Transmitator Radio Brick RFM12B + Arduino

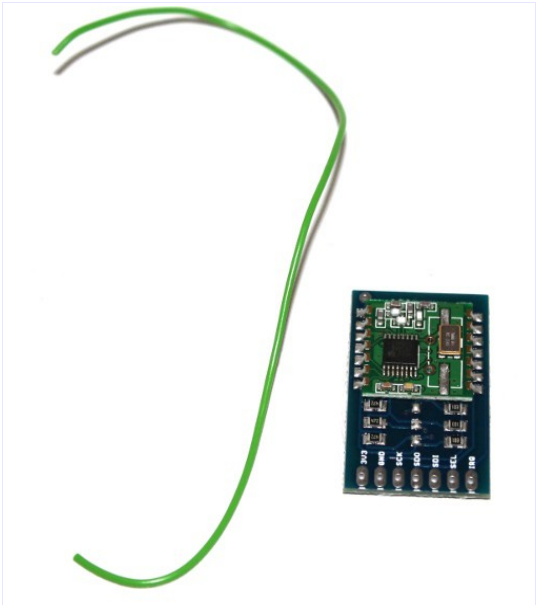
Transmitatorul radio este un modul brick de emisie/receptie radio ISM ce poate functiona in 3 game de frecvente: 433, 868 si 915 Mhz. Varianta actuala a modulului brick lucreaza in frecventa de 433 Mhz si poate realiza transfer de date la viteze de pana la 115Kbps, pe o distanta de cel mult 300m.

Modulul se alimenteaza la 3.3V si se conecteaza impreuna cu placa Arduino printr-un set de fire de conexiune mama-tata.

Transmitatorul radio este disponibil in 2 variante, prima varianta compatibila cu dispozitivele ce functioneaza la o tensiune de 5V, iar a doua varianta compatibila cu dispozitivele ce lucreaza la 3.3V. Varianta care functioneaza la 5V poate fi adaptata foarte usor pentru a functiona la 3.3V, iar varianta care functioneaza la 3.3V poate fi adaptata sa lucreze la 5V, in aceeasi maniera.

Pentru ca transmitatorul sa functioneze la 3.3V trebuie sa lipesti cei 3 jumperi de pe placa. Pentru ca transmitatorul sa functioneze la 5V trebuie sa dezlipesti sau sa separi cei 3 jumperi de pe placa. Poti realiza lipirea si dezlipirea cu ajutorul unui starter kit basic.

http://www.robofun.ro/bricks/starter-kit-electronica



Practic, modulul iti permite sa transferi date dintr-o placa Arduino catre alta in ambele sensuri. Acest lucru devine foarte util atunci cand doresti sa realizezi o retea wireless intr-un mod simplu si ieftin.

http://www.robofun.ro/bricks/radio-rfm12b-5v

sau

http://www.robofun.ro/bricks/radio-rfm12b-3v3

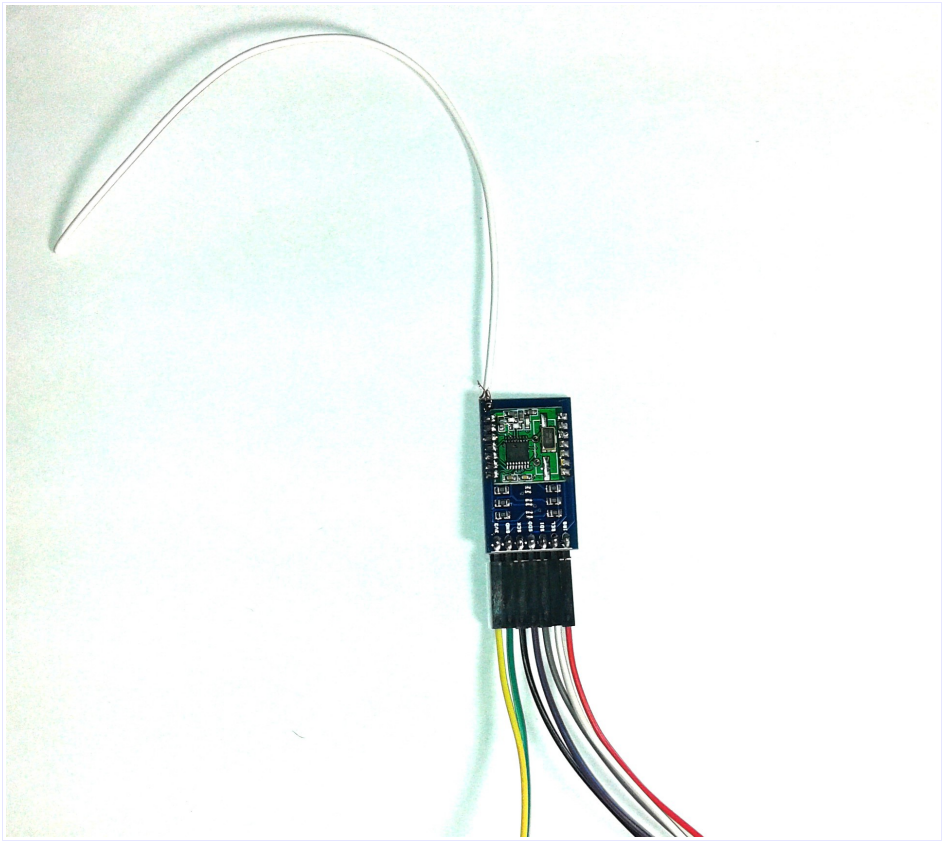
Pentru a realiza o comunicatie intre 2 puncte vei avea nevoie de urmatoarele componente:

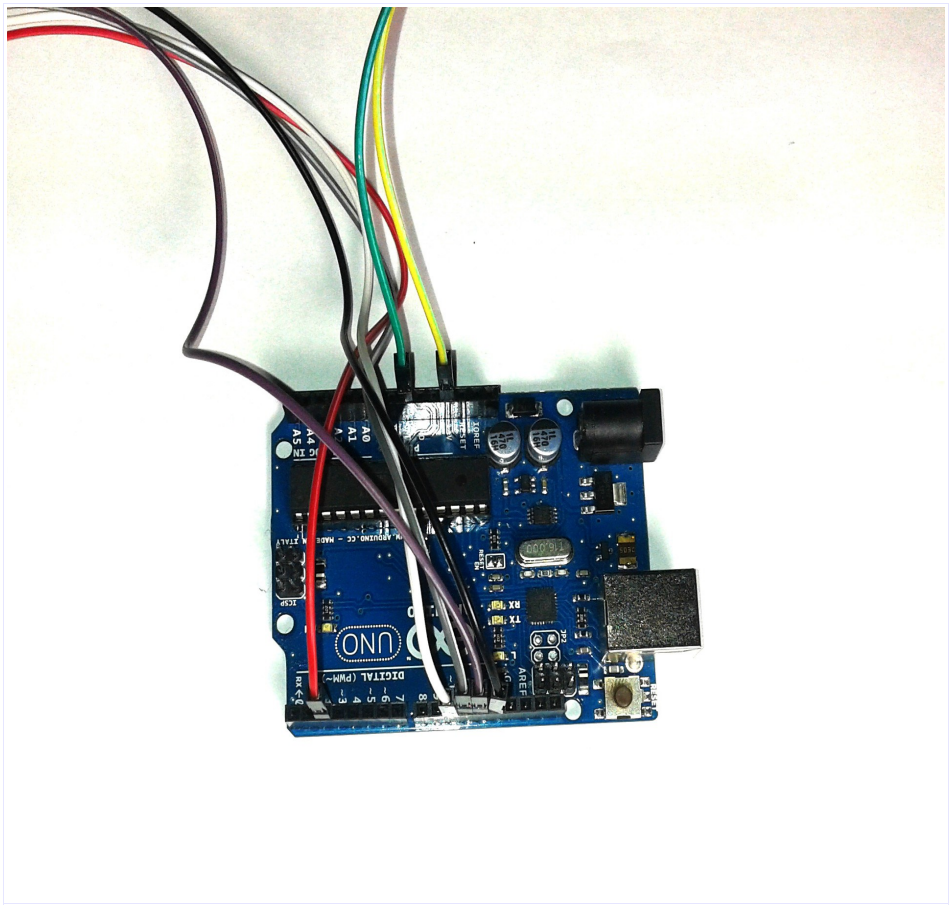
- 2 placi Arduino - http://www.robofun.ro/arduino
- 2 sau mai multe transmitatoare radio brick - http://www.robofun.ro/bricks/radio-rfm12b-5v
- Fire de conexiune mama-tata - http://www.robofun.ro/cabluri

Cum se conecteaza un transmitator brick ?

Conexiunile electrice dintre un transmitator si o placa Arduino se realizeaza foarte usor, folosind firele de conexiune mama-tata. Urmeaza tabelul de mai jos:

Transmitator Brick pin 3.3V	Arduino pin 3.3V
Transmitator Brick pin GND	Arduino pin GND
Transmitator Brick pin SCK	Arduino pin digital 13
Transmitator Brick pin SDO	Arduino pin digital 12
Transmitator Brick pin SDI	Arduino pin digital 11
Transmitator Brick pin nSEL	Arduino pin digital 10
Transmitator Brick pin nIRQ	Arduino pin digital 2

