ROBOFUN.RO

LECȚIA XX

# CURS GRATUIT

## ARDUINO ȘI ROBOTICĂ

# Placa Relee MOD-IO2 şi Arduino

# Arduino UNO si MOD-IO2

In acest tutorial vom exemplifica functionarea modulului MOD-IO2 impreuna cu Arduino UNO. Modulul MOD-IO2 este un modul cu microcontroller dedicat, comunica cu Arduino prin protocolul I2C, si dispune de doua relee si mai multe porturi GPIO (similare cu porturile digitale din Arduino).

Vei avea nevoie de urmatoarele componente:

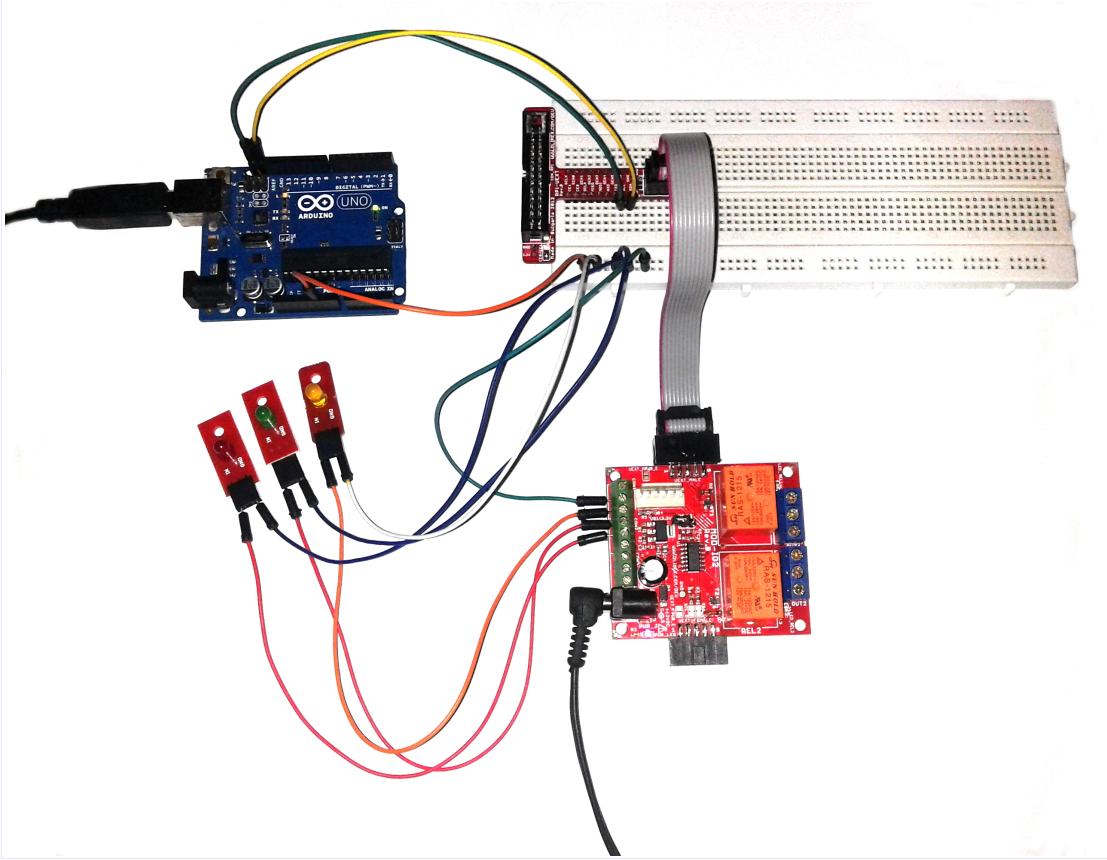
- Arduino UNO
- Placa cu relee MOD-IO2
- Adaptor Rpi-UEXT
- 3 led-uri brick de culori diferite
- Breadboard
- Fire pentru conexiuni

Cateva lucruri despre MOD-IO2 :

- contine pe placa un microcontroller dedicat care controleaza cele doua relee de pe placa, precum si mai multe porturi GPIO (similare cu porturile digitale din Arduino).
- se conecteaza cu Arduino doar prin 2 fire (SCL, SDA – protocol I2C) la Arduino.
- se alimenteaza de la o sursa de tensiune stabilizata de 12V.
- este adresabil, ceea ce inseamna ca poti inlantui mai multe module MOD-IO2.

