

Proiectare cu Microprocesoare

Curs 9

Motoare

An 3 CTI

Semestrul I

Lector: Răzvan Itu

Motoare de curent continuu (DC)

Motor DC cu reductor (cutie de viteze)

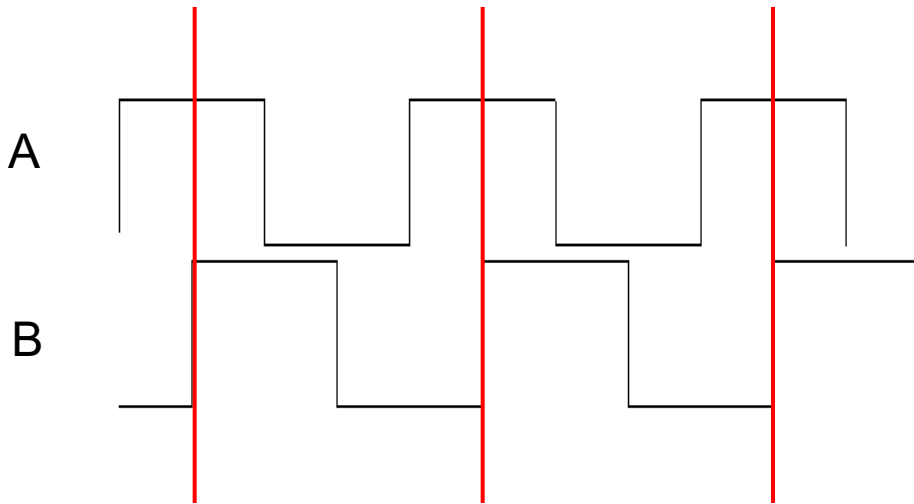
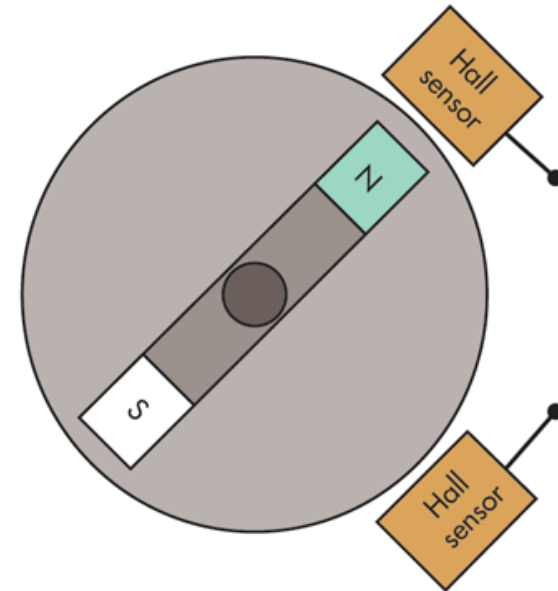
- Motor clasic DC, viteza e data de tensiune, direcția de polaritate
- Rotație continuă, cât timp motorul este sub tensiune
- Cutie de viteze (angrenaj de roți dințate) cu diferite rapoarte (1:19, 1:53, 1:48, etc)
 - Majorare a forței (cuplu) în dauna vitezei de rotație



Motoare de curent continuu (DC)

Măsurarea turației motorului

- Senzor Hall (magnetic) în cuadratură

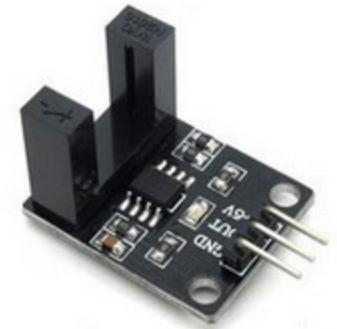
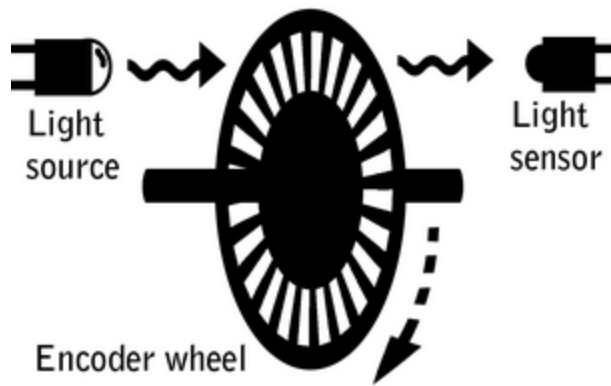


- Orientare: se monitorizează fronturile crescătoare sau descrescătoare ale unui semnal
- Starea celuilalt semnal în momentul tranziției dă orientarea

Motoare de curent continuu (DC)

Măsurarea turației motorului

- Roată cu perforații + senzor de lumină IR

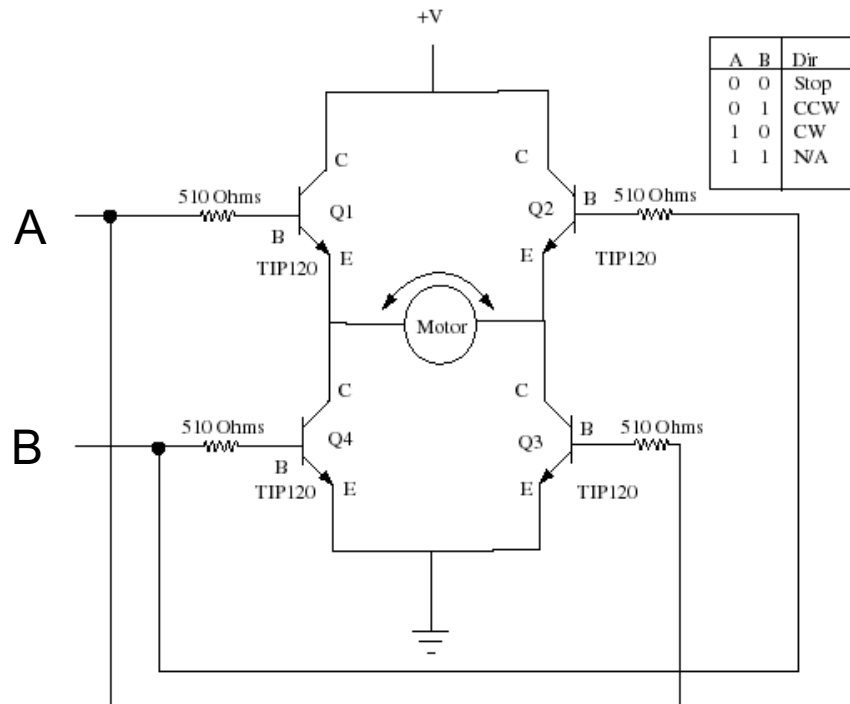
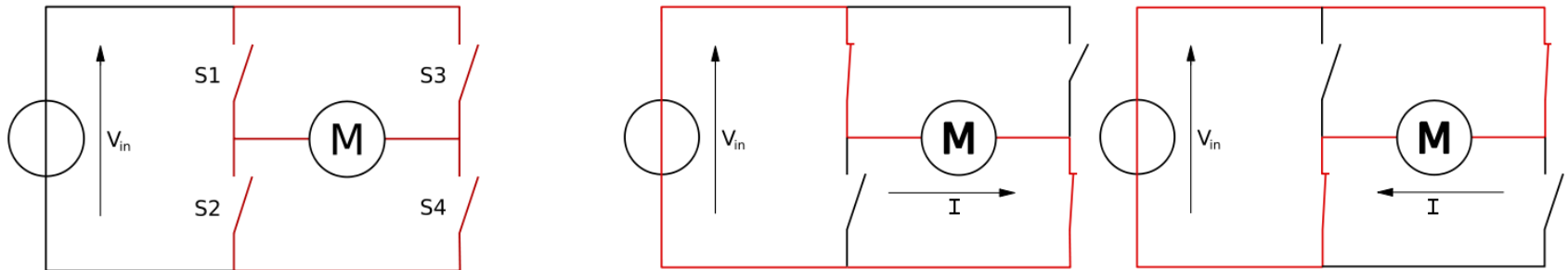


- Trecerea sau blocarea razei IR produce un tren de pulsuri pentru măsurarea turației.
- Cum putem măsura și orientarea ?

Motoare de curent continuu (DC)

Puntea H

- Controlul pornirii-opririi și a direcției unui motor



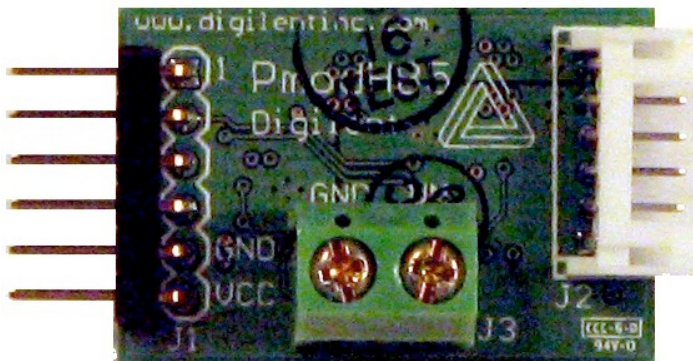
A	B	Dir
0	0	Stop
0	1	CCW
1	0	CW
1	1	N/A

- Scurt-circuit!

