Starter Kit for Arduino Uno – Manual Vol.1

1.

Introducere în Arduino IDE (Integrated Development Environment)

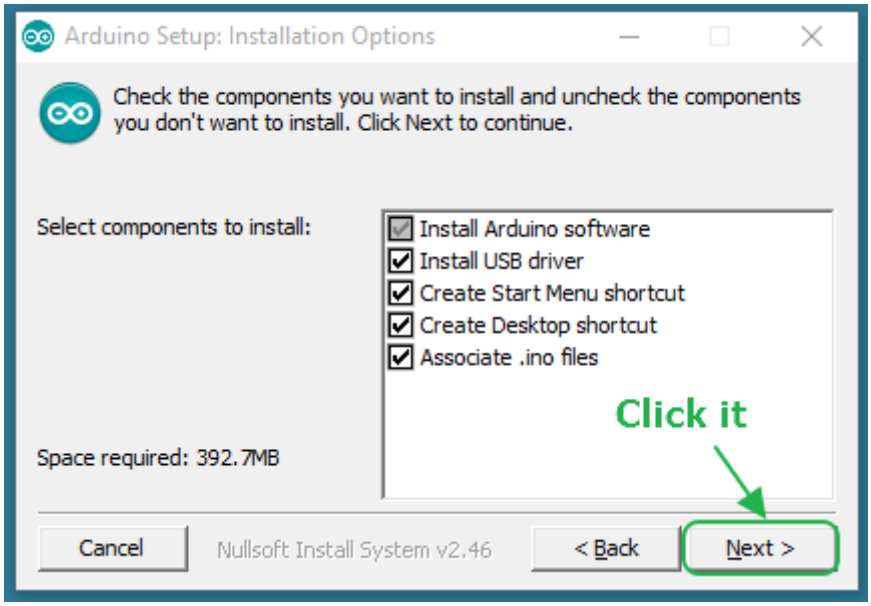
Software-ul Arduino (IDE) este ușor de utilizat pentru începători, dar suficient de flexibil și pentru avansați. Se bazează pe „Processing programming environment”, astfel încât elevii să se familiarizeze cum să învețe să programeze în acest mediu și modul în care funcționează ID-ul Arduino.

Pasul 1 : Install the Arduino Software (IDE)

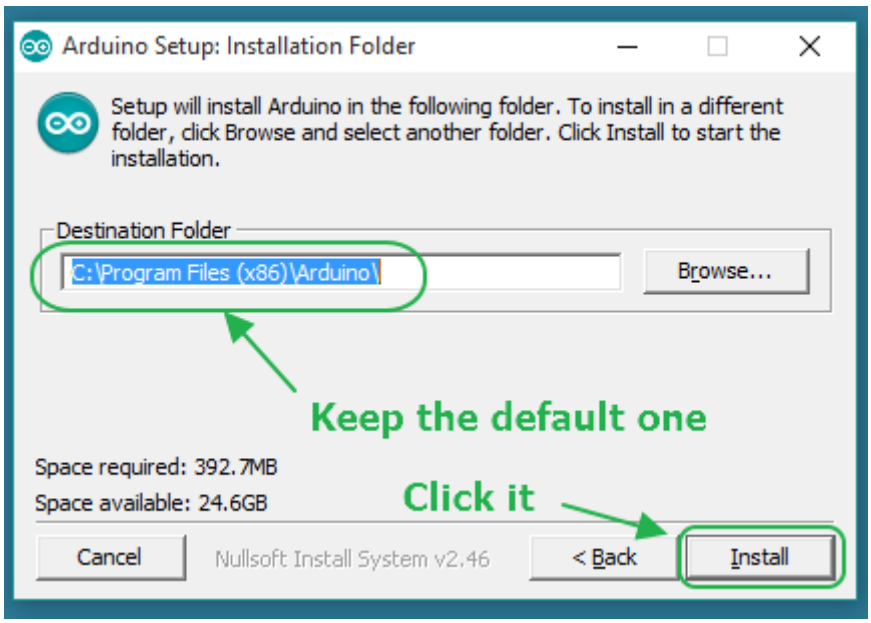
Download ultima versiune din pagina : http://arduino.cc/en/Main/Software

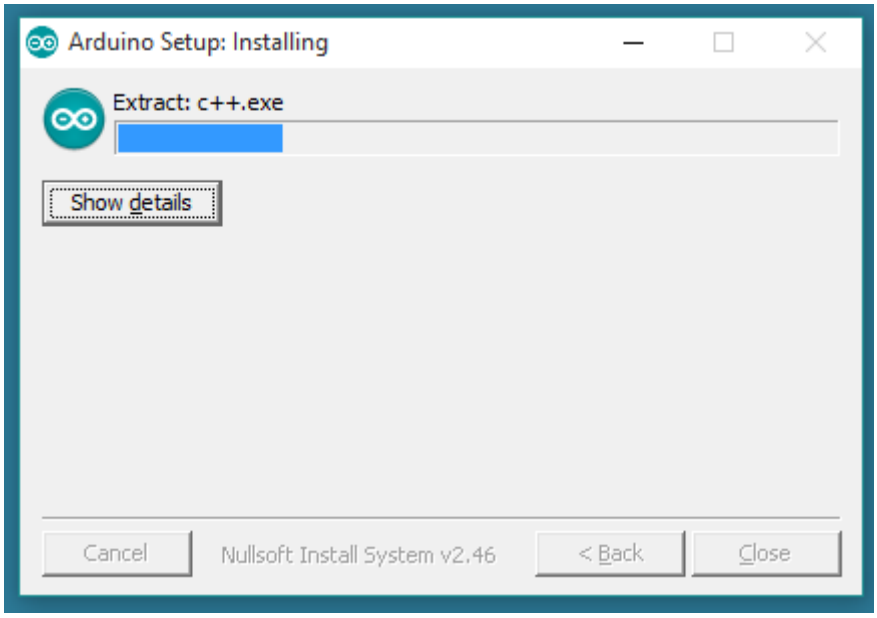
Apoi urmeaza instalarea driverului din pagina : http://oboromania.ro/drivere/
Dacă Windowsul din PC nu îl instaleaza automat, descărcați driverul din pagina indicată și instalați (.exe).

Sau puteți folosi instalarea din ID :



Alegeți componentele pe care doriți să le instalați și faceți clic pe butonul "Next".

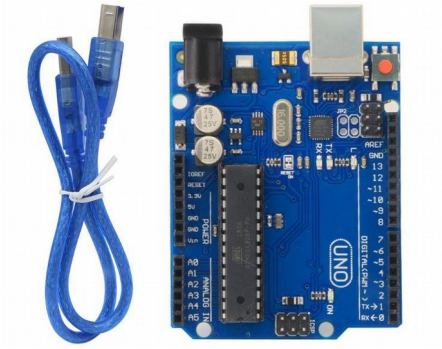


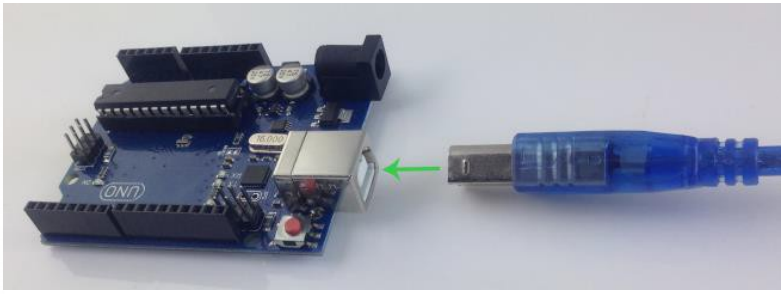


Procesul va extrage și instala toate fișierele necesare pentru a executa corect software-ul Arduino (IDE).

Pasul 2 : Conectarea unui Uno R3 prin cablul USB

În acest tutorial, folosiți un Uno R3. De asemenea, aveți nevoie de un cablu USB standard (USB-A la USB-B): de exemplu tipul care se folosește la conectarea unei imprimante.





Conexiunea USB cu PC-ul este necesară pentru programarea plăcii cât și pentru alimentarea acesteia. Uno și Mega extrag automat puterea fie de la USB, fie de la o sursă de alimentare externă. Conectați placa la computer utilizând cablul USB. LED-ul de putere verde (denumit PWR PWR) ar trebui să se aprindă.

Pasul 3 : Descrierea pinilor unui Uno R3

Începând de sus.

