**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**FACULTATEA DE SISTEME INFORMATICE ȘI SECURITATE CIBERNETICĂ**

**Specializarea: Calculatoare și sisteme informatice pentru apărare și securitate națională**



***Push button + BCD***

***Platforma de dezvoltare FRDM-KL25Z***

Stud. Sg. Maj. Bujdea Liviu-Andrei (C114 B)

Stud. Sg. Maj. Bilțan Ilie Cosmin (C114 B)

Stud. Sg. Maj. Ceauș Alexandru (C114 B)

Cuprins

[1. Prezentarea senzorului de tip push button 3](#_Toc124416199)

[2. Prezentarea senzorului BCD 4](#_Toc124416200)

[3. Scopul proiectului 5](#_Toc124416201)

[4. Conectarea senzorilor la placa de dezvoltare FRDM-KL25Z 6](#_Toc124416202)

[4.1. Conectarea push button 6](#_Toc124416203)

[4.2. Conectarea modulului BCD 7](#_Toc124416204)

[5. Descrierea programului 8](#_Toc124416205)

[5.1. Funcția main 8](#_Toc124416206)

[5.2. Push button 8](#_Toc124416207)

[5.3. Modulul GPIO 9](#_Toc124416208)

[5.3.1. Inițializarea ledurilor din componența modulului BCD 9](#_Toc124416209)

[5.4. Modulul UART 11](#_Toc124416210)

[6. Generare grafic în python în timp real 12](#_Toc124416211)

[6.1 Librariile necesare 12](#_Toc124416212)

[6.2. Preluarea datelor din portul serial 12](#_Toc124416213)

[6.3. Initializare si afisare grafic 13](#_Toc124416214)

[7. Dificultăți întâmpinate 18](#_Toc124416215)

[8. Referințe 18](#_Toc124416216)

# Prezentarea senzorului de tip push button

Butonul este un tip de comutator utilizat pentru controlul unor aparate sau numai a unor funcții ale acestora. Butoanele sunt de diferite forme și dimensiuni și se găsesc în tot felul de dispozitive, în principal, în echipamentele electrice și electronice. Butoanele sunt în general activate atunci când sunt apăsate. De asemenea, acesta este un senzor digital.

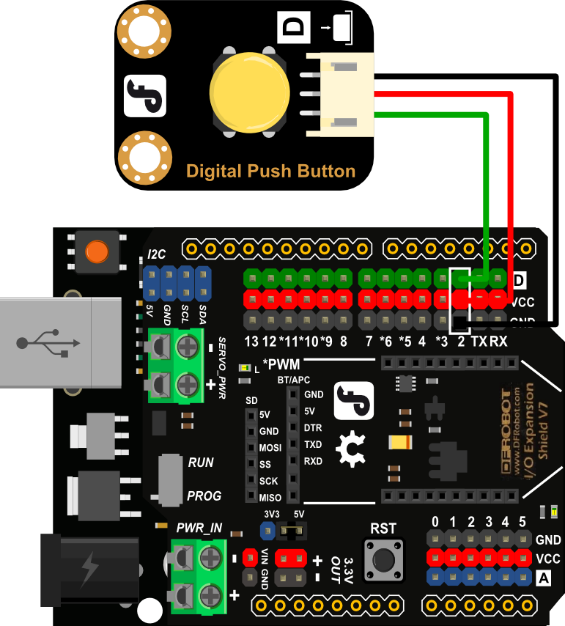


*Figura 1. Push button*

Pinii senzorului:

1. Signal output (pinul unde este conectat firul verde din fig. 2)
2. GND (pinul unde este conectat firul negru din fig. 2)
3. Power (pinul unde este conectat firul roșu din fig. 2)