Motoare de curent continuu (DC)

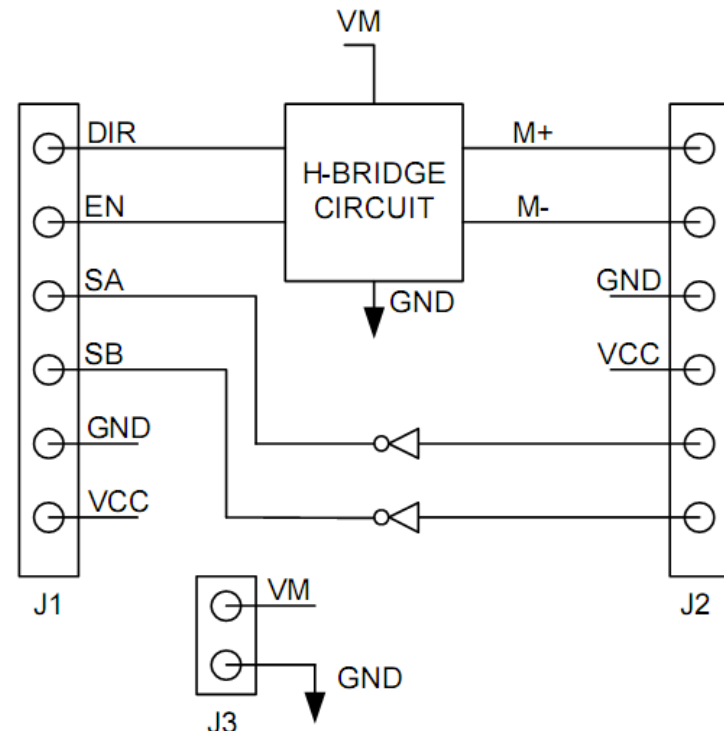
Puntea H

- Digilent PMOD HB5
- DIR – control direcție
- EN – dacă e '1', motorul funcționează – se poate atașa PWM pentru viteză variabilă
- A = EN and DIR, B = EN and (not DIR) – **Previne scurtcircuitul.**
- SA, SB – semnale de la motor, pentru a monitoriza starea acestuia



<input type="checkbox"/>	Direction
<input type="checkbox"/>	Enable
<input type="checkbox"/>	Sensor A
<input type="checkbox"/>	Sensor B
<input type="checkbox"/>	GND
<input type="checkbox"/>	Vcc (3.3 - 5v)

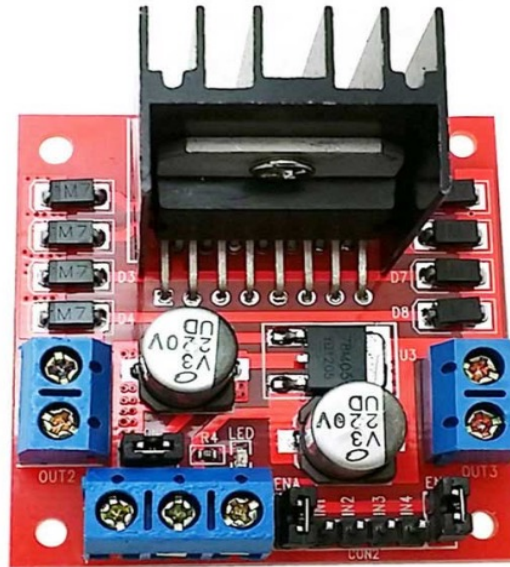
HB5 6-Pin Header, J1



Motoare de curent continuu (DC)

Driver-ul de motoare L298N Dual H-Bridge

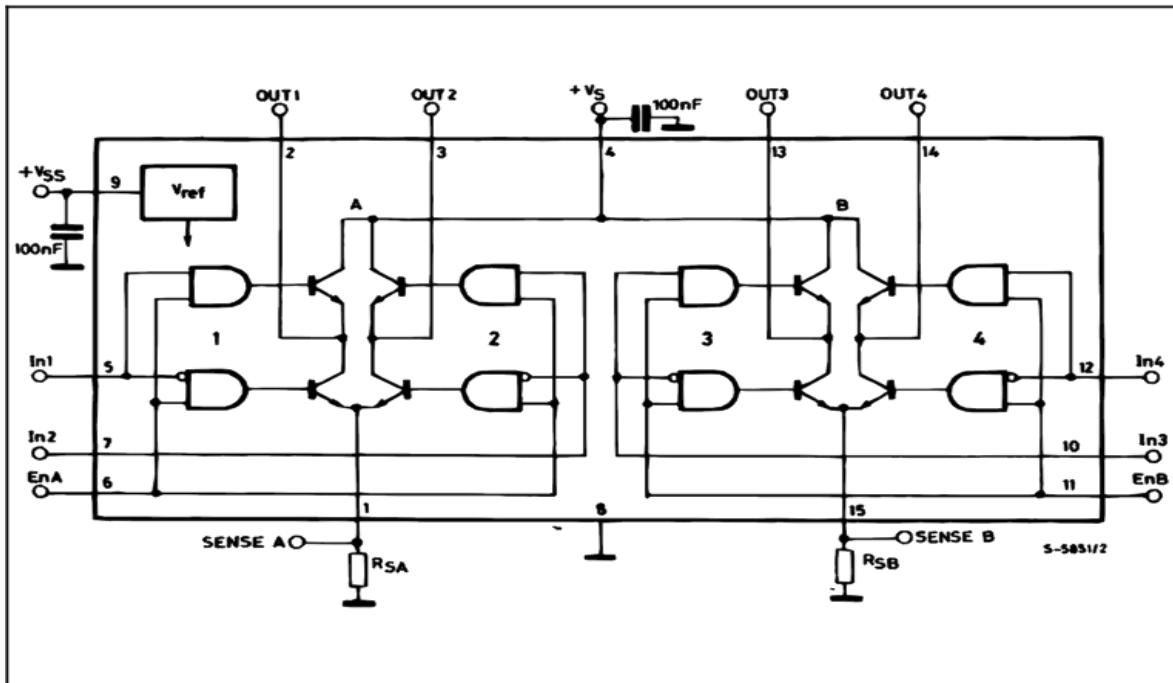
- <https://ardushop.ro/en/electronics/84-dual-h-bridge-for-dc-and-stepper-motors.html>
- Driver-ul se conectează la platforma Arduino folosind 4 pini digitali, conectați la pinii In1, In2, In3 și In4.
- Tensiune de alimentare motoare: 5... 35 V
- Tensiune circuit logic: 5 V (poate genera această tensiune pentru alimentare Arduino)
- Poate controla motoare care necesita cel mult 2 Amperi (2000 mA).
- 2 Motoare DC, sau un motor pas cu pas (Stepper)



Motoare de curent continuu (DC)

Driver-ul de motoare L298N Dual H-Bridge

- Schema circuitului L298N



In3	In4	Efect
0	0	Motor 2 oprit (frână)
0	1	Motor 2 pornit – înainte
1	0	Motor 2 pornit – înapoi
1	1	Motor 2 oprit (frână)

In1	In2	Efect
0	0	Motor 1 oprit (frână)
0	1	Motor 1 pornit – înainte
1	0	Motor 1 pornit – înapoi
1	1	Motor 1 oprit (frână)

Motoare de curent continuu (DC)