Există **14 pini digitali** de intrare / ieşire (I/O sau input/output).

Aceştia operează la o tensiune de 5 volţi şi pot fi controlaţi cu una din funcţiile pinMode(), digitalWrite() şi digitalRead().

Fiecare pin poate primii sau trimite o intensitate de maxim 40 mA şi au o

rezistenţă internă între 20-50 kOhmi (default deconectată). În afară de semnalul standard I/O, unii dintre pini mai au şi alte funcţii specializate.

- 0 (serial) **RX** – pin serial, utilizat în special pentru recepţia (intrare – Rx) datelor seriale asincrone (asynchronous serial communication) Protocolul serial asincron este o metodă foarte răspândită în electronică pentru a trimite şi recepţiona date între dispozitive. Acest protocol este implementat în dispozitiv numit UART (Universal Asynchronous Receiver/Transmitter)
- 1 (serial) **TX** – pin serial, utilizat pentru trimiterea datelor asincrone (ieşire – **Tx**). TTL vine de la transistor-transistor logic.
- 2 (External Interrupts) întrerupere externă. Acest pin poate fi configurat pentru a declanșa o întrerupere la o valoare mică, un front crescător sau descrescător, sau o schimbare în valoare. Vezi detalii despre posibile comenzi la attachInterrupt()
- 3 (External Interrupts + PWM) întrerupere externă. Identic cu pinul 2. Suplimentar, toţi pinii marcaţi cu semnul ~ pot fi folosiţi şi pentru **PWM** (pulse with modulation)
- 4 (**I/O**) pin standard intrare/iesire
- 5 (**PWM**) poate furniza control de ieşire pe 8-bit pentru controlul PWM. Vezi detalii despre posibile comenzi la analogWrite()
- 6 (**PWM**)
- 7 (**I/O**) pin standard intrare/ieşire
- 8 (**I/O**) pin standard intrare/ieşire
- 9 (**PWM**)
- 10 (**PWM** + SPI) – suportă comunicare prin interfaţa serială (Serial Peripheral Interface). SPI-ul are patru semnale logice specifice iar acest pin se foloseste pentru **SS** – Slave Select (active low; output din master). Pinii SPI pot fi controlaţi folosind libraria SPI.
- 11 (**PWM** + SPI) – suportă SPI, iar acest pin se foloseşte pentru **MOSI**/SIMO – Master Output, Slave Input (output din master)
- 12 (SPI) – suportă SPI, iar acest pin se foloseşte pentru **MISO**/SOMI – Master Input, Slave Output (output din slave)
- 13 (LED + SPI) – suportă SPI, iar acest pin se foloseşte pentru **SCK**/SCLK – Ceas serial (output din master). De asemenea, pe placă este încorporat un LED care este conectat la acest pin. Când pinul este setat pe valoarea HIGH este pornit, când are valoarea LOW este oprit.
- 14 (**GND**) – împământare. Aici se pune negativul.
- 15 (**AREF**) – Analog REFference pin – este utilizat pentru tensiunea de referinţă pentru intrările analogice. Se poate controla folosind funcţia analogReference().
- 16 (**SDA**) – comunicare I2S
- 17 (**SCL**) – comunicare I2S

În partea de jos.

Există o serie de 6 pini pentru semnal analogic, numerotaţi de la A0 la A5.

Fiecare din ei poate furniza o rezoluţie de 10 biţi (adică maxim 1024 de valori diferite). În mod implicit se măsoară de la 0 la 5 volţi, deşi este posibil să se schimbe limita superioară a intervalului lor folosind pinul 15 AREF şi funcţia analogReference(). De asemenea, şi aici anumiţi pini au funcţii suplimentare descrise mai jos:

- **A0** standard analog pin
- **A1** standard analog pin
- **A2** standard analog pin
- **A3** standard analog pin
- **A4** (**SDA)** suportă comunicarea prin 2 fire (**I2C** (I-two-C) sau **TWI** (Two wire interface)). Acest pin este folosit pentru SDA (Serial Data) la TWI.
- **A5** (**SCL**) identic cu pinul 4, doar că acest pin este folosit pentru **SCL** (Serial Clock) la TWI. Pentru controlul TWI se poate folosi librăria Wire.

Lângă pinii analogici arătaţi mai există o secţiune de pini notată **POWER.**

Acestia sunt (începând de lângă pinul analog A0) :

- 1 **Vin** – intrarea pentru tensiune din sursă externă (input Voltage) 6...9v DC
- 2 **GND** – negativul pentru tensiune din sursă externă (ground Voltage)
- 3 **GND** – negativ. Se foloseşte pentru piesele şi componentele montate la arduino ca şi masă/împământare/negativ.
- 4 **5V** – ieşire pentru piesele şi componentele montate la arduino. Scoate fix 5V dacă placa este alimentată cu tensiune corectă (între 7 şi 12 v)
- 5 **3,3V** – ieşire pentru piesele şi senzorii care se alimentează la această tensiune. Tensiunea de ieşire este 3.3 volţi şi maxim 50 mA.
- 6 **RESET** – se poate seta acest pin pe LOW pentru a reseta controlerul de la Arduino. Este de obicei folosit de shield-urile care au un buton de reset şi care anulează de obicei butonul de reset de pe placa Arduino.
- 7 **5VREF** – este folosit de unele shield-uri ca referinţă pentru a se comuta automat la tensiunea furnizată de placa arduino (5 volţi sau 3.3 volţi) (Input/Output Refference Voltage)
- 8 **pin** neconectat, este rezervat pentru utilizări ulterioare (la reviziile următoare ale plăcii probabil).

Comunicarea cu calculatorul, altă placă arduino sau alte microcontrolere se poate realiza fie prin portul USB (şi este văzut ca un port standard serial COMx), fie prin pinii 0 şi 1 (RX şi TX) care facilitează comunicarea serială UART TTL (5V). Folosind librăria SoftwareSerial.

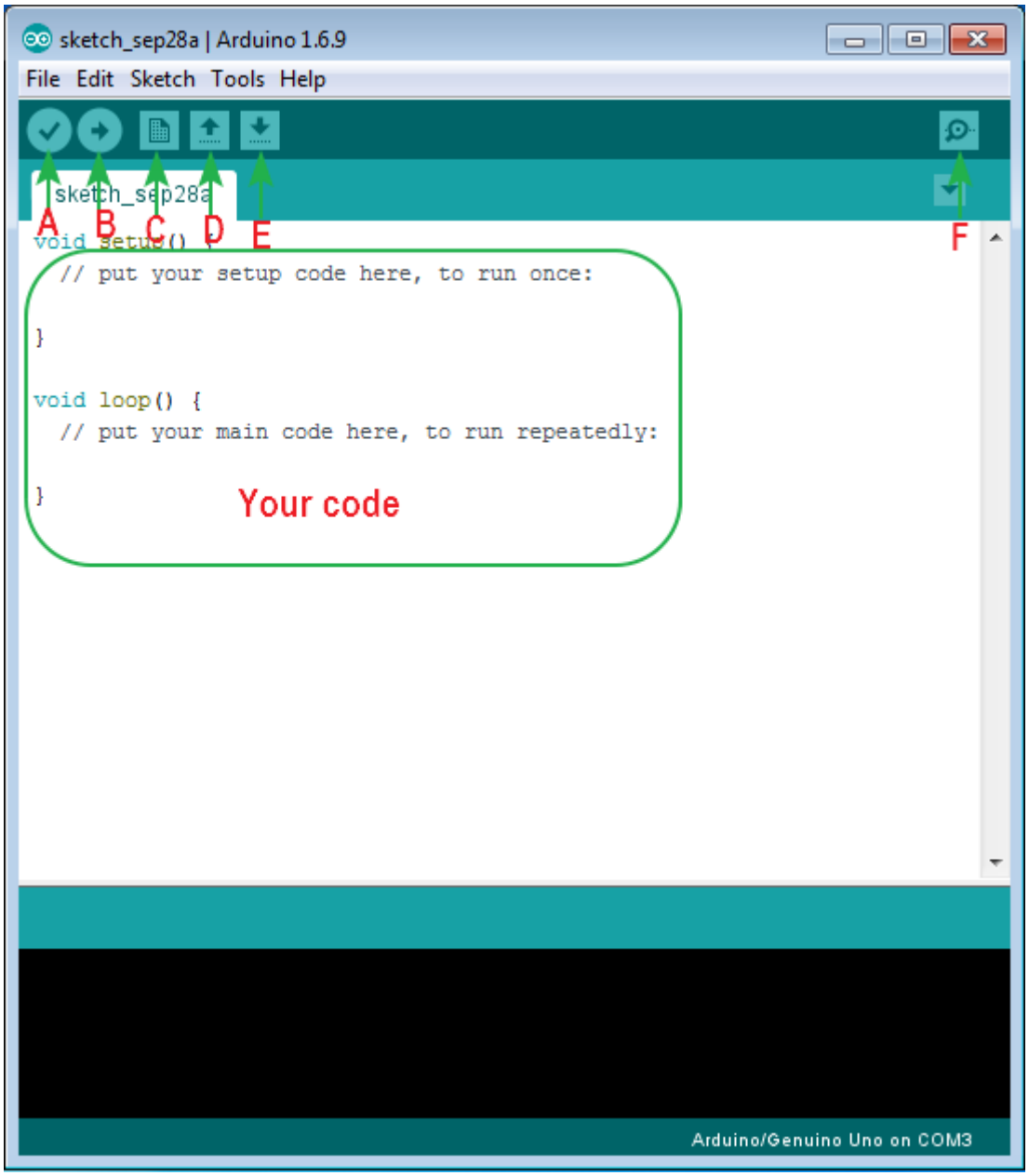
Se poate face comunicaţii seriale folosind oricare din pinii digitali.

