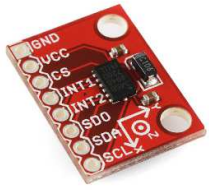
Pedometru IoT

Dispozitivele de tip pedometru (dispozitive ce contorizează numărul de pași) sunt din ce în ce mai populare. La acest lucru a condus pe de o parte scăderea prețului dispozitivelor electronice dar și necesitatea psihologică de a avea un ”martor” la eforturile personale de a ne menține în formă. Pedometrele variază ca funcționalitate de la simple dispozitive ce afișează pe un ecran numărul de pași până la dispozitive ce raportează prin protocoale fără fir către telefon sau tabletă numărul de pași permițând astfel generarea de rapoarte pe diverse intervale de timp și chiar clasamente între mai multe persoane.

În cadrul lecției de față vă propunem implementarea unui dispozitiv de tip pedometru ce raportează numărul de pași prin Internet către serviciul cloud Robofun IoT (1). Serviciul cloud Robofun IoT este gratuit și permite înregitrarea și vizualizarea datelor prin intermediul unei interfețe web, pentru mai multe detalii puteți consulta documentația oficială a serviciului (2).

Există mai multe exemple de proiecte ce își propun implementarea funcționalității de pedometru dar nu și funcționalitatea de înregistrare online a datelor. Totuși, putem parcurg câteva astfel de proiecte pentru a înțelege diverse soluții de implementare: „Arduino Pedometer” (3), „Simple, Easy and Cheap DIY Pedometer With Arduino” (4), „Arduino Pedometer Watch, With Temperature, Altitude and Compass!” (5).

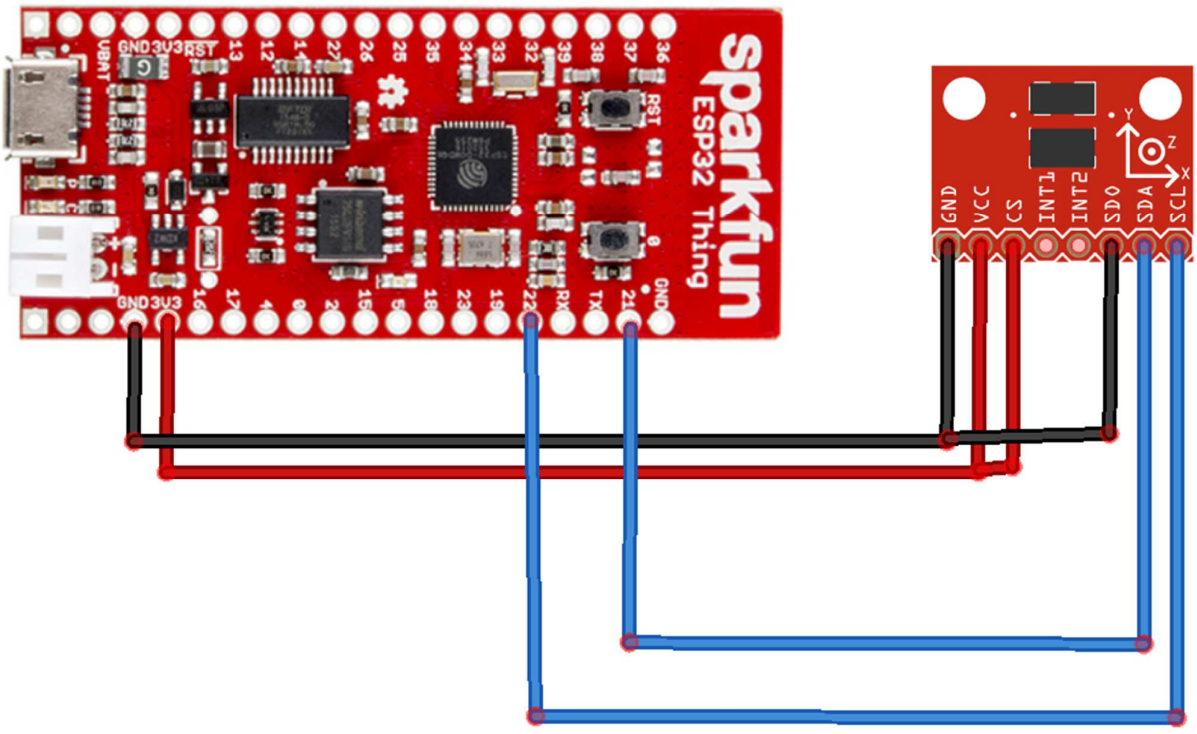


Dispozitivul prezentat va utiliza, pentru a putea sesiza fenomenul de ”pas”, un accelerometru digital ADXL345 (6). Analiza fenomenelor fizice implicate (accelerații, recunoașterea tiparului generat de efectuarea unui pas) nu fac subiectul acestei lecții dar recomandăm parcurgerea suplimentară a următorului material: „Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer” (7).