Exemplu: Rotatație a două motoare, în ambele sensuri

```
int MOTOR2_PIN1 = 3; // fiecare motor are 2 pini. Diferenta de polaritate dintre ei
int MOTOR2_PIN2 = 5; // cauzeaza motorul sa se deplaseze, intr-un sens sau in celalalt
int MOTOR1_PIN1 = 6;
int MOTOR1_PIN2 = 9;
```

```
void setup() {
  // pinii motor, configurati ca iesire
  pinMode(MOTOR1_PIN1, OUTPUT);
  pinMode(MOTOR1_PIN2, OUTPUT);
  pinMode(MOTOR2_PIN1, OUTPUT);
  pinMode(MOTOR2_PIN2, OUTPUT);
}

void loop() {
  // 2 motoare, 2 directii, 4 combinatii de cate 1 secunda
  go(255,-255);
  delay(1000);
  go(-255,-255);
  delay(1000);
  go(-255,255);
  delay(1000);
  go(255,255);
  delay(1000);
}
```

```
// functie control, viteza pentru M1 si pentru M2
```

```
void go(int speedLeft, int speedRight) {

  if (speedLeft > 0) { // viteza pozitiva, pe pin 1
    analogWrite(MOTOR1_PIN1, speedLeft);
    analogWrite(MOTOR1_PIN2, 0);
  }
  else
  {
    analogWrite(MOTOR1_PIN1, 0);
    analogWrite(MOTOR1_PIN2, -speedLeft); // viteza negativa,
                                           // val absoluta pe pin2
  }

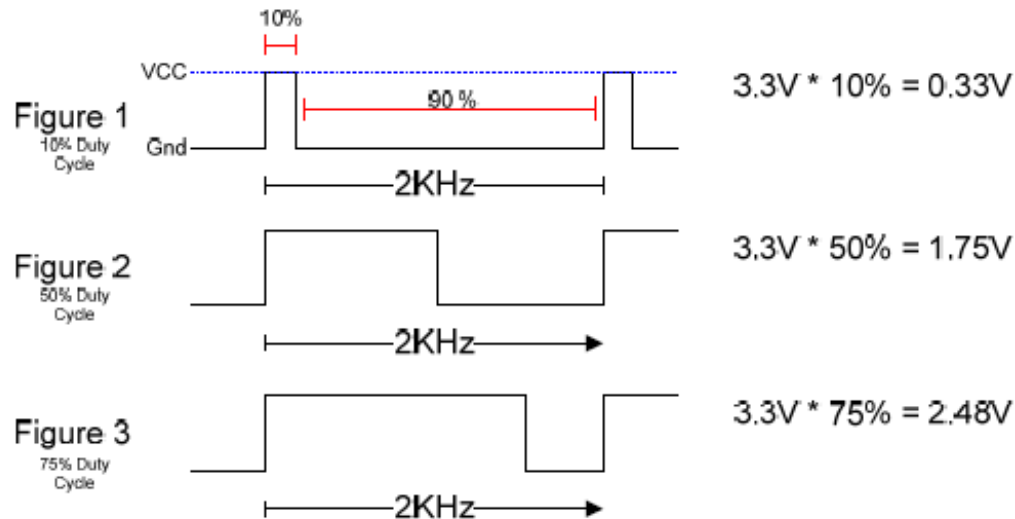
  if (speedRight > 0) {
    analogWrite(MOTOR2_PIN1, speedRight);
    analogWrite(MOTOR2_PIN2, 0);
  }
  else
  {
    analogWrite(MOTOR2_PIN1, 0);
    analogWrite(MOTOR2_PIN2, -speedRight);
  }
}
```

Motoare de curent continuu (DC)

Controlul vitezei folosind PWM

Într-un circuit analogic, viteza motorului este controlată de nivelul tensiunii. Într-un circuit digital, avem doar două soluții:

- Folosirea unui circuit de rezistență variabilă pentru a controla tensiunea aplicată motorului (soluție complicată, care irosește energie sub formă de căldură)
- Aplicarea intermitentă a tensiunii sub forma PWM.

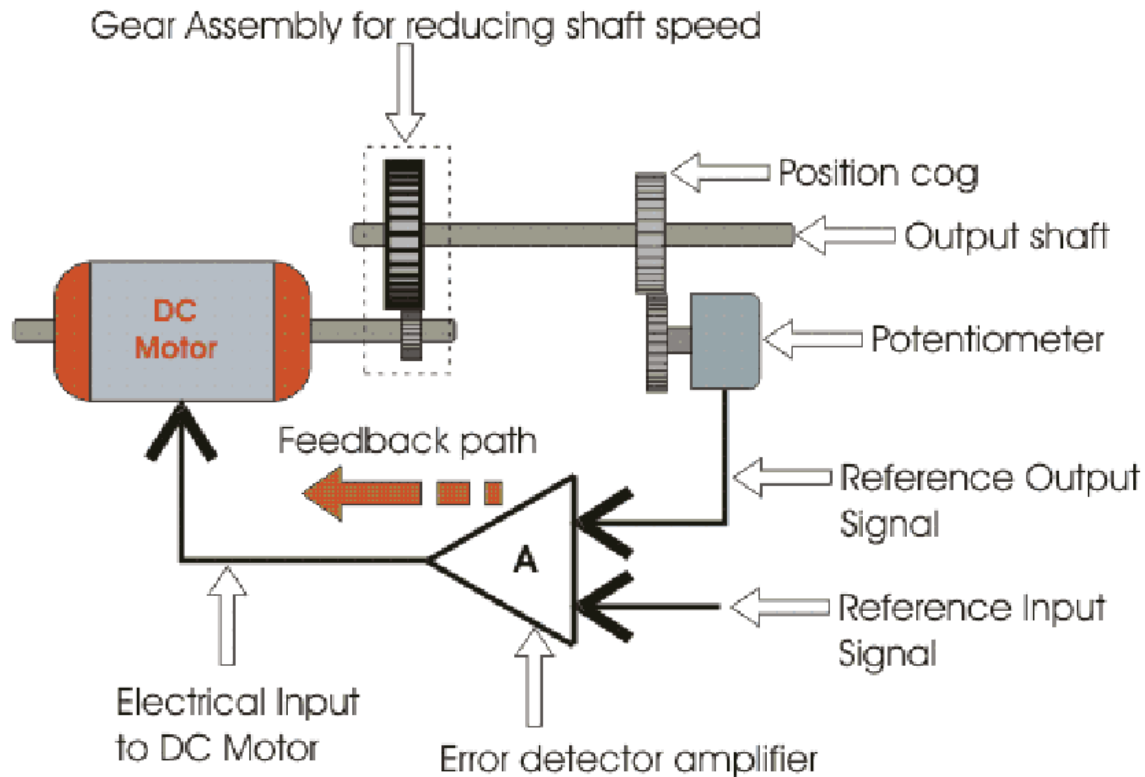


- Când tensiunea este aplicată, motorul este acționat de forța electromagnetică.
- Când tensiunea e oprită, inerția cauzează motorul să continue rotația pentru scurt timp.
- Dacă frecvența pulsurilor este suficient de mare, acest proces de pornire+mers din inerție permite motorului o rotație uniformă, controlabilă prin logica digitală.

Motoare Servo

Motorul servo

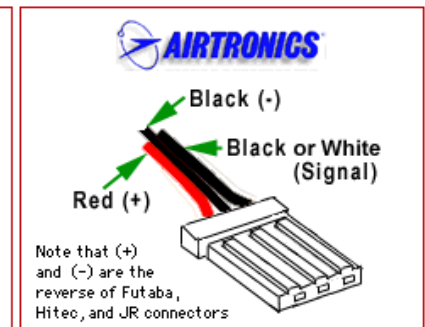
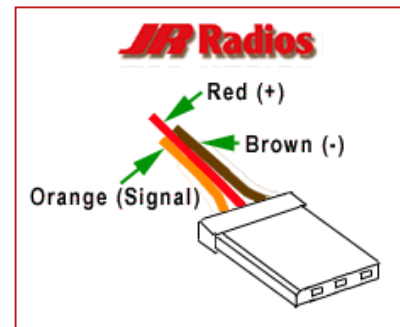
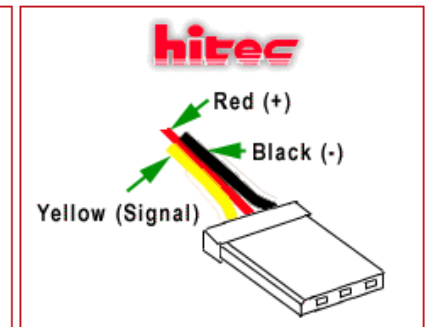
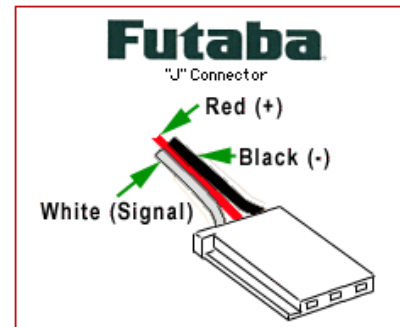
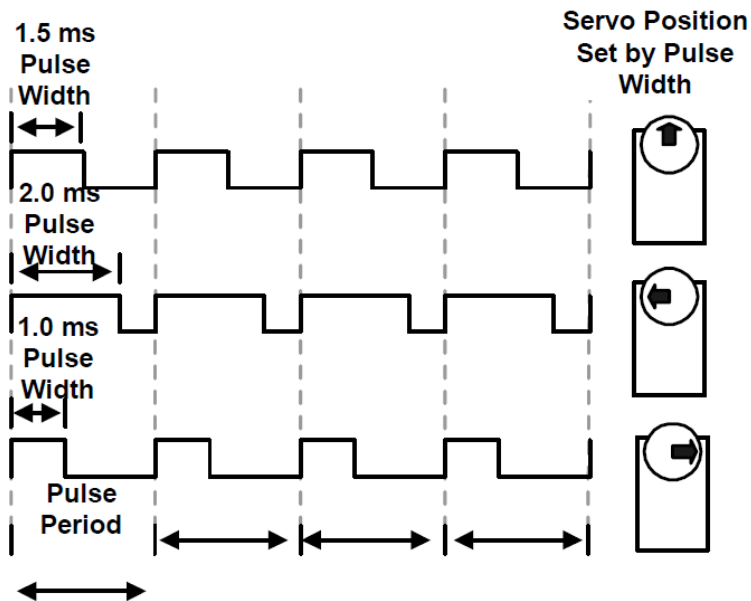
- Folosește un mecanism de feedback (reacție negativă) pentru a menține o poziție dată printr-un semnal de control (analog sau digital)
- Conține un motor DC, un angrenaj de roți dințate și un circuit de control.



