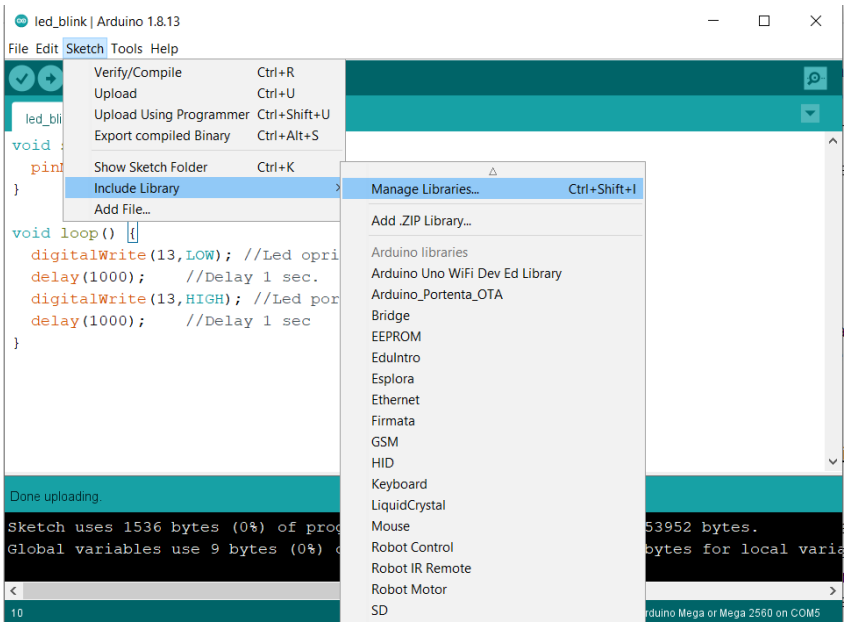


Fig. 1.24 IR remote + LED

D. Conexiunea se face in felul urmator:

- Anodul(+) se conecteaza la o rezistenta de 220Ω, dupa care in continuare la D3;
- Catodul(-) se conecteaza la GND;
- Terminalul (+) al senzorului IR se conecteaza la GND, iar (-) la GND;
- Pinul ramas se conecteaza la D11.

Inainte de a incepe sa scriem codul trebuie sa instalam o anumita librarie. Urmati pasii urmatori.



In bara de meniu selectam “Sketch” → “Include Library” → “Manage Libraries”

Fig. 1.25 Manage library

Se deschide o noua fereastra, iar la search scriem “IRremote” si instalam libraria din poza spre exemplu.

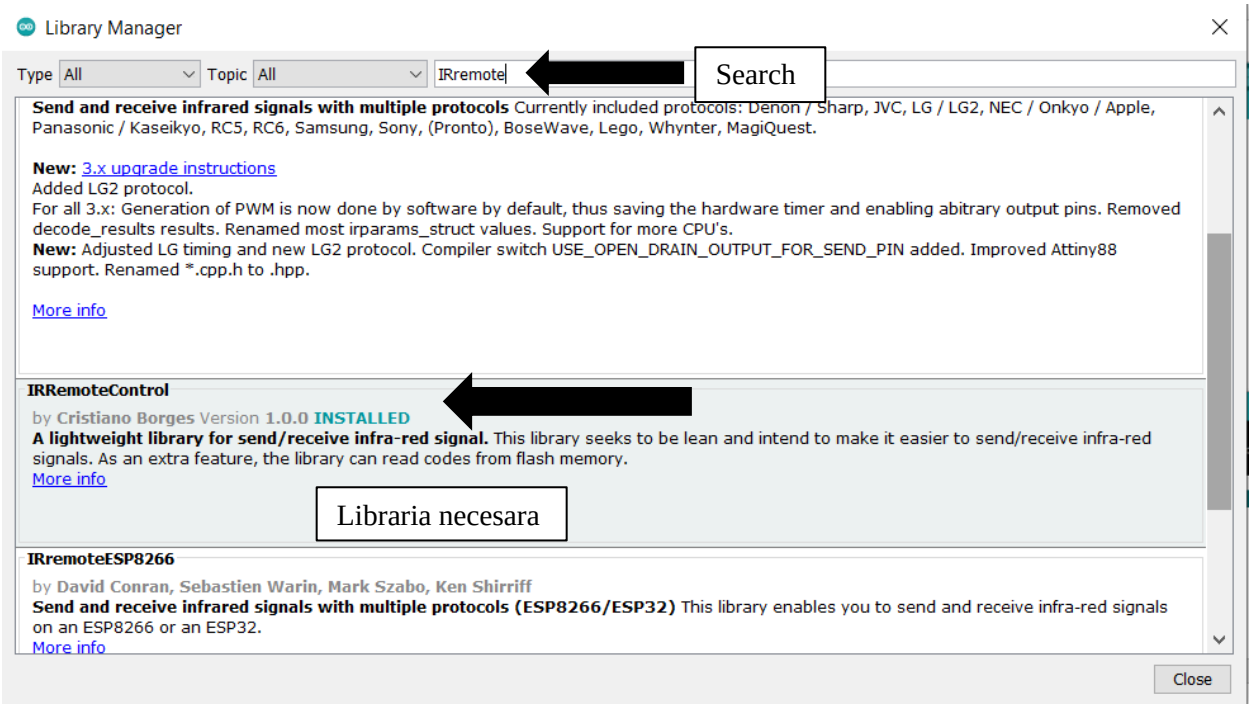


Fig. 1.26 Instalarea librariei

E. Codul sursa

```
#include <IRremote.h> //Includem libraria instalata anterior
const IR_RECEIVE_PIN = 11;  //Senzor IR pe pinul 2
#define LED1 3 // LED-ul pe pinul 11
void setup(){
	Serial.begin(115200);
	Serial.println("IR Receive test");
	IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);
	pinMode(LED1, OUTPUT);
}
void loop(){
  if (IrReceiver.decode()){
  String ir_code = String(IrReceiver.decodedIRData.command, HEX);
  //Imi decodeaza butonul in baza hexazecimala.
  Serial.println(ir_code);
  if(ir_code == "c")
	digitalWrite(LED1, HIGH); //Apasam tasta 1 se aprinde LED-ul
  else if(ir_code == "18")
	digitalWrite(LED1, LOW); //Apasam tasta 2 se stinge LED-ul
  IrReceiver.resume();
}
}
```

F. Informatii

Pentru a stii ce sa scriem in cod la sectiunea de “if” trebuie mai intai sa ne uitam in “serial monitor” pentru ca acolo se decodifica butonul pe care il apasam. De

exemplu “c” este tasta 1 de la telecomanda si imi aprinde LED-ul, iar tasta 2 este “18” care imi opreste LED-ul.

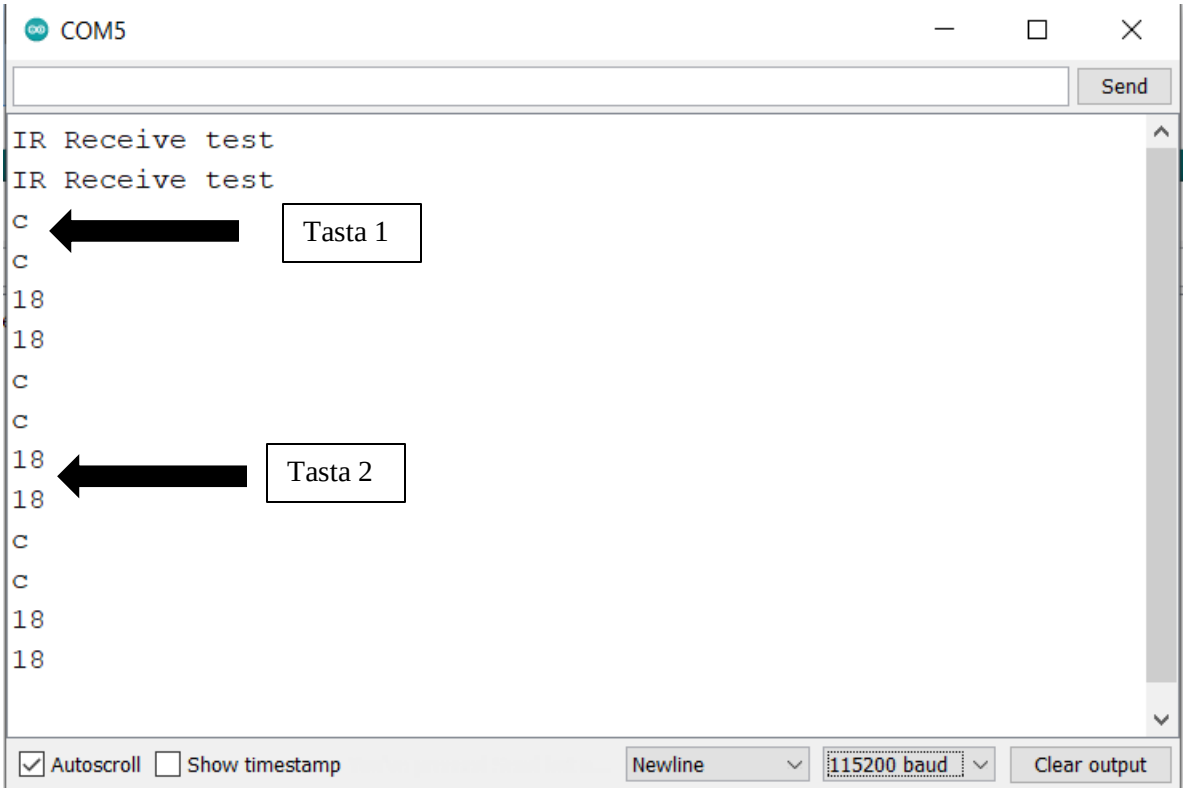


Fig. 1.27 Valoarea butoanelor de la telecomanda

Este foarte interesant si te poti juca cu astfel de lucruri si sa dai mai multor butoane diferite sarcini care nu necesita neaparat LED-uri.

La secventa “String ir_code = String(IrReceiver.decodedIRData.command, HEX);” in loc de “HEX”, scrie-ti “DEC”, iar apoi uitati-va in serial monitor si vei observa o schimbare. Faptul ca nu mai decodeaza din punct de vedere hexazecimal ci in decimali. Putem observa totusi ca daca lasam cu “HEX” ne apar si litere si cifre in schimb la “DEC” sunt doar cifre.

10. Joystick

A. Ce reprezinta acest proiect?

Cu un joystick asemanator cu cel de la PlayStation, variantele mai vechi vom afisa in “serial monitor” anumite valori in functie de pozitia sa.

B. Componente necesare:

- Arduino Uno;
- Joystick;
- Fire.

C. Diagrama

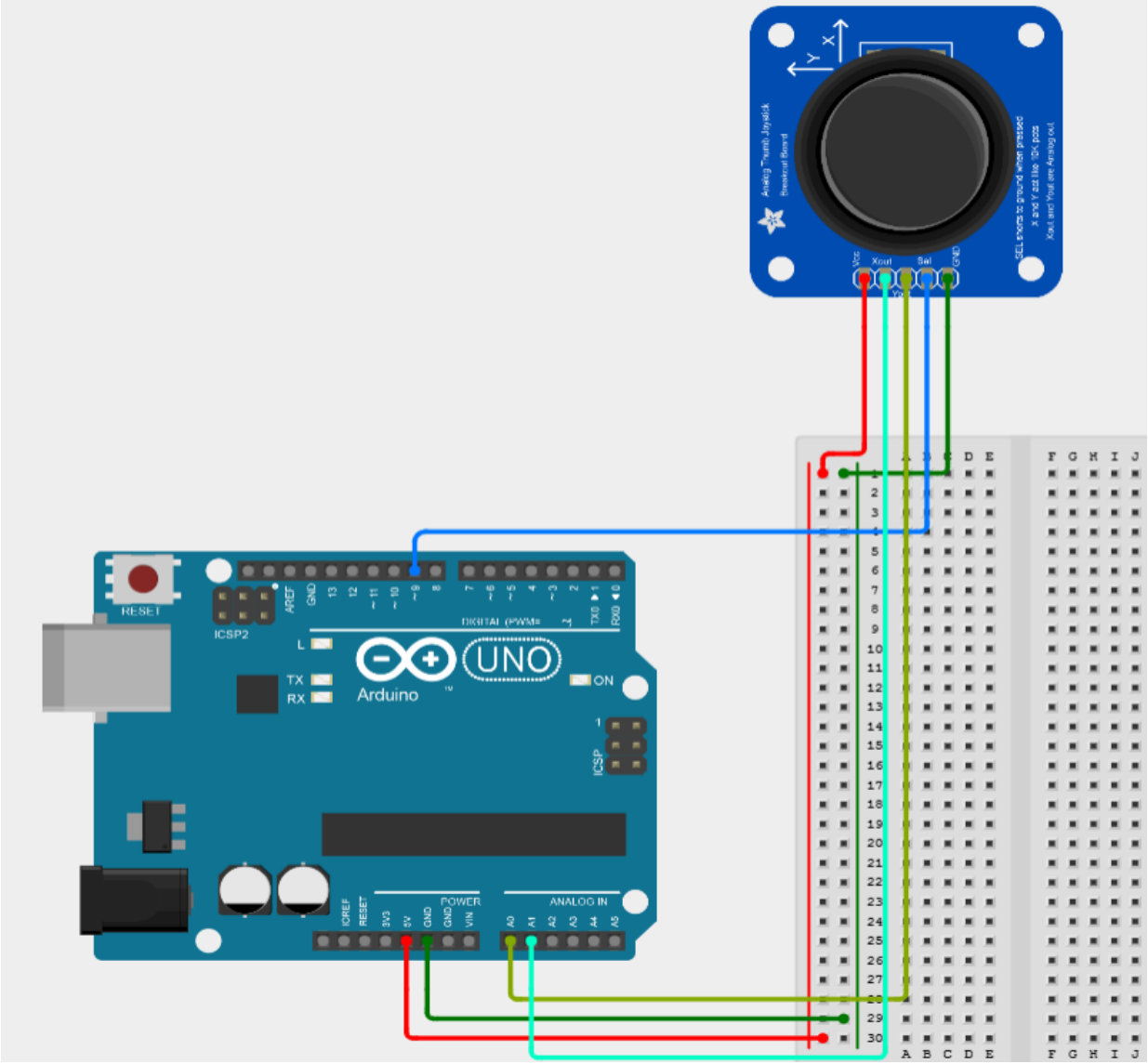


Fig. 1.28 Joystick

D. Conexiunea se face in felul urmator:

 GND la GND, 5V la 5V.

