

Traian Anghel

PROGRAMAREA PLĂCII ARDUINO

Ediția a II-a

Editura Paralela 45

*Editura Paralela 45 este recunoscută de
Consiliul Național al Cercetării Științifice (CNCS)*

Redactare: Bianca Vișan
Tehnoredactare & pregătire de tipar: Marius Badea
Design copertă: Ionuț Broșțianu

Descrierea CIP a Bibliotecii Naționale a României
ANGHEL, TRAIAN

Programarea plăcii Arduino / Traian Anghel. - Ed. a 2-a. - Pitești :
Paralela 45, 2020
Conține bibliografie
ISBN 978-973-47-3204-3

004

Cuprins

Introducere	9
Capitolul 1. Noțiuni de electricitate	14
1.1. Structura corpurilor	14
1.2. Sarcina electrică	18
1.3. Conductorii, izolatorii, semiconductori.....	20
1.4. Legea lui Coulomb. Câmpul electric.	23
1.5. Potențialul electric. Tensiunea electrică.....	27
1.6. Condensatorul. Capacitatea electrică	29
1.7. Curentul electric. Circuitul electric.....	31
1.8. Intensitatea curentului electric	33
1.9. Rezistența electrică. Rezistivitatea.....	36
1.10. Surse electrice. Tensiunea electromotoare	38
1.11. Legile circuitelor electrice	40
1.12. Energia și puterea electrică	44
1.13. Potentiometrul.....	47
1.14. Fotorezistorul	47
1.15. Sistemul Internațional. Multipli și submultipli	50
Capitolul 2. Noțiuni de programare.....	52
2.1. Sisteme de numerație	52
2.2. Algoritmi.....	55
2.3. Limbajul pseudocod.....	57
2.4. Programarea structurată	61
2.5. Introducere în limbajul C++.....	66
2.6. Instrucțiunile limbajului C++.....	73
2.7. Funcții în C++	79
2.8. Programarea orientată pe obiect în C++.....	83
Capitolul 3. Prezentarea plăcii și a mediului de dezvoltare Arduino	94
3.1. Placa de dezvoltare Arduino	94
3.2. Instalarea IDE Arduino Software.....	100

3.3. Conectarea plăcii Arduino la PC.....	101
3.4. Biblioteci Arduino.....	102
3.5. Software suplimentar	105
3.6. Bootloader-ul Arduino	106
Capitolul 4. Bazele programării plăcii Arduino	109
4.1. Structura unui program Arduino	109
4.2. Compilarea și încărcarea sketch-ului pe placa Arduino	111
4.3. Primul sketch Arduino	112
4.4. Breadboard, fire de legătură și LED-uri.....	115
4.5. Intrări și ieșiri digitale	118
4.6. Generarea semnalelor PWM	123
4.7. Utilizarea întreruperilor.....	125
4.8. Intrări analogice	128
4.9. Afișarea datelor folosind LCD-uri	137
4.10. Generarea sunetelor.....	143
Capitolul 5. Interfețe de comunicație	146
5.1. Porturi analogice și digitale.....	146
5.2. Interfața serială.....	147
5.3. Interfața SPI	150
5.4. Interfața I ² C.....	152
Capitolul 6. Aplicații bazate pe senzori.....	156
6.1. Măsurarea temperaturii și umidității	156
6.2. Măsurarea nivelului de iluminare.....	159
6.3. Măsurarea distanței	162
6.4. Măsurarea nivelului de fum	165
6.5. Măsurarea forței	167
6.6. Măsurarea inducției magnetice	169
6.7. Detectarea mișcării.....	178
6.8. Detectarea înclinării	181
6.9. Detectarea ploii	183
6.10. Măsurarea presiunii atmosferice	186
6.11. Detectarea sunetului.....	189