Motoare Servo

Motorul servo (ex: GWS Servo Kit)

- Lățimea pulsului controlează amplitudinea rotației
- 1.5 ms – poziția neutră
- 1 ms – poziție maxim stânga (dreapta)
- 2 ms – pozitie maxim dreapta (stânga)
- Codificare PWM, frecvența purtătoare între 30 și 60 Hz



Controlul Motoarelor Servo - Arduino

Biblioteca **Servo**:

- Poate controla până la **12 motoare pe majoritatea placilor Arduino**
- **48 motoare** pe placa Arduino Mega.
- Folosirea bibliotecii va dezactiva analogWrite() (PWM) pe pinii 9 și 10, indiferent dacă există sau nu motor servo conectat la acești pini (**exceptand placa Arduino Mega**).
- La Arduino Mega Mega, se pot utiliza până la 12 motoare servo fără a afecta funcționarea PWM; folosirea mai multor motoare va dezactiva PWM pe pinii 11 și 12.

Conectarea Servo la Arduino (3 fire): Vcc, Gnd, semnal.

- **Vcc** , → la pinul de 5V al placii.
- Gnd (negru sau **maro**) → la GND de pe Arduino.
- Pinul de semnal (**galben**, **portocaliu** sau **alb**) conectat la un pin digital.

Nota: motoarele necesita putere considerabilă! Pentru a acționa mai mult de 2 motoare servo, folosiți o sursă de alimentare externă.

Controlul Motoarelor Servo - Arduino

Metode ale bibliotecii Servo:

`servo.attach(pin)` / `servo.attach(pin, min, max)` – atașează obiectul Servo la pini

- `servo`: un obiect instanță a clasei Servo
- `pin`: numărul pinului digital unde va fi atașat semnalul pentru motorul Servo
- `min` (optional): lățimea pulsului, în microsecunde, corespunzătoare unghiului minim (0 grade) al motorului servo (implicit 544)
- `max` (optional): lățimea pulsului, în microsecunde, corespunzătoare unghiului maxim (180 grade) al motorului servo (implicit 2400)

`servo.detach()` – detașează obiectul de tip Servo de la pin.

`boolean val servo.attached()` – verifică dacă obiectul de tip Servo este atașat unui pin. Returnează adevărat sau fals.

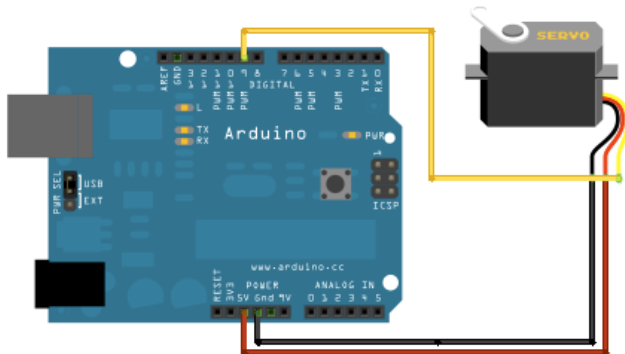
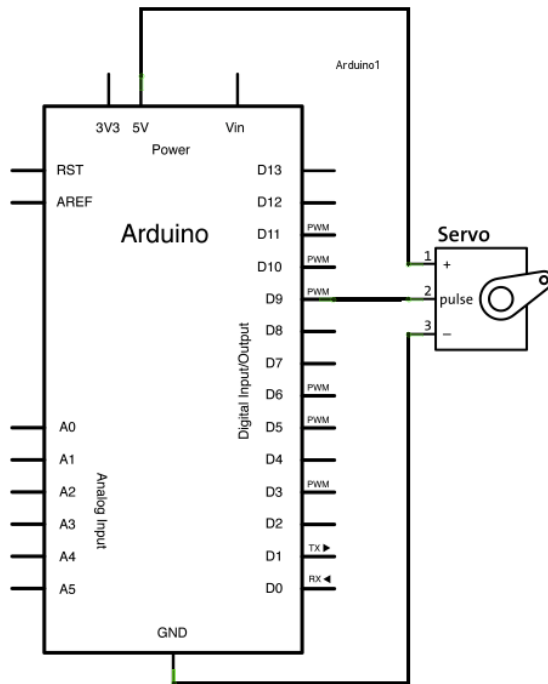
`servo.write (angle)` – scrie o valoare (0 .. 180) către servo, controlând mișcarea:

- *Servo standard* ⇒ setează unghiul axului [grade] cauzând motorul să se orienteze în direcția specificată.
- *Servo cu rotație continuă* ⇒ configurează viteza de rotație (0: viteza maximă într-o direcție; 180: viteza maximă în direcția opusă; ≈ 90: oprit)

`int val servo.read()` – citește unghiul curent al servo, configurat la ultimul apel al `write()`.

Controlul Motoarelor Servo - Arduino

Exemplu: Orientează axul unui servo parcurgând înainte și înapoi 180 grade (<http://arduino.cc/en/Tutorial/Sweep>)



```
#include <Servo.h>
```

```
Servo myservo; // obiect pentru controlul servo  
int pos = 0; // variabila ce tine pozitia curenta a axului
```

```
void setup()
```

```
{  
  myservo.attach(9); // ataseaza obiectul servo la pinul 9  
}
```

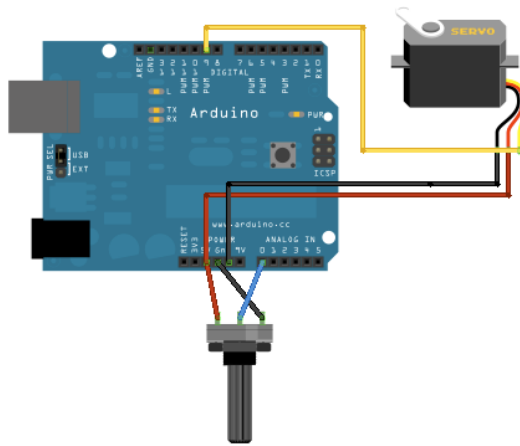
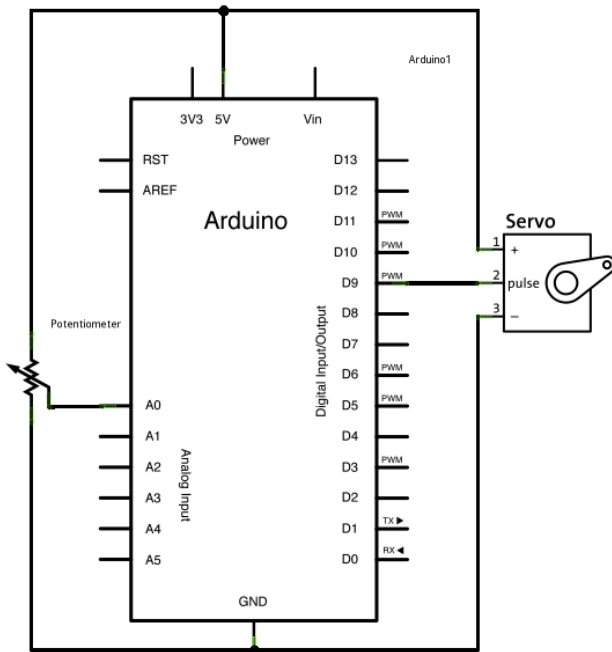
```
void loop()
```

```
{  
  for(pos = 0; pos < 180; pos += 1) // de la 0 la 180 grade  
  {  
    myservo.write(pos); // configureaza pozitia dorita  
    delay(15); // asteapta 15 ms pentru ca motorul sa se  
 // pozitioneze  
  }  
  for(pos = 180; pos >= 1; pos -= 1) // baleiere inapoi
```

```
{  
  myservo.write(pos);  
  delay(15);  
}  
}
```

Controlul Motoarelor Servo - Arduino

Exemplu: Controlul poziției unui servo motor cu Arduino și un potențiomtru (<http://arduino.cc/en/Tutorial/Knob>)



```
#include <Servo.h>
```

```
Servo myservo; // obiect pentru controlul motorului servo
```

```
int potpin = 0; // pin analogic pentru citirea potentiometrului  
int val; // variabila in care se va citi starea pinului analogic  
int angle; // unghiul servo
```

```
void setup()
```