In cele de mai jos vom utiliza un breadboard, impreuna cu adaptorul UEXT pentru a simplifica lucrurile si (astfel vom avea masa comuna GND intre led-uri, Arduino si MOD-IO2).

Pentru exemplificare, vom folosi trei led-uri brick conectate la placa MOD-IO2. Nu vom conecta nimic la relee, zgomotul pe care il fac acestea la cuplare si decuplare va fi suficient pentru a ne da seama ca acestea functioneaza. In proiectul tau, poti folosi doar releele (daca nu ai nevoie de porturi GPIO suplimentare; ceea ce inseamna ca in cele ce urmeaza esti liber sa ignori led-urile) conectate la porturile GPIO.



Tabelul de conexiuni:

| Arduino UNO SCL | Rpi-UEXT SCL |
|---|---|
| Arduino UNO SDA | Rpi-UEXT SDA |
| Led brick galben IN | MOD-IO2 GPIO 0 |
| Led brick verde IN | MOD-IO2 GPIO 1 |
| Led brick rosu IN | MOD-IO2 GPIO 2 |
| Led brick galben GND | Breadboard GND |
| Led brick verde GND | Breadboard GND |
| Led brick rosu GND | Breadboard GND |
| Arduino UNO GND | Breadboard GND |
| MOD-IO2 GND | Breadboard GND |

Pentru comunicarea I2C intre Arduino si MOD-IO2, Arduino este master iar placa MOD-IO2 functioneaza pe post de slave. Arduino transmite comenzi prin magistrala I2C la frecventa de 100Khz. Din pacate, MOD-IO2 intelege doar comenzi trimise la cel mult 50KHz, asa ca va fi necesar sa modifici frecventa placii Arduino in fisierului de configurare de mai jos.

**arduino-1.0.2/libraries/Wire/utility**

Deschide fisierul **twi.h** si cauta liniile:

```
#ifndef TWI_FREQ
#define TWI_FREQ 100000L
#endif
```

Schimba valoarea 100000L in 50000L.

Cand vrei sa abordezi alte proiecte, nu uita sa schimbi la loc valoarea.