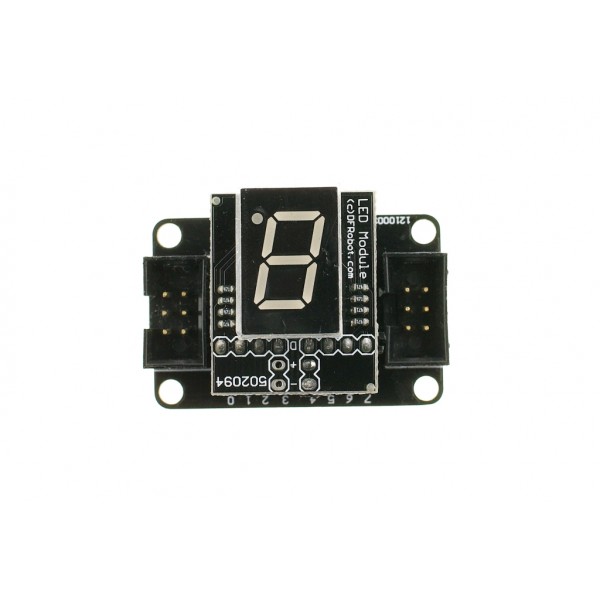
Exemplu de conectare la placa de dezvoltare:



*Figura 2. Exemplu conectare senzor*

# Prezentarea senzorului BCD

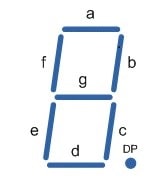
Senzorul BCD este de fapt un modul de afișare LED foarte simplu și mic.



*Figura 3. BCD decoder*

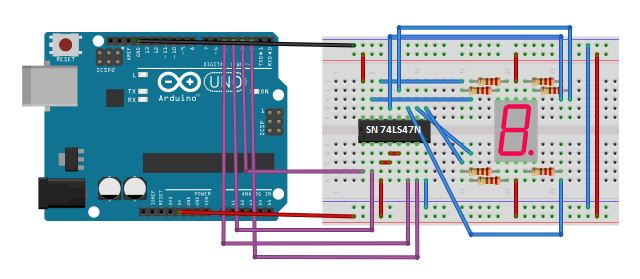
Pinii modului LED:

1. Pinul pentru masă (-)
2. Pinul pentru tensiune (+)
3. Pinii 1-8 corespunzători fiecărui segment (include și punctul)



*Figura 4. Segmentele BCD-ului*

Exemplu de conectare la o plăcuță arduino:



*Figura 5. Exemplu conectare BCD*

# Scopul proiectului

Scopul proiectului este acela de a schimba segmentul aprins din cadrul BCD-ului prin apăsarea push button-ului. Se va face un baleiaj al ledurilor, astfel nu vor exista două segmente aprinse simultan, ci se va stinge un segment și se aprinde altul.

Pentru a exemplifica un caz, pentru cifra 0, la apăsarea butonului, se va aprinde segmentul a (fig. 4). La apăsarea butonului, se va stinge segmentul a și se va aprinde segmentul b, și tot așa până la terminarea cifrei 0, apoi se va trece la 1, după care prin toate cifrele.

Proiectul se imparte in trei componente principale:

1.Partea de GPIO, scrisa in limbajul de programare C;

2.Partea de Uart, scrisa in limbajul de programare C;

3.Interfata grafica, scrisa cu ajutorul limbajului python.

Prin functionarea sincrona a celor trei parti am reusit sa trimitem semnal digital prin actionarea senzorului de push button pentru a aprinde pe rand led-urile senzorului BCD si a afisa in interfata, sub forma de grafic balejul acestora.

