Proiect Sisteme incorporate

Implementarea jocului Hangman (Spânzurătoarea) pe placa Nucleo

Andron Iulia-Maria

Bolbotină Ioana-Flavia

# 

# 

[**I. Descriere generala proiect 3**](#_rbn2ubf3aley)

[A. Caracteristici: 3](#_zgcpz8bvpo6a)

[**II. Descriere placa de dezvoltare EduBase-V2 3**](#_iife24jhdfcw)

[A. Componente: 3](#_a8n176chr4hz)

[B. Descriere placa Nucleo-64 STM32F446RE 5](#_5oqqtvprl9te)

[**III. Componentele utilizate 6**](#_a5c94yity1gy)

[A. Tabel componente 6](#_e4rbieehwztn)

[B. Descriere Componente 6](#_27tsglgux9rl)

[1. Display LCD 16X2 6](#_gq7fsuxeank2)

[1) Descrierea pinilor 6](#_3rkk6ccddqw5)

[2) Coduri de comandă pentru LCD 8](#_xwzq7vwkwzh)

[3) Conectarea la microcontroler 8](#_rx94z9ul4834)

[2. Display 7- segmente 9](#_n3fi4puyrxqh)

[1) Descrierea segmentelor 9](#_4iynxffteapo)

[2) Configurații numere în catod comun 9](#_qklvjwabbd8r)

[3. Butoane de intrare 10](#_kdvbsvkrsuxp)

[C. Configurația Pinilor 10](#_eh9lf5d3p7c8)

[**IV. Schemă Hardware 11**](#_req1q26s43bg)

[**V. Implementarea software 11**](#_tdo1fsadavc6)

[A. Inițializare componente și variabile globale 11](#_rc9jkaq8ptyf)

[B. Modul de funcționare a jocului: 12](#_7lqy3wkon32s)

[**VI. Bilbiografie 15**](#_uimvzlwvvib8)

# 

# **Descriere generala proiect**

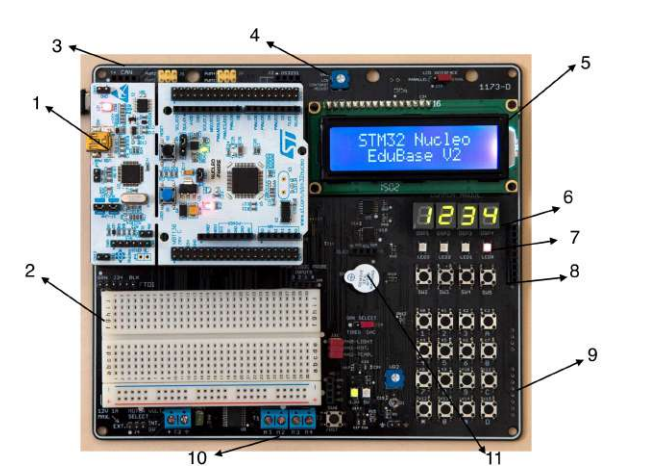
## **Caracteristici:**

* La începutul jocului se generează un cuvânt aleator de lungime variabilă (aleasă de studenţi) şi, pentru fiecare caracter din cuvânt, se afişează „\_” pe prima linie a afişajului LCD.
* Utilizatorul trebuie să ghicească cuvântul, având 6 încercări.
* Caracterul dorit este ales de utilizator, la un moment dat, prin folosirea a 3 butoane (pushbuttons): un buton este folosit pentru parcurgerea caracterelor în ordine alfabetică (fiecare apăsare determină trecerea la următoarea literă), un buton va parcurge caracterele în ordine inversă, iar cel de-al treilea buton va confirma selecţia caracterului dorit.
* Dacă respectiva literă se află în cuvânt, pe poziţiile corespunzătoare din cuvânt se va afişa acea literă, in loc de „\_”.
* În cazul în care caracterul nu se află în cuvânt, numărul de încercări rămase este decrementat.
* Numărul de încercări se afişează pe LED-uri sau pe un rang din afişajul cu 7 segmente.
* În momentul în care numărul de încercări rămase ajunge la 2 0, afișajele cu 7 segmente se vor aprinde într-o configurație specială: de exemplu se aprind segmentele de pe exterior (a,b,c,d,e,f) pe rând, cu viteză mare.
* Jocul se termină în momentul în care cuvântul a fost ghicit sau numărul de încercări a ajuns la 0, afişându-se un mesaj corespunzător pe LCD.