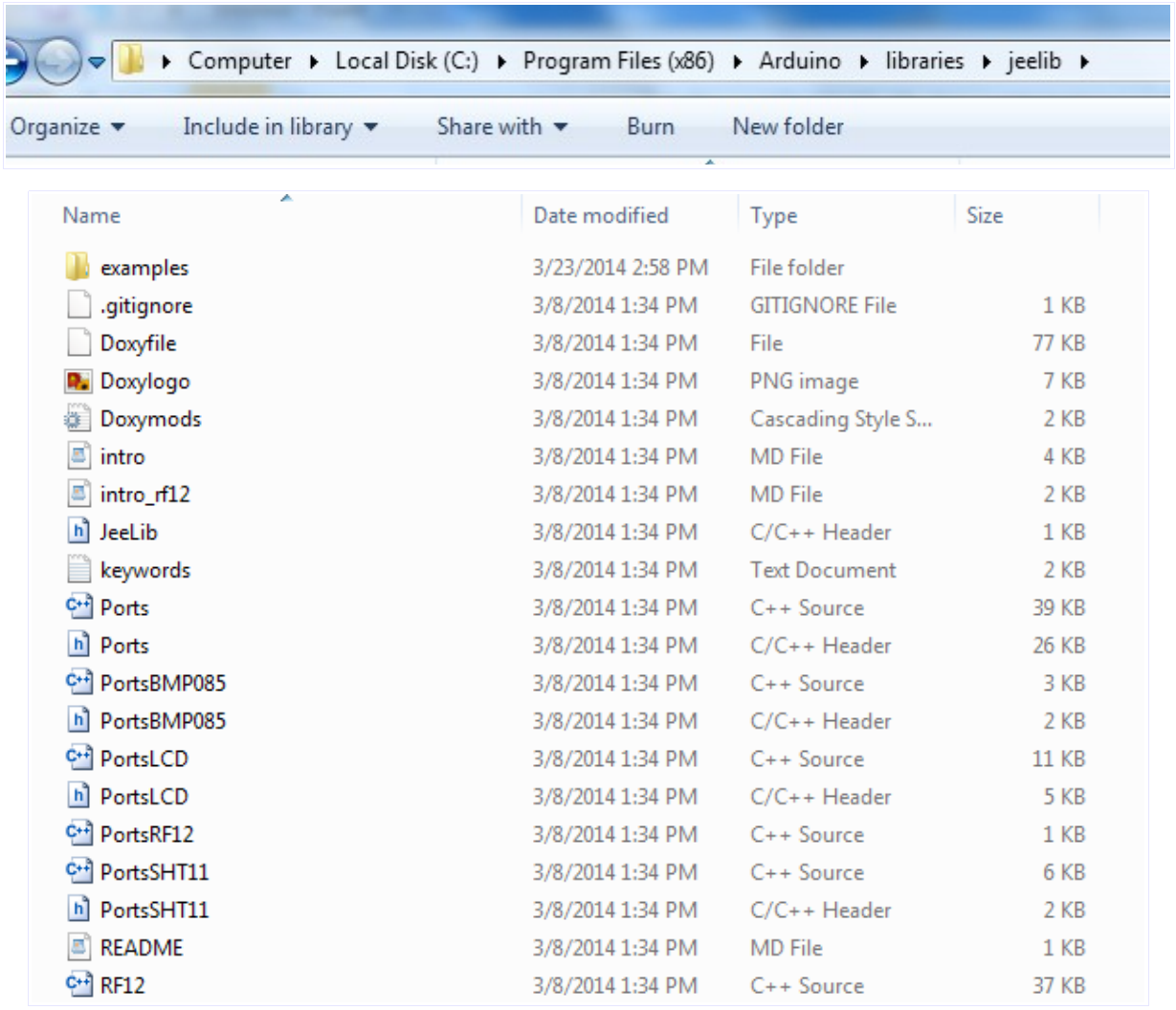



UNO
DIGITAL (PWM~)

Cum se realizeaza o conexiune de date?

Transmitatoarele radio iti permit sa transferi date intre doua sau mai multe placi Arduino, intr-un mod simplu si usor. Modulele radio necesita biblioteca JeeLib, a celor de la JeeLabs. Tot ce trebuie sa faci este sa descarci libraria si sa o instalezi in mediul Arduino.

Descarca libraria jeelib accesand link-ul de mai jos, dezarhiveaza fisierul si copiaza folderul „jeelib-master“ in folderul „libraries“ din Arduino. Nu uita sa redenumesti folderul copiat in „jeelib“.