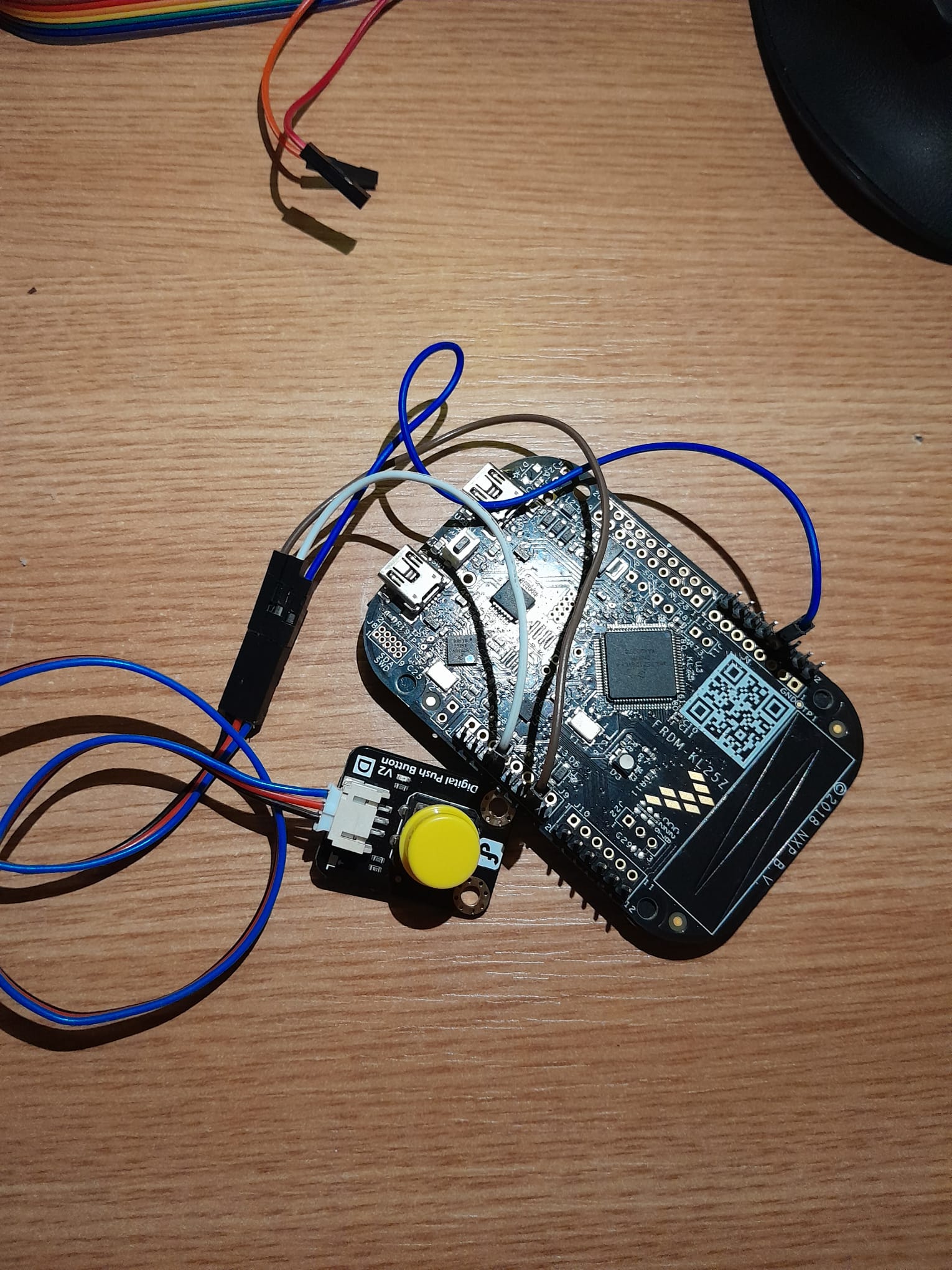
Astfel, graficul nostru va lua pe rand valorile tuturor segmentelor aprinse pentru fiecare caracter reprezentat in baza16.

# Conectarea senzorilor la placa de dezvoltare FRDM-KL25Z

Vom conecta senzorii astfel:

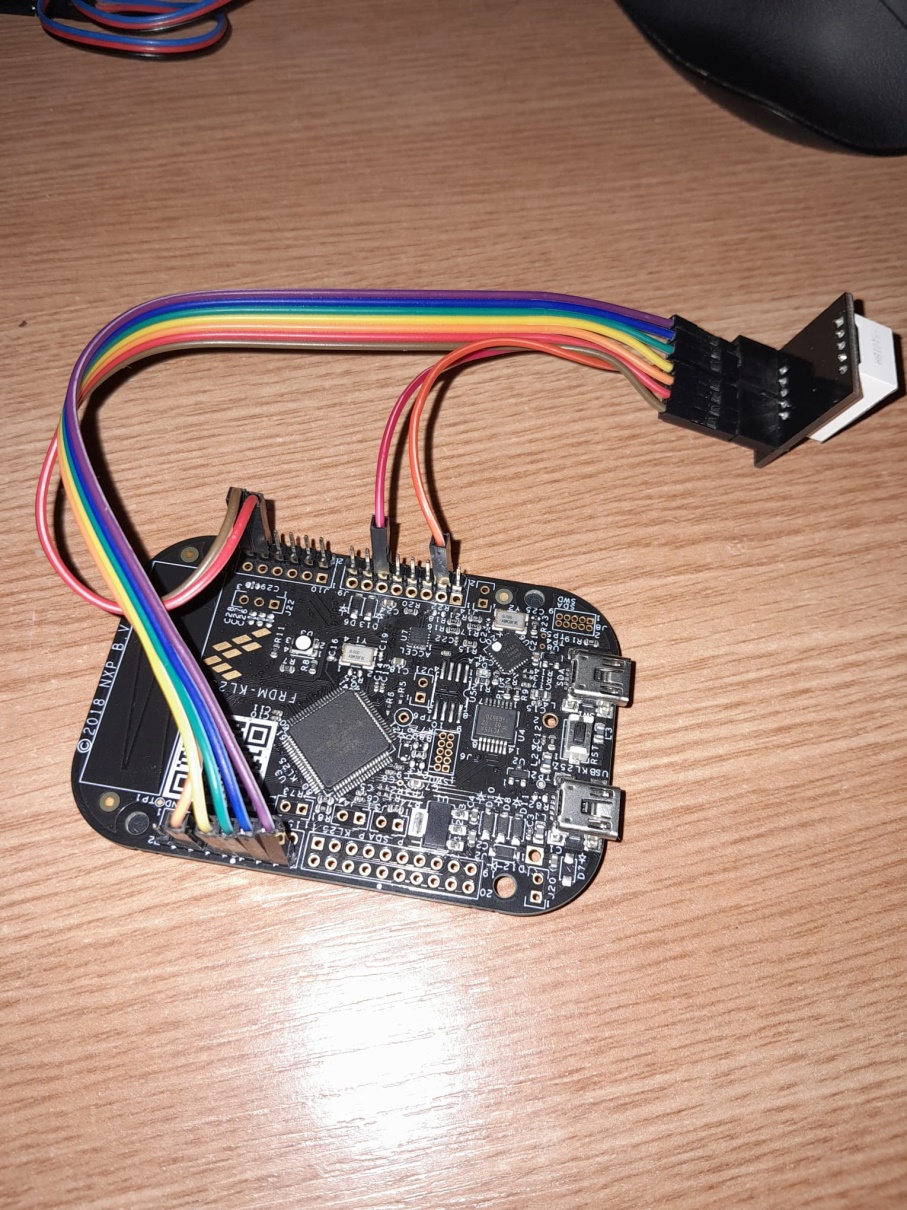
## Conectarea push button

* Firul albastru – albastru conectează BCD-ul la PTA12
* Firul roșu – alb conectează BCD-ul la P3V3
* Firul negru – maro conectează BCD-ul la GND



## Conectarea modulului BCD

* Firul portocaliu scurt conectează modulul la P3V3
* Firul roșu scurt conectează modulul la GND
* Firul maro conectează modulul la PTC1
* Firul roșu lung conectează modulul la PTC2
* Firul portocaliu lung conectează modulul la PTD4
* Firul galben conectează modulul la PTA4
* Firul verde conectează modulul la PTA5
* Firul albastru conectează modulul la PTC8
* Firul mov conectează modulul la PTC9