Partea de achiziție a datelor și de comunicație în rețea va fi asigurată de o placă de dezvoltare Sparkfun ESP32 Thing (8) – placă echipată cu microprocesorul ESP32 de la Espressif ce asigură conectivitate WiFi și BLE (chiar dacă dispozitivul prezentat

exemplifică doar partea de conectivitate WiFi asta nu înseamnă că acesta nu poate fi dezvoltat suplimentar pentru a putea asigura conectivitate BLE cu dispozitive mobile de tip telefon sau tabletă). Pe lângă puterea de procesare și conectivitatea WiFi placa de dezvoltare oferă avantajul de a funcționa alimentată de la un acumulator LiPo de 3.7V permițând astfel ca sistemul să fie portabil.

Schema de interconectare între placa de dezvoltare și accelerometru este următoarea (accelerometrul se va alimenta la 3.3V, pinul SDA se va conecta la pinul 21 al plăcii de dezvoltare iar SCL la pinul 22, pinul CS se va conecta la 3.3V iar SD0 la GND):



Pentru integrarea plăcii Sparkfun ESP32 Thing în mediul Arduino IDE este necesar să parcurgeți materialul „ESP32 Thing Hookup Guide” (9) deoarece sunt necesare operații manuale de copiere și instalare a utilitarelor aferente.

Programul va utiliza, pentru detecția numărului de pași, bibliotecile Pedometer (10) și Accelerometer_ADXL345 (11) a celor de la Seeed-Studio.

Programul a fost dezvoltat și testa utilizând Arduino IDE 1.8.3. În program trebuie personalizate datele de conectare la rețeaua WiFi (*networkName* și *networkPswd*).

```
#include <WiFi.h>
#include "pedometer.h"
#include <ADXL345.h>

Pedometer pedometer;

#include <Wire.h>

const char * networkName = "…";
const char * networkPswd = "…";

const char * hostDomain = "iot.robofun.ro";
const int hostPort = 80;

const int BUTTON_PIN = 0;
const int LED_PIN = 5;
```

În cadrul secțiunii *setup()* se vor realiza inițializările necesare. Placa de dezvoltare are integrate un led (pinul 5) și un buton (pinul 0). Ledul va indica (se va aprinde) activitățile de rețea din program (conectare AP, postare IoT) iar butonul va fi utilizat pentru a reseta numărul de pași efectuați.

```
void setup() {
  Serial.begin(115200);
  pinMode(BUTTON_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, HIGH);
  pedometer.init();
  connectToWiFi(networkName, networkPswd);
  digitalWrite(LED_PIN, LOW); }
```

Constanta *INTERVAL* va indica perioda de timp (în milisecunde) la care se va face transmiterea numărului de pași efectuați către serviciul IoT.

```
#define INTERVAL 60000
unsigned long lastPost = 0;

void loop()
{
  if (digitalRead(BUTTON_PIN) == LOW)
            pedometer.stepCount = 0;
  if((millis() - lastPost) > INTERVAL) {
    digitalWrite(LED_PIN, HIGH);
    if (WiFi.status() != WL_CONNECTED)
            connectToWiFi(networkName, networkPswd);
    requestURL(hostDomain, hostPort);
    digitalWrite(LED_PIN, LOW);
    lastPost = millis();
  }
  delay(10);
}
```

Procedura *connectToWiFi()* este responsabilă cu conectarea la rețeaua WiFi.

```
void connectToWiFi(const char * ssid, const char * pwd)
{ int ledState = 0;
  Serial.println("Connecting to WiFi network: " +
                          String(ssid));

  WiFi.begin(ssid, pwd);

  while (WiFi.status() != WL_CONNECTED)    {
    digitalWrite(LED_PIN, ledState);
```

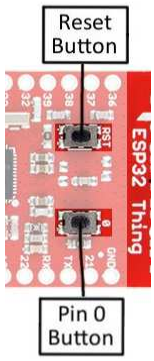
```
    ledState = (ledState + 1) % 2; // Flip ledState
    delay(500);
    Serial.print(".");
  }

  Serial.println();
  Serial.println("WiFi connected!");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
}
```