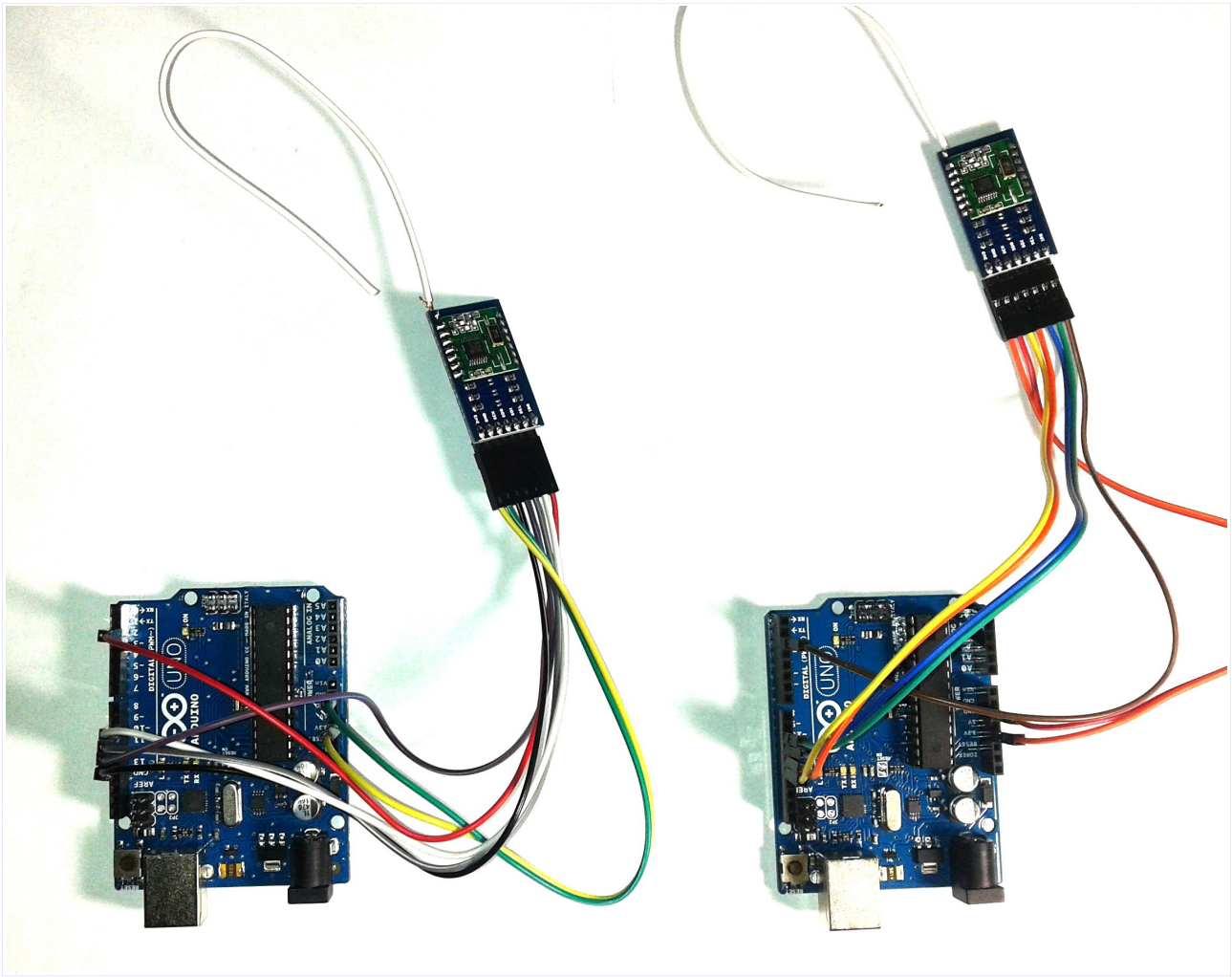


https://github.com/jcw/jeelib

Pentru o documentatie completa referitoare la libraria jeelib, acceseaza link-ul de mai jos:

http://jeelabs.net/pub/docs/jeelib/md_intro_rf12.html

O conexiune de date necesita cel putin 2 transmitatoare pe care le vei conecta la placile Arduino, in aceeasi maniera urmand tabelul de mai sus. Practic, pentru o conexiune wireless simpla, vei avea nevoie de 2 placi Arduino, 2 transmitatoare si cateva fire de conexiune mama-tata.



Vei programa o placa Arduino sa se comporte ca un receptor, iar cealalta ca un emitator. Practic, emitatorul sau prima placa Arduino va transfera informatii catre receptor, adica cealalta placa Arduino. Receptorul iti va afisa informatiile primite prin intermediul Monitorului Serial. Ramane la alegerea ta sa decizi ce fel de informatie vrei sa transmiti, dar ca si exemplu aceasta poate sa fie o temperatura, o valoare ce reprezinta presiunea, umiditatea sau nivelul de inclinatie al unui obiect. Mai jos, ai la dispozitie codurile sursa pentru emitator si pentru receptor.

Sketch-ul Arduino

Pentru a programa prima placa Arduino care se comporta ca si receptor, copiaza sketch-ul de mai jos si incarca-l in placa.

```
//Simple RFM12B wireless demo - Receiver - no ack
//Glyn Hudson openenergymonitor.org GNU GPL V3 12/4/12
//Credit to JCW from Jeelabs.org for RFM12

#include <JeeLib.h>

#define myNodeID 30          //node ID of Rx (range 0-30)
#define network     210      //network group (can be in the range
1-250).
#define freq RF12_433MHZ     //Freq of RF12B can be RF12_433MHZ,
RF12_868MHZ or RF12_915MHZ. Match freq to module

typedef struct { int power1, power2, power3, battery; } PayloadTX;
// create structure - a neat way of packaging data for RF comms
PayloadTX emontx;

const int emonTx_NodeID=10;            //emonTx node ID

void setup() {

  rf12_initialize(myNodeID,freq,network);   //Initialize RFM12 with
settings defined above
  Serial.begin(9600);
  Serial.println("RF12B demo Receiver - Simple demo");

 Serial.print("Node: ");
 Serial.print(myNodeID);
 Serial.print(" Freq: ");
 if (freq == RF12_433MHZ) Serial.print("433Mhz");
 if (freq == RF12_868MHZ) Serial.print("868Mhz");
 if (freq == RF12_915MHZ) Serial.print("915Mhz");
 Serial.print(" Network: ");
 Serial.println(network);
}

void loop() {


 if (rf12_recvDone()){
  if (rf12_crc == 0 && (rf12_hdr & RF12_HDR_CTL) == 0) {

    int node_id = (rf12_hdr & 0x1F);         //extract nodeID from
payload

    if (node_id == emonTx_NodeID)  {             //check data is
coming from node with the corrct ID
        emontx=*(PayloadTX*) rf12_data;            // Extract the
data from the payload
       Serial.print("power1: "); Serial.println(emontx.power1);
```

```
        Serial.print("power2: "); Serial.println(emontx.power2);
        Serial.print("power3: "); Serial.println(emontx.power3);
        Serial.print("battery: "); Serial.println(emontx.battery);
        Serial.println("  ");
  }
 }
}
}
```