# Descrierea programului

## 5.1. Funcția main

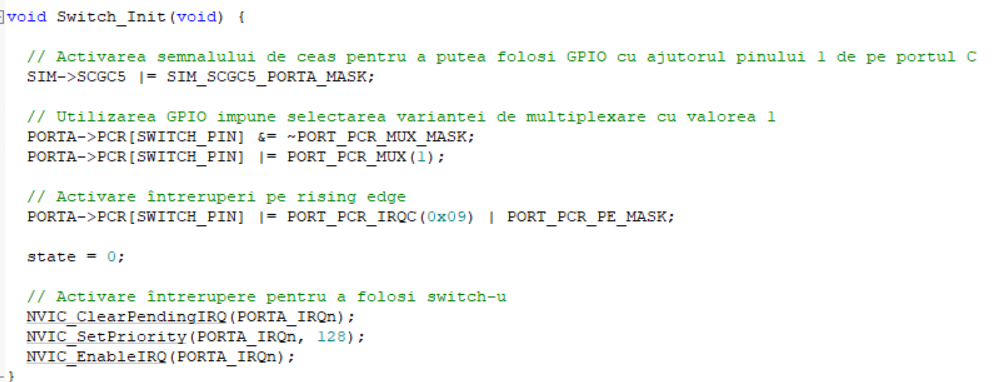
În fișierul main.c am inclus fișierele header în care sunt declarate funcții și variabile ce urmează a fi folosite: *gpio.h* și *uart.h.* Dupa care inițializez modulul UART și switch-ul, adică push button-ul.

## 5.2. Push button

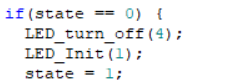
Pinul folosit este următorul:



Vom inițializa push button-ul prin intermediul funcției Switch\_Init().



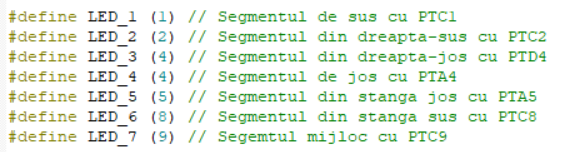
De asemenea, la fiecare apăsare de buton, variabila state se va modifica pentru a ști în ce stadiu se află BCD-ul.



## 5.3. Modulul GPIO

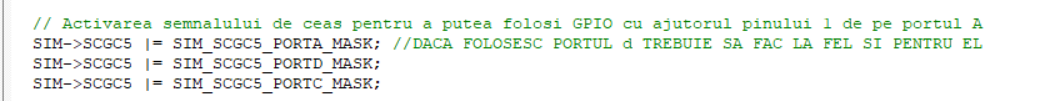
GPIO (General Purpose Input-Output) este o interfață programabilă pe un dispozitiv de care ne folosim pentru a transmite primi date cu alte dispozitive sau componente. De exemplu, putem folosi pinii GPIO pentru a citi date de la senzori sau a controla niște motoare. În proiectul nostru am folosit Modulul GPIO pentru a comunica cu modulul BCD.

Pinii folosiți la nivelul aplicației sunt următorii:

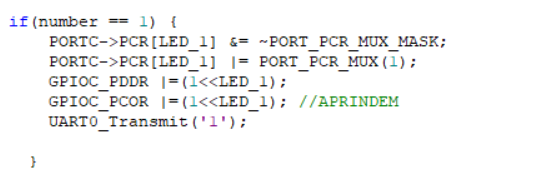


### 5.3.1. Inițializarea ledurilor din componența modulului BCD

Folosim registrul SCGC5 din cadrul SIM (System Integration Module) pentru a active semnalele de ceas pentru a putea folosii pinii declarați anterior. Am activat semnale de ceas pentru porturile A, B, C și D.