Pentru comunicarea **I2C** (**TWI**) este inclusă o librărie Wire. Pentru comunicarea **SPI** se poate folosi librăria SPI.

După cum vedeţi în imagine,in partea dreapta, placa mai are o serie de pini marcaţi **ICSP** (In-Circuit Serial Programming). Aceşti pini pot fi folosiţi pentru reprogramarea microcontrolerului, sau ca pini de expansiune cu alte microcontrolere compatibile. Sunt conectaţi standard şi se poate folosi un cablu de 6 fire (**MOSI, MISO, SCK, VCC, GND,** şi pinul **RESET**).

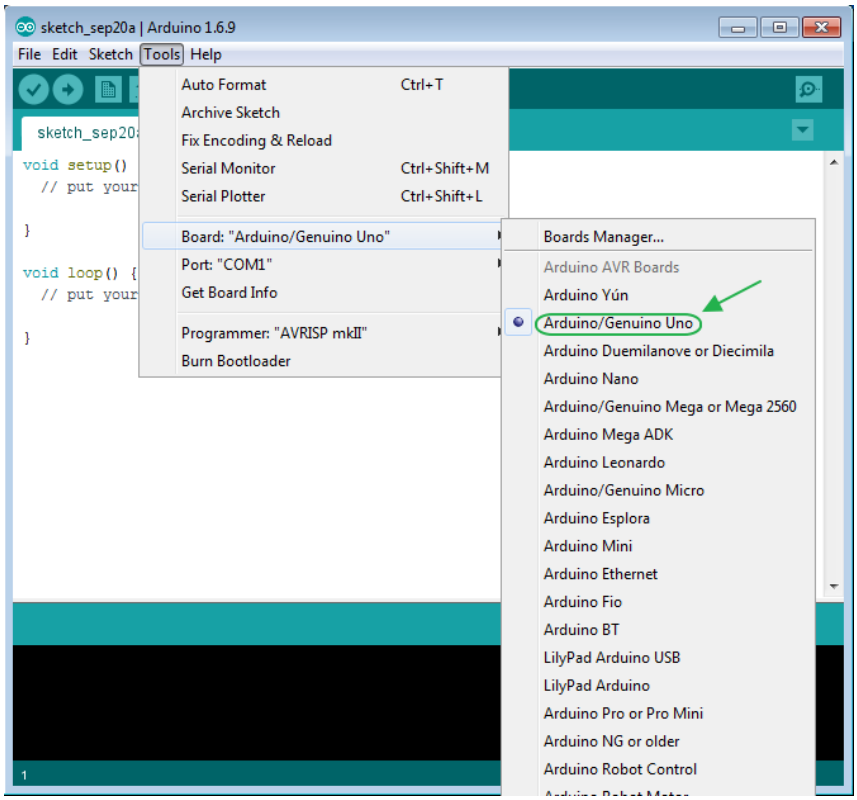
Pasul 4 : Setarea ID pentru Uno

Arduino interface introduction :

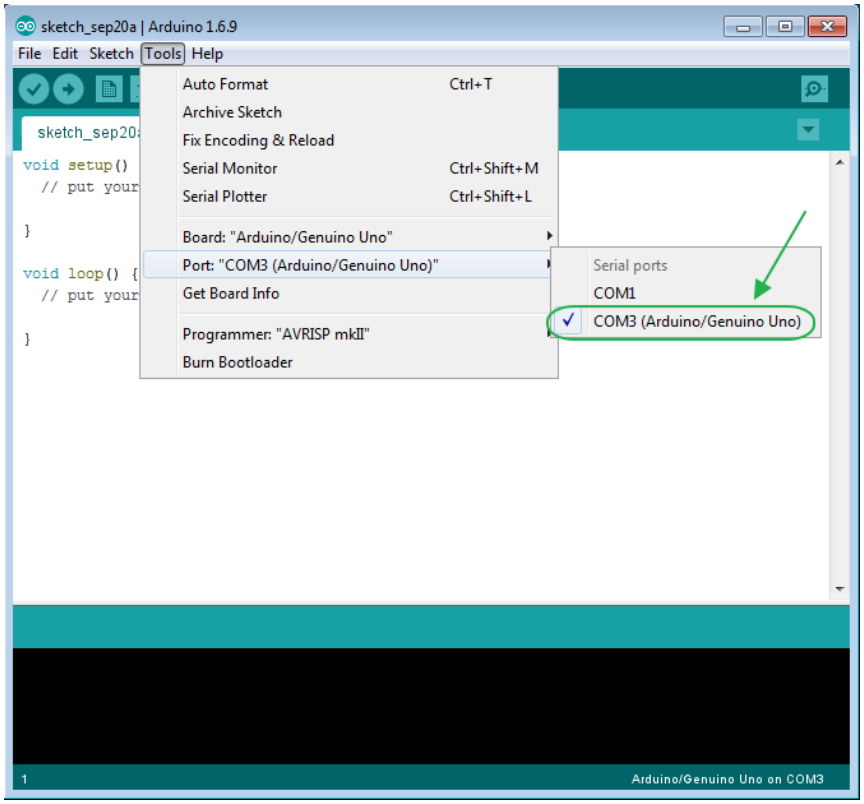


A -> Compile
B -> Upload
C -> New
D -> Open
E -> Save
F -> Serial monitor

Selectare board :



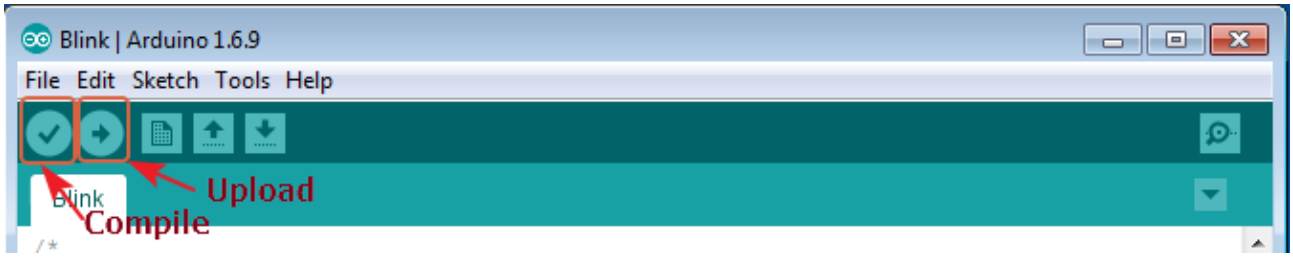
Selectare COM :



Selectați dispozitivul serial al plăcii din Serial Port meniu. Aceasta este probabil să fie COM3 sau mai mare (COM1 și COM2 sunt de obicei rezervate pentru porturile seriale hardware). Pentru a afla, puteți să deconectați UNO din USB și să deschideți din nou meniul; intrarea care dispare ar trebui să fie modulul Arduino. Reconectați placa și selectați portul serial.