Pentru a programa a doua placa Arduino care se comporta ca si emitator, copiaza sketch-ul de mai jos si incarca-l in placa.

```
//Simple RFM12B wireless demo - transimtter - no ack
//Glyn Hudson openenergymonitor.org GNU GPL V3 7/7/11
//Credit to JCW from Jeelabs.org for RFM12

#include <JeeLib.h>  //from jeelabs.org

#define myNodeID 10          //node ID of tx (range 0-30)
#define network     210      //network group (can be in the range
1-250).
#define freq RF12_433MHZ     //Freq of RF12B can be RF12_433MHZ,
RF12_868MHZ or RF12_915MHZ. Match freq to module


typedef struct { int power1, power2, power3, battery; } PayloadTX;
// create structure - a neat way of packaging data for RF comms
PayloadTX emontx;

void setup() {
  rf12_initialize(myNodeID,freq,network);   //Initialize RFM12 with
settings defined above
Serial.begin(9600);
Serial.println("RFM12B Transmitter - Simple demo");


 Serial.print("Node: ");
 Serial.print(myNodeID);
 Serial.print(" Freq: ");
 if (freq == RF12_433MHZ) Serial.print("433Mhz");
 if (freq == RF12_868MHZ) Serial.print("868Mhz");
 if (freq == RF12_915MHZ) Serial.print("915Mhz");
 Serial.print(" Network: ");
 Serial.println(network);


}
```

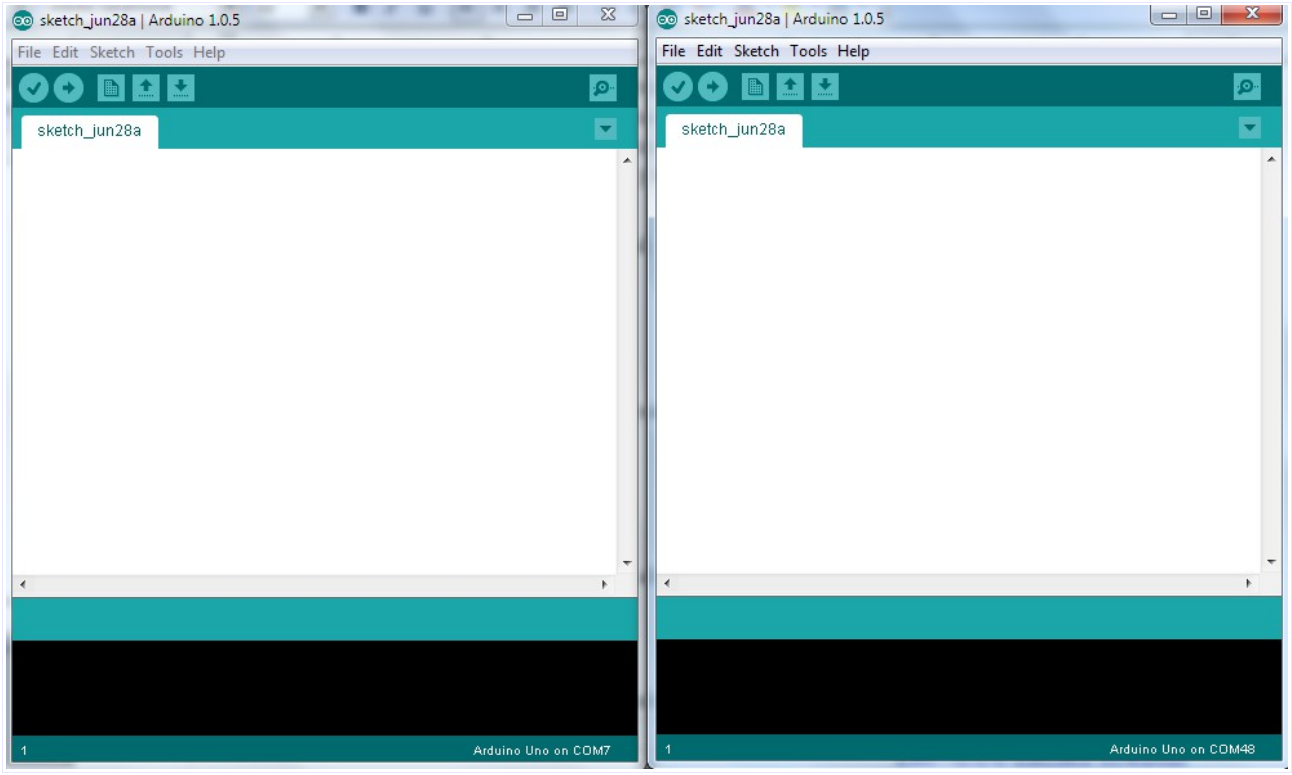
```
void loop() {
  emontx.power1=emontx.power1+1;
  emontx.power2=emontx.power2+2;
  emontx.power3=emontx.power3+3;
  emontx.battery=emontx.battery+4;

  int i = 0; while (!rf12_canSend() && i<10) {rf12_recvDone(); i+
+;}
    rf12_sendStart(0, &emontx, sizeof emontx);

  Serial.print("power1: "); Serial.println(emontx.power1);
  Serial.print("power2: "); Serial.println(emontx.power2);
  Serial.print("power3: "); Serial.println(emontx.power3);
  Serial.print("battery: "); Serial.println(emontx.battery);
  Serial.println("  ");


  delay(2000);
}
```