## Programul.

Incarca in Arduino urmatorul sketch.

```
#include <Wire.h>
String inputString = "";
boolean stringComplete = false;
int led = 0;
int whichRelay = 0;

void setup() {
  Serial.begin(9600);
  inputString.reserve(200);
  Wire.begin();
  setareGPIO();
  setGPIOState(0,0);
  setGPIOState(1,0);
  setGPIOState(2,0);
  setRelayState(0,0);
  setRelayState(1,0);
}

void loop() {

 if (stringComplete) {
    if (inputString == "led 1 on\n") {
     setGPIOState(0,1);
     Serial.println("GPIO Led 1 ON");
    } else if (inputString == "led 1 off\n") {
       setGPIOState(0,0);
       Serial.println("GPIO Led 1 OFF");
    } else if (inputString == "led 2 on\n") {
       setGPIOState(1,1);
       Serial.println("GPIO Led 2 ON");
    } else if (inputString == "led 2 off\n") {
       setGPIOState(1,0);
       Serial.println("GPIO Led 2 OFF");
    } else if (inputString == "led 3 on\n") {
       setGPIOState(2,1);
       Serial.println("GPIO Led 3 ON");
    } else if (inputString == "led 3 off\n") {
       setGPIOState(2,0);
       Serial.println("GPIO Led 3 OFF");
    } else if (inputString == "relay 1 on\n") {
       setRelayState(0,1);
       Serial.println("RELAY 1 ON");
    } else if (inputString == "relay 1 off\n") {
       setRelayState(0,0);
       Serial.println("RELAY 1 OFF");
    } else if (inputString == "relay 2 on\n") {
       setRelayState(1,1);
       Serial.println("RELAY 2 ON");
```

```
    } else if (inputString == "relay 2 off\n") {
       setRelayState(1,0);
       Serial.println("RELAY 2 OFF");
    } else if (inputString == "relays on\n") {
       setRelayState(0,1);
       setRelayState(1,1);
       Serial.println("BOTH RELAYS ON");
    } else if (inputString == "relays off\n") {
       setRelayState(0,0);
       setRelayState(1,0);
       Serial.println("BOTH RELAYS OFF");
    }
    inputString = "";
    stringComplete = false;
  }
}

void serialEvent() {
  while (Serial.available()) {
    char inChar = (char)Serial.read();
    inputString += inChar;
    if (inChar == '\n') {
      stringComplete = true;
    }
  }
}


void setareGPIO() {
 Wire.beginTransmission(0x48);
 Wire.write(0x02);
 Wire.write(0xA0);
 Wire.write(0x01);
 Wire.write(0x00);
 Wire.endTransmission();
}


/* Exemple de situatii
 setGPIOState(0,1) – led-ul conectat la GPIO 0 devine ON; restul
led-urilor nu isi modifica starea
 setGPIOState(1,1) – led-ul conectat la GPIO 1 devine ON
 setGPIOState(0,0) – led-ul conectat la GPIO 0 devine OFF
 setGPIOState(1,0) – led-ul conectat la GPIO 1 devine OFF
 setGPIOState(2,1) – led-ul conectat la GPIO 2 devine ON
 setGPIOState(2,0) – led-ul conectat la GPIO 2 devine OFF
*/
```

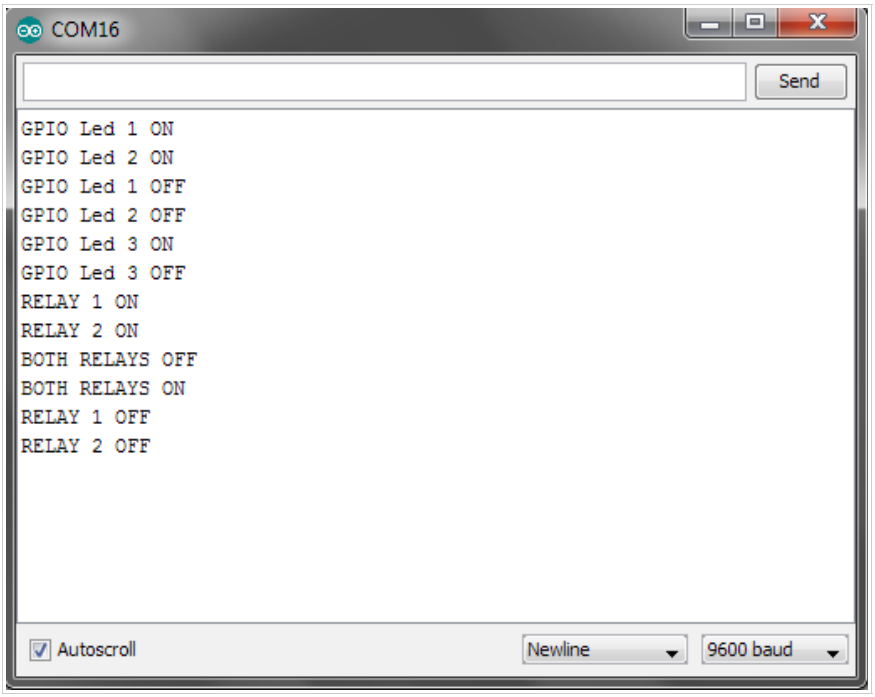
```
void setGPIOState(int gpio, int state) {
 if (state == 1) bitSet(led, gpio);
 if (state == 0) bitClear(led, gpio);
 Wire.beginTransmission(0x48);
 Wire.write(0x02);
 Wire.write(0xA0);
 Wire.write(0x02);
 Wire.write(led);
 Wire.endTransmission();
}


/* Exemple de situatii:
setRelayState(0,1) – releu 1 cuplat; releu 2 decuplat
setRelayState(1,1) – releu 2 cuplat; releu 1 ramane cuplat(din
instructiunea anterioara)
setRelayState(0,0) – releu 1 decuplat; releu 2 ramane cuplat(din
instructiunea anterioara)
setRelayState(1,0) – releu 2 decuplat; releu 1 ramane decuplat(din
instructiunea anterioara)
*/


void setRelayState(int relay, int state) {
 if (state == 1) bitSet(whichRelay, relay);
 if (state == 0) bitClear(whichRelay, relay);
 Wire.beginTransmission(0x48);
 Wire.write(0x02);
 Wire.write(0xA0);
 Wire.write(0x40);
 Wire.write(whichRelay);
 Wire.endTransmission();
}
```