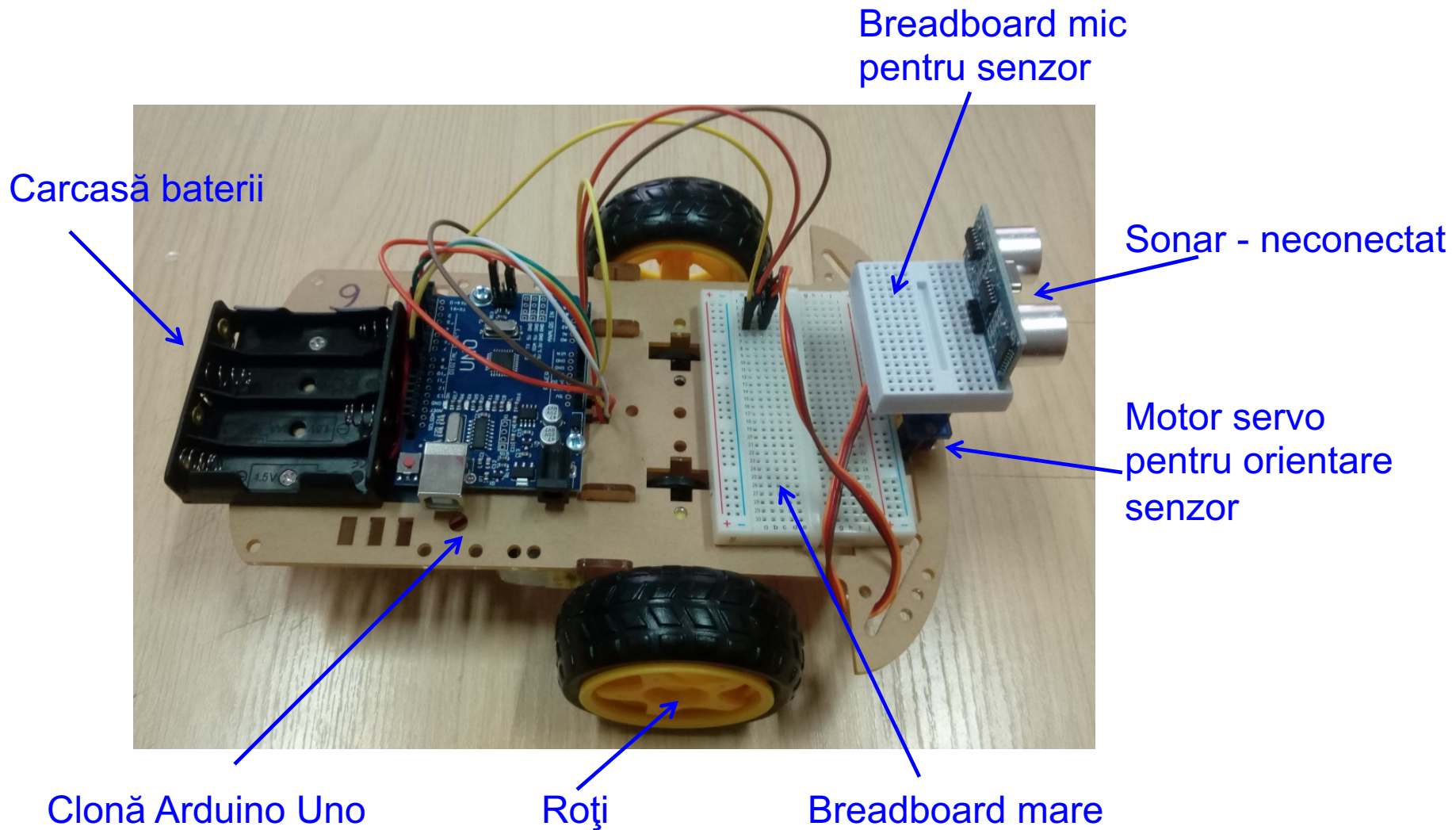
```
{  
  myservo.attach(9); // ataseaza obiectul servo la pinul 9  
}
```

```
void loop()
```

```
{  
  val = analogRead(potpin); // citeste stare potentiometru  
  angle = map(val, 0, 1023, 0, 179); // scalarea valorii citite (0...1023)  
  // in domeniul 0... 179  
  myservo.write(angle); // scrie noua pozitie pentru servo  
  delay(15); // asteapta pozitionarea motorului  
}
```


Robotul experimental

Robot pre-asamblat pentru desfășurarea activităților de proiect

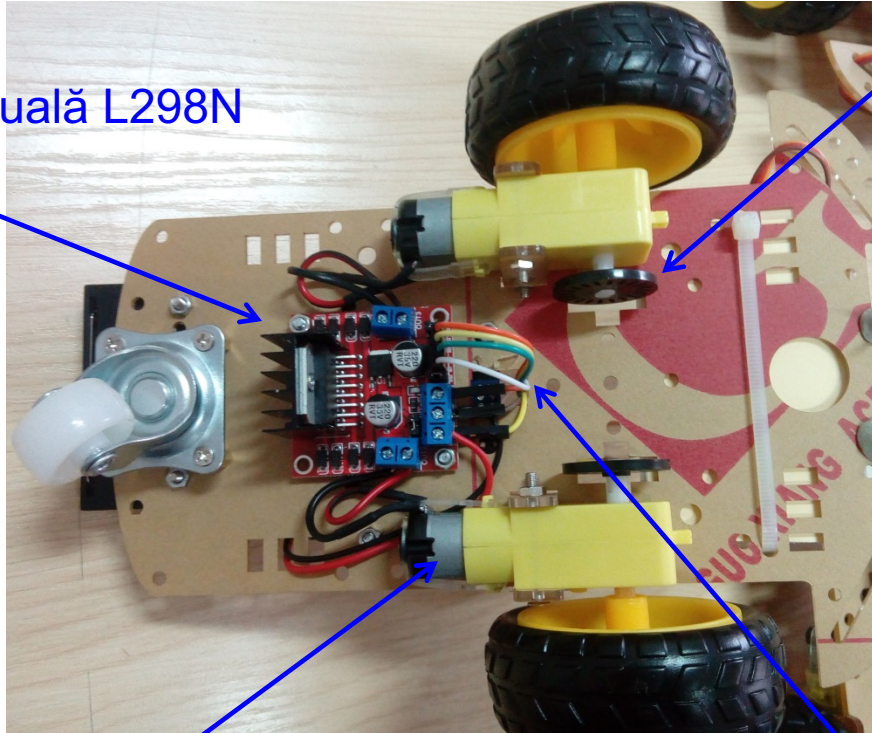


Robotul experimental

Robot pre-asamblat pentru desfășurarea activităților de proiect

Roată perforată pentru turație
Fără senzor IR !

Punte H duală L298N



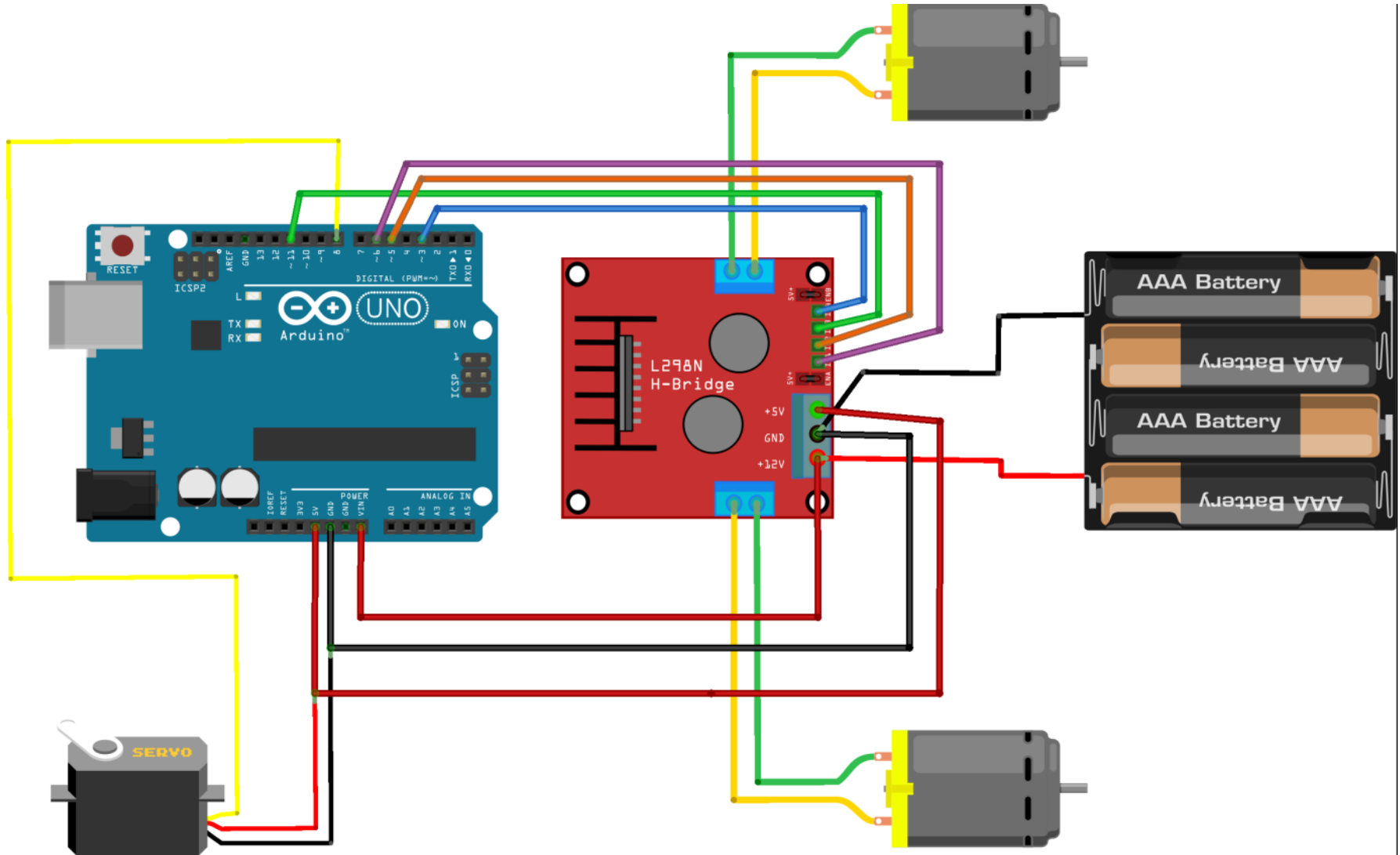
Motor DC

Roți

Fire control (In1, ... In4)