Capitolul 7. Comunicația prin Bluetooth	193
7.1. Standardul Bluetooth.....	193
7.2. Conectarea modului HC-06 la placa Arduino UNO R3.....	195
7.3. Proiect: comunicarea între Arduino și calculator prin Bluetooth	195
7.4. Proiect: comunicarea între Arduino și telefon prin Bluetooth ...	201
7.5. Proiect: telefonul, telecomandă Bluetooth pentru Arduino.....	204
Capitolul 8. Comunicația în rețele Ethernet și în Internet	205
8.1. Standardul Ethernet.....	205
8.2. Internet și Web: organizare și accesare	211
8.3. Shield Ethernet.....	219
8.4. Conectarea și testarea shield-ului Ethernet	220
8.5. Client Web cu Arduino	226
8.6. Server Web cu IP dinamic	228
8.7. Proiect: server Web pentru monitorizarea camerei.....	230
8.8. Proiect: postarea unui tweet folosind Arduino.....	234
8.9. Proiect: salvarea datelor măsurate folosind PHP și MySQL.....	237
8.10. Proiect: server Web care folosește un card micro-SD.....	250
Capitolul 9. Comunicația în rețele Wi-Fi și în Internet	256
9.1. Rețele Wi-Fi.....	256
9.2. Conectarea plăcii Arduino la o rețea Wi-Fi	258
9.3. Scurtă prezentare a modului ESP8266.....	258
9.4. Prezentarea plăcii NodeMCU	260
9.5. Proiect: client Web cu NodeMCU	261
9.6. Proiect: server Web cu NodeMCU	264
9.7. Proiect: monitorizarea nivelului de iluminare cu NodeMCU	269
9.8. Proiect: stație meteo cu NodeMCU.....	272
9.9. Instalarea firmware-ului LUA pe NodeMCU	275
9.10. Proiect: server Web cu NodeMCU și LUA Script.....	279
Capitolul 10. Comunicația prin GSM.....	282
10.1. Standardul GSM.....	282
10.2. Shield GSM.....	282
10.3. Proiect: sistem de alarmare prin GSM	283

Capitolul 11. Controlul motoarelor	291
11.1. Motoare de curent continuu	291
11.2. Proiect: comanda a două motoare de curent continuu.....	294
11.3. Proiect: modificarea dinamică a vitezei motoarelor DC	297
11.4. Motoare pas cu pas.....	299
11.5. Proiect: controlul rotației unui motor pas cu pas.....	300
11.6. Servomotoare	302
11.7. Proiect: controlul unui servomotor.....	304
Capitolul 12. Programarea plăcii Arduino cu LabVIEW	307
12.1. Introducere în LabVIEW	307
12.2. Cum se utilizează LabVIEW cu Arduino.....	308
12.3. Analiza unui instrument virtual.....	310
12.4. Proiect: măsurarea temperaturii și a iluminării	314
Capitolul 13. Internetul lucrurilor	318
13.1. Avantaje și dezavantaje ale utilizării IoT.....	318
13.2. Servicii IoT	320
13.3. Proiect: monitorizarea temperaturii cu ThingSpeak.....	321
13.4. Proiect: monitorizarea nivelului de iluminare cu Ubidots	335
13.5. Proiect: monitorizarea temperaturii cu NodeMCU și Ubidots.	343
13.6. Proiect: utilizarea platformei Thethings.....	345
Bibliografie.....	348
Cărți.....	348
Web.....	349

Introducere

„Lucrul cel mai important este să nu încetezi să pui întrebări.
Curiozitatea are propria rațiune de a exista.”

Albert Einstein

Astăzi, singura constantă este schimbarea, lumea transformându-se într-un ritm nemaîntâlnit. În zilele noastre, suntem martorii începutului celei de-a patra revoluții industriale¹, care constă în pătrunderea rapidă a tehnologiilor digitale în toate domeniile. „Viitorul este aici, doar că nu este uniform distribuit”, spunea cândva William Gibson, scriitor canadian de origine americană. La fel se întâmplă și acum, când noua revoluție industrială se manifestă deja în zone diverse ale activității omului, cuprinzând treptat un număr mare de sectoare ale activității economice, sociale, dar și personale a acestuia.

Tehnologia avansată pătrunde în toate domeniile, inclusiv în cele care până nu de mult păreau a fi „ferite” de aceasta. Automatizarea casei (*home automation*) este un concept care se dezvoltă extrem de rapid. De exemplu, se pot controla luminile din casă, precum și sistemele de încălzire, de aer condiționat și de securitate ale acesteia. De asemenea, se pot utiliza diverse *gadgeturi* în apropierea casei (cu care, de exemplu, sunt controlate temperatura și umiditatea aerului din sera de flori sau de legume) sau în drumul către casă (e.g. se pornește mașina de spălat).