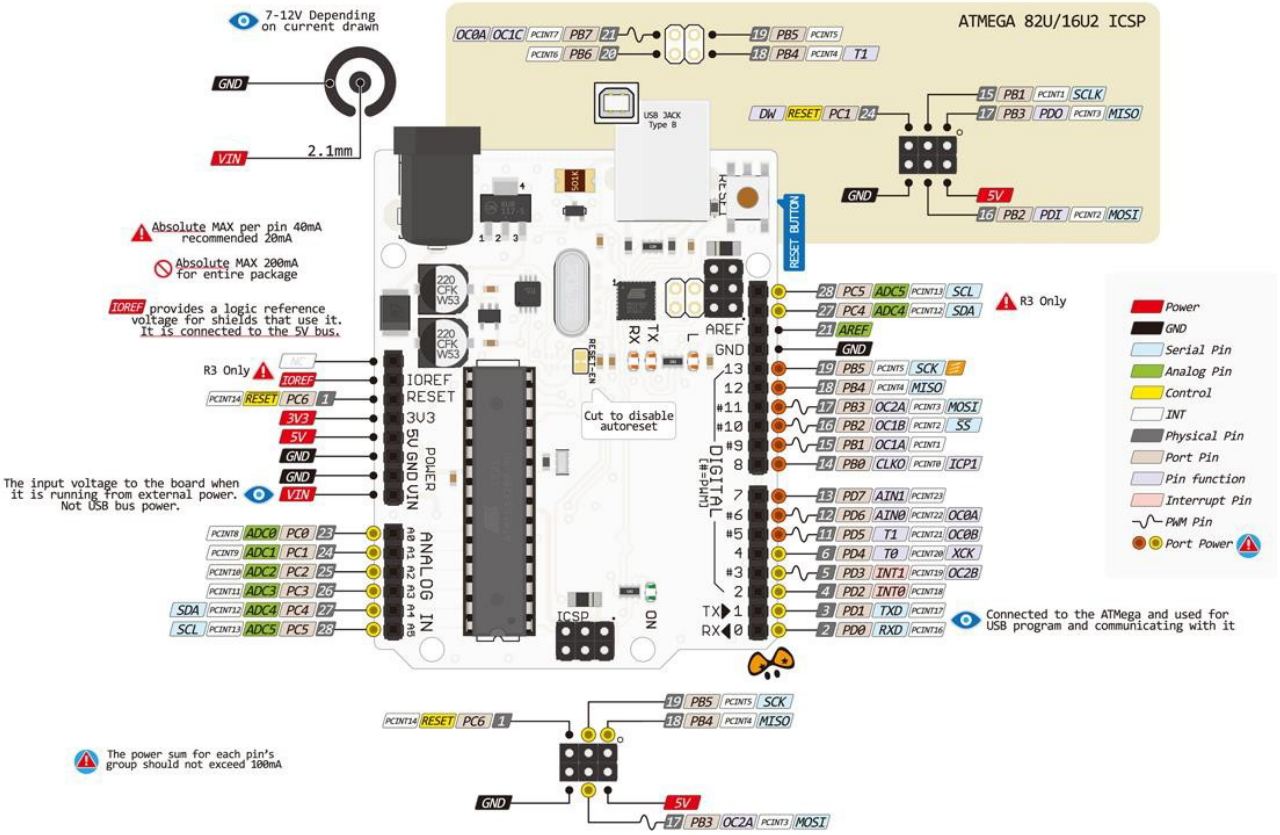
Pasul 5 : **Upload-area unui program** (cod sau sketch)

După ce ați scris sau copiat un cod, faceți clic pe butonul "Upload" din bara ID. Așteptați câteva secunde - ar trebui să vedeți LED-urile RX și TX de pe panou care clipesc. În cazul în care încărcarea are succes, mesajul "Done uploading". va apărea în bara de stare (în partea de jos).



În câteva secunde după finalizarea încărcării, ar trebui să vedeți LED-ul 13 (L) de pe UNO că începe să clipească.