Cum apar datele in monitorul serial ?

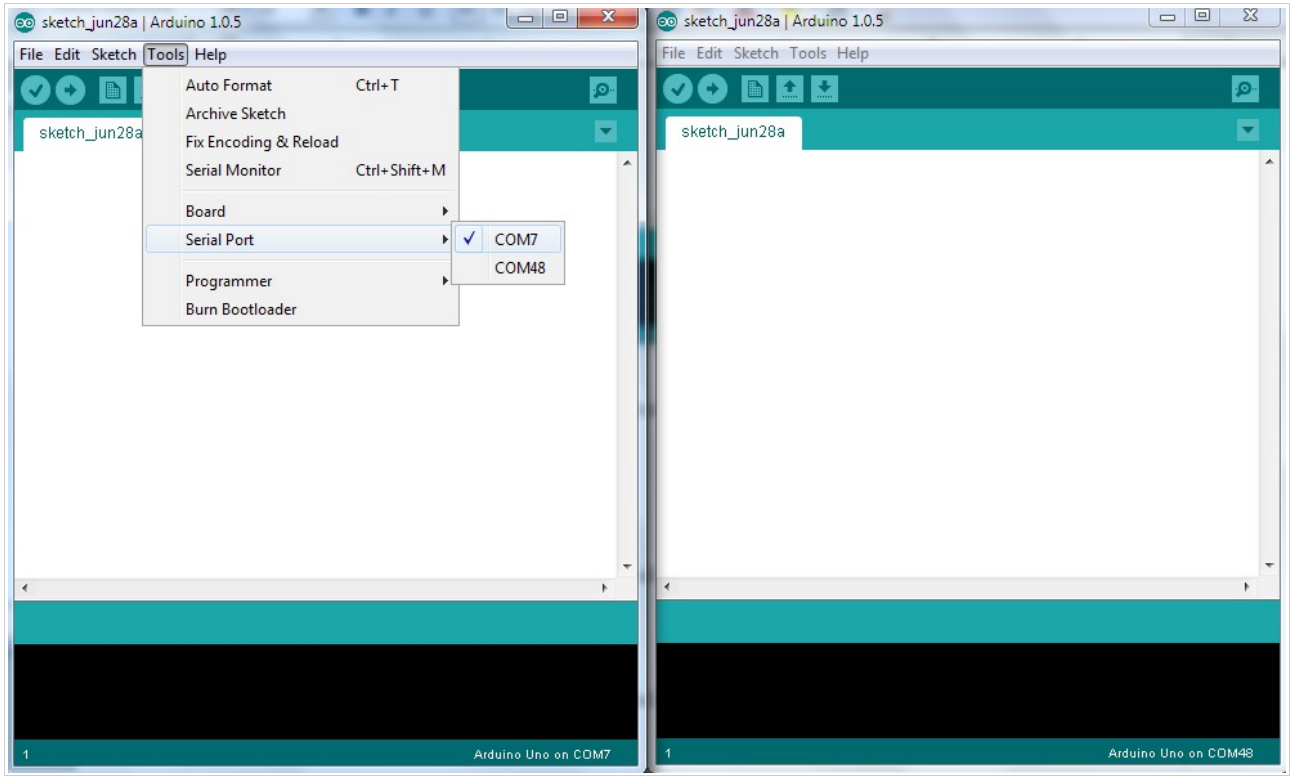
Pentru a observa daca transferul de date se realizeaza corect, adica daca ceea ce transmite emitatorul este primit si afisat de catre receptor trebuie sa urmezi pasii de mai jos.

1. Lanseaza separat 2 ferestre Arduino. Poti realiza acest lucru din butonul Start sau dand dublu click pe iconita Arduino. Nu lansa prima fereastra, iar apoi cea de-a doua folosindu-te de prima (adica din meniul File-New) deoarece, in aceasta maniera, nu vei putea selecta 2 porturi seriale diferite. Daca abordezi prima varianta, poti efectua acest lucru si poti deschide 2 Monitoare Seriale separate unul fata de celalalt.