În următorul deceniu, mai mult de un trilion de senzori vor fi conectați la Internet, conform afirmațiilor făcute la Forumul Economic Mondial de la Davos, 2016. Aceștia vor trimite diverse informații omului, dar vor comunica și între ei, scopul acestei uriașe „desfășurări de forțe” fiind de a oferi omului un trai mai bun, mai confortabil și mai sigur.

¹ Această idee a fost avansată la cea de-a 46-a ediție a Forumului Economic Mondial, desfășurată la Davos (Elveția), în perioada 20-23 ianuarie 2016.

Conform *Business Insider* (www.businessinsider.com/), în 2015 erau conectate la Internet circa 15 miliarde de dispozitive (dintre care numai jumătate erau calculatoare, tablete și telefoane mobile), iar în 2020 vor fi conectate 34 de miliarde. O parte dintre acestea sunt ceasuri inteligente, frigider, brățări de fitness, camere de supraveghere, televizoare *smart*, dar și o mulțime de dispozitive construite chiar de către utilizatori. Acea parte a Internetului la care sunt conectate dispozitivele enumerate este cunoscută sub numele de *Internetul lucrurilor* (*Internet of Things*, IoT).

Una dintre caracteristicile definitorii ale tehnologiei actuale constă în aceea că este accesibilă practic oricărei persoane. O altă caracteristică importantă este adaptabilitatea ei la necesitățile personale. Din acest motiv, este posibilă construirea unor dispozitive electronice în propria casă (garaj²) sau în laboratorul de fizică al școlii³. Astfel, bazându-se pe cunoștințele și deprinderile obținute în gimnaziu și liceu, dar și pe cele dobândite din diverse surse aflate la îndemână, oricine are posibilitatea de a construi dispozitive pe care le poate utiliza în cele mai variate scopuri.

Este adevărat că multe dintre dispozitivele de care avem nevoie la un moment dat pot fi cumpărate. Dar ne putem imagina unele extrem de personalizate pe care ori nu le putem găsi în comerț, ori – dacă le găsim sau le putem obține la comandă – sunt destul de scumpe. Iată argumente care vin în sprijinul ideii de a ne crea propriul „laborator” în care să realizăm dispozitivele care ne sunt necesare sau măcar o parte dintre ele.

Deoarece aceste dispozitive electronice îndeplinesc sarcini repetitive, ele trebuie să funcționeze pe baza unor programe. De asemenea, dacă se dorește schimbarea sarcinilor îndeplinite sau cel puțin a unor parametri ai acestora, programele pe care se bazează funcționarea lor trebuie înlocuite sau cel puțin modificate, ceea ce înseamnă că dispozitivele trebuie să fie programabile. Iar dacă programarea lor poate fi realizată de către utilizator, atunci dispozitivele pot fi adaptate sau modificate în funcție de nevoile acestuia. Este situația cea mai convenabilă pentru utilizatorii interesați de crearea propriilor dispozitive electronice.

² Steve Jobs a început să asambleze calculatoare, la mijlocul anilor 1970, împreună cu Steve Wozniak, în garajul casei părintești. Ambii sunt cofondatori ai companiei Apple, înființată în anul 1976.

³ Construirea unor dispozitive chiar de către utilizatori este facilitată în cea mai mare măsură de structura modulară a acestora. Adesea, modulele componente pot fi tratate drept cutii negre, fiind considerate importante pentru utilizatori numai operațiile pe care acestea le efectuează asupra datelor de intrare.

Capitolul 1. Noțiuni de electricitate

„Studiază mai întâi știința și continuă apoi cu practica născută din această știință!”

Leonardo da Vinci

În primul capitol sunt prezentate succint o serie de noțiuni de electricitate necesare pentru buna înțelegere a lucrării, în special a exemplelor practice oferite în cadrul acesteia. Precizăm că noțiunile prezentate în capitolul de față se regăsesc între cele incluse în programele de fizică pentru gimnaziu și de liceu, fiind astfel accesibile tuturor persoanelor interesate de buna înțelegere a lucrării.