# **Descriere placa de dezvoltare EduBase-V2**

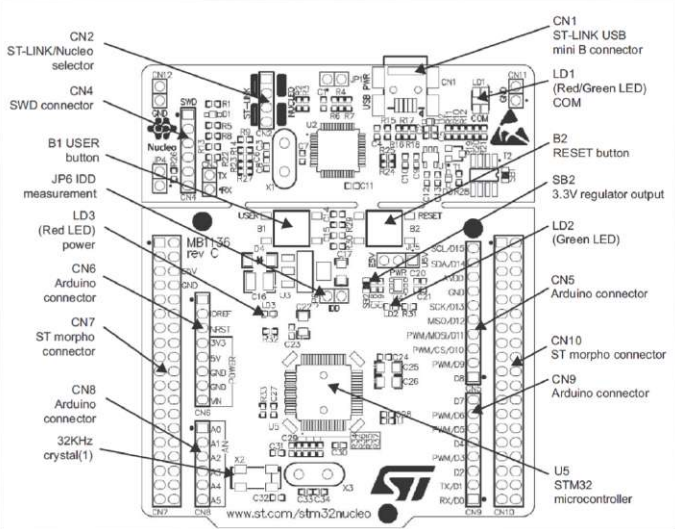
## **Componente:**

* + 1. Placa Nucleo-64 STM32F446RE;
    2. Extensie de dezvoltare de tip breadboard;
    3. Conectori interfețe comunicare serială (UART, CAN);
    4. Potențiometru și control LCD;
    5. LCD 2 linii 16 coloane;
    6. Module de afisaje cu segmente controlate prin serial shifters;
    7. Leduri;
    8. Input buttons;
    9. Keyboard 4x4;
    10. Motor Bridge connector;
    11. Zona analogica, senzori lumina, temperatura, buzzer



## **Descriere placa Nucleo-64 STM32F446RE**

## Procesorul STM32 este bazat pe arhitectura ARM Cortex-M și include gamele M0, M0+, M3, M4 și M7. Aceste procesoare dispun de memorie RAM variind de la câțiva kB până la câțiva MB. Toate microcontrolerele STM sunt echipate cu un set de periferice, incluzând 8 porturi GPIO, temporizatoare (timers), module de conversie analog-digital (ADC), conversie digital-analog (DAC), interfețe I2C, SPI, UART și CAN. Microcontrolerul utilizat în acest laborator este STM32Fxx, bazat pe nucleul Cortex-M4. Placa de dezvoltare care conține acest microcontroler se numește Nucleo-64 STM32F446RE. Principalele componente ale acestei plăci sunt ilustrate în imaginea de mai jos:



# **Componentele utilizate**

## **Tabel componente**

În cadrul proiectului s-a utilizat doar Placa de dezvoltare EduBase-V2. Componentele din tabelul de mai jos sunt incorporate pe această placă.

| Componenta | Cantitate |
| --- | --- |
| Display LCD 16x2 | 1 |
| Display 7- segmente | 1 |
| Butoane de intrare | 3 |