VRX la A0, VRY la A1, iar SW(switch) la D8.

D. Codul sursa

```
int xValue = 0 ; // Axa x
int yValue = 0 ;  //Axa y
int bValue = 0 ;  //Val. butonului

void setup()
{
  Serial.begin(9600) ;
  pinMode(9,INPUT);
  digitalWrite(9,HIGH);
}
void loop()
{
  xValue = analogRead(A0);  //Citim axa x
  yValue = analogRead(A1);  //Citim axa y
  bValue = digitalRead(8);  //Citim val. butonului 1/0
  Serial.print(xValue,DEC); //Afisam coordonata x
  Serial.print(",");
  Serial.print(yValue,DEC); //Afisam coordonata y
  Serial.print(",");
  Serial.print(!bValue); //Afisam val butonului
  Serial.print("\n"); //Rand nou
  delay(1000);
```

}

E. Informatii

Posibil sa va intrebati care este butonul la acest joystick. SW este iesirea de la buton. Cand apasati pe componenta cand axele x si y nu sunt diferite de pozitia initiala atunci se va actiona mecanic un buton care va conecta dispozitivul la GND oferind o iesire LOW.

Ideea de baza a acestui proiect a fost sa traducem pozitia joystick-ului fata de punctele sale de repaus. Cand acest dispozitiv este neschimbat atunci valorile sale sunt x=512, y=512 si butonul are valoarea 0. Cand actionam cu degetul asupra stick-ului valorile se vor schimba datorita pozitiei celor 2 potentiometre incorporate in el. Axa x este stanga dreapta, y este sus si jos.

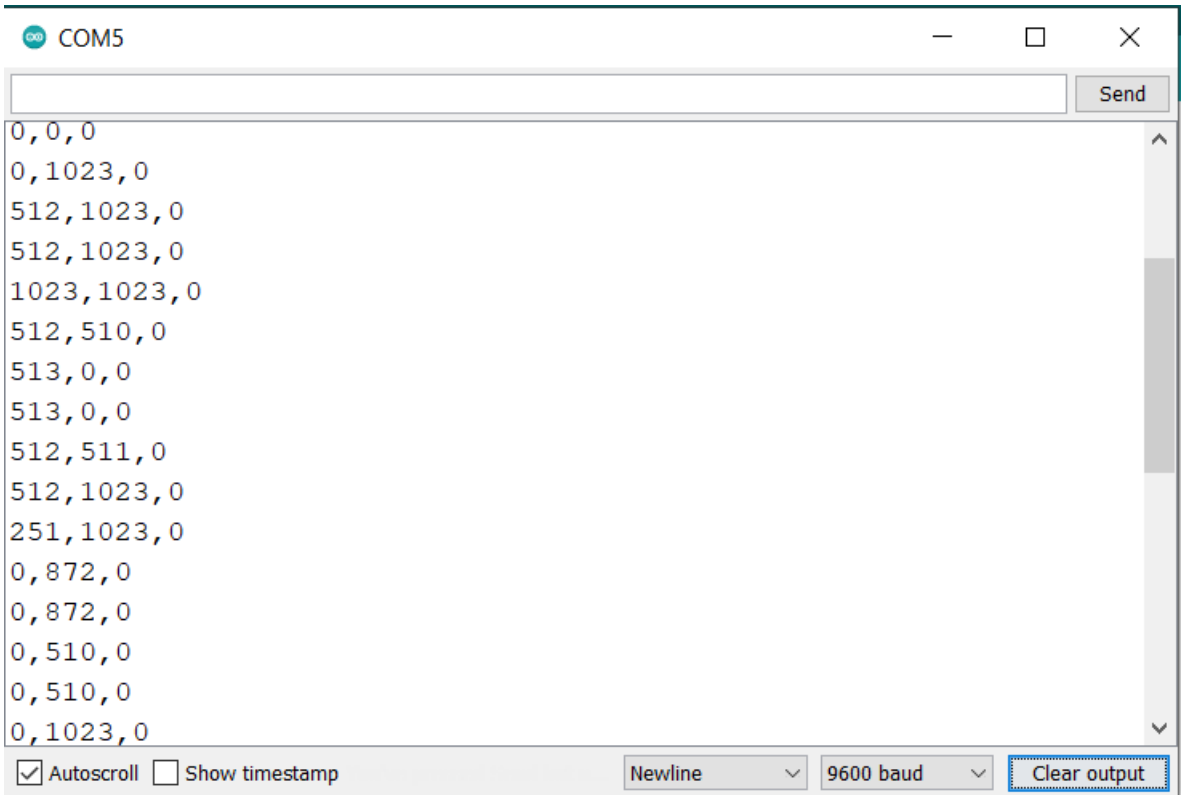


Fig. 1.29 Valorile afisate de joystick in “Serial Monitor”

Se pot realiza multe proiecte cu un astfel de componenta de exemplu sa controlam un robot sau un rover sau mai putem misca de exemplu o camera de supravegheat.

11. Senzor de prezenta umana

A. Ce reprezinta acest proiect?

Folosind un senzor de prezenta umana vom aprinde un LED.

B. Componente necesare:

- Arduino Uno;
- Senzor de prezenta umana(PIR);
- LED;
- Rezistor(220Ω);
- Fire.

C. Diagrama

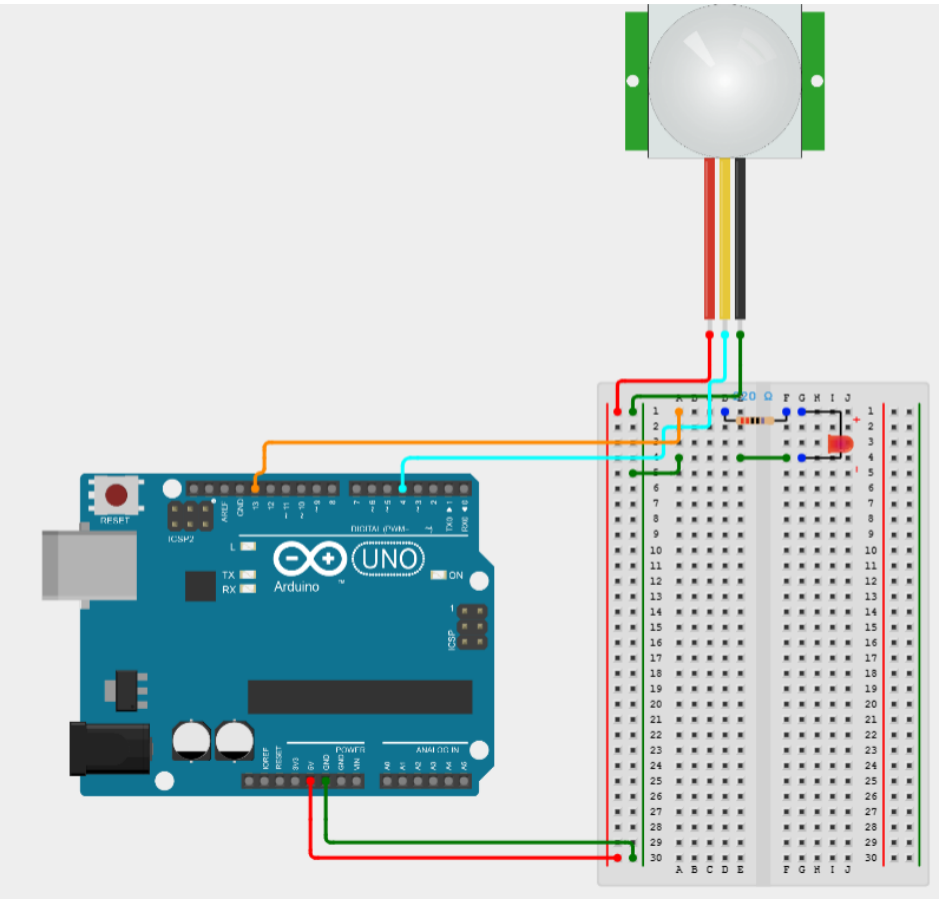


Fig. 1.30 PIR sensor + LED

D. Conexiunea se face in felul urmator:

- Anodul(+) se conecteaza la o rezistenta de 220Ω, iar apoi la pinul D13;
- Catodul(-) se conecteaza la GND;
- Terminalele GND si 5V ale senzorului se vor conecta la placa de dezvoltare la GND si 5V;
- Pinul de “out” se va conecta la D13.

E.Codul sursa

```
int led = 13;  //LED-ul pe pinul 13
int pirSensor = 4; //Senzorul PIR pe pinul 4
void setup() {
  pinMode(pirSensor, INPUT_PULLUP);
  pinMode(led, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  if (digitalRead(pirSensor)) { //Citim senzorul PIR daca sesizeaza prezenta
       Serial.println("Sensed Something");  //  Se va afisa in S.M. prezenta umana
       digitalWrite(led, HIGH);  //Se aprinde LED-ul
  }
  else { //Daca nu sesizeaza nimic
     digitalWrite(led, LOW); //LED-ul se opreste
     Serial.println("Sensed Nothing"); //Se afiseaza in S.M. ca nu a detectat nimic
  }
delay(3000);}
```

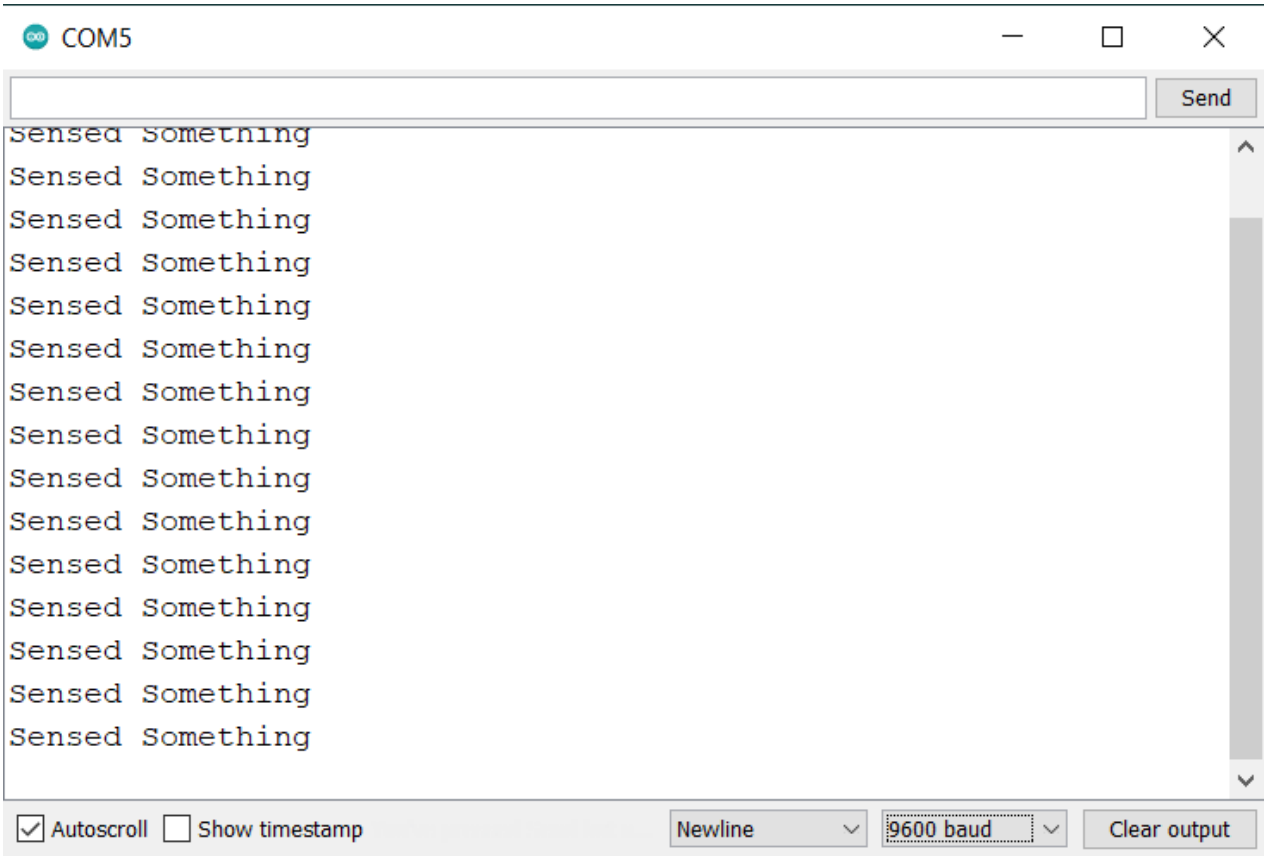


Fig. 1.31 Valoarea sesizata de senzorul PIR

F. Informatii

Senzor PIR vine de la pasiv infrarosu, cum ii spune si numele asa detecteaza el miscarea. Fiindca noi oamenii, dar si animalele emitem energie termica intr-o forma de radiatie infrarosie, iar senzorul PIR va detecta miscare.

Dupa cum puteti observa in “Serial Monitor” se afiseaza doar textul “Sensed Something”, pentru ca eu sunt prezent in incapere cu senzorul. Chiar daca raman nemiscat asta nu schimba cu nimic faptul ca eu voi emite in continuare caldura detectabila de senzorul PIR.

In interiorul capacelului alb se poate observa un senzor care se numeste “senzor piroelectric”. Acest senzor este acoperit de un capac alb numit “lentile Fresnel” care au rolul de a concentra semnalele infrarosii.

12. Magnetic sensor

A. Ce reprezinta acest proiect?

Ne vom familiariza cu un senor magnetic si vom invata cateva lucruri interesante despre el.

B. Componente necesare:

- Arduino Uno;
- Senzor magnetic
- Fire.