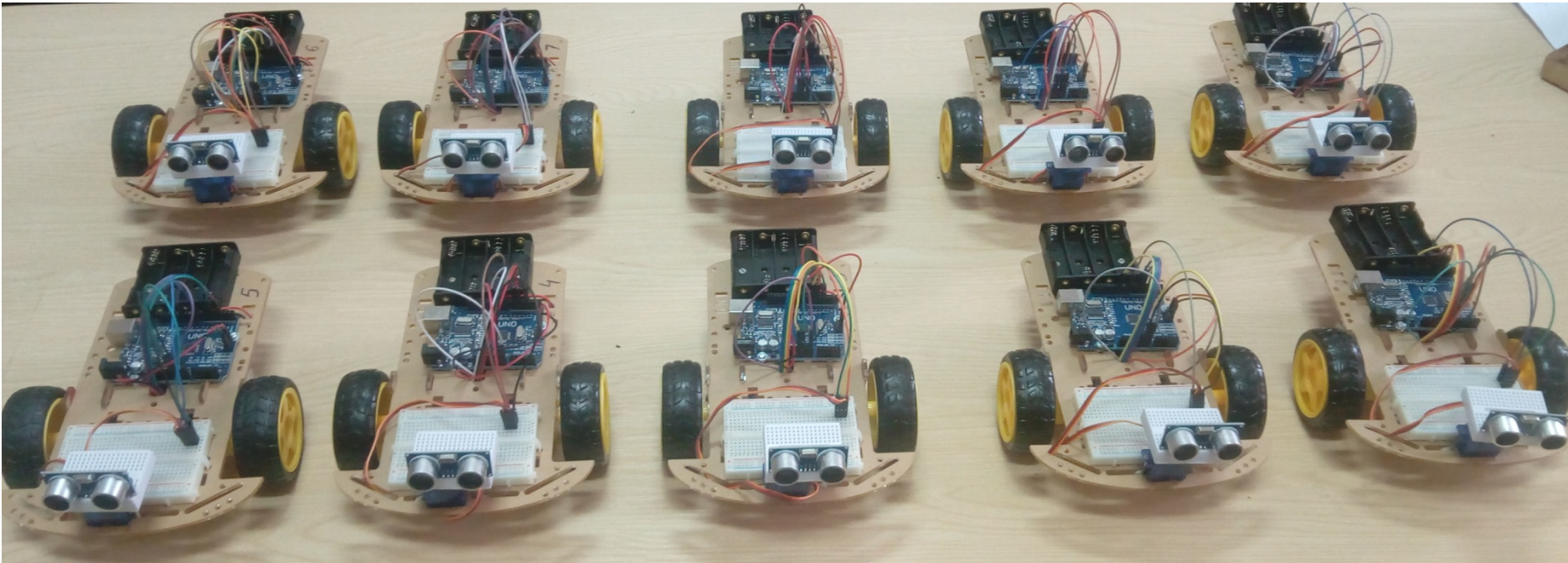
Robotul experimental

Robot pre-asamblat pentru desfășurarea activităților de proiect



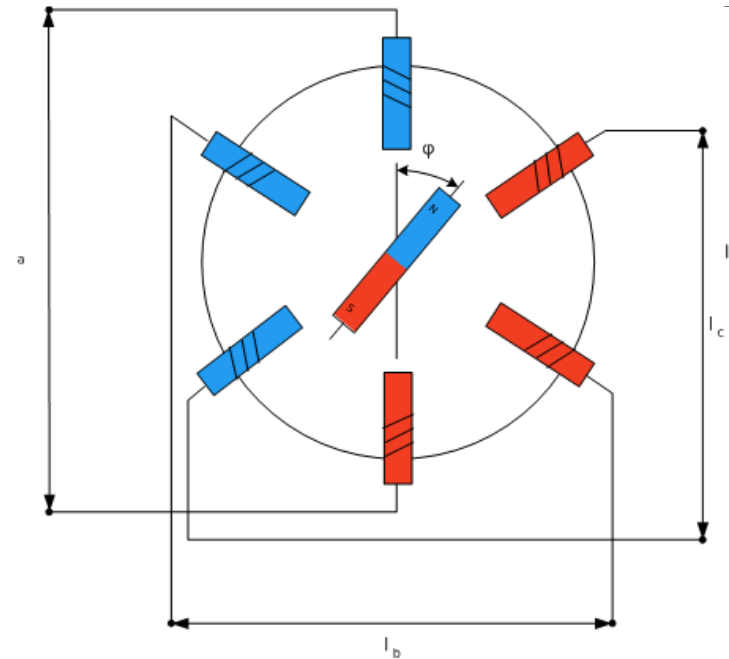
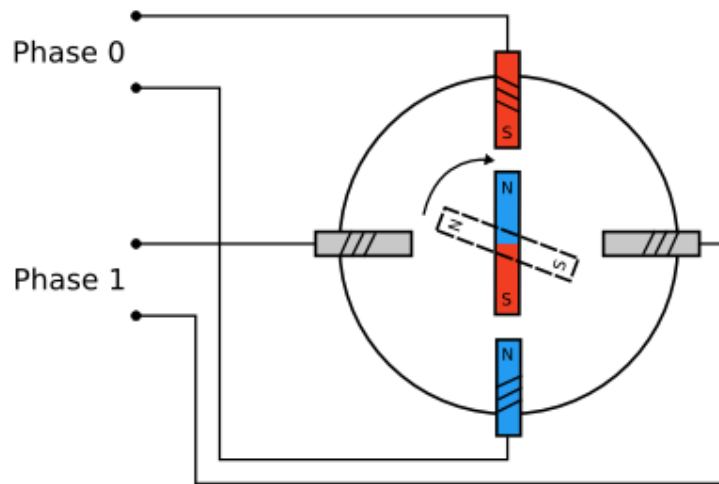
Robotul experimental

Robot pre-asamblat pentru desfășurarea activităților de proiect



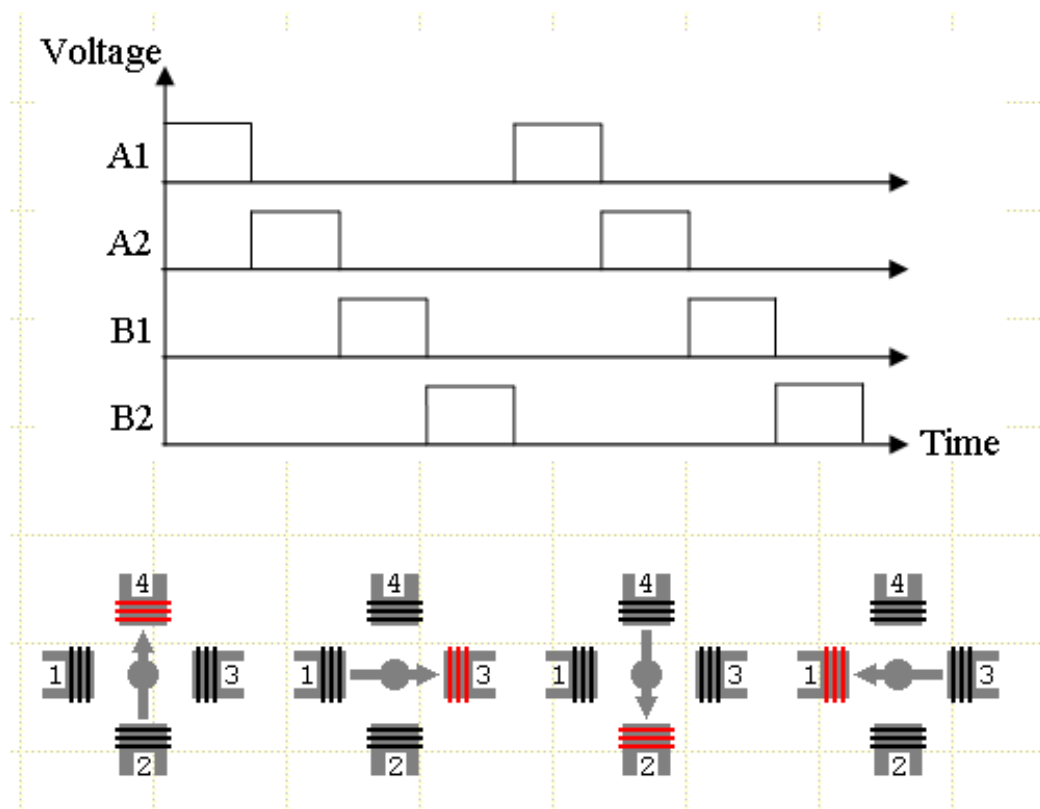
Motor Pas cu Pas

- Rotația se face pas cu pas, prin activarea selectivă a bobinelor de pe stator
- Curenții din bobine se schimbă prin control electronic, spre deosebire de motoarele clasice, la care schimbarea se face prin control mecanic, cu perii