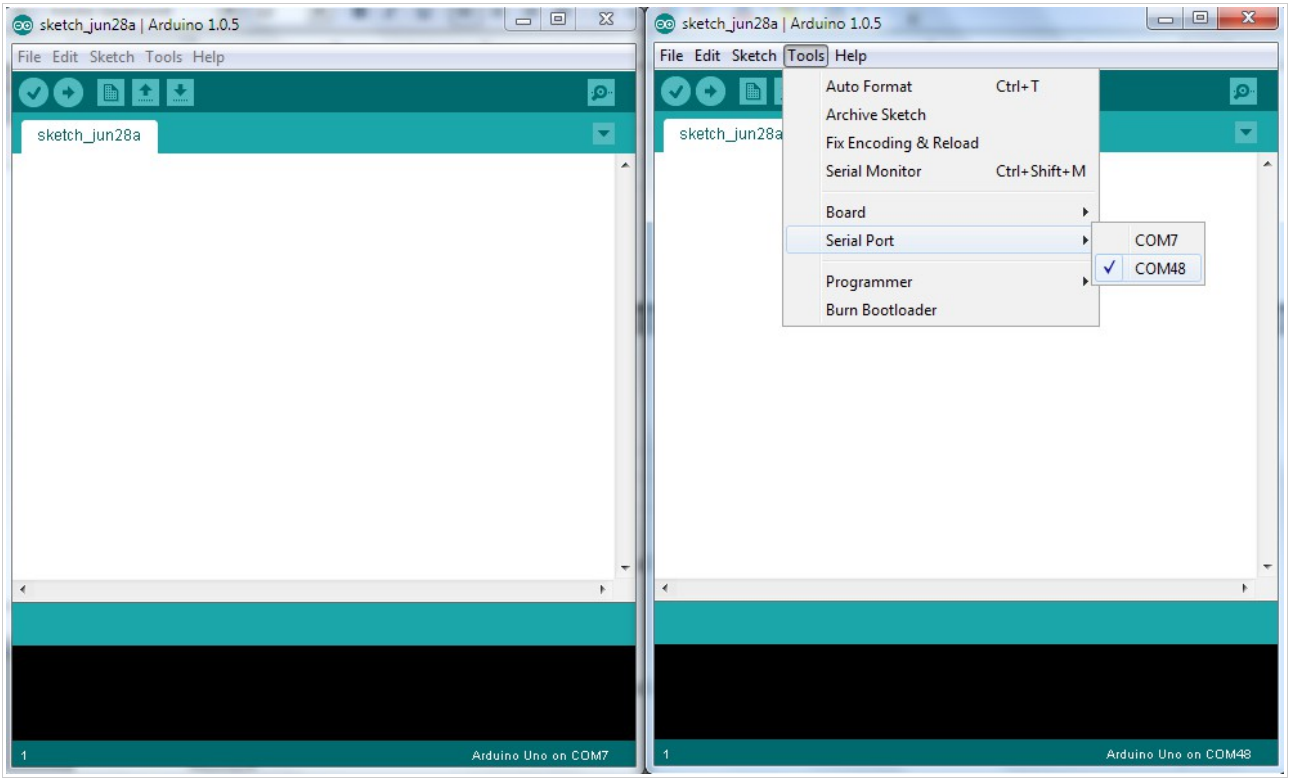
2. Imparte ferestrele pe ecran, in mod egal.