C. Diagrama

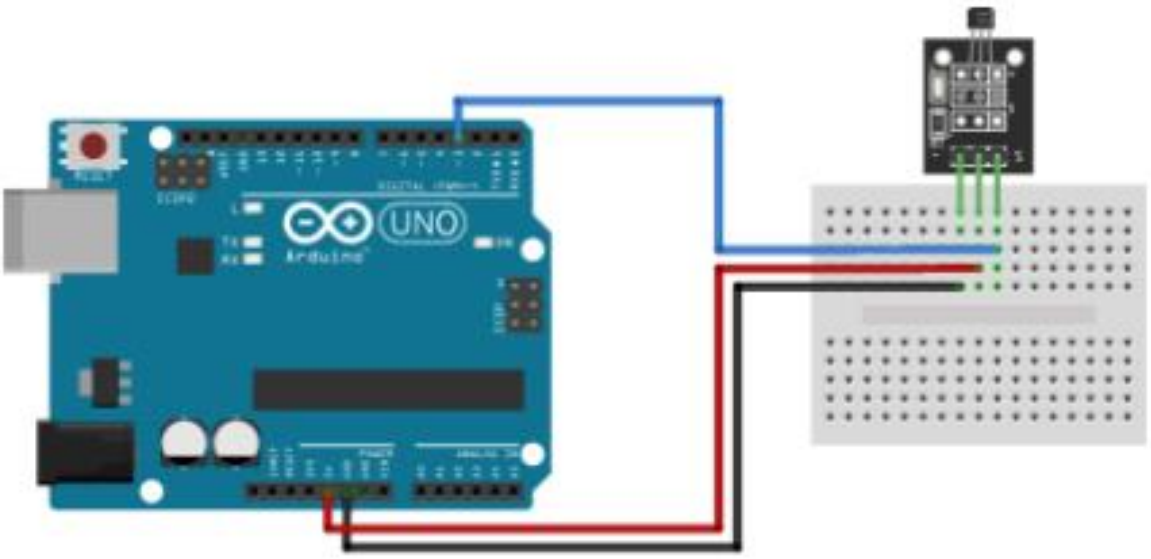


Fig. 1.32 Senzor magnetic

D. Conexiunea se face in felul urmator:

 Pinii GND si 5V ai senzorului vor ocupa pe placa Arduino pinii GND si 5V;

 Pinul de “out” va prelua D4.

E. Codul sursa

```
int hallSensorPin = 4;
int hallSensorValue = 0;
void setup() {
```

```
Serial.begin(9600);
pinMode(hallSensorPin,INPUT);
}
void loop() {
hallSensorValue = digitalRead(hallSensorPin);
Serial.print("hallSensorValue: ");
Serial.println(hallSensorValue);
delay(1000);}
```

F. Informatii

Un lucru foarte interesant cu acest senzor este faptul ca citeste valorile in “1” si “0”, adica in sitem binar. In momentul in care senzorul nostru sesizeaza un camp magnetic in jurul sau, in “Serial Monitor” ni se va afisa valoarea 0.

Depinde insa de puterea magnetului, cu cat este mai puternic cu atat senzorul il va sesiza de la o distanta mai mare. Daca magnetul este mai slab trebuie sa-l aducem mai aproape de senzor ca sa il citeasca.

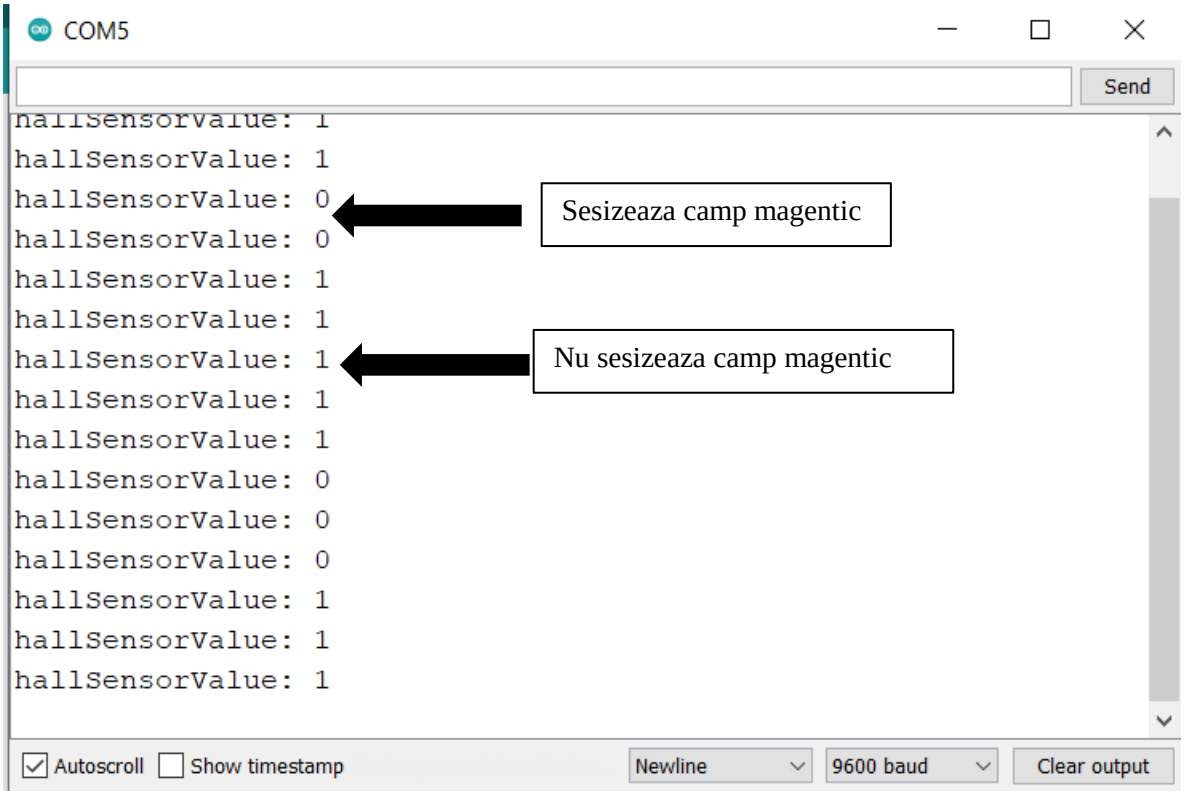


Fig. 1.33 Datele citite de senzor afisate in “Serial Monitor”

13. Termometru

A. Ce reprezinta acest proiect?

Este un circuit simplu, utilizam doar un senzor de temperatura si umiditate de tip DHT11. Codul va fi scris astfel incat sa citim si sa afisam valorile in “Serial monitor”.

B. Componente necesare:

- Arduino Uno;
- Senzor de temperature si umiditate(DHT11/DHT22);
- Fire.

C. Diagrama

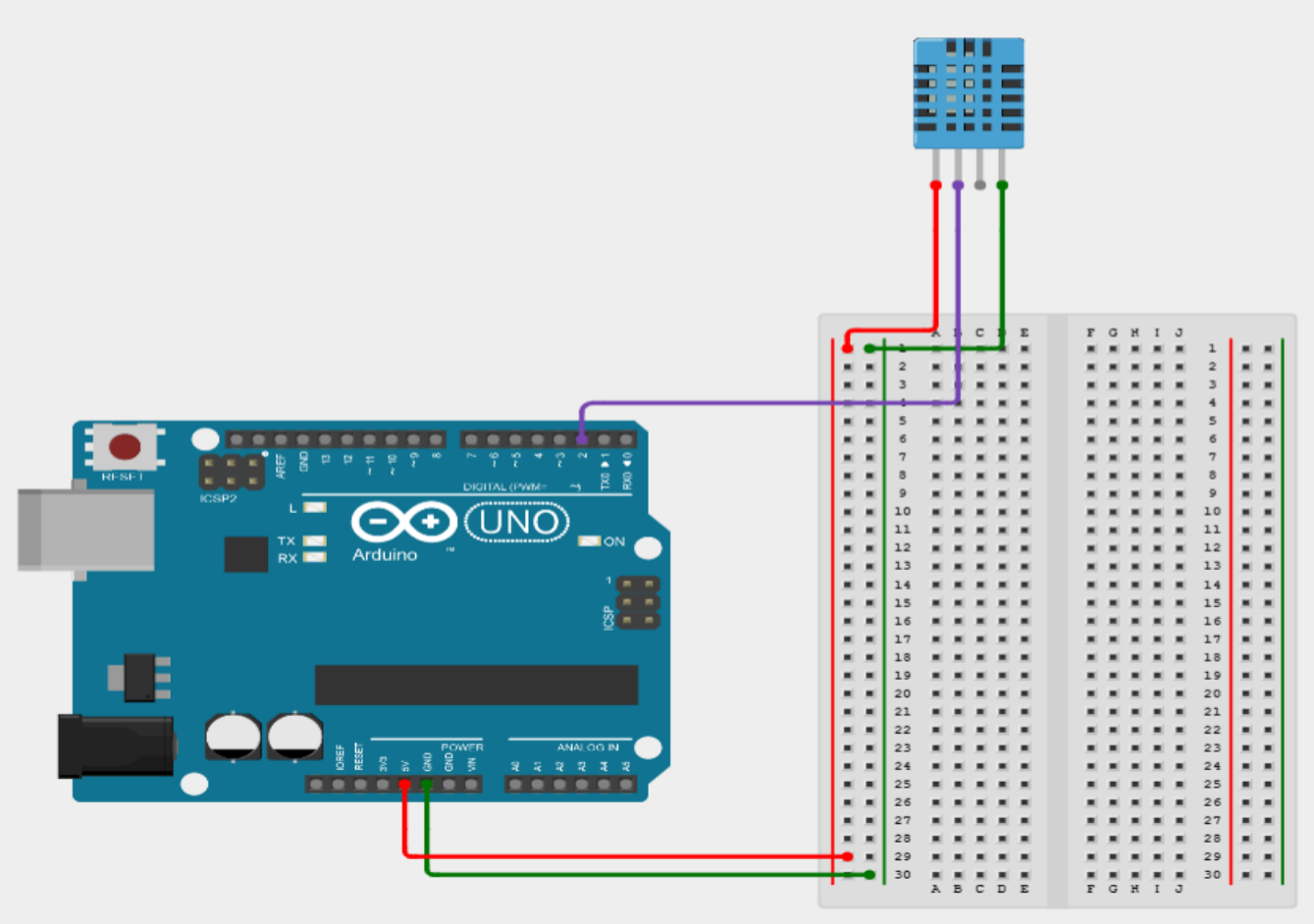
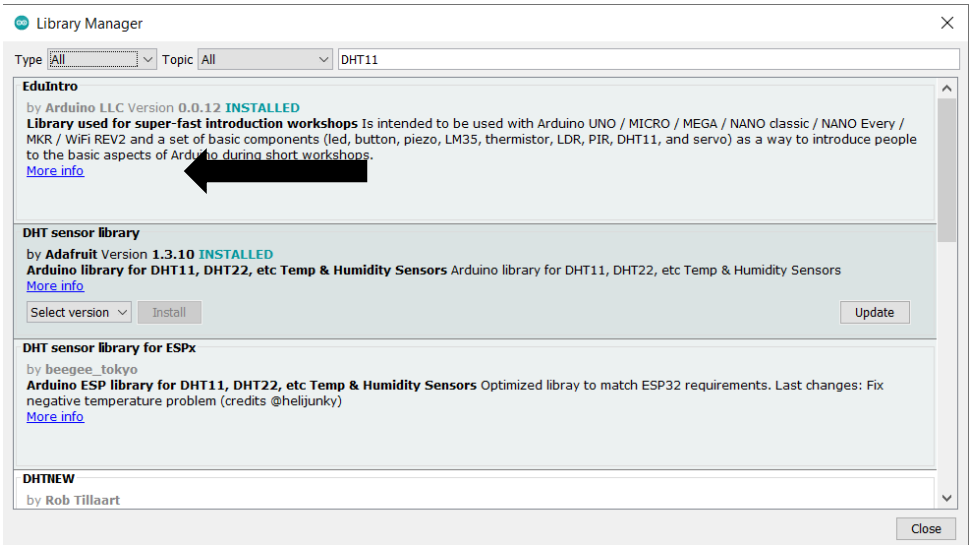


Fig. 1.34 Senzor de temperatura si umididate

D. Conexiunea se face in felul urmator:

 GND la GND, 5V la 5V, iar pinul de “out” il conectam la “D2”.



Inainde de a incepe sa scriem codul, trebuie sa instalam libraria pentru senzorul DHT.

Fig. 1.35 Instalarea librariei DHT

E.Codul sursa

```
#include "DHT.h" // Includem libraria DHT
#define DHTPIN 2     // Senzorul DHT il initializam pe pinul 2
#define DHTTYPE DHT11   // Definim tipul senzorului DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(9600);
  Serial.println(F("DHT11 test!"));
  dht.begin();
}
void loop() {
  // Delay de 2 secunde pentru masurarea temperaturii
  delay(2000);
//Citim umiditate
```

```
float h = dht.readHumidity();
// Citim temperature in grade Celsius
float t = dht.readTemperature();
// Citim temperature in Fahrenheit (isFahrenheit = true)
float f = dht.readTemperature(true);
// Verificam daca exista erori la citirea datelor
if (isnan(h) || isnan(t) || isnan(f)) {
  Serial.println(F("Failed to read from DHT sensor!"));
  return;
}
// Calculam indicele de caldura in Fahrenheit (implicit)
float hif = dht.computeHeatIndex(f, h);
// Calculam indicele de caldura in Celsius (isFahreheit = false)
float hic = dht.computeHeatIndex(t, h, false);
// Afisam valorile citite de senzor.(Umiditatea, temperatura in grade C si F)
Serial.print(F(" Humidity: "));
Serial.print(h);
Serial.print(F("%  Temperature: "));
Serial.print(t);
Serial.print(F("C "));
Serial.print(f);
Serial.print(F("F  Heat index: "));
Serial.print(hic);
Serial.print(F("C "));
Serial.print(hif);
```

```
  Serial.println(F("F"));
}
```

F. Informatii

Senzorul DHT este un senzor de temperatura si umiditate alcatuit dintr-un senzor de umiditate capacitive si un termistor. Este un dispozitiv usor de utilizat, dar necesita o sincronizare atenta pentru a culege informatiile.