1.1. Structura corpurilor

Eforturile depuse de savanți de-a lungul secolelor – în special fizicieni și chimiști – au permis obținerea unei imagini bogate și complexe a realității înconjurătoare, în particular a structurii corpurilor.

Astăzi, se știe că toate corpurile, indiferent de starea de agregare a acestora, sunt compuse din atomi și molecule. Moleculele sunt structuri alcătuite din atomi uniți prin câteva tipuri de legături chimice (e.g. covalentă, ionică). Un tip special de legătură este cea metalică.

După încercarea de pionierat¹ din 1904 a lui J.J. Thomson (care a elaborat un model simplificat al atomului, supranumit „cozonacul cu stafide”), structura atomului a fost dezvăluită în 1911 de către fizicianul Ernest Rutherford (1871-1937), laureat al premiului Nobel pentru chimie (1908), considerat „părintele” fizicii nucleare și cel mai mare experimentator după Michael Faraday. Rutherford a formulat, ca urmare a celebrelor sale experiențe de împrăștiere a particulelor alfa² pe foițe metalice subțiri,

¹ Menționăm că primul model atomic a fost cel al lui Dalton (1766-1844). Conform acestuia, atomul este sferic și fără structură internă, atomii aceluiași element fiind identici, dar diferiți de atomii altor elemente.

² Particulele alfa (α) sunt nuclee de heliu (i.e. atomi de heliu care au cedat prin ionizare ambii electroni), cu o sarcină pozitivă de două ori mai mare decât cea a electronului și cu o masă de circa 7300 de ori mai mare decât masa unui electron.

modelul planetar al atomului (figura 1). În cadrul acestui model, atomul este similar cu sistemul planetar, în cadrul căruia – după cum se știe – planetele (inclusiv Pământul) se rotesc pe orbite aproximativ circulare în jurul Soarelui.

În cadrul modelului elaborat de Rutherford, atomul (având raza de circa $10^{-10} \div 10^{-9}$ m) este alcătuit dintr-un înveliș electronic și un nucleu (având raza de circa $10^{-15} \div 10^{-14}$ m), în jurul căruia se rotesc pe orbite circulare electronii. În modelul planetar al atomului a fost introdusă prima oară în fizică noțiunea de nucleu atomic.

Electronul (cu simbolul e^-) este o particulă elementară, cu sarcina electrică negativă și egală în modul cu sarcina electrică elementară (pentru detalii, a se citi secțiunea următoare), având masa foarte mică³:

$$e = -1,6 \cdot 10^{-19} \text{ C} \quad m = 9,1 \cdot 10^{-31} \text{ kg} \quad (1)$$

Electronii unui atom formează învelișul electronic al acestuia. Numărul electronilor unui atom se numește număr atomic și se notează cu Z .

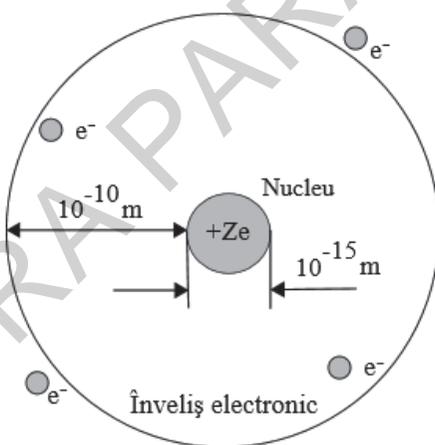


Figura 1. Modelul planetar al atomului

Nucleul atomic, conținând aproape întreaga masă a atomului⁴, este încărcat pozitiv, fiind alcătuit din protoni⁵ (Rutherford, 1920) și neutroni

³ Masa unui electron este de circa 1840 de ori mai mică decât masa nucleului celui mai ușor atom din natură (atomul de hidrogen).

Capitolul 3. Prezentarea plăcii și a mediului de dezvoltare Arduino

„Fără o mare perseverență nu există talente, nici genii.”