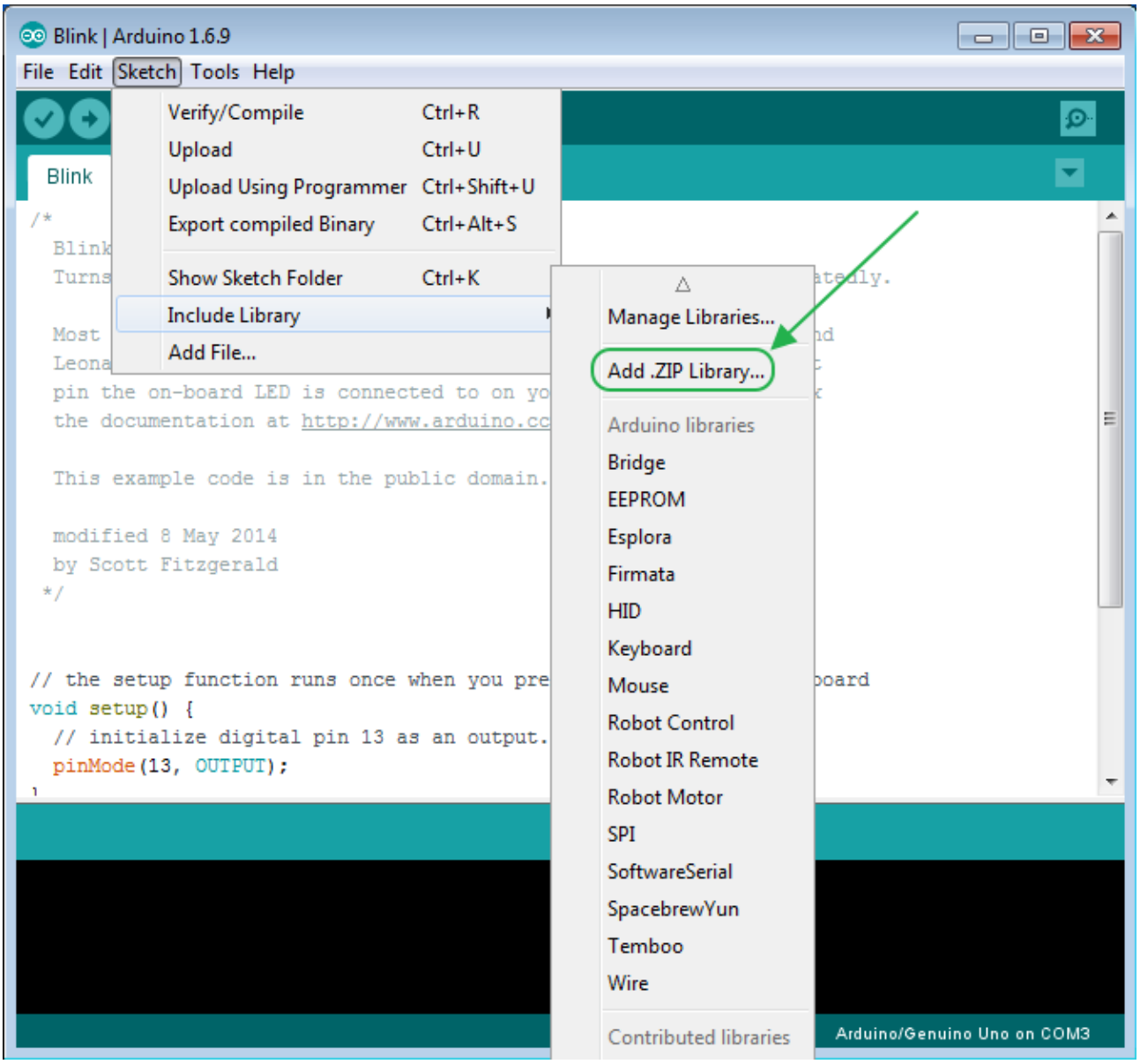
UNO R3 hardware



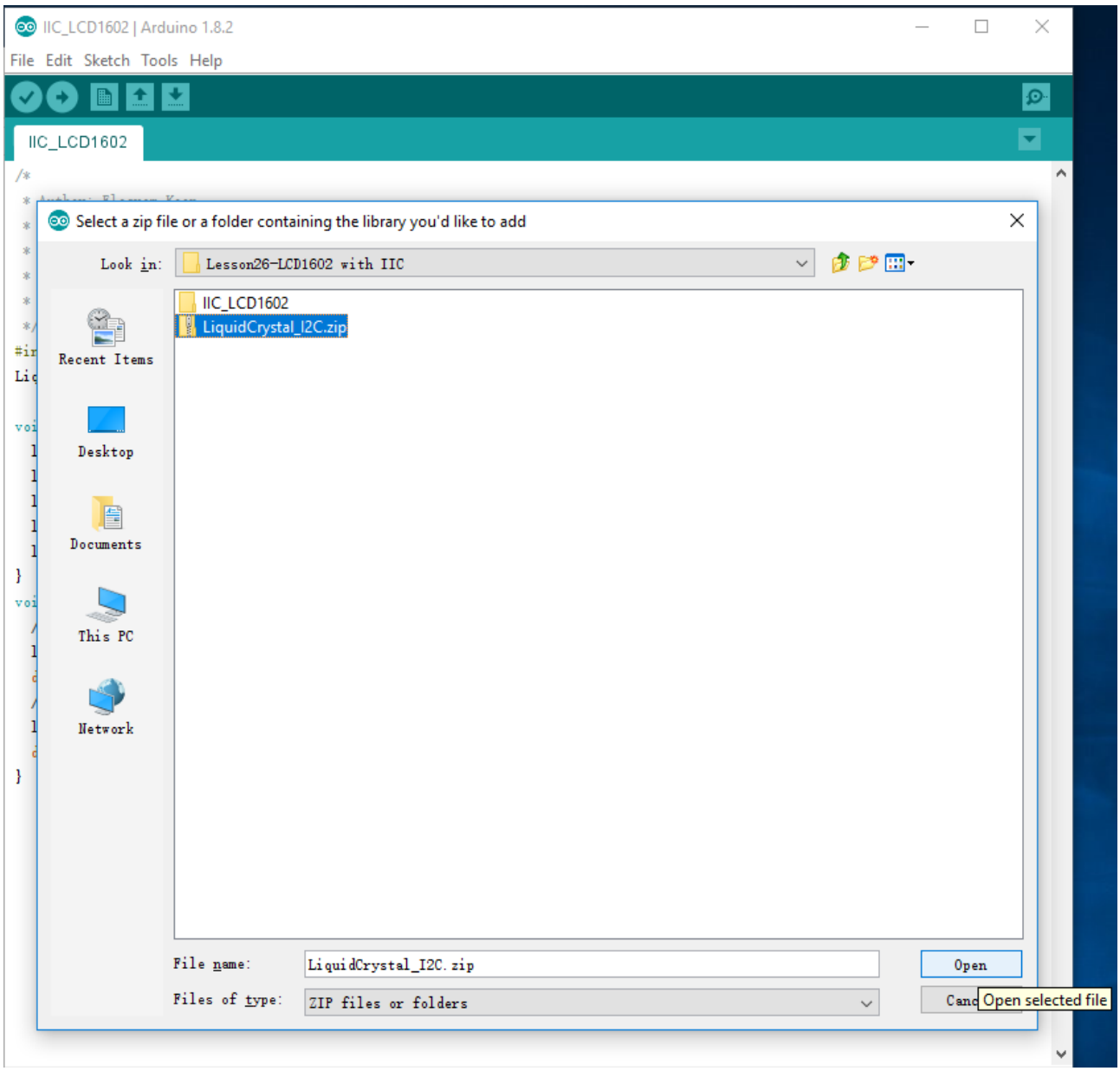
2.

Cum instalam o librărie (add library files)

Add library file: Sketch > Include Library > Add.zip Library



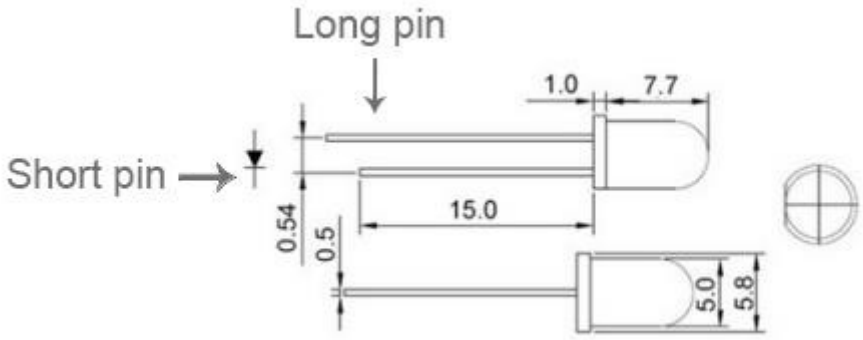
Selectați pachetul de compresie a fișierelor de librarie din fișierul cu codul demo, după cum urmează :


IIC_LCD1602 | Arduino 1.8.2
File Edit Sketch Tools Help
IIC_LCD1602
Select a zip file or a folder containing the library you'd like to add
Look in:
Lesson26-LCD1602 with IIC
IIC_LCD1602
LiquidCrystal_I2C.zip
Recent Items
Desktop
Documents
This PC
Network
File name:
LiquidCrystal_I2C.zip
Files of type:
ZIP files or folders
Open
Open selected file

3.

Exemple

Exemplul 1 LED Blink

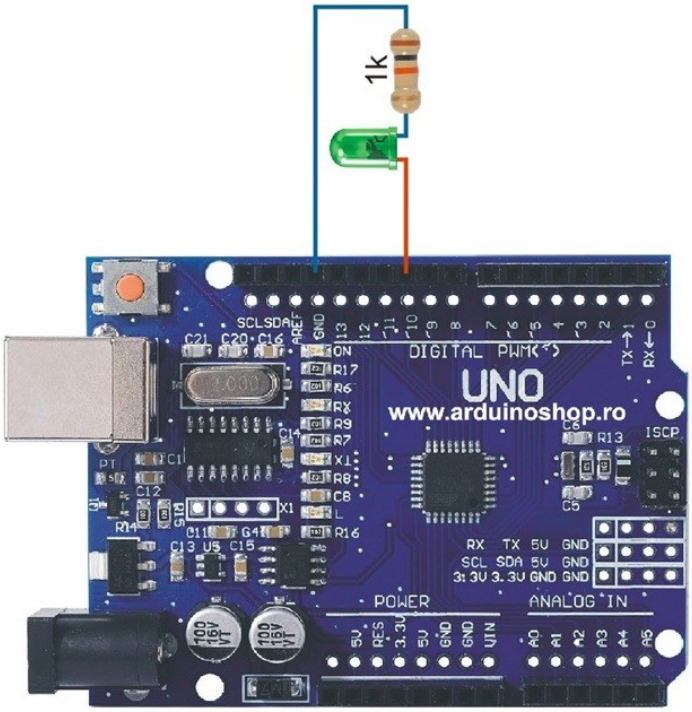


LED
Long pin -> +5V
Short pin -> GND

Atenție : Întotdeauna în serie cu LED-ul se montează o rezistenta de minimum 220Ω (recomandat intre 330Ω și 1k)

Cel mai simplu program pentru Arduino.

Voi folosi un LED de 3v pe care îl leg la pinul 10 cu un rezistor de 1kΩ.



Acest led se aprinde atunci când **pinul 13 digital** este pus in **HIGH** şi se stinge atunci când **pinul 13** este pus în **LOW**.

Vom scrie un cod :

```
void setup() {
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH);   // led aprins
delay(1000);
digitalWrite(13, LOW);    // led stins
delay(1000);
}
```

Acum voi face upload la program pe placa UNO prin USB.

Dacă mă uit la led-ul conectat la pinul 13 voi vedea că el clipeşte o dată pe secundă.

Să analizăm codul de mai sus.

La ” void setup”.

Avem o singură instrucţiune, care declară că **pinul 13** digital va fi folosit ca ieşire (OUTPUT).

La ” void loop”.

Avem o serie de instrucţiuni :

Prima care aprinde led-ul conectat la pinul 13 adică „digitalWrite(13, HIGH)”

A doua care “aşteaptă o secundă” adică “delay(1000)” înseamnă 1000 ms.

A treia îl stinge adică „digitalWrite(13, LOW)”

Şi a patra iaraşi “aşteaptă o secundă”.

Instrucţiunile din „loop” se execută ciclic.

Exemplul următor este extrem de similar cu acesta, doar că în locul led-ului montat din fabrică voi folosi un led exterior plăcii.

Voi folosi un **LED** de 3v pe care îl leg la **pinul 10** cu un **rezistor de 1kΩ** (foarte important limitează curentul prin LED).