Acum aveti propriul termometru in camera facut de voi. Super tare nu?! Se mai poate adauga eventual un ecran LCD pentru afisarea valorilor senzorului pentru a fi independent de calculator spre exemplu.

Limita acestor proiecte esti chiar tu insuti!

14. Senzor de alcool

A. Ce reprezinta acest proiect?

Acest proiect este foarte asemanator cu cel precedent. Acum vom citi anumite valori cu un dispozitiv care detecteaza nivelul de alcool din aer. Daca doriti sa va creati propriul etilotest aceasta este sansa voastra.

B. Componente necesare:

- Arduino Uno;
- Senzor alcool brick;
- Fire.

C.Diagrama

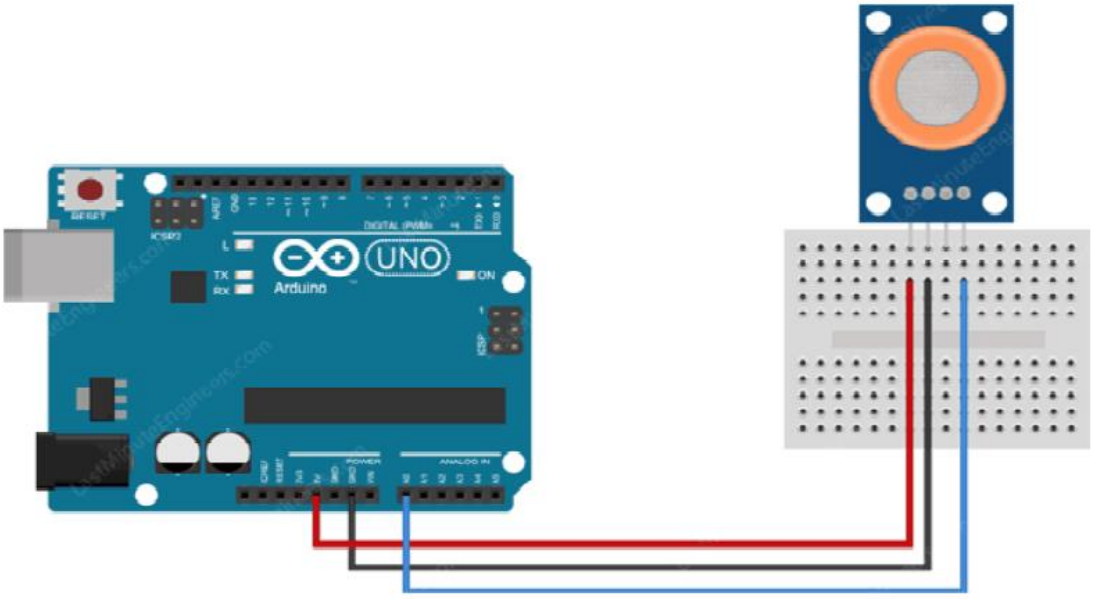


Fig. 1.36 Senzor de alcool

D. Conexiunea se face in felul urmator:

 GND la GND, 5V la 5V, iar “out” la A0;

E. Codul sursa

a.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int nivelAlcool = analogRead(A0);
  Serial.print("Nivelul de alcool este: ");
  Serial.println(nivelAlcool);
  delay(1000);}
```

b.

```
#define MQ3pin 0 //Definim senzorul MQ3
float sensorValue;  //variabila pentru valoarea senzorului
void setup() {
  Serial.begin(9600);
  Serial.println("MQ3 warming up!");  // Scriem in S.M. mesajul
  delay(20000); // Asteptam 20 de secunde pentru ca senzorul nostru sa se incalzeasca
}
void loop() {
  sensorValue = analogRead(MQ3pin); //Citim val. senzorului
  Serial.print("Sensor Value: "); // Afisam in S.M. val.
  Serial.println(sensorValue);
  delay(2000); // Asteptam 2 secunde pentru urmatoarea citire
}
```

c.

```
#define Sober 350   // Definim val. max in care nu suntem cu alcoolemie
#define Drunk 450   // Definim val. Min in care suntem beat
#define MQ3pin 0
float sensorValue;  //variabila pentru senzor
void setup() {
  Serial.begin(9600);
  Serial.println("MQ3 warming up!");
  delay(20000); // alocam 20 de secunde pentru incalzirea senzorului
}
void loop() {
  sensorValue = analogRead(MQ3pin); // Citim val. senzorului
```

```
  Serial.print("Sensor Value: ");
  Serial.print(sensorValue);
  // Afisam statusul
  if (sensorValue < Sober) {
    Serial.println("  |  Status: Stone Cold Sober");
  } else if (sensorValue >= Sober && sensorValue < Drunk) {
    Serial.println("  |  Status: Drinking but within legal limits");
  } else {
    Serial.println("  |  Status: DRUNK");
  }
  delay(2000); // Asteptam 2 secunde pentru urmatoarea citire
}
```

F. Informatii

In interiorul sau senzorul este actionat de un incalzitor. Dispozitivul este alcatuit din carcasa, in cazul nostrum partea portocalie, si din 2 straturi de plasa din otel inoxidabil fina, care se mai numeste si anti-exlozie, deoarece umblam cu alcool care este inflamabil.

Fiind un senzor analogic, chiar daca nu exista alcool in respiratia noastra valoarea senzorului nu va fii 0. Cat timp este sub 500 inseamna ca nu exista alcool in respiratie, ce trece de valoarea 500 inseamna ca senzorul nostru sesizeaza un anumit grad de alcool.

De obicei modelele de tip “brick” vin incorporate cu un mic potentiometru atasate de ele pentru a ajusta sensibilitatea dispozitivului.

Pentru o precizie mai buna de citire a informatiilor de pe senzor trebuie sa asteptam 24-48 de ore pentru incalzirea acestuia.

15. Senzor de distanta

A. Ce reprezinta acest proiect?

Cu ajutorul unui senzor ultrasonic(HC-SR04) vom masura distanta de la un obiect la dispozotiv.

B. Componente necesare:

- Arduino Uno;
- Senzor ultrasonic;
- Fire.

C. Diagrama

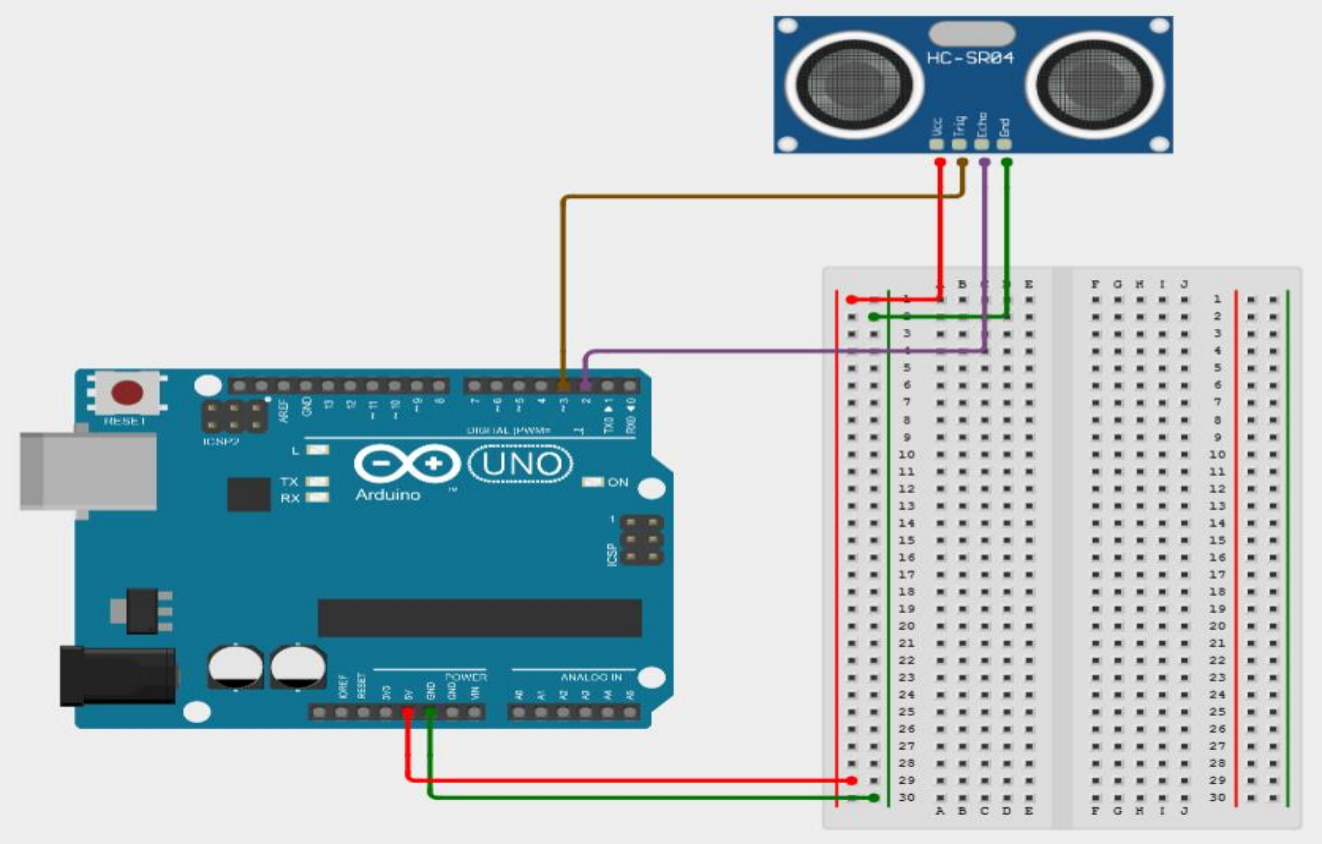


Fig. 1.37 Senzor ultrasonic

D. Conexiunea se face in felul urmator:

 GND la GND, 5V la 5V, Trig la D3 si Echo la D2.

E. Codul sursa

```
#define echoPin 2 // Pinu ECHO va prelua D2
#define trigPin 3 //Pinul TRIG va prelua D3
// definim variabilele
long duration; // variabilă pentru durata deplasării undelor sonore
int distance; // variabila pentru masurarea distantei
void setup() {
  pinMode(trigPin, OUTPUT); // Setam pinul TRIG ca OUTPUT
  pinMode(echoPin, INPUT); // Setam pinul ECHO ca INPUT
  Serial.begin(9600);
  Serial.println("Ultrasonic Sensor HC-SR04 Test"); // Scriem in S.M.
  Serial.println("with Arduino UNO R3");
}
void loop() {
  // Stergem conditia TRIG
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Setam pinul TRIG sa functioneze 10 milisecunde
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  //Citește echoPin, returnează timpul de deplasare al undei sonore în microsecunde
  duration = pulseIn(echoPin, HIGH);
  // Calculeaza distanta
  distance = duration * 0.034 / 2; // Viteza undei Sonore impartita la 2
  // Afisam distanta in S.M.
```

```
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
}
```

F. Informatii

Acest senzor ultrasonic(HC-SR04) este un senzor care poate masura distanta. Dispozitivul emite ultrasunete la 40 kHz care se deplaseaza in aer, iar daca undele acestea se “lovesc” de un obstacol se vor intoarce inapoi la senzor. Este alcatuit dintr-un transmitator (trimite semnal) si un receptor(primeste semnal).a

Distanta maxima pe care o poate masura acest tip de senzor este de 450 cm, la un unghi mai mic de 15 grade fata de senzori.

Exemplu:

Daca un obiect este situat la 20 de cm fata de senzor, iar viteza sunetului este de 340 m/s sau 0.034 cm/μs, unda sonora ar trebui sa se deplaseze cu 588 microsecunde. Insa ceea ce va citi pinul ECHO va fi o valoare dubla deoarece unda sonora trebuie sa se deplaseze inainte, dar trebuie sa si revina inapoi.

Formule:

v = 340 m/s; v – viteza d – distanta

v = 0.034 cm/μs; t – timpul s – viteza suntetului

t = d/s => 20/0.034 = 588 μs;

d = t * 0.034/2 cm

16. Pianul electronic

A. Ce reprezinta acest proiect?

Va voi arata cum sa faceti un pian folosind cateva butoane si un buzzer.

B. Componente necesare:

- Arduino Uno;

- 8 butoane;
- Buzzer;
- Fire.

C. Diagrama

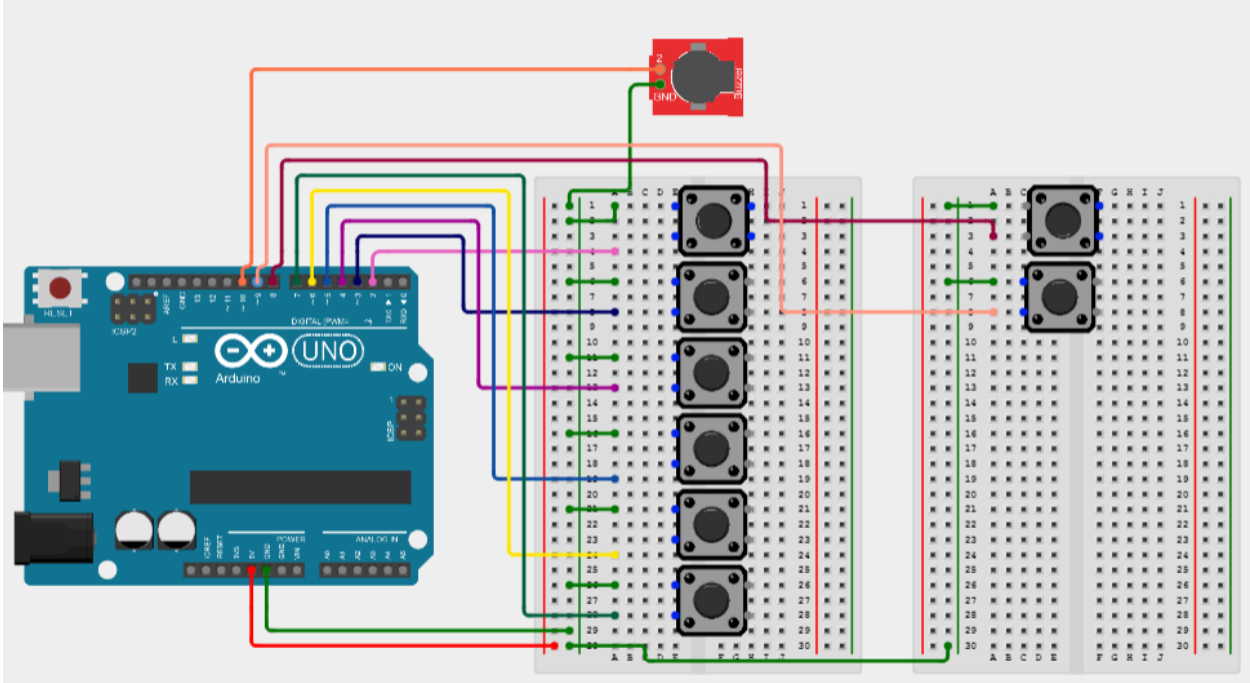


Fig. 1.38 Pianul electronic

D. Conexiunea se face in felul urmator:

Un pin de la fiecare buton il vom conecta la GND;

Pe rand in ordine crescatoare vom conecta celalalt pin al butonului incepand de la D3 pana la D9;

Buzzerul este conectat la GND si la pinul D10.

Pinul conectat la D2 este pentru a putea fi actionat ca pin de intrerupere.

E. Codul sursa