Dmitri Mendeleev

Pentru a programa plăcile de dezvoltare din familia Arduino este necesar ca pe calculator să se instaleze mediul de dezvoltare Arduino IDE (*Integrated Development Environment*), dar și o serie de biblioteci, precum și software suplimentar. Arduino IDE are versiuni pentru platformele software uzuale (Windows, Linux, Mac OS). Înainte de analiza procedurii de instalare a software-ului, este realizată o prezentare a familiei de plăci Arduino, în special a versiunii Arduino UNO¹ R3 (Revision 3), utilizată cu precădere în lucrarea de față.

3.1. Placa de dezvoltare Arduino

Arduino este o familie de plăci de dezvoltare (*development boards*) cu microcontroler, comercială, accesibilă ca preț, care poate fi programată folosind un limbaj de programare similar cu C/C++, cu ajutorul mediului de dezvoltare integrat Arduino IDE.

Arduino a fost dezvoltată începând cu anul 2005, sub forma unui proiect al lui Massimo Banzi, student italian în Ivrea, Italia. Din echipa inițială au mai făcut parte încă patru persoane. Numele Arduino provine de la un bar în care aceștia obișnuiau să se întâlnească.

Plăcile de dezvoltare Arduino pot fi echipate cu un număr impresionant de senzori (e.g. de temperatură, umiditate, presiune, forță, distanță, nivel de iluminare), actuatori (e.g. motoare de curent continuu, motoare pas cu pas sau servomotoare) și plăci de extensie (cunoscute sub denumirea de *shields*), cu scopul de a construi dispozitive (pe care le vom numi în cele ce urmează proiecte) pentru obținerea informațiilor din mediul extern, pentru controlul acestuia sau pentru comunicarea cu alte echipamente, folosind diverse medii de comunicație (Bluetooth, Ethernet, WI-FI, GSM,

¹ Numele UNO a fost dat plăcii pentru a celebra prima versiune a Arduino IDE (1.0), care – până atunci – era în versiune Beta.

GPS). Pentru afișarea informațiilor primite din mediul extern sau a altor tipuri de informații, plăcile din familia Arduino folosesc ecrane LCD (*Liquid-Crystal Display*), de la cele mai simple (pe care se pot afișa informații text), până la cele grafice.

Se poate spune – fără teama de a greși – că limitele care pot să apară în dezvoltarea proiectelor cu plăcile din familia Arduino sunt determinate numai de imaginația celor care le folosesc, acestea putând fi utilizate de profesioniști care activează într-un număr mare de domenii, inclusiv de artiști, dar și de către oricine altcineva, în scop personal sau ca hobby.

Arduino este un echipament cu sursă deschisă (*open-source hardware*). Din acest motiv, designul plăcii, schemele și codul sursă sunt disponibile în mod gratuit oricui dorește să le folosească. Printre altele, aceasta înseamnă că, alături de producătorul italian al plăcilor Arduino, există numeroși alți fabricanți de clone (ale căror prețuri sunt adesea mai avantajoase), dar și de variante îmbunătățite. Din punct de vedere legal, numai plăcile fabricate de firma mamă din Italia pot purta numele Arduino. Alegerea între placa originală și o clonă este determinată în primul rând de bugetul de care dispune cel care dezvoltă proiecte cu Arduino. Pentru dezvoltarea proiectelor incluse în lucrarea de față au fost utilizate atât plăci originale Arduino, cât și clone, fără a observa diferențe semnificative între performanțele și funcționarea acestora.

Arduino este fabricată în două formate: formatul de dezvoltare (folosit în lucrarea de față) și formatul de producție. Plăcile realizate în cele două formate sunt total compatibile din punct de vedere funcțional, ele fiind deosebite numai în ceea ce privește dimensiunile, plăcile de producție având dimensiuni mai mici decât cele de dezvoltare. De asemenea, plăcile de producție nu conțin modulul de comunicație USB și sunt mai ieftine. În general, proiectele sunt create pe o placă de dezvoltare, fiind transferate după finalizare pe o placă de producție (e.g. Arduino Nano, Arduino Mini).