Vom scrie un cod :

```
void setup() {
pinMode(10, OUTPUT);
}
void loop() {
digitalWrite(10, HIGH);   // led aprins
delay(1000);
digitalWrite(10, LOW);    // led stins
delay(1000);
}
```

Acum vom face upload la program pe placa UNO prin USB.

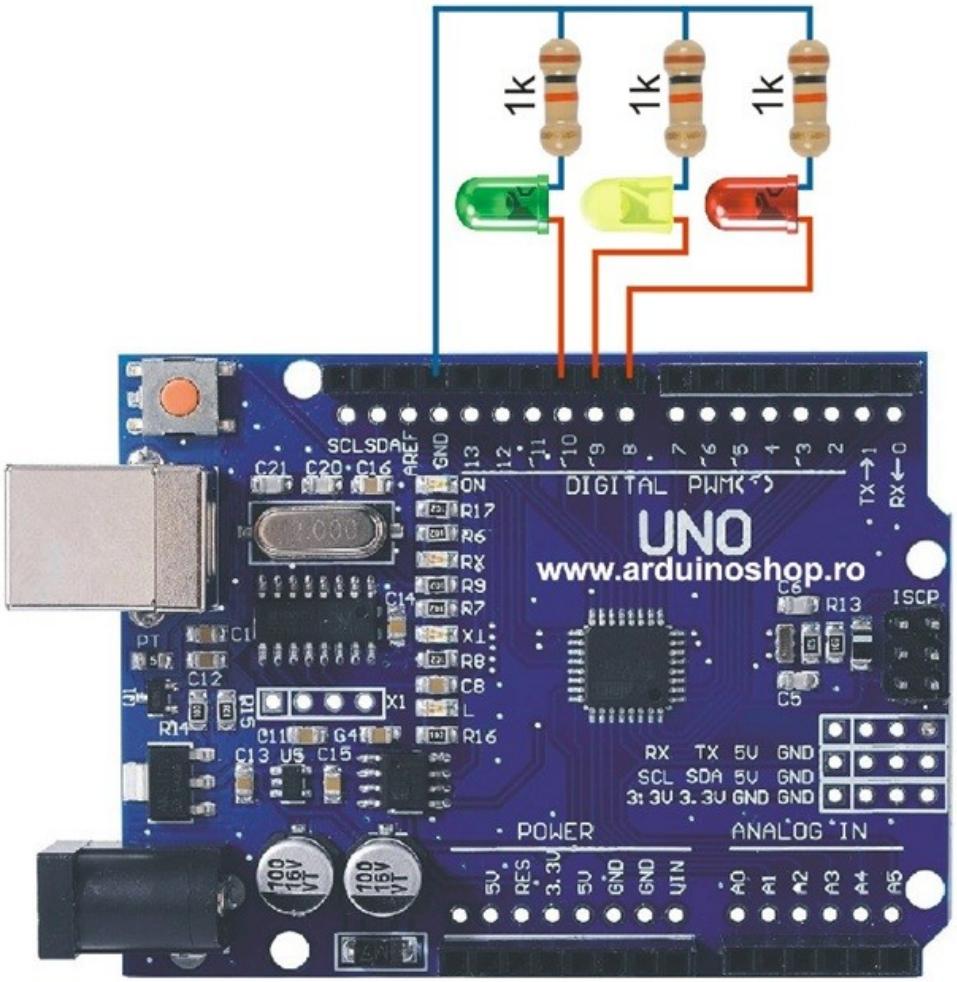
Dacă mă uit la led-ul conectat la pinul 10 voi vedea că el clipeşte o dată pe secundă.

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

Exemplul 2 UNO Semafor cu trei LED-uri verde-galben-rosu

Înainte ar fi bine sa reciteşti capitolul “Descrierea pinilor UNO R3”

Vom folosi trei LED-uri de 3v
pe care le leg la pinii 8, 9, 10 cu rezistoare de 1kΩ.



Vom scrie un cod :

```
void setup() {
pinMode(8, OUTPUT);    // LED verde
pinMode(9, OUTPUT);    // LED galben
pinMode(10, OUTPUT);   // LED rosu
}
```

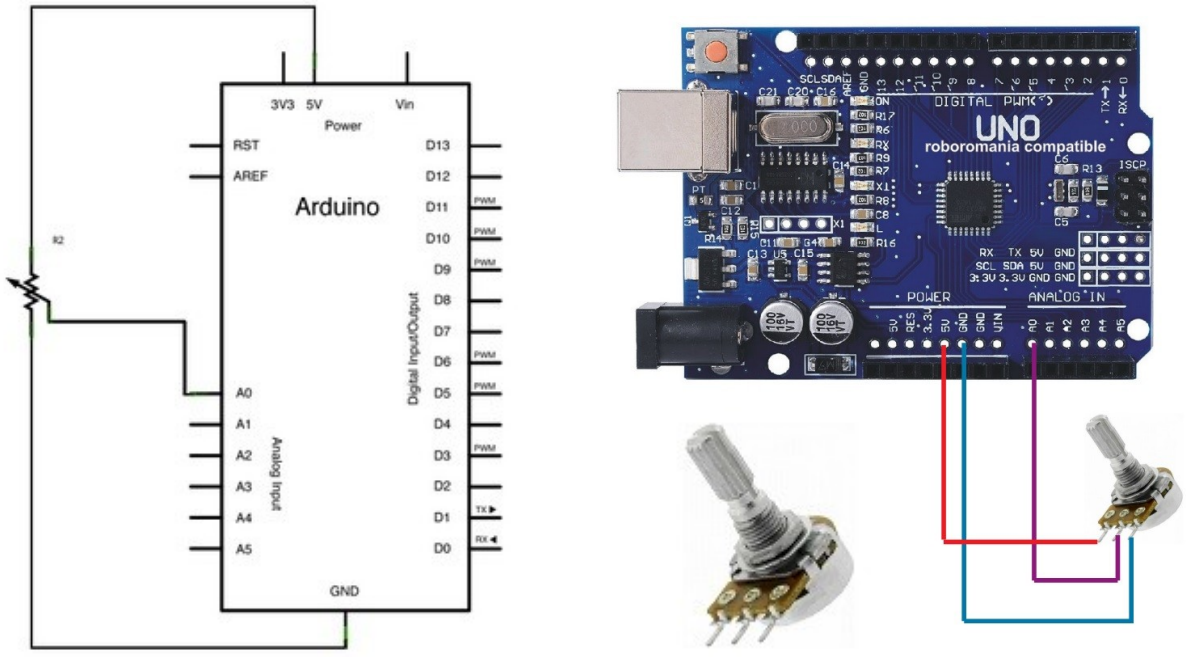
```
void loop() {
digitalWrite(10, HIGH);   // verde aprins
delay(2000);
digitalWrite(10, LOW);    // verde stins
digitalWrite(9, HIGH);    // galben aprins
delay(1000);
digitalWrite(9, LOW);     // galben stins
digitalWrite(8, HIGH);    // rosu aprins
delay(2000);
digitalWrite(8, LOW);     // rosu stins
digitalWrite(9, HIGH);    // galben aprins
delay(1000);
digitalWrite(9, LOW);     // galben stins
}
```

Acum vom face upload la program pe placa UNO prin USB.

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

Exemplul 3 Analog Input potențiometru

În acest exemplu folosim un rezistor variabil (un potențiometru de 10k), citim valoarea sa utilizând o intrare analogică a unei plăci Arduino și o citim pe Serial Monitor. Valoarea analogică a rezistorului este citită ca o tensiune deoarece acesta este modul în care funcționează intrările analogice.



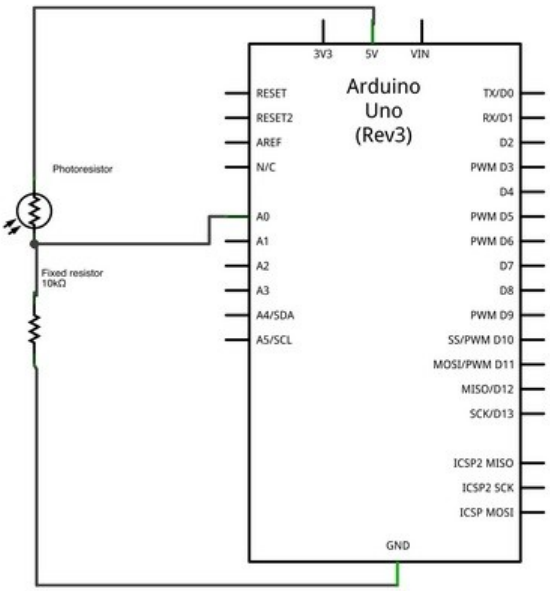
Conectați trei fire la placa Arduino Primul trece la GND de la unul dintre pinii exteriori ai potențiometrului. Al doilea este de la +5 volți la celălalt capăt al potențiometrului. Al treilea merge de la intrarea analogică A0 la pinul central al potențiometrului.