```
int val=0;

int buzzer = 10;

unsigned long on_time=0;
```

```
unsigned long off_time=0;
unsigned long button_ontime[20];
unsigned long button_offtime[20];
int button_seq[20];
int button1=3;
int button2=4;
int button3=5;
int button4=6;
int button5=7;
int button6=8;
int button7=9;
int button8=10;
int frequency[] = {300, 550, 800, 1000, 1250, 1400, 1550};
int buttonPin = 2;
int previousState = HIGH;
unsigned int previousPress;
volatile int buttonFlag;
int buttonDebounce = 20;
int path=1;
int i=0;
int led=13;
void playback (void);
void setup() {
Serial.begin(9600);
pinMode(buzzer,OUTPUT);
```

```
pinMode(led,OUTPUT);
pinMode(button1,INPUT_PULLUP);
pinMode(button2,INPUT_PULLUP);
pinMode(button3,INPUT_PULLUP);
pinMode(button4,INPUT_PULLUP);
pinMode(button5,INPUT_PULLUP);
pinMode(button6,INPUT_PULLUP);
pinMode(button7,INPUT_PULLUP);
pinMode(button8,INPUT_PULLUP);
pinMode(buttonPin,INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(2), button_ISR, CHANGE);
analogWrite(buzzer,0);
digitalWrite(led,HIGH);
}
void loop() {
if(path==0){
  Serial.println("playback");
  playback();
}
if((millis() - previousPress) > buttonDebounce && buttonFlag){
    previousPress = millis();
    if(digitalRead(buttonPin) == LOW && previousState == HIGH){
      path =! path;
      previousState = LOW;
    }
```

```
    else if(digitalRead(buttonPin) == HIGH && previousState == LOW){
      previousState = HIGH;
    }
    buttonFlag = 0;
  }
if(digitalRead(button1)==LOW){
  analogWrite(buzzer,frequency[0]);
  on_time=millis();
  if(i>0){
    button_offtime[i-1]=on_time-off_time;
  }
  while(digitalRead(button1)==LOW);
  if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=0;
    i++;
    Serial.println("button 1 stored");
  }
}
else if(digitalRead(button2)==LOW){
analogWrite(buzzer,frequency[1]);
on_time=millis();
if(i!=0)
button_offtime[i-1]=on_time-off_time;
```

```
while(digitalRead(button2)==LOW);
if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=1;
     i++;
     Serial.println("button 2 stored");}
}
else if(digitalRead(button3)==LOW){
analogWrite(buzzer,frequency[2]);
on_time=millis();
if(i!=0)
button_offtime[i-1]=on_time-off_time;
while(digitalRead(button3)==LOW);
if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=2;
     i++;
     Serial.println("button 3 stored");
  }
}
else if(digitalRead(button4)==LOW){
analogWrite(buzzer,frequency[3]);
on_time=millis();
```

```
if(i!=0)
button_offtime[i-1]=on_time-off_time;
while(digitalRead(button4)==LOW);
if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=3;
     i++;
     Serial.println("button 4 stored");
  }
}
else if(digitalRead(button5)==LOW){
analogWrite(buzzer,frequency[4]);
on_time=millis();
if(i!=0)
button_offtime[i-1]=on_time-off_time;
while(digitalRead(button5)==LOW);
if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=4;
     i++;
     Serial.println("button 5 stored");
  }
}
```

```
else if(digitalRead(button6)==LOW){
analogWrite(buzzer,frequency[5]);
on_time=millis();
if(i!=0)
button_offtime[i-1]=on_time-off_time;
while(digitalRead(button6)==LOW);
if(path==1){
    off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=5;
     i++;
     Serial.println("button 6 stored");
  }
}
else if(digitalRead(button7)==LOW){
analogWrite(buzzer,frequency[6]);
on_time=millis();
if(i!=0)
button_offtime[i-1]=on_time-off_time;
while(digitalRead(button7)==LOW);
if(path==1)
  {off_time=millis();
    button_ontime[i]=(off_time-on_time);
    button_seq[i]=6;
     i++;
```

```
    Serial.println("button 7 stored");}}
analogWrite(buzzer,0);}
void playback (void){
 digitalWrite(led,LOW);
 for(int j=0;j<i;j++){
  analogWrite(buzzer,frequency[button_seq[j]]);
  delay(button_ontime[j]);
  analogWrite(buzzer,0);
  delay(button_offtime[j]);}
 i=0;
 off_time=0;
 on_time=0;
 path=1;
 digitalWrite(led,HIGH);}
void button_ISR()
{buttonFlag = 1;}
```

F. Informatii

Stiu la ce te gandesti… este intr-adevar un cod lung care aparent nu face prea multe. Doar apasam niste butoane iar buzzerul scoate sunete la diferite frecvente. Totusi astfel inveti sa programezi, scriind iar si iar linii de cod. Dupa cum puteti observa sunt cam asemanatoare liniile de cod intre ele.

17. Controlarea motoarelor DC

A. Ce reprezinta acest proiect?

Vom scrie un cod astfel incat sa pornim motorul DC sa mearga inainte si inapoi pentru o secunda in fiecare directie (inainte si inapoi).

B. Componente necesare:

- Arduinio Uno;
- Shield pentru motoare (L298N);
- 2 motoare DC;
- 2 condensatoare ceramice (104);
- 1 baterie de 9V;
- Fire.

C. Diagrama

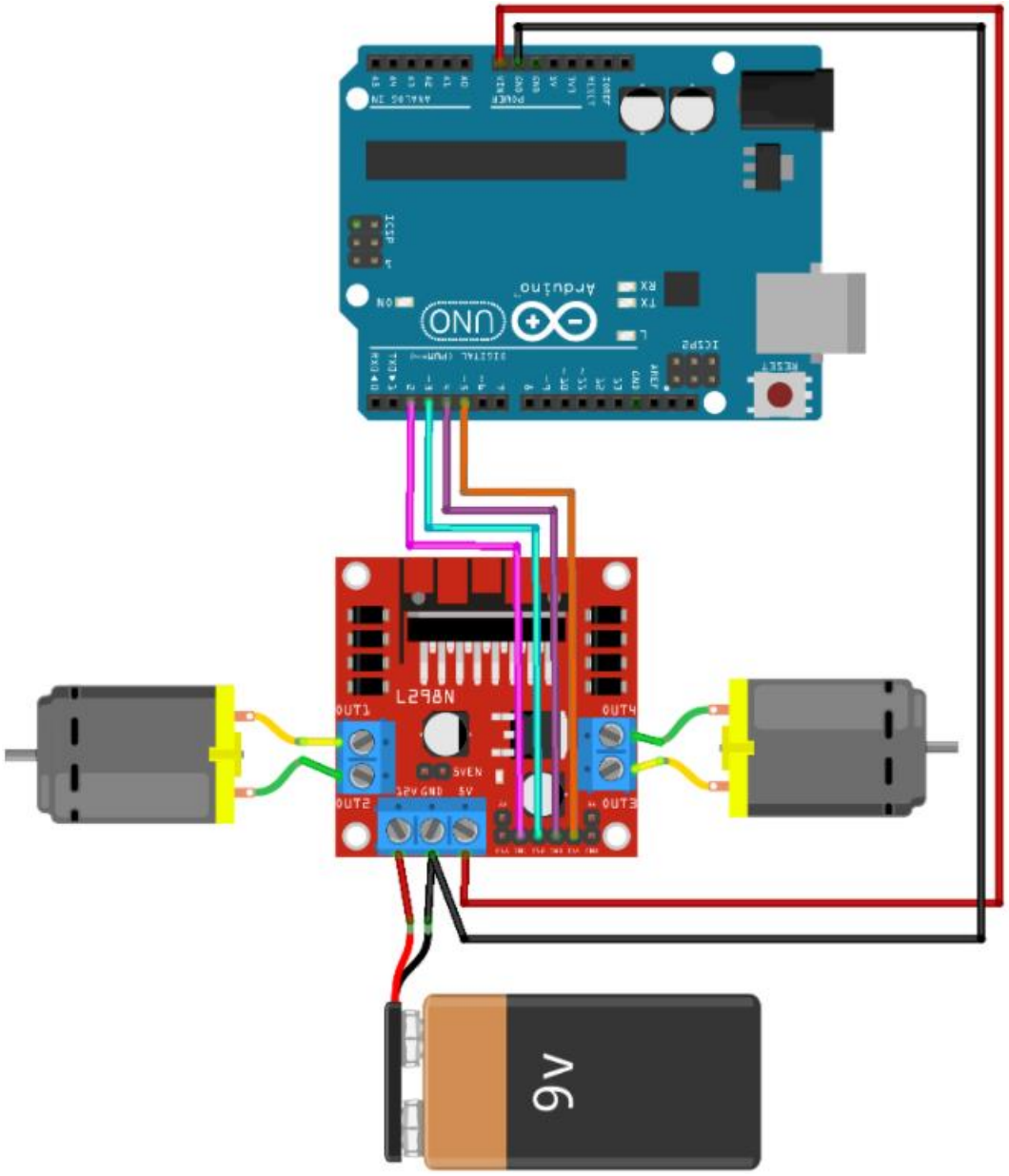


Fig. 1.39 Shield de motoare(L298) + 2 motoare DC

D. Conexiunea se face in felul urmator:

- Shield-ul L298 se insereaza in placa de dezvoltare Arduino Uno;
- Bateria se va introduce la Vin(+) si GND(-);
- Cele 2 motoare se vor conecta la “MOTOR1” si “MOTOR2”.

E. Codul sursa