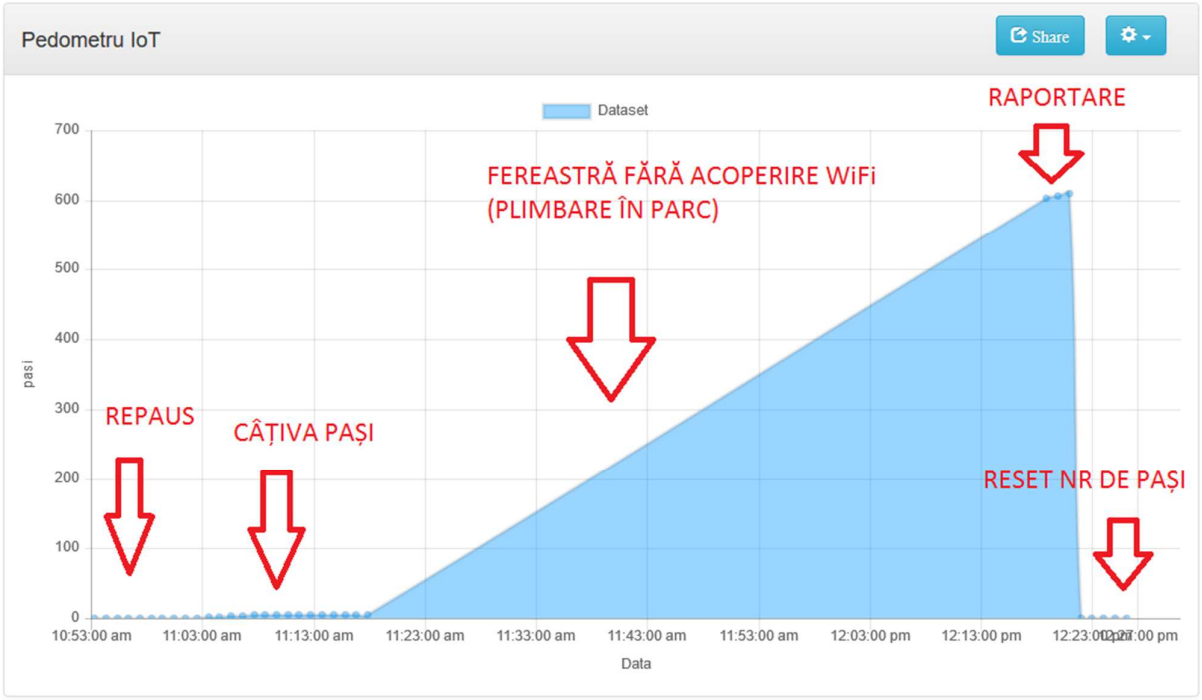
Procedura *requestURL()* este responsabilă cu postarea către serviciul IoT. În cadrul acesteia trebuie completată valoarea cheii de autentificare (*TOKEN*) obținute în urma înregistrării în cadrul serviciului Robofun IoT.

```
void requestURL(const char * host, uint8_t port)
{
  Serial.println("Connecting to domain: " + String(host));

  WiFiClient client;
  if (!client.connect(host, port))
  {
    Serial.println("connection failed");
    return;
  }
  Serial.println("Connected!");

  pedometer.stepCalc();

  client.print((String)"GET
  /api/v1/senzor/TOKEN/input?value=" +
  String(pedometer.stepCount) +
  "HTTP/1.1\r\n" +
```

```
  "Host: " + String(host) + "\r\n" +
  "Connection: close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0)
  {
    if (millis() - timeout > 5000)
    {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }

  while (client.available())
  {
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }

  Serial.println();
  Serial.println("closing connection");
  client.stop();
}
```

Sistemul va posta în mod automat numărul de pași la un interval de 10 minute. În cazul în care ieșim din aria de acoperire WiFi sistemul va contoriza în continuarea numărul de pași efectuați și la reîntoarcere va realua raportarea. Astfel vom putea evalua numărul de pași efectuați și ”în casă” și afară. Resetarea numărului de pași se va face apăsând butonul de pe pinul 0 al plăcii de dezvoltare.





Referințe on-line

(1) Robofun IoT
http://iot.robofun.ro/

(2) Robofun IoT API
http://iot.robofun.ro/doc

(3) Arduino Pedometer
http://www.instructables.com/id/Arduino-Pedometer/

(4) Simple, Easy and Cheap DIY Pedometer With Arduino
http://www.instructables.com/id/Simple-Easy-and-Cheap-DIY-Pedometer-with-Arduino/

(5) Arduino Pedometer Watch, With Temperature, Altitude and Compass!
http://www.instructables.com/id/Arduino-Watch-With-Altitude-Temperature-Compass-An/

(6) Accelerometru ADXL345

https://www.robofun.ro/senzori/accelerometru/accelerometru_adxl345?utm_medium=email&utm_content=productLink&utm_campaign=CURS_EMAIL

(7) Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer

http://www.analog.com/en/analog-dialogue/articles/pedometer-design-3-axis-digital-acceler.html

(8) ESP32 Thing

https://www.robofun.ro/iot/sparkfun-esp32-thing?utm_medium=email&utm_content=productLink&utm_campaign=CURS_EMAIL

(9) ESP32 Thing Hookup Guide

https://learn.sparkfun.com/tutorials/esp32-thing-hookup-guide

(10) Pedometer: this is a easy pedometer made by using seeed Xadow products

https://github.com/Seeed-Studio/Pedometer

(11) Accelerometer_ADXL345: Seeed 3-Axis Digital Accelerometer library

https://github.com/Seeed-Studio/Accelerometer_ADXL345