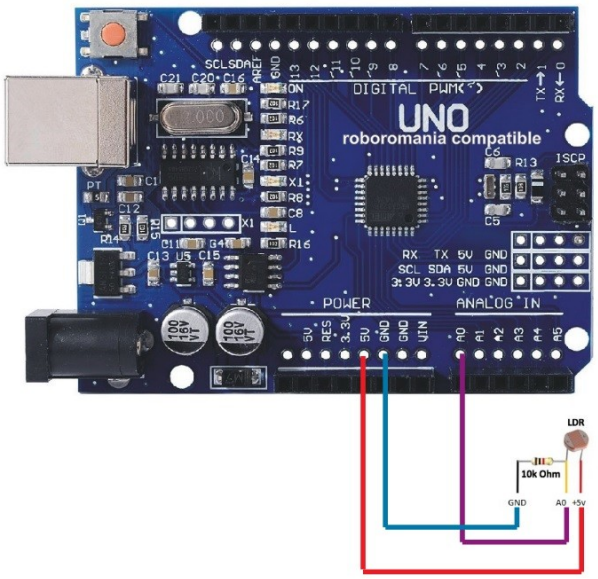
Codul :

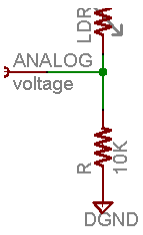
```
void setup() {
  Serial.begin(9600);
}
void loop() {
  // selectați pinul de intrare pentru senzor potențiometru, respectiv A0
  // sensorValue este o variabilă pentru a stoca valoarea provenită de la senzor
  // citire valoare de la senzor
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);  // trimite pe serial (USB) valoarea (intre 0 și 1024)
  delay(1000); // timp de pauză 1 secunde
}
```

Exemplul 4 Analog Input fotorezistenta GL5528

În acest exemplu folosim un rezistor variabil (o fotorezistenta GL5528), citim valoarea sa utilizând o intrare analogică a unei plăci Arduino și o citim pe Serial Monitor. Valoarea analogică a rezistorului este citită ca o tensiune deoarece acesta este modul în care funcționează intrările analogice.







Codul :

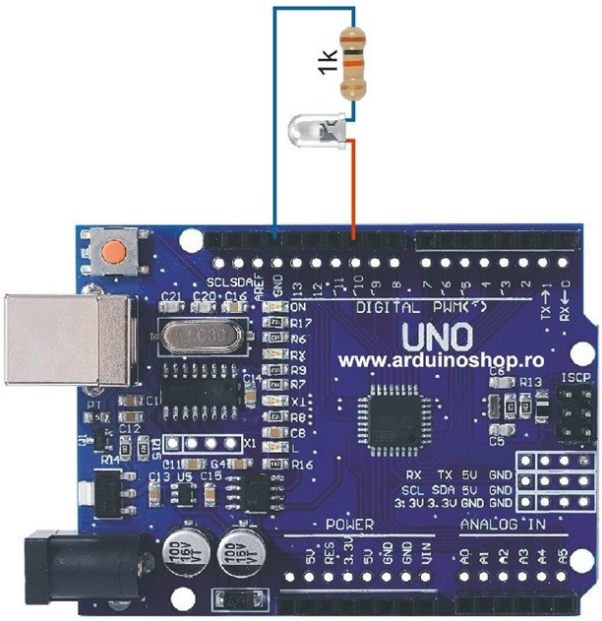
```
void setup() {
  Serial.begin(9600);
}
void loop() {
  // selectați pinul de intrare pentru senzor fotorezistenta, respectiv A0
  // sensorValue este o variabilă pentru a stoca valoarea provenită de la senzor
  // citire valoare de la senzor
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);  // trimite pe serial (USB) valoarea (intre 0 și 1023)
  delay(1000); // timp de pauză 1 secunde
}
```

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

Exemplul 5 LED cu intensitate variabilă (Fading)

Un program simplu pentru Arduino.

Voi folosi un LED de 3v pe care îl legăm la pinul 10 cu un rezistor de 1kΩ.



Acest led se aprinde atunci când pinul 10 digital este pus in **HIGH** şi se stinge atunci când pinul 10 este pus în **LOW**.

Acum îi voi varia aprinderea.

Deşi UNO nu scoate tensiune variabilă pe porturile digitale, exista o posibilitate de a genera un semnal dreptunghiular între 0V și 5V, foarte rapid, şi în funcție de cât timp stă în 5V și cat timp sta in 0V, puterea semnalului variază.

Numele acestui gen de semnal este „**PWM**„.

De remarcat faptul că doar pinii 3, 5, 6, 9, 10 şi 11 sunt capabili să genereze semnal PWM.

Codul :

```
void setup() {
pinMode(10, OUTPUT);
}
void loop() {
for (int i = 0; i < 255; i++) {
```

```
analogWrite(10, i);
delay(50);
}
for (int i = 255; i > 0; i–) {
  analogWrite(10, i);
  delay(50);
  }
}
```

Acum vom face upload la program pe placa UNO prin USB.

Să analizăm codulul de mai sus.

La ” void setup”.

Avem o singură instrucţiune, care declară că pinul 10 digital va fi folosit ca ieşire (**OUTPUT**).

La ” void loop”.

Avem instrucţiunea **analogWrite**, care defineşte puterea semnalului **PWM** de ieşire.

Ca parametri, instrucţiunea analogWrite primeste pinul (10, în cazul meu), şi puterea

semnalului (variabilă, de la 0 la 255).

Aceasta instrucţiune este apelată într-un ciclu “for”, care modifică

valoarea variabilei “i” între 0 si 255.

Efectul va fi ca led-ul se va aprinde gradat pană la maxim, iar apoi se va stinge treptat.

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

Exemplul 6

LED ON/OFF comandat la bătaie din palme „CLAPS”

Puteți să aprindeți/stingeți lumina din camera la o bătaie din palme

Avem nevoie de :

Placă de dezvoltare UNO R3 (sau oricare)

Modul senzor de detectare a sunetului

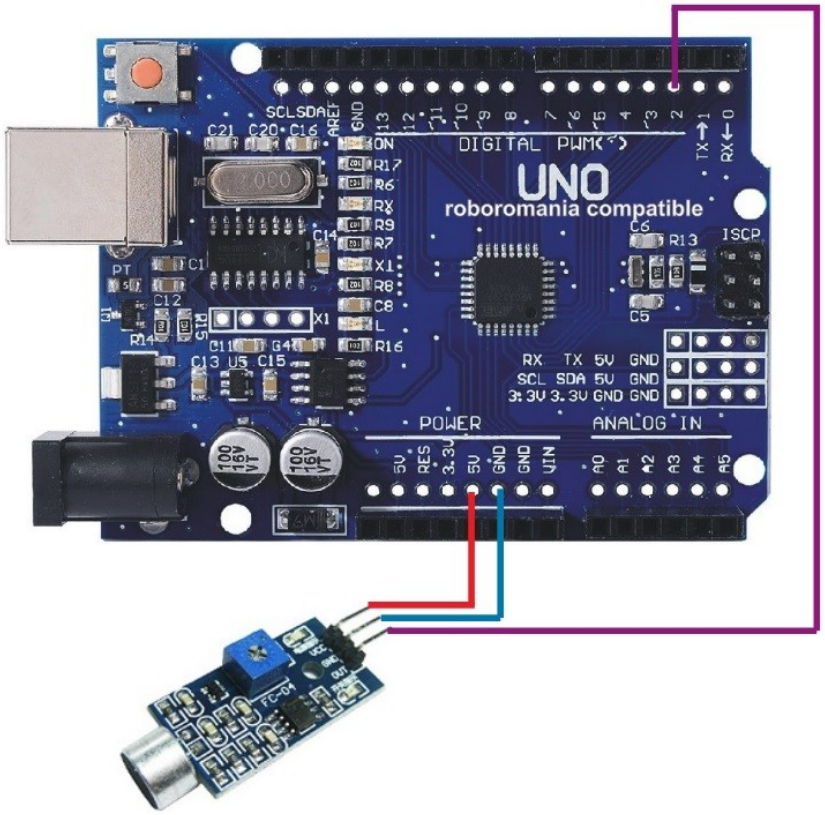
Ca **LED** îl vom folosi pe cel de la placa de dezvoltare : **Led13**