```
int MOTOR2_PIN1 = 3; // Motorul 2 pinul 1 pe pinul 3 al placii
int MOTOR2_PIN2 = 5; // Motorul 2 pinul 2 pe pinul 5 al placii
int MOTOR1_PIN1 = 6; // Motorul 1 pinul 1 pe pinul 6 al placii
int MOTOR1_PIN2 = 9; // Motorul 1 pinul 2 pe pinul 9 al placii
void setup() {
  pinMode(MOTOR1_PIN1, OUTPUT);
  pinMode(MOTOR1_PIN2, OUTPUT);
  pinMode(MOTOR2_PIN1, OUTPUT);
  pinMode(MOTOR2_PIN2, OUTPUT);
  Serial.begin(9600);
}
void loop() {
    go(255,-255);
    delay(1000);
    go(-255,-255);
    delay(1000);
    go(-255,255);
    delay(1000);
```

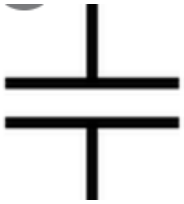
```
   go(255,255);
   delay(1000);
}
void go(int speedLeft, int speedRight) {
  if (speedLeft > 0) {
    analogWrite(MOTOR1_PIN1, speedLeft);
    analogWrite(MOTOR1_PIN2, 0);}
  else {
    analogWrite(MOTOR1_PIN1, 0);
    analogWrite(MOTOR1_PIN2, -speedLeft);}
  if (speedRight > 0) {
    analogWrite(MOTOR2_PIN1, speedRight);
    analogWrite(MOTOR2_PIN2, 0);
  }else {
    analogWrite(MOTOR2_PIN1, 0);
    analogWrite(MOTOR2_PIN2, -speedRight);}}
```

F. Informatii

Un shield Arduino este o placa de circuit care asigura o functionalitate speciala placii pentru a spori capacitatile Arduino. In cazul nostru un shield de motoare (L298N) asigura controlul motoarelor. Ea se introduce direct in placa de dezvoltare si este gata pentru programat.

Am scris ca in componentele de care avem nevoie se afla si 2 condensatori ceramici. Ei se vor conecta la bornele motorului DC pentru a inmagazina energie electrica.

Simbol

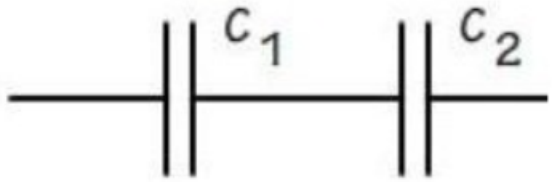


Un condensator este un dispozitiv care stocheaza energie electrica intr-un camp electric. Unitatea de masura pentru aceasta componenta este Faradul(F). Un condensator ramane incarcat cu energie electrica chiar si dupa deconectarea sursei, asa ca mare atentie cand folositi un astfel de dispozitiv pentru ca va puteti curenta. Mereu sa verificati daca este descarcat!

Folosim o astfel de componenta la motoarele DC pentru ca motorul sa isi extraga dintr-o data toata toata energia inmagazinata in el. Este ca si cum ai fii obosit, esti epuizat, dar mai poti functiona normal intr-o oarecare masura, in schimb daca bei cafea sau o bautura energizanta iti dai un impuls pentru a mai functiona o perioada de timp in parametrii normali si tot asa.

Atunci cand incerci sa realizezi astfel de proiecte sunt mari sanse sa incurci directia de deplasare a motorului. Asta nu necesita sa schimbi codul, cel mai usor lucru pe care poti sa il faci este sa schimbi cele 2 de la motor fire intre ele.

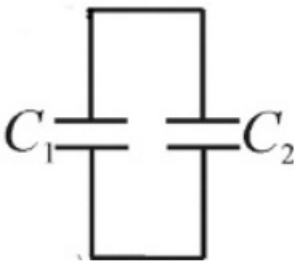
Formula pentru a calcula valoarea a doua condensatoare legate in serie este:



$1/C_T = 1/C_1 + 1/C_2$

Formula generala este:

$1/C_T = 1/C_1 + 1/C_2 + \ldots + 1/C_n$



Formula de calcul pentru 2 condensatoare in paralel este:

$C_T = C_1 + C_2$

Formula generala este:

$C_T = C_1 + C_2 + \ldots + C_3$

Dupa cum putem observa aceste formule sunt exact opuse cu cele de la rezistenta.

18. Utilizarea unui motor stepper(pas cu pas)

A. Ce reprezinta acest proiect?

Vom programa un motor stepper(pas cu pas) sa se roteasca cu 1,8 grade in directia acelor de ceasornic si cu 3,6 grade in sens invers.

B. Componente necesare:

- Arduino Uno;
- Motor stepper;
- Modul stepper ULN2003;
- Fire.

C. Diagrama

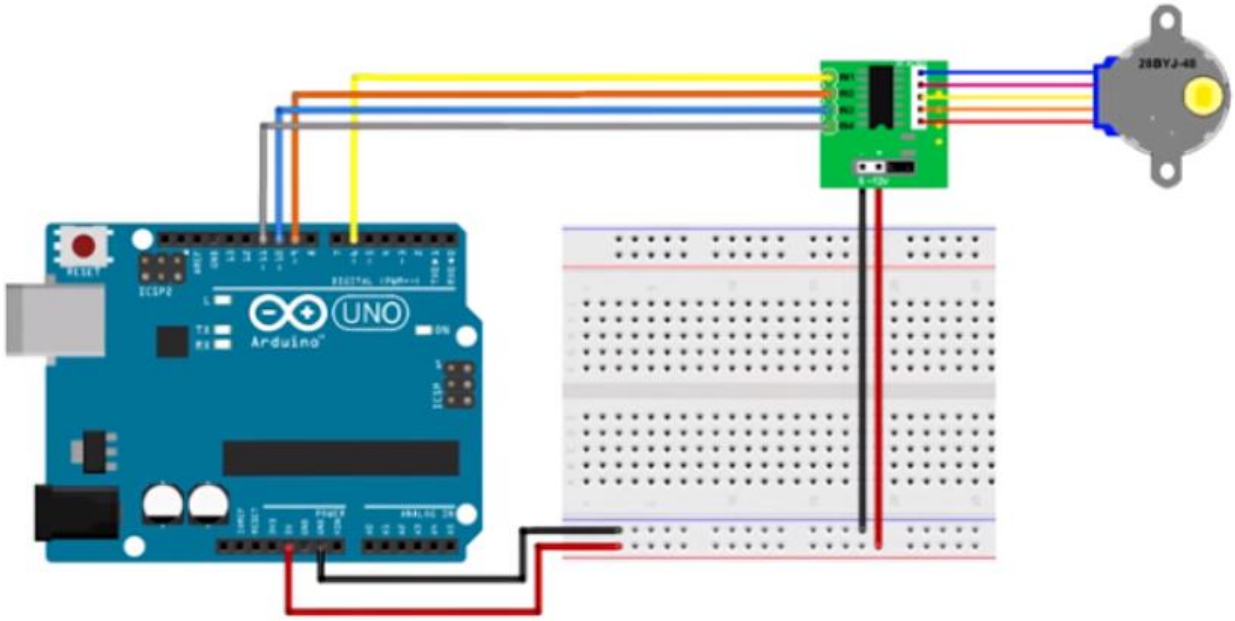


Fig 1.40 Step motor + modul UNL2003

D. Conexiunea se face in felul urmator:

 Introducem firele de la motorul stepper in modului UNL2003;

 Dupa care conectam “IN1” la D7, “IN2” la D9, “IN3” la D10, “IN4” la D11;

 Primii 2 pini de alimentare se vor conecta la GND si 5V.

E. Codul sursa

```
#include <Stepper.h>  //Includem libraria
```

```
int StepNum=2050;   // Numărul de pași ai motorului. Verificați fișa tehnică a motorului și înlocuiți-l dacă este necesar

Stepper SurtrStepper(StepNum,9,11,10,6); //Definim pinii pentru motor

void setup() {

    SurtrStepper.setSpeed(9); //Viteza motorului este de obicei 9

}

void loop() {

   SurtrStepper.step(200);  //Pozitiv si negativ isi schimba directia

   SurtrStepper.step(-400);

}
```

F. Informatii

Un motor pas cu pas este un dispozitiv electromecanic care transforma impulsuri electrice in miscari mecanice. Aceste motoare sunt folosite in special la imprimantele 3D, datorita preciziei lor.

19. Controlul unui servo motor

A. Ce reprezinta acest proiect?

Vom programa un servo motor astfel incat sa se roteasca 180 de grade si sa revina la pozitia initiala.

B. Componente necesare:

- Arduino Uno;
- Servo motor;
- Fire.

C. Diagrama

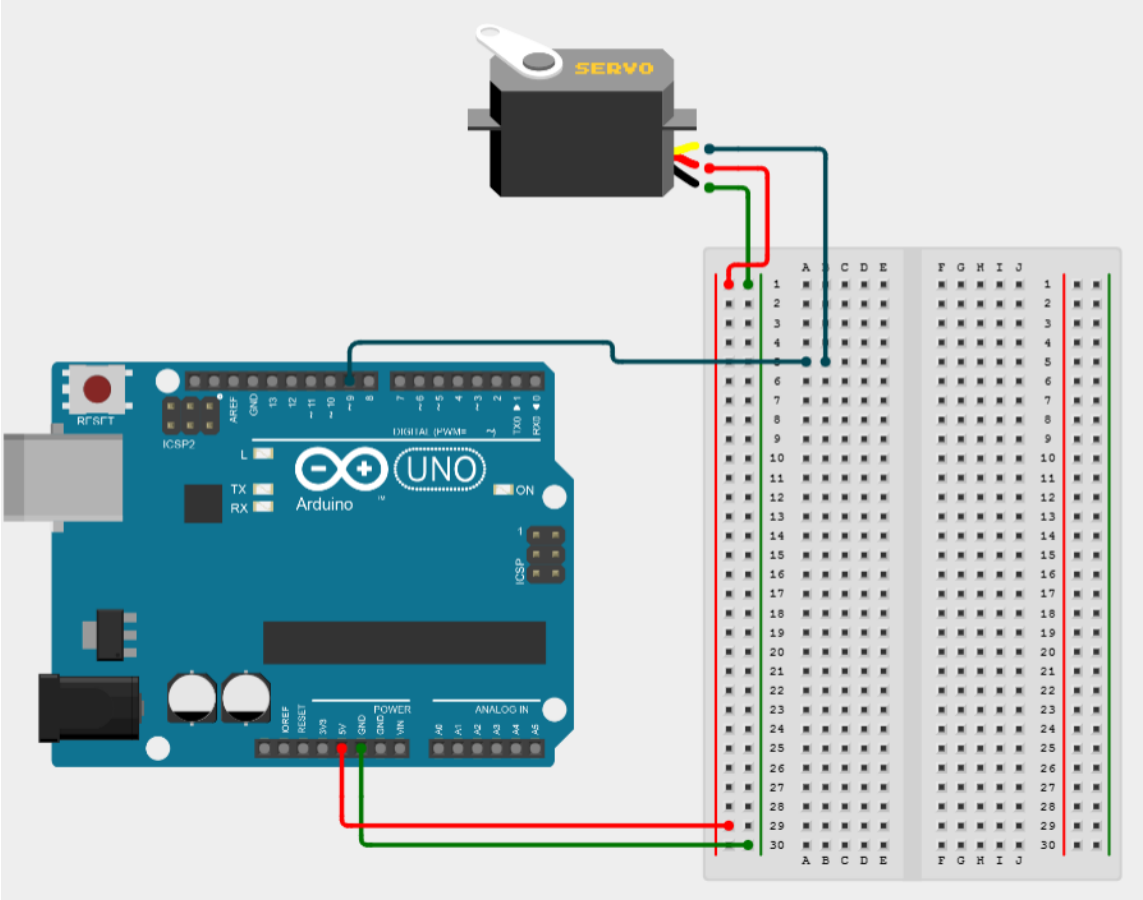


Fig. 1.41 Micro servo

D. Conexiunea se face in felul urmator:

 Firul maro/negru la GND, firul rosu la 5V, iar ultimul fir ramas il conectam la D9.

E. Codul sursa

```
#include <Servo.h>
Servo myservo;  // Cream un obiect servo pentru a controla un servo
// 12 obiecte servo pot fi create pe majoritatea panourilor
int pos = 0;    // variabila pentru pozitionarea servo motorului
void setup() {
  myservo.attach(9);  // Atasam servo pe pinul 9
```