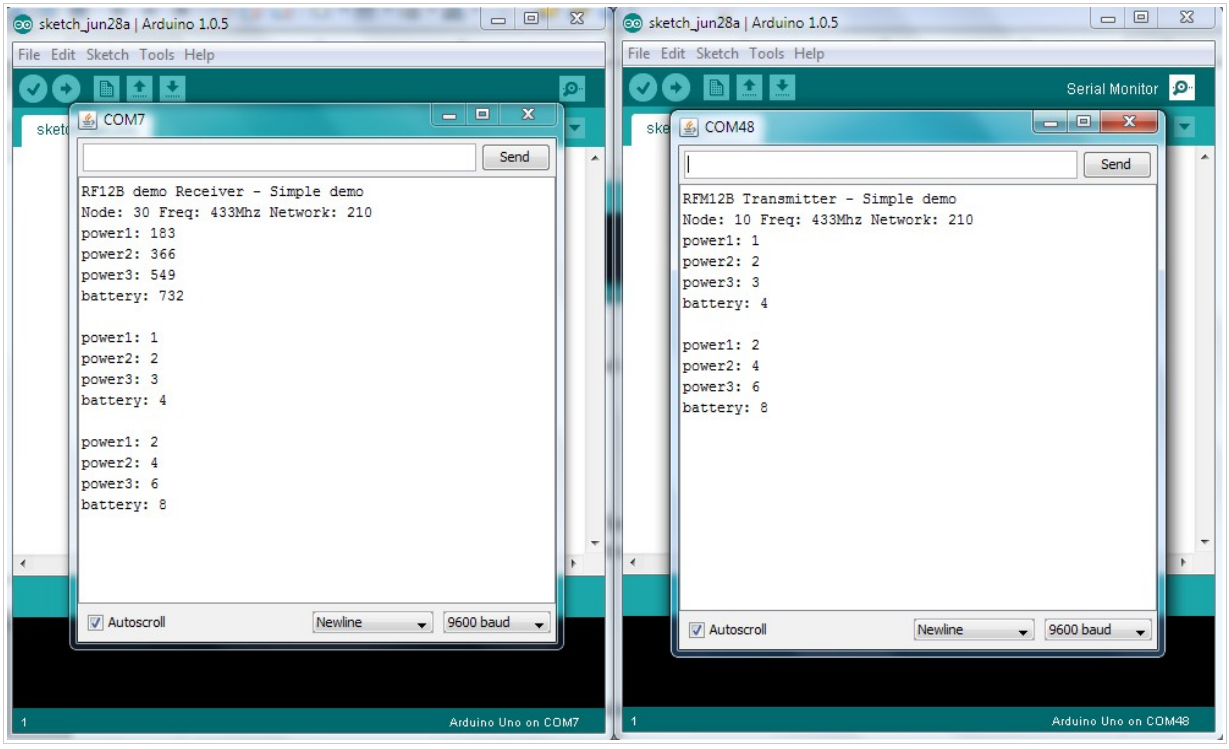
3. Pentru fereastra din stanga selecteaza primul port serial si deschide Monitorul Serial.



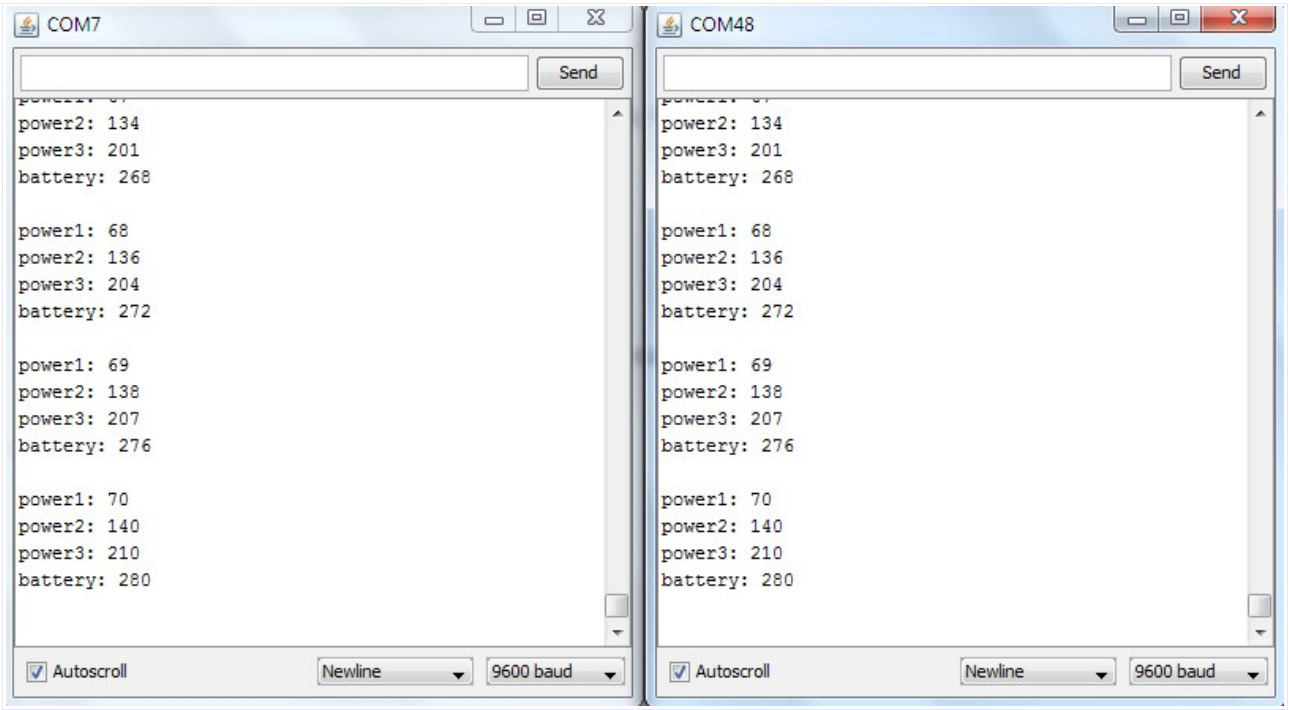
4. Pentru fereastra din dreapta selecteaza al doilea port serial si deschide Monitorul Serial.



5. Imparte cele 2 Monitoare Seriale in mod egal pe ecran.



6. Imediat ce ai deschis cele 2 Monitoare Seriale, vei observa o serie de informatii transmise de catre emitator si primite de catre receptor.



7. Informatiile sunt pur si simplu 4 variabile ce sunt incontinuu incrementate.