Codul :

```
//--------------------------------
const int sound_pin = 2;
const int led_pin = 13;
const unsigned long min_time_distance_between_signals = 100UL;
const unsigned long max_time_distance_between_signals = 500UL;

const unsigned long delay_after_triggered = 2000UL;

bool led_state;
unsigned long last_time_since_clap;
unsigned long last_action_time;

void setup ()
{
 Serial.begin(9600);
 pinMode (sound_pin, INPUT) ;
 pinMode(led_pin, OUTPUT);
 digitalWrite(led_pin, LOW);

 led_state = false;
 last_action_time = 0;
```

```
 last_time_since_clap = 0;
}

void update_state(bool new_state, unsigned long current_time) {
 led_state = new_state;
 last_action_time = current_time;
 digitalWrite(led_pin, led_state ? HIGH : LOW);
 Serial.print("state = ");
 Serial.println(led_state);
}

void loop () {
 int sound_detected_state = digitalRead(sound_pin);

 unsigned long current_time = millis();
 unsigned long delta_time_since_last_action = current_time - last_action_time;

 if ((delta_time_since_last_action > delay_after_triggered) && ( sound_detected_state == HIGH )) {
 if (!led_state) {
 unsigned long current_delta_time_between_signals = current_time - last_time_since_clap;
 if (current_delta_time_between_signals > max_time_distance_between_signals) {
 last_time_since_clap = current_time;
 } else if (current_delta_time_between_signals > min_time_distance_between_signals) {
 update_state(true, current_time);
 }
 } else {
 update_state(false, current_time);
 }
 }
}

//--------------------------------
```

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

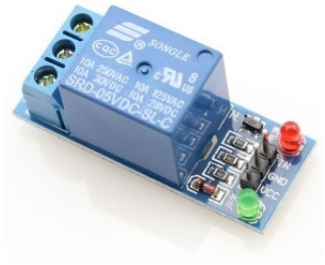


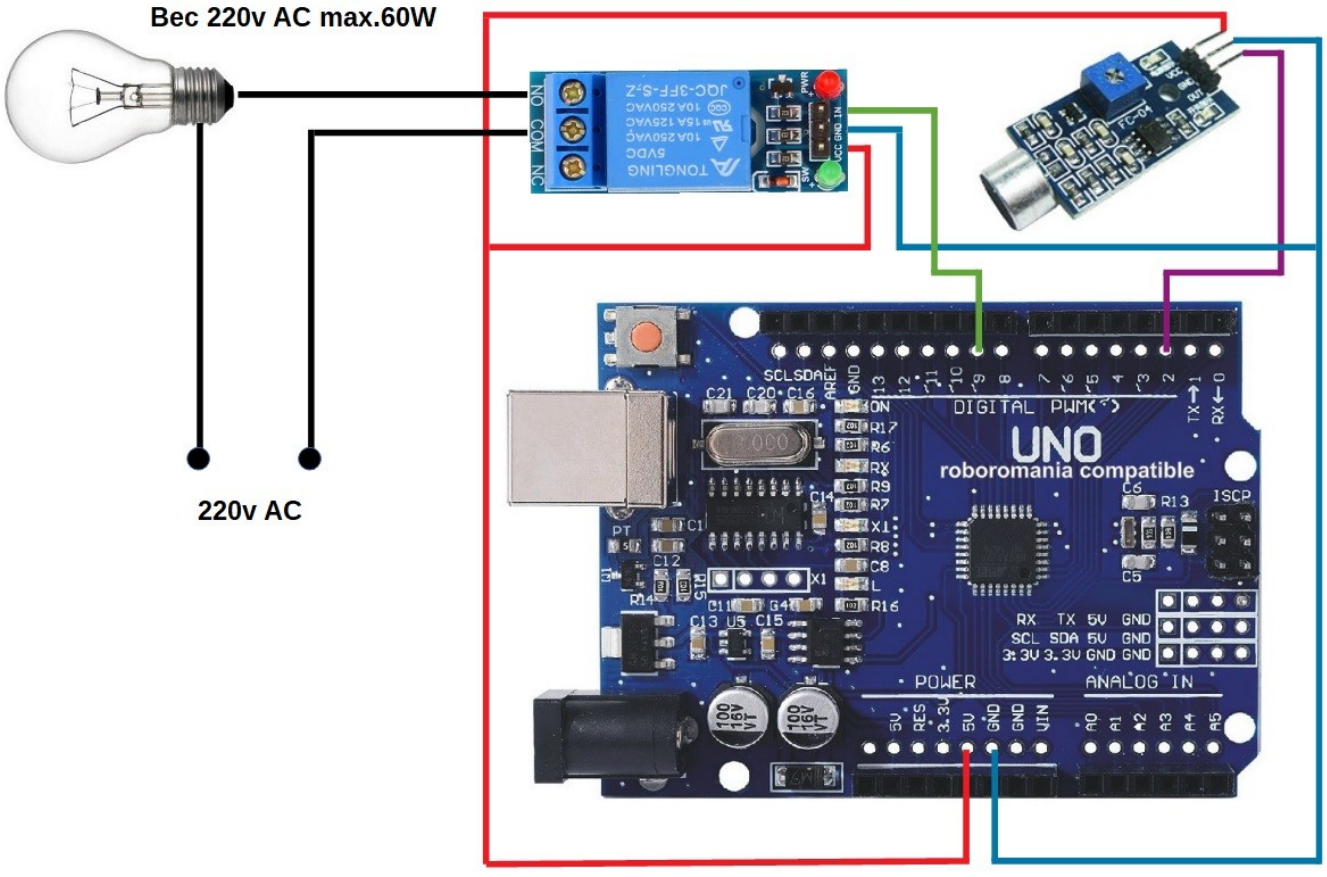
Exemplul 7

LED ON/OFF comandat la bătaie din palme „CLAPS” folosim și un modul releu și un bec la 220v

Avem nevoie de UNO, Modul sunet similare cu exemplul anterior și

Un modul releu





Atenție când realizați montajul sa aveți grija mare la firele de 220v AC, pericol de electrocutare.

Codul :

```
//------------------------------
const int sound_pin = 2;
const int rel_pin = 9;
const unsigned long min_time_distance_between_signals = 100UL;
const unsigned long max_time_distance_between_signals = 500UL;

const unsigned long delay_after_triggered = 2000UL;

bool rel_state;
unsigned long last_time_since_clap;
unsigned long last_action_time;

void setup ()
{
 Serial.begin(9600);
 pinMode (sound_pin, INPUT) ;
 pinMode(rel_pin, OUTPUT);
 digitalWrite(rel_pin, LOW);

 rel_state = false;
 last_action_time = 0;
 last_time_since_clap = 0;
}

void update_state(bool new_state, unsigned long current_time) {
 rel_state = new_state;
 last_action_time = current_time;
 digitalWrite(rel_pin, rel_state ? HIGH : LOW);
 Serial.print("state = ");
 Serial.println(rel_state);
}

void loop () {
 int sound_detected_state = digitalRead(sound_pin);

 unsigned long current_time = millis();
 unsigned long delta_time_since_last_action = current_time - last_action_time;

 if ((delta_time_since_last_action > delay_after_triggered) && ( sound_detected_state == HIGH )) {
 if (!rel_state) {
 unsigned long current_delta_time_between_signals = current_time - last_time_since_clap;
 if (current_delta_time_between_signals > max_time_distance_between_signals) {
 last_time_since_clap = current_time;
 } else if (current_delta_time_between_signals > min_time_distance_between_signals) {
 update_state(true, current_time);
 }
 } else {
 update_state(false, current_time);
 }
 }
}

//------------------------------
```

Atenție dacă copiati codul din acest manual în ID poate sa apara erori de la „fonturi”, pe care trebuie să le corectați.

Urmează Vol.2