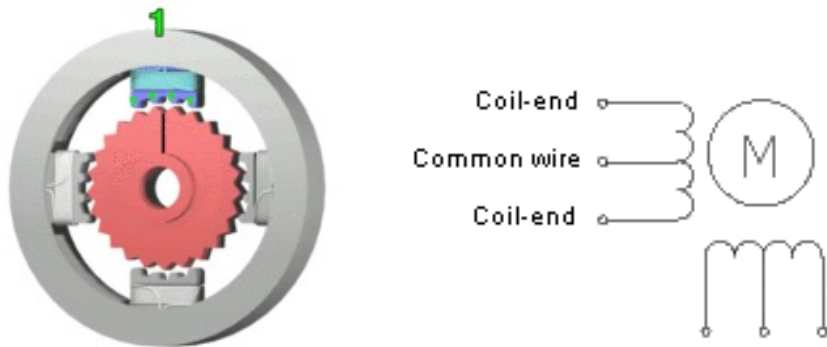
Motor Pas cu Pas

- Un controller de motor pas cu pas trebuie sa genereze secvența corectă pentru activarea bobinelor
- Se pot efectua rotații complete, sau părți de rotație, în funcție de numărul de pulsuri – control precis al cantității de rotație !



Motor Pas cu Pas

Motor pas cu pas unipolar



Comandă tip undă , sau Pas întreg cu o singură fază

- Cuplu motor mic, se folosește rar. 25 dinți / 4 pași pentru a roti o poziție a unui dinte $\Rightarrow 25 \cdot 4 = 100$ pași pentru o rotație completă \Rightarrow fiecare pas va avea $360/100 = 3.6$ \square

Comandă cu pas întreg cu două faze

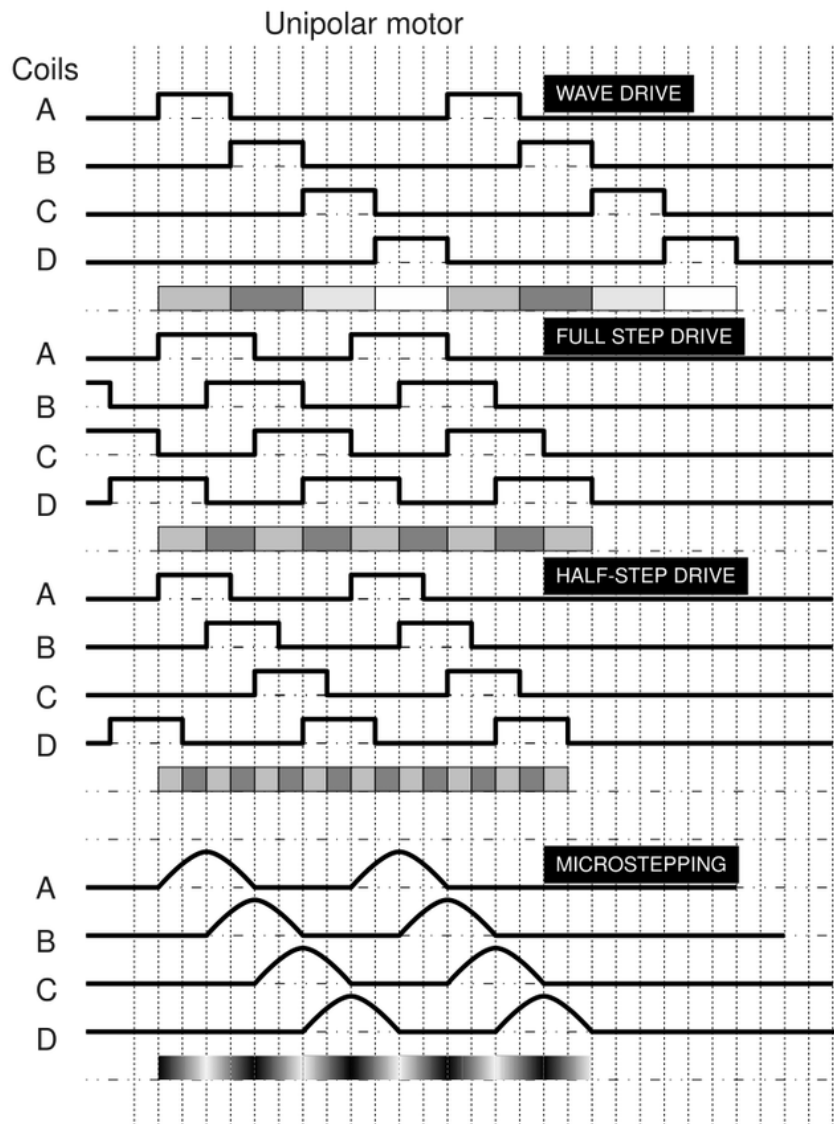
- Cuplu motor maxim, comanda cea mai folosită

Comandă cu jumătate de pas

- Cuplu mai mic (70%) / rezoluție x2 (8 pași pentru a deplasa un dinte $\Rightarrow 25 \cdot 8 = 200$ pași pentru rotație întreaga \Rightarrow un pas = 1.8 \square

Micro-pas

- Operare mai fină



Motor Pas cu Pas - Arduino

Biblioteca Arduino Stepper (<http://arduino.cc/en/reference/stepper>)

- Permite controlul motoarelor pas cu pas unipolare sau bipolare. Pentru a putea folosi această bibliotecă, e nevoie de un motor pas cu pas și de o interfață hardware pentru acesta.

Pentru a crea un obiect de clasa Stepper, se apelează constructorul:

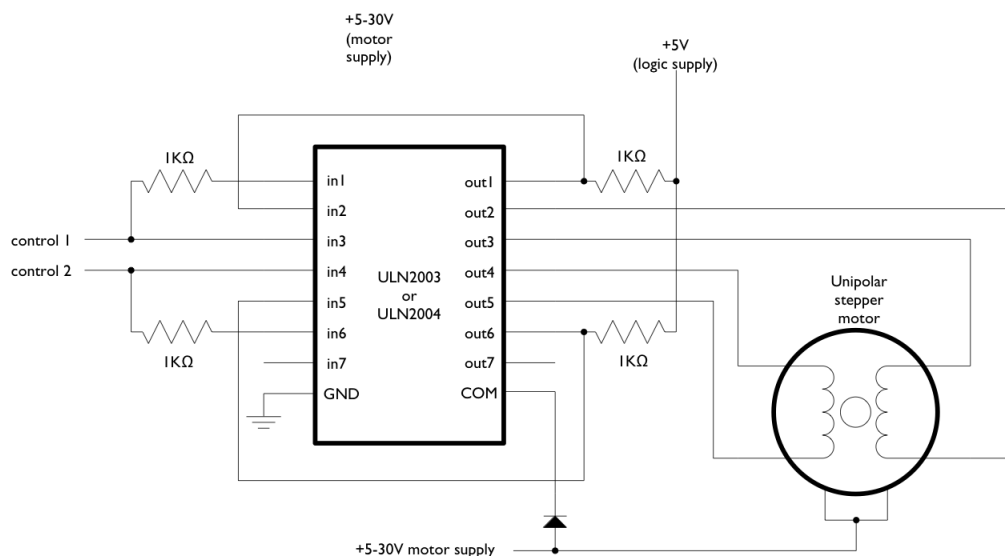
Stepper(steps, pin1, pin2) - ex: Stepper myStepper = Stepper(100, 5, 6);

Stepper(steps, pin1, pin2, pin3, pin4)

int steps: numărul de pași per rotație completă (ex: $360 / 3.6 = 100$ pași)

int pin1, pin2: doi pini atașati interfeței hardware (montaj cu 2 pini)

int pin3, pin4: *optional* pini atașati interfeței hardware, pentru montaj cu 4 pini



2 pin/wire setup

Secvența de control (2 pini):