```
}
void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // Merge de la 0-180 de grade
    myservo.write(pos);  // Spune servo sa mearga pe pozitia variabilei pos
    delay(15);}
  for (pos = 180; pos >= 0; pos -= 1) { // opusul actiunii primului ciclu for
    myservo.write(pos);
    delay(15);}}
```

20. Sursa de alimenare

A. Ce reprezinta acest proiect?

Vreau sa va exemplific cum functioneaza un HW-131(sursa de alimentare). Asa ca vom porni un LED folosind aceasta placuta de alimentare fara a scrie cod.

B. Componente necesare:

- HW-131
- Breadboard;
- LED
- Rezistor 220Ω
- Baterie 9V;
- Fire.

C. Componentele HW-131

1 – Sursa de alimentare Jack

2 – Buton ON/OFF

3 – Sursa de alimentare USB

4 si 5 – Sursa de 0/3.3/5 V

6 – Surse suplimentare

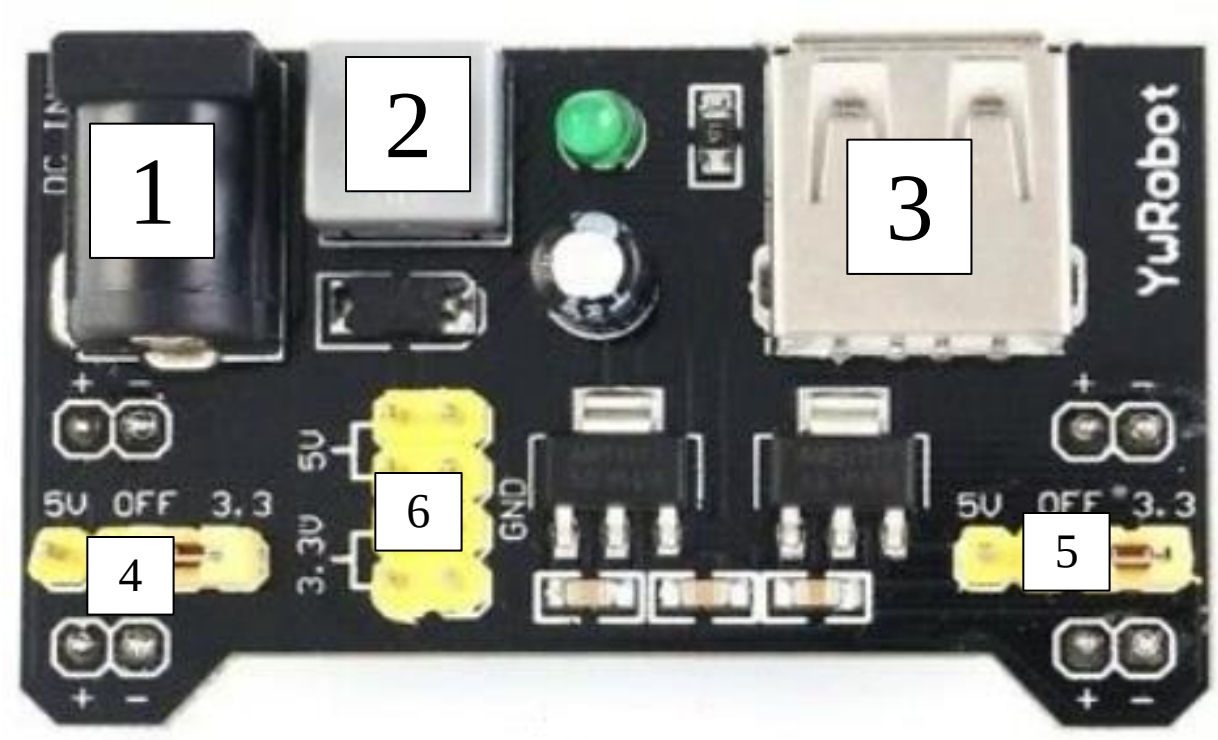
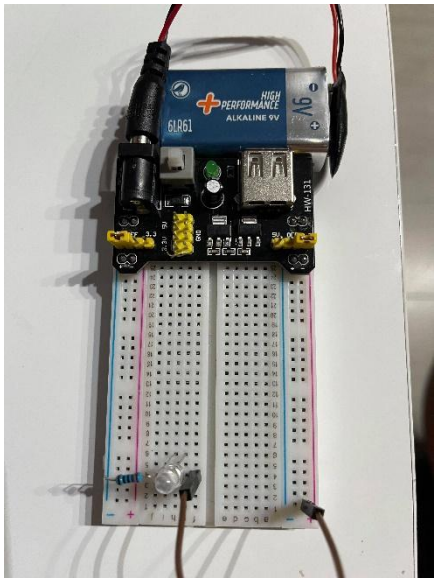


Fig. 1.42 Sursa de alimentare



Inserati sursa de alimentare in bread board. Introdu bateria de 9V in mufa jack(atentie ca butonul sa fie pe pozitia deschis, daca butonul imi inchide circuitul atunci LED-ul verde de pe placa va fi pornit). Pozitioneaza un LED pe breadboard iar apoi la anod ii pui o rezistenta de 220Ω care se conecteaza la 5V, iar catodul la GND. Acum puteti actiona butonul pentru a inchide circuitul, astfel LED-ul va lumina

Astfel ai reusit sa faci un circuit simplu fara a scrie cod. Felicitari!

Fig. 1.43 LED + sursa de alimentare

21. Logic level converter bidirectional

A. Ce reprezinta acest proiect?

O sa va explic cum functioneaza un convertor bi-directional.

B. Componente necesare:

- Arduino Uno;
- Convertor logic bi-directional;
- Multimetru;
- Fire.

C. Diagrama

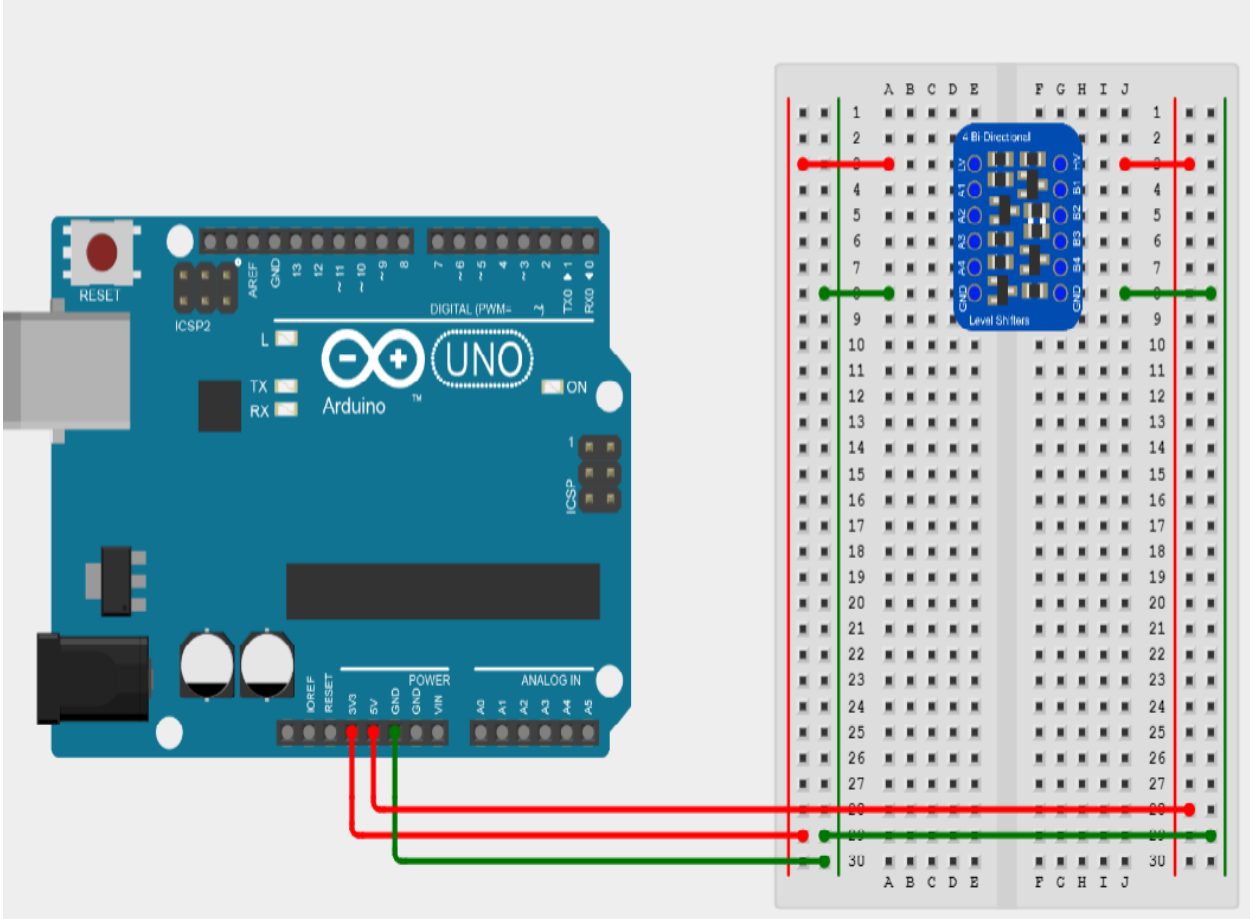
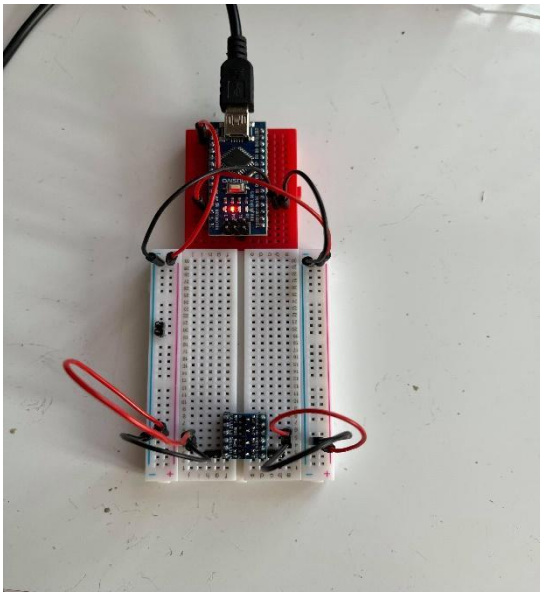


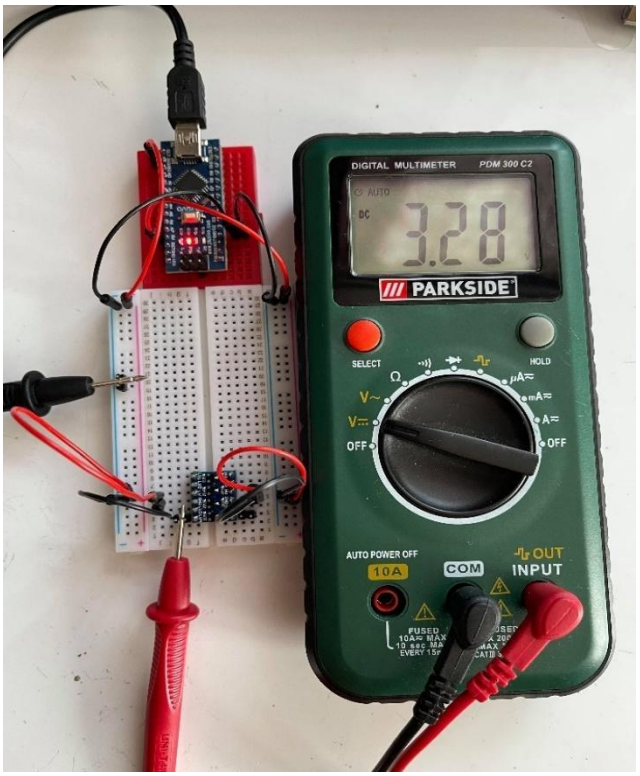
Fig. 1.44 Logic level converter bidirectional



In acest proiect eu folosesc un Arduino Nano. Nu trebuie sa scriem nici un cod aici. Un convertor bidirectional are 2 parti cu cate 6 pini fiecare parte. Partea pe care vedem initialele “LV” vine de la “low voltage”, iar HV “high voltage”. Practic aceasta componenta imi converteste de la 3.3V la 5V si invers.

In figurile de mai jos sunt afisate masuratorile facute cu multimetrul. Valorile acestora nu sunt fixe deoarece exista o anumita toleranta deoarece cum spuneam la inceput pana si un fir are o rezistenta.

Fig. 1.45 Cablarea convertorului bidirectional



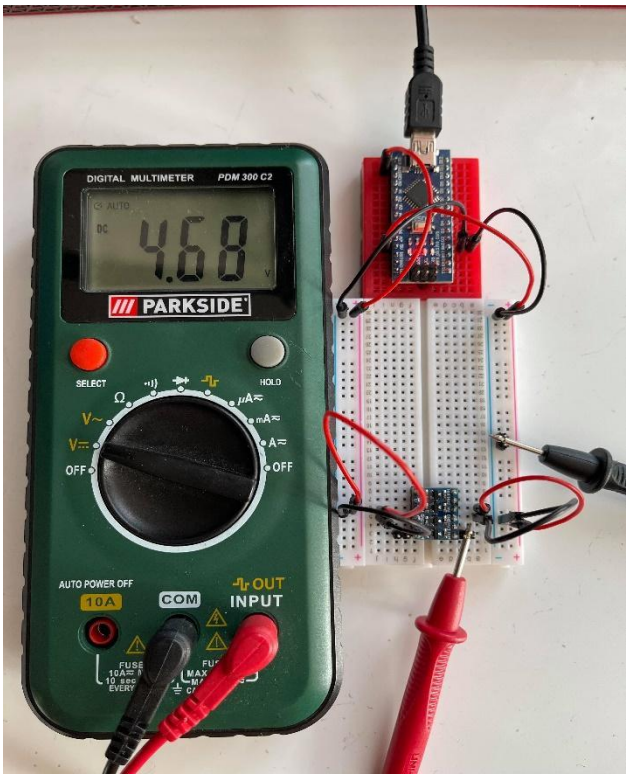


Fig. 1.46 + Fig. 1.47 Masuratori cu multimetrul

22. Detectarea nivelul de apa dintr-un recipient

A. Ce reprezinta acest proiect?

Vom face un detector de nivel de apa dintr-un containter. Acest proiect nu necesita codare, dar folosim un Arduino Uno pentru pinii de 5V si GND.

B. Componente necesare:

- Arduino Uno;
- 3 tranzistori NPN(BC174A); //ar trebui sa functioneze si cu 2N2222
- 3 rezistori (2.2kΩ);
- 3 rezistori (220Ω);
- Fire.

C.Conectarea se face in felul urmator:

 Cei trei tranzistori(NPN) se plaseaza in breadboard.

 La bazele acestora se vor pune rezistete de 2.2kΩ, iar apoi cate un fir trasat de la baza se introduce in container la diferite inaltimi.

 Firele trebuie sa fie dintr-un material conductor.

 In functie de nivelul acestora in compartiment se va detecta nivelul de apa.

 Un fir este conectat la 5V se va amplasa pe fundul recipientului.

 La fiecare colector este amplasat cate o rezistenta de 220Ω.

 LED-ul se va conecta cu anodul la GND si cu catodul la colector.

D. Diagrama

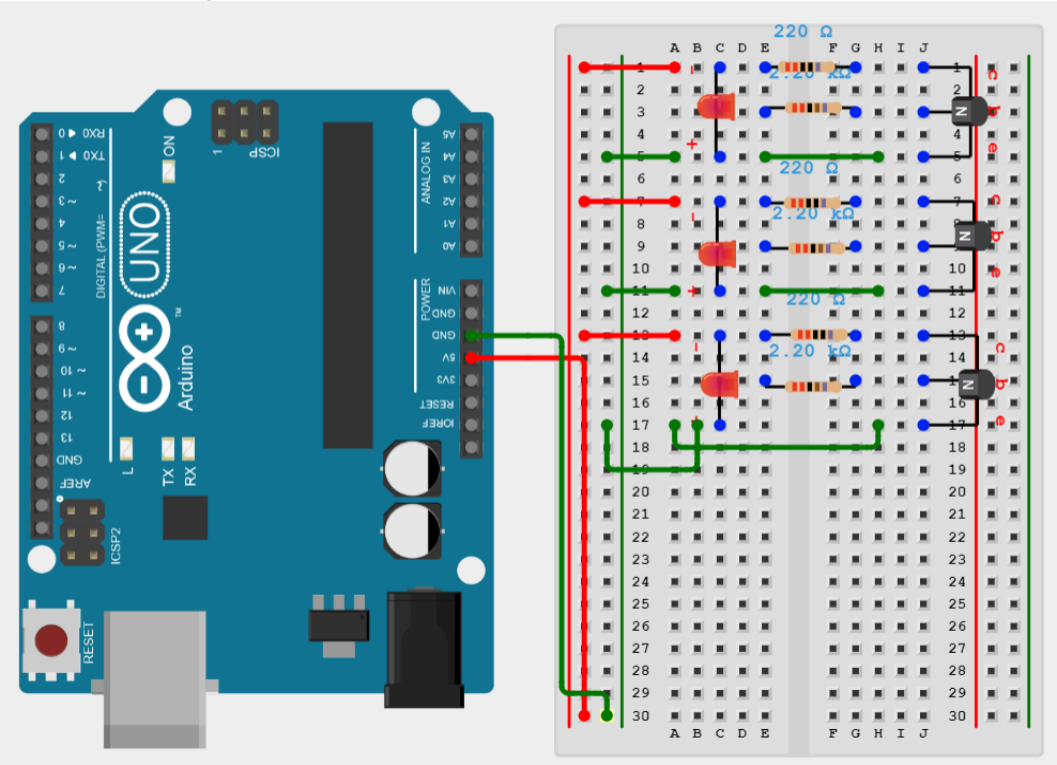


Fig. 1.48 Cablarea tranzistorilor si a LED-urilor

E. Informatii

Acesta este un proiect simplu de realizat si interactiv. Poti folosi alt tip de tranzistor daca doresti atata timp cat cauti in datasheet informatiile pentru a-l utiliza.

Avem un recipient gol cu 4 fire in el la diferite inaltimi. Cel care nu este conectat la baza unui tranzistor acela se va pozitiona exact pe fundul bazinului. Pe masura ce umplem recipientul cu apa atunci cand lichidul atinge un fir care este atasat de peretele bazinului ne indica nivelul apei cu ajutorul LED-ului care se va aprinde.

Foarte important ca firele sa fie din material conductor.

23. Ghiveciul inteligent

A. Ce reprezinta acest proiect?

Practic vom face un dispozitiv care te atentioneaza atunci cand o floare spre exemplu are nevoie de apa.

B. Componente necesare:

- Arduino Uno;
- Senzor pentru sol;
- Releu;
- Fire.

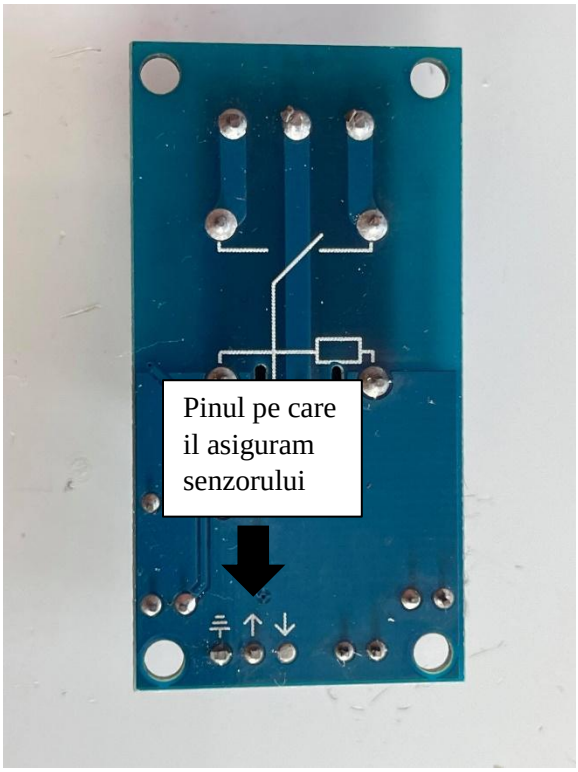


Fig. 1.49 Pinul de “out”

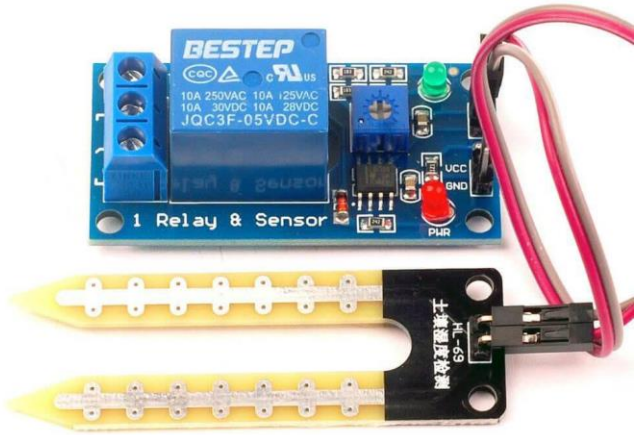


Fig. 1.50 Senzorul de sol

Imi cer iertare, dar din pacate nu am reusit sa realizez o diagrama cu astfel de dispositive, asa ca va voi scrie cum se fac conexiunile intre ele.

 Cu fire de tip mama-mama vom conecta senzorul de sol la releu astfel;

 Unul din fire il vom conecta la impamantarea releului;

 iar celalalt la VCC;

De pe placa Arduino conectam la releu astfel;

 GND la GND;

 VCC la 5V;

 Pinul asigurat pentru senzorul de sol(Fig. 1.49) il conectam la placa Arduino la pinul A0.

C.Codul sursa