„Creierul” unei plăci Arduino este microcontrolerul (MCU, *microcontroller unit*). Cele mai multe plăci Arduino, inclusiv versiunea UNO, folosesc un microcontroler AVR ATmega², dezvoltat de compania Atmel. Acesta este bazat pe o arhitectură Harvard RISC pe opt biți, modificată. De exemplu, Arduino UNO, dar și Arduino Nano folosesc microcontrolerul

² Arduino Due folosește un microcontroler ARM Cortex, pe 32 de biți.

Capitolul 4. Bazele programării plăcii Arduino

„Mai multe lucruri a dus la capăt iscusința decât forța.”

Baltasar Gracián

Interacțiunea dintre Arduino și dispozitivele conectate la pinii acesteia este comandată prin intermediul unui program încărcat pe placă. În capitolul de față sunt abordate subiecte care se pot constitui ca punct de plecare în studiul programării plăcii Arduino (e.g. structura unui program, intrări/ieșiri digitale, intrări analogice, generarea semnalelor PWM, utilizarea întreruperilor, afișarea datelor).

4.1. Structura unui program Arduino

Un program scris pentru placa Arduino poartă denumirea de *sketch* și are extensia *.ino*. După lansarea Arduino IDE, este creat automat un *sketch* „gol”, al cărui nume implicit este *sketch_MonthDayChar*, unde *Month* este luna curentă, *Day* este ziua curentă și *Char* este o literă (e.g. *sketch_mar20a*). Oricare *sketch* are următoarea structură:

```
void setup() {  
    // put your setup code here, to run once  
}  
void loop() {  
    // put your main code here, to run repeatedly  
}
```

În programul anterior au fost păstrate comentariile în limba engleză, așa cum apar ele în *sketch*-ul creat automat la pornirea Arduino IDE. Iată și traducerea în limba română:

```
void setup() {  
    /* pune aici codul de initializare, pentru a rula  
       o singura dată: */  
}  
void loop() {  
    /* pune aici codul principal, pentru a rula în  
       mod repetat: */  
}
```

După cum se poate observa, spre deosebire de un program scris în limbajul C/C++ standard, un *sketch* care rulează pe o placă Arduino are două secțiuni sau blocuri standard, obligatorii: *setup* și *loop*.

În secțiunea *setup* este realizată inițializarea conținutului variabilelor și este configurat comportamentul porturilor plăcii. Codul inclus în această secțiune va fi executat o singură dată (la alimentarea plăcii sau la apăsarea butonului *Reset* al acesteia). În schimb, secțiunea *loop* include partea principală a codului, care va fi executată repetitiv (acesta este motivul pentru care secțiunea poartă numele *loop*, adică buclă). Cele două secțiuni sunt implementate sub forma a două rutine (funcții), `setup()` și `loop()`, ambele fiind de tipul `void` (i.e. nu întorc nimic).

Dacă *sketch*-ul folosește variabile partajate între secțiunile *setup* și *loop*, acestea trebuie definite în afara lor, deoarece sunt variabile globale. Definierea lor se va face într-o a treia secțiune, situată înaintea celor două secțiuni standard. De asemenea, în cadrul acesteia se pot folosi directive standard de preprocesor (e.g. `#define`, `#include`, `#if`, `#ifdef`, `#ifndef`, `#endif`, `#else`).

Directiva de preprocesor `#include` face posibilă includerea unor fișiere și este utilizată astfel:

```
#include "numeFișier"
```

sau

```
#include <numeFișier>
```

O linie `#include` este înlocuită cu conținutul integral al fișierului specificat. Un fișier inclus poate conține la rândul său alte directive `#include`.

Folosind directiva de preprocesor `#define` se definesc macrouri, care, ulterior, pot fi apelate când se dorește. Un macro constituie o modalitate de prelucrare a textelor bazată pe substituție, ceea ce înseamnă că în definiția unui macro se specifică textul care urmează a se substitui la fiecare apel al acestuia. În particular, se recomandă definirea constantelor simbolice folosind directiva `#define` și nu ca variabile, deoarece în ultimul caz se consumă mai multă memorie din memoria SRAM.

Pentru compilarea condiționată se folosesc directivele de preprocesor `#if`, `#ifdef`, `#ifndef`, `#endif`, `#else` etc. Acest tip de compilare permite să se aleagă dintr-un program părțile care se vor compila împreună.