Pas	pin 1	pin 2
1	low	high
2	high	high
3	high	low
4	low	low

Motor Pas cu Pas - Arduino

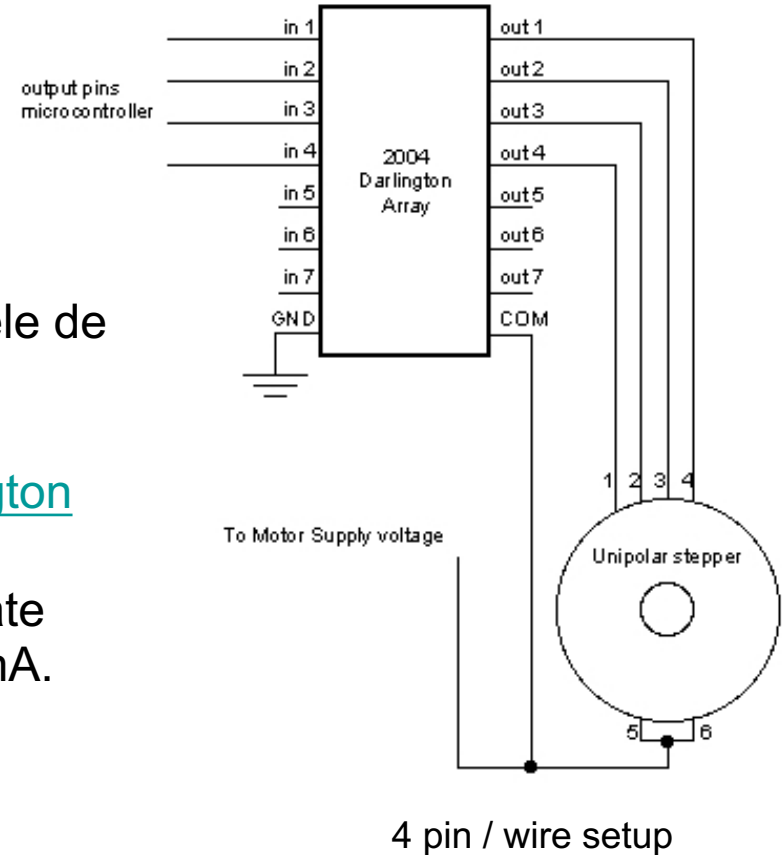
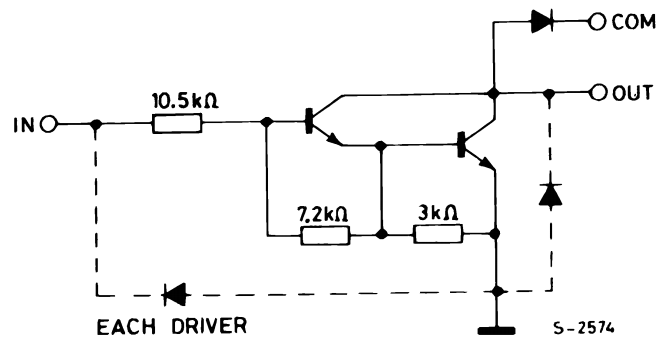
Secvența de control (4 pini):

Pas	pin 1	pin 2	pin 3	pin 4
1	High	low	high	low
2	low	high	high	low
3	low	high	low	high
4	high	low	low	high

Dacă se folosește biblioteca Stepper, semnalele de control sunt generate de către bibliotecă!

Exemplu de interfață hardware: [U2004 Darlington Array](#)

- Tensiune și amperaj mare. Fiecare canal poate susține 500 mA, cu vârfuri acceptate de 600 mA.



Motor Pas cu Pas - Arduino

Funcții ale bibliotecii Stepper (<http://arduino.cc/en/reference/stepper>)

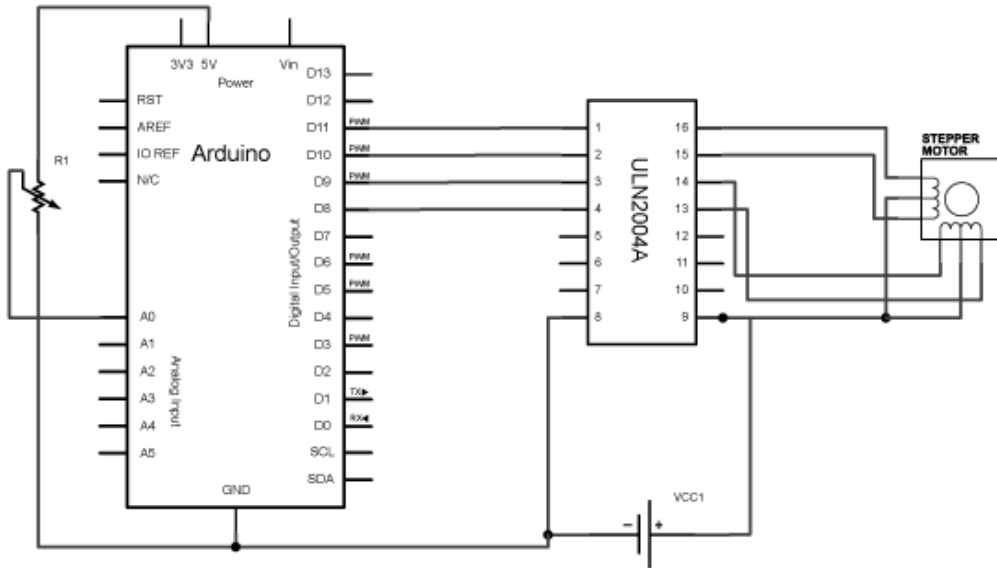
setSpeed(long *rpms*) – configurează viteza de rotație a motorului, în rotații pe minut (RPMs). Această funcție nu pornește motorul, ci doar configurează viteza cu care se va roti când se apelează funcția `step()`.

step(int *steps*) – Rotește motorul un număr specificat de pași, cu viteza configurată.

- int *steps*: numărul de pași pe care motorul îi va executa – pozitiv (+) rotație într-o direcție, negativ (-) rotație în direcția opusă
- **Funcția este blocantă**: va aștepta până când motorul va termina rotația, pentru a ieși. (Ex: la viteza = 1 RPM, apelata cu parametrul `steps = 100` pentru un motor cu rotație completă în 100 de pași, funcția va bloca programul timp de 1 minut).
- Pentru un control mai bun, apăsați doar un număr mic de pași odată, cu o viteză mare.

Motor Pas cu Pas - Arduino

Exemplu: Motor pas cu pas controlat cu potentiometru Knob
(<http://arduino.cc/en/Tutorial/MotorKnob>)



```
#include <Stepper.h>
#define STEPS 100
```

```
Stepper stepper(STEPS, 8, 9, 10, 11);
```

```
// citirea anterioara de la potentiometru
int previous = 0;
```

```
void setup()
```

```
{
  // viteza motor, 30 RPM
  stepper.setSpeed(30);
}
```

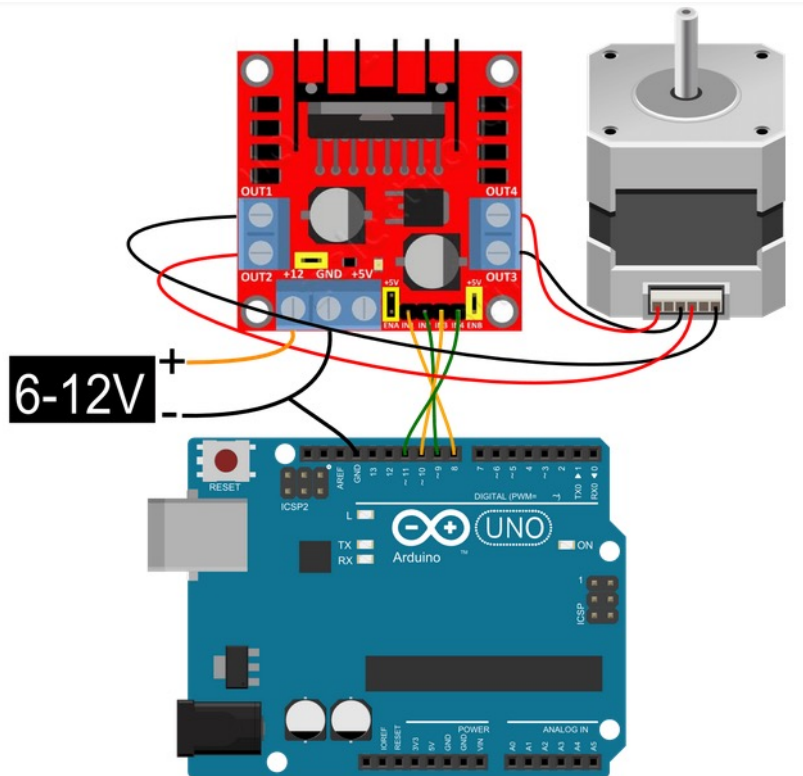
```
void loop()
```

```
{
  // citire stare potentiometru
  int val = analogRead(0);
  // deplasare motor cu diferenta dintre citiri
  stepper.step(val - previous);
  // valoarea curenta devine valoare anterioara
  previous = val;
}
```

Motor Pas cu Pas și L298N

Motorul pas cu pas se poate controla și prin puntea H duală

Sursa: <https://coeleveld.com/arduino-stepper-l298n/>



```
#include <Stepper.h>
```

```
const int stepsPerRevolution = 200; // modificati pentru  
// specificatiile motorului propriu
```

```
// se initializeaza biblioteca stepper pe pinii 8 ...11:  
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);
```

```
void setup() {  
  // regleaza viteza de rotatie la 60 rpm:  
  myStepper.setSpeed(60);  
  // initializeaza interfata seriala  
  Serial.begin(9600);  
}
```

```
void loop() {  
  // o rotatie completa in directia orara  
  Serial.println("clockwise");  
  myStepper.step(stepsPerRevolution);  
  delay(500);  
  
  // o rotatie completa in directia antiorara  
  Serial.println("counterclockwise");  
  myStepper.step(-stepsPerRevolution);  
  delay(500);  
}
```