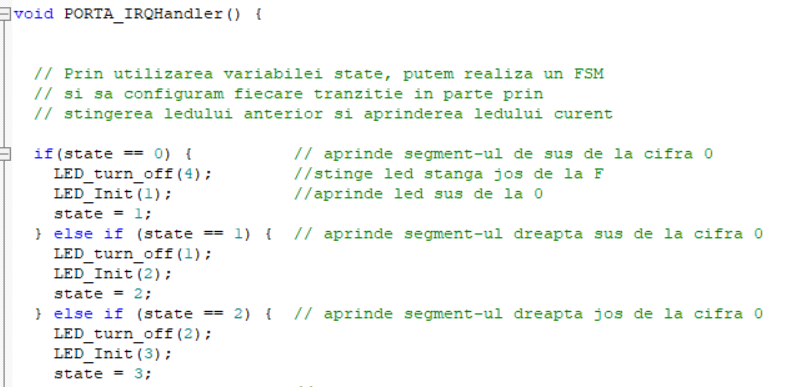
Cele 7 segmente din componența Modulului BCD sunt aprinse unul câte unul pentru a putea crea baleiajul dorit. Având în vedere că într-un anumit moment de timp se stinge un segment și se aprinde un altul, am creat o funcție ce aprinde un segment, precum și o funcție ce stinge un segment. Aici avem un exemplu ce aprinde segmentul 1 al BCD-ului:



Pentru stingerea segmentului se folosește registrul PSOR astfel:



În funcția PORTA\_IRQHandler() avem toate stările prin care poate trece BCD-ul folosind un state care se incrementează la fiecare apăsare a push button-ului.



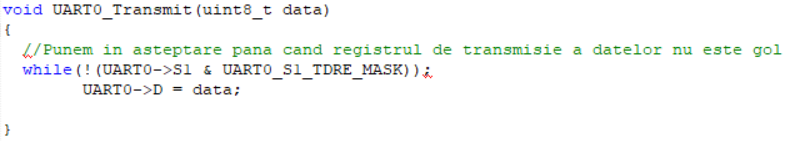
## 5.4. Modulul UART

Acest mod de comunicație este numit și start-stop. Controlerul UART se ocupă de comunicarea serială asincronă dintre un computer și un dispozitiv periferic conectat la portul serial al calculatorului și convertește datele de la serial la paralel și invers.

Transmisiile seriale de date au la bază transmiterea pachetelor de informații printr-un mediu de transmisie a semnalelor electrice (în general cablu coaxial) într-un singur șir sau serial. Pentru interconectarea dispozitivelor ce constituiau rețeaua nu este necesară folosirea unui switch, conexiunea făcându-se de la un dispozitiv la celălalt, iar la capătul șirului se montează un dispozitiv terminal pentru adaptarea de impedanță.

Utilizăm funcția UART\_Init pentru a inițializa modulul UART si pentru a specifica baud rate-ul dorit comunicării.

Ne vom folosi de UART pentru a putea transmite valori, astfel putând crea graficul din python.



Avem și un exemplu prin care transmitem un semnal prin UART în funcție de stadiul în care se află BCD-ul



# Generare grafic în python în timp real

## 6.1 Librariile necesare

* **serial**, care este folosit pentru a comunica cu portul serial
* **numpy**, care este folosit pentru a lucra cu array-uri de date
* **tkinter**, care este folosit pentru a crea interfața grafică (GUI)
* **FigureCanvasTkAgg** - importă clasa FigureCanvasTkAgg din biblioteca **matplotlib**, care este utilizată pentru a afișa un grafic într-o fereastră tkinter
* **Figure** - importă clasa Figure din biblioteca **matplotlib**, care este utilizată pentru a crea un grafic

## 6.2. Preluarea datelor din portul serial

## 

Datele din portul serial le-am manipulat cu ajutorul functiei plot() care are rolul de a actualiza graficul cu noi date prin citirea unui nou byte din portul serial, a-l decoda prin conversia la un sir de caractere, ca mai apoi sa-l convertim in int.