## Cum functioneaza?

Dupa ce ai incarcat sketch-ul in Arduino, deschide monitorul serial si tasteaza in casuta alaturi de buton Send urmatoarele comenzi:

- led 1
- led 2
- led 3

- relay 1
- relay 2
- relay on
- relay off




Vei observa ca led-urile si releele isi schimba starea in functie de comanda utilizata (pentru led-uri, vei sesiza vizual modificarea starii, iar pentru relee vei auzi click-ul pe care il fac cand isi modifica starea).

## Ce se intampla in program ?

Functia **setup()** initializeaza portul serial la 9600 baud, interfata I2C la frecventa de 50Khz si aduce modulul MOD-IO2 in starea in care led-urile si releele sunt oprite, prin cele 3 functii: **setareGPIO, setLedGPIO, setRelay.** Se aloca din memorie 200 de caractere pentru variabila **inputString.**

In bucla **loop()** se testeaza daca **stringComplete** este true. Aceasta variabila este legata de functia **serialEvent(),** ea fiind executata automat de

Arduino, adica nu trebuie sa o apelezi undeva in program.

In functia **serialEvent()** se receptioneaza caracterele de la monitorul serial si se stocheaza in **inputString.** Tot aici se cauta daca s-a tastat Enter ( echivalentul lui '\n' sau new line). Daca s-a tastat inseamna ca s-a introdus comanda completa.

```
if (inChar == '\n') {
      stringComplete = true;
    }
```

Cand s-a receptionat o comanda (un Enter), **stringComplete** devine *true*.

Revenind in bucla loop(), daca variabila este true atunci se cauta comanda (se testeaza inputString de mai multe ori) iar pentru fiecare comanda se apeleaza o functie (fie ea setGPIOState sau setRelayState).

Functia **setareGPIO()** seteaza portul GPIO al placii MOD-IO, ca iesire (output). Se comporta exact ca porturile lui Arduino (INPUT/OUTPUT).

Functia **setGPIOState(int gpio, int state)** primeste doua argumente si anume portul GPIO si starea in care va comuta. In interiorul functiei, Arduino ii transmite placii MOD-IO, comanda de a aprinde led-ul conectat la portul GPIO.

Asemanator functioneaza si functia **setRelayState(int relay, int state).** Primeste doua argumente: relay si state respectiv „care dintre relee“ si in ce stare va comuta.

Detaliile despre placa MOD-IO2 si anume, setul de comenzi, formatul adreselor, etc sunt explicate in manualul acesteia.