```
// cu cat este mai ud solul senzorul va lua o valoare mai mica
#define led 12 //Definim LED-ul
#define ALARM 450 //Definim valoarea maxima
void setup() {
  Serial.begin(9600);
  pinMode(led, OUTPUT);
}
void loop() {
  int v = analogRead(A0); // Citim valoarea senzorului
//Daca val. senzorului < val. max atunci LED-ul se aprinde
  if (v > ALARM){digitalWrite(12,HIGH);}
  else{digitalWrite(12,LOW);}
  delay(3000);
  Serial.println(v); //Afisam valoarea in S.M.
}
```

D. Informatii

Poti face un proiect interesant de irigare cu un astfel de senzor. Mai poti adauga eventual o pompa de apa ca sa ude in locul tau planta.

Eu am scris codul ca sa iti exemplific cum functioneaza acest tip senzor si cum se scrie un cod usor pentru el. Insa tu poti deveni un programator-gradinar. De exemplu sa vezi ce fel de sol trebuie unei anumite plante cata apa are nevoie, sa incerci sa faci tu cercetari acasa si sa faci tot felul de experimente. Sa observi de cata apa/soare/caldura etc are nevoie planta ta.

Sper ca ti-am dat o idee de proiect care sa te puna pe ganduri.

24. Rover Avoiding Stuff(RAS)

A. Ce reprezinta acest proiect?

Este vorba despre o masinuta care evita obstacolele din calea ei folosind diferite dispozitive care sunt compatibile cu placa de dezvoltare Arduino Uno.

B. Componente necesare:

- Sasiu;
- Arduino Uno;
- Shield pentru motoare(L293D);
- 2 motoare
- Servo motor;
- Senzor ultrasonic;
- Buton switch;
- 2 condensatoare;
- Baterie 9V;
- Roti;
- Fire.

Introducem shield-ul pentru motoare in placa Arduino, iar apoi se vor conecta astfel componentele intre ele:

C. Diagrama

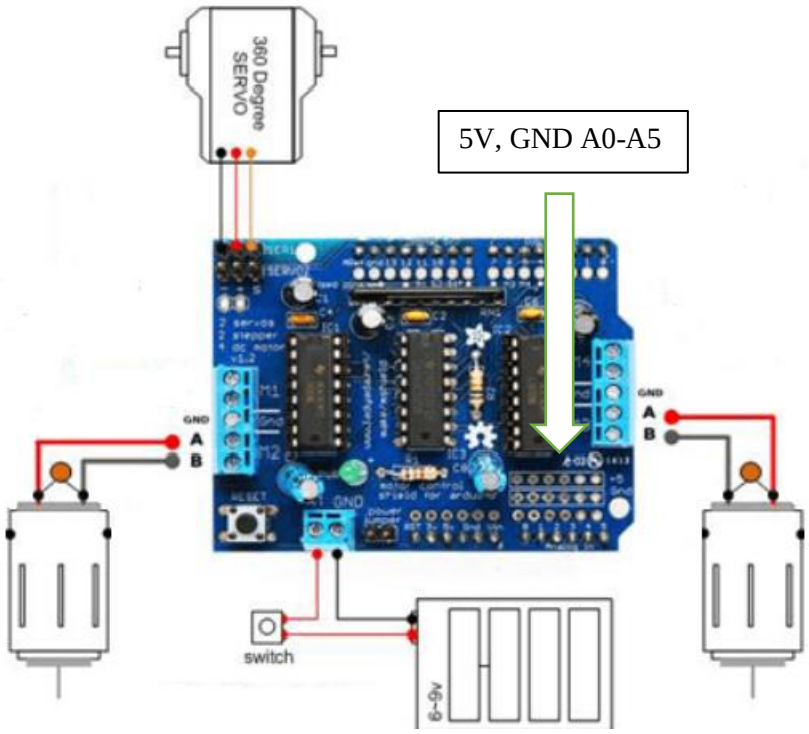


Fig. 1.51 Shield L293D conexiuni

Senzorul ultrasonic va fi pozitionat deasupra servo motorului se se va conecta pe shield astfel: GND □ GND, ECHO □ A1, TRIG □ A0, VCC □ 5V.

Citirea pinilor analogici pentru senzorul ultrasonic se face de la dreapta la stanga, adica pinul din dreapta jos va fii A0 apoi A1 si tot asa pana la A5.

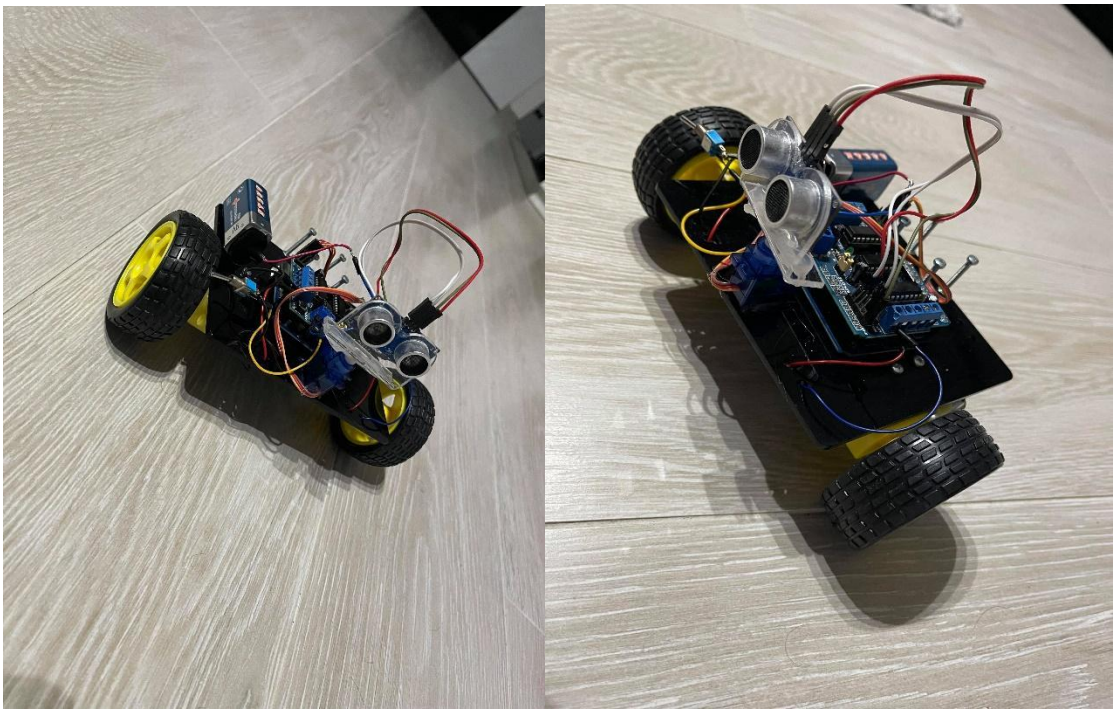


Fig. 1.52 Avoiding stuff car

D.Codul sursa

```
#include "AFMotor.h"  // includem libraria
#include <Servo.h>
#define echopin A4 // echo pin pe A4
#define trigpin A5 // Trigger pin pe A5
Servo myservo;
const int MOTOR_2 = 2; // pini pentru motor
const int MOTOR_3 = 3;
AF_DCMotor motor2(MOTOR_2, MOTOR12_64KHZ);
AF_DCMotor motor3(MOTOR_3, MOTOR12_64KHZ); // cream obiect pentru motor, 64KHz pwm
int distance_L, distance_F, distance_R; // L-left, F-forward, R-right
long distance;
int set = 20;
```

```
void setup() {
Serial.begin(9600);
  Serial.println("Start");
  myservo.attach(10); //Servo motorul pe pinul 10
  myservo.write(100);  // Unghiul la care sa se pozitioneze din start
  pinMode (trigpin, OUTPUT);
  pinMode (echopin, INPUT );
  // Setarea motorului cu val intre 0-255
  motor2.setSpeed(120);
  motor3.setSpeed(120);
}
void loop() {
 distance_F = data();
 Serial.print("S=");
 Serial.println(distance_F);
  if (distance_F > set){
   Serial.println("Forward");
  // Ambele motoare pornesc in aceiasi directie
  motor2.run(FORWARD);
  motor3.run(FORWARD);
    }
    else{hc_sr4();}
}
//Citim datele de la senzorul ultrasonic
long data(){
```

```
  digitalWrite(trigpin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  distance = pulseIn (echopin, HIGH);
  return distance / 29 / 2;
}
//Comparam distanta dintre dreapta si stanga
void compareDistance(){
  if (distance_L > distance_R){
   // Intoarce la stanga
  motor2.run(BACKWARD);
  motor3.run(FORWARD);
    delay(350);
  }
  else if (distance_R > distance_L){
 // Intoarce la dreapta
  motor2.run(FORWARD);
  motor3.run(BACKWARD);
    delay(350);
  }
  else{
  // Merge cu spatele
  motor2.run(BACKWARD);
  motor3.run(BACKWARD);
```

```
   delay(300);
  // Vireaza la stanga
  motor2.run(BACKWARD);
  motor3.run(FORWARD);
    delay(500);
  }
}
void hc_sr4(){
    Serial.println("Stop");
    motor2.run(RELEASE);  //Pauza la motoare
    motor3.run(RELEASE);
    myservo.write(0); // intors la 0 de grade
    delay(300);
    distance_R = data(); // distanta dreapta
    delay(100);
    myservo.write(170); // intors la 170 de grade
    delay(500);
    distance_L = data(); // distanta stanga
    delay(100);
    myservo.write(90); //intors la 90 de grade
    delay(300);
    compareDistance();
}
```

E. Informatii

Aceasta masinuta este una obisnuita in randul oamenilor carora le place sa faca proiecte cu Arduino. Ii pune la incercare atentia mai ales, deoarece trebuie sa fii

foarte precis si exact in momentul in care incepi sa il construiesti de la 0. Sunt sanse sa incurcati firele sau sa gresiti ceva in program, desi nu exista nici o eroare vizibila. Dupa parerea mea, sa faci o astfel de masinuta este chiar foarte eficient pentru cei ce vor sa invete despre acest domeniu, iar proiectul asta este un bun inceput.

Anual se organizeaza evenimente care necesita proiecte facute in Arduino. Aceste concursuri se organizeaza pe mai multe categorii precum: roboti line-follower, avoiding stuff, free style etc.

Depinde in ce categorie vrei sa participi sau care te pasioneaza. Ar fi de preferat sa incerci intr-o oarecare masura sa faci cumva sa participi la un astfel de eveniment pentru ca te va ajuta pe tine mai mult in viitor.

Cu cat vei exersa mai mult cu atat mai priceput sim ai remarcat vei deveni. Aceste proiecte chiar daca sunt simple tu poti oricand sa le imbunatatesti si vei iesi in evidenta fata de alte persone cand este vorba de un loc de munca.

Pentru ca masinuta sa functioneze nu uitati sa conectati si jumper-ul la locul lui pentru a inchide circuitul.