Byte-ul primit de la portul serial reprezinta segmentul aprins pe BCD, acesta schimbandu-se la fiecare actionare a push buton-ului. Astfel linia graficului fluctueaza intr-un interval de la [1,7], in functie de schimbarile ledurilor, culoarea acestei linii schimbandu-se in functie de valoarea segmentului care este aprins:

* Dacă valoarea primită din portul serial se află între 1 și 3,inclusiv 0, atunci culoarea liniei trasate va fi **verde**
* Dacă valoarea primită din portul serial se află între 3 și 5, atunci culoarea liniei trasate va fi **galben**
* Dacă valoarea primită din portul serial se află între 5 și 7, atunci culoarea liniei trasate va fi **roșu**

În final, actualizăm datele care sunt trasate pe grafic și redesenăm canvasul cu ajutorul functiei draw oferită de o instanta a clasei FigureCanvasTkAgg declarată în main. Pentru a cauza efectul de interpretare a datelor in timp real ne vom folosi de funcția after pusă la dispoziție de api-ul bibliotecii tkinter și vom seta ca funcția plot() să se reapeleze după 1,5 secunde.

## 6.3. Initializare si afisare grafic

Cu ajutorul pachetului tkinter am creat o fereastra pe care va fi desenat graficul care are urmatoarele caracteristici:

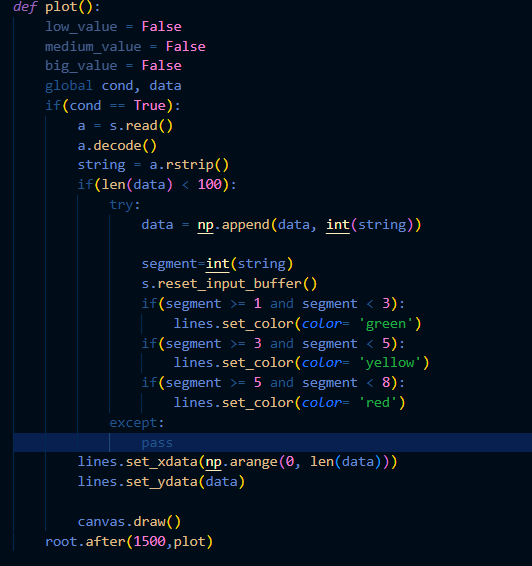
* titlul „Real Time Plot”
* backround „powderblue”
* dimensiunea/geometria de „700x500” pixeli

Cu ajutorul pachetului Figure creem un subplot cu urmatoarele caracteristici:

* titlul „Baleaj BCD”
* pe axa x masuram numarul de actionari a push buton-ului
* pe axa y este reprezentat segmentul aprins de pe BCD

Se creeaza un canvas pentru a afisa figura in fereastra, se deschide portul serial „COM9” cu o baude rate de 115200, se reseteasa buffer-ul de imput, iar apoi se face un apel la functia plot(), dupa 1.5 secunde si se lanseaza bucla principala a ferestrei.

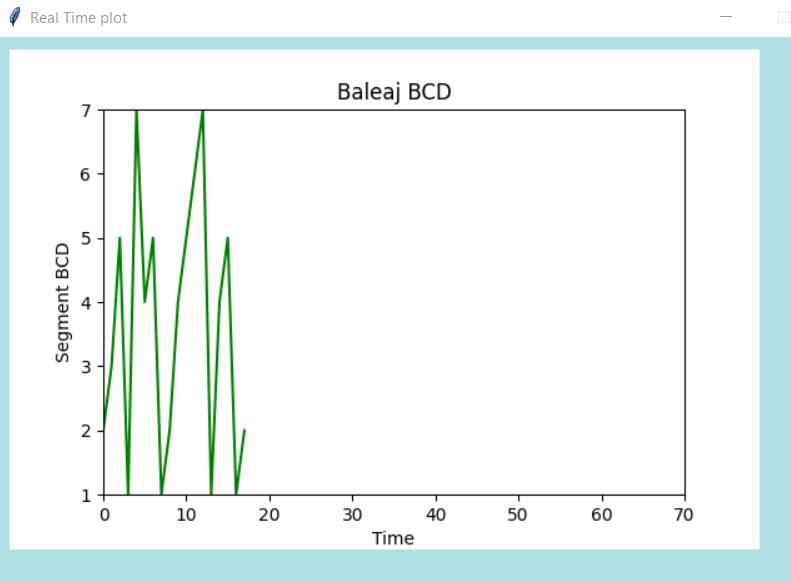
* Functia plot()



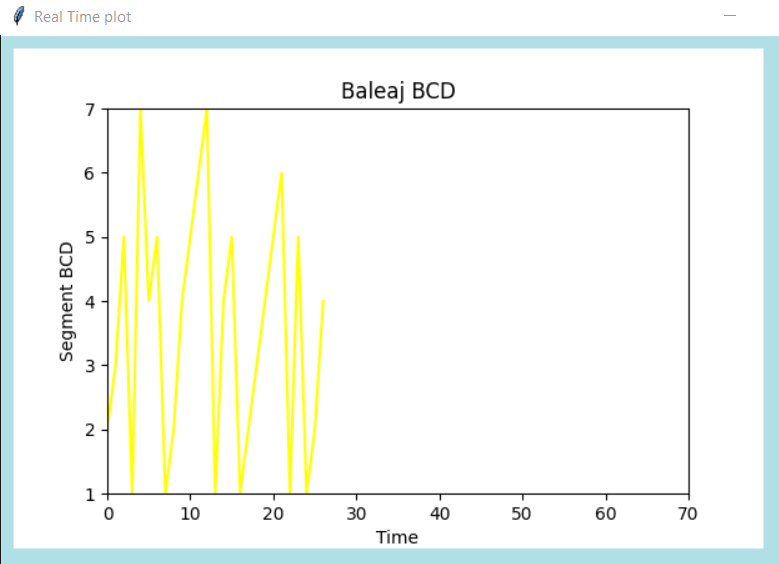
* Creeare si afisare grafic



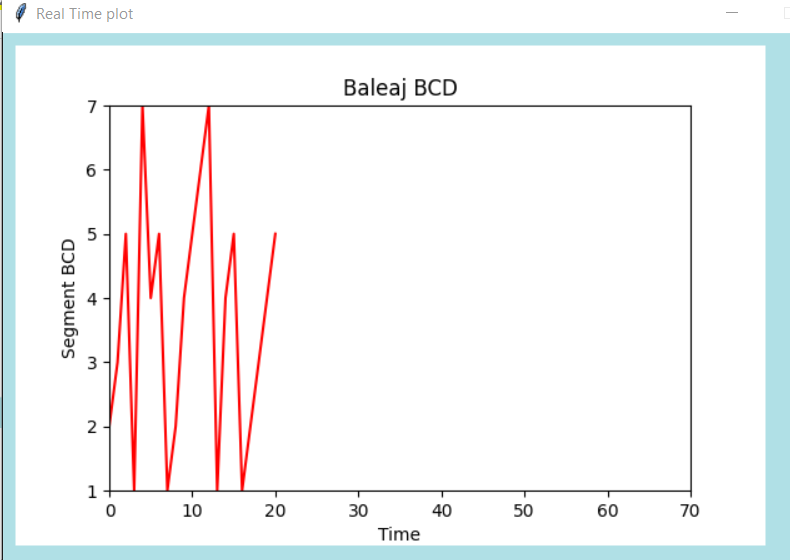
* Pentru segmentele cu valoarea mai mica decat 3.



* Pentru valori intre 3 si 5



* Pentru segmentele cu valoare mai mare decat 5



## 

## 

# Dificultăți întâmpinate

UART necesită porturile PTA1 și PTA2 pentru a funcționa. Problema întâmpinată a fost că am folosit aceste porturi pentru două din segmentele BCD-ului. Pentru a rezolva problema, am mutat acele două fire pe porturile PTC1 și PTC2.

Din cauza faptului ca push buton-ul este sensibil la atingere, uneori cand acesta se actioneaza de mai mult de 3 ori, valoarea de pe grafic nu corespunde in totdeauna cu valoarea ledului aprins.

# Referințe

https://ro.wikipedia.org/wiki/Transmisie\_de\_date\_serial%C4%83

https://wiki.dfrobot.com/DFRobot\_Digital\_Push\_Button\_SKU\_DFR0029

FRDM-KL25Z\_UserManual.pdf