## **Descriere Componente**

### **Display LCD 16X2**

#### **Descrierea pinilor**

| Pin | Simbol | I/O | Descriere |
| --- | --- | --- | --- |
| 1 | VSS | - | GND |
| 2 | VCC | - | Alimentare +5V |
| 3 | VEE | - | Alimentare pentru controlul contrastului |
| 4 | RS | I | RS = 0 pentru a selecta registrul de comenzi, RS = 1 pentru a selecta registrul de date |
| 5 | R/W | I | R/W = 0 pentru scriere, R/W = 1 pentru citire |
| 6 | E | I | Enable |
| 7 | DB0 | I/O | Magistrala de date pe 8 biti |
| 8 | DB1 | I/O | Magistrala de date pe 8 biti |
| 9 | DB2 | I/O | Magistrala de date pe 8 biti |
| 10 | DB3 | I/O | Magistrala de date pe 8 biti |
| 11 | DB4 | I/O | Magistrala de date pe 4/8 biti |
| 12 | DB5 | I/O | Magistrala de date pe 4/8 biti |
| 13 | DB6 | I/O | Magistrala de date pe 4/8 biti |
| 14 | DB7 | I/O | Magistrala de date pe 4/8 biti |

##### VCC, VSS și VEE

* VCC și VSS furnizează alimentarea de +5V și, respectiv, împământarea.
* VEE este folosit pentru controlul contrastului LCD-ului.

##### RS (Register Select)

* RS = 0: Se selectează registrul de comandă, permite trimiterea unei comenzi.
* RS = 1: Se selectează registrul de date, permite trimiterea de date pentru a fi afișate pe LCD.

##### R/W (Read/Write)

* R/W = 0: Scriere
* R/W = 1: Citire

##### E (Enable)

* Pinul Enable este folosit de LCD pentru a prelua informațiile prezentate la pini săi de date. Când datele sunt furnizate la pini de date, un impuls (Low-to-High-to-Low) trebuie aplicat acestui pin pentru ca LCD-ul să preia datele prezente la pinii de date.

##### Pini de date D0-D7

* Pinii de date pe 8 biți (D0-D7) sunt folosiți pentru a trimite informații la LCD sau pentru a citi conținutul registrelor interne ale LCD-ului. Controlerul LCD este capabil să funcționeze cu date pe 4 biți, caz în care sunt folosiți doar pinii D4-D7.
* Pentru a afișa litere și cifre, se trimit codurile ASCII corespunzătoare literelor (A–Z, a–z), cifrelor (0–9) și semnelor de punctuație la acești pini în timp ce RS = 1.VCC și VSS furnizează alimentarea de +5V și, respectiv, împământarea.

#### **Coduri de comandă pentru LCD**

| Cod(hex) | Comanda |
| --- | --- |
| 01 | Șterge ecranul |
| 02 | Aduce cursorul în poziția de start |
| 06 | Mută cursorul la dreapta |
| 0F | Afișare, cursorul clipind |
| 80 | Forțează cursorul la începutul primei linii |
| C0 | Forțează cursorul la începutul celei de-a doua linii |
| 38 | 2 linii și caractere 5x7 (date pe 8 biți, D0-D7) |
| 28 | 2 linii și caractere 5x7 (date pe 4 biți, D4-D7) |

### Conectarea la microcontroler

### Display 7- segmente

Afișajul cu șapte segmente poate avea anod comun sau catod comun. Cu anod comun, anodul unui LED este conectat la tensiunea de alimentare pozitivă, iar microcontrolerul coboară catozii individual la nivel LOW pentru a permite curentului să circule prin LED-uri și să le aprindă. Cu catod comun, catodul unui LED este conectat la pământ, iar microcontrolerul ridică anozi individual la nivel HIGH pentru a aprinde LED-ul. În oricare configurație, dacă microcontrolerul nu are capacitatea suficientă de a furniza sau scădea curent, trebuie să adăugăm un buffer între afișajul cu șapte segmente și microcontroler.

#### Descrierea segmentelor

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| . | g | f | e | d | v | b | a |

#### Configurații numere în catod comun

| Număr | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Val(HEX) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0x3F |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0x06 |
| 2 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0x5B |
| 3 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0x4F |
| 4 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0x66 |
| 5 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0x6D |
| 6 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0x7D |
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0x07 |
| 8 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0x7F |
| 9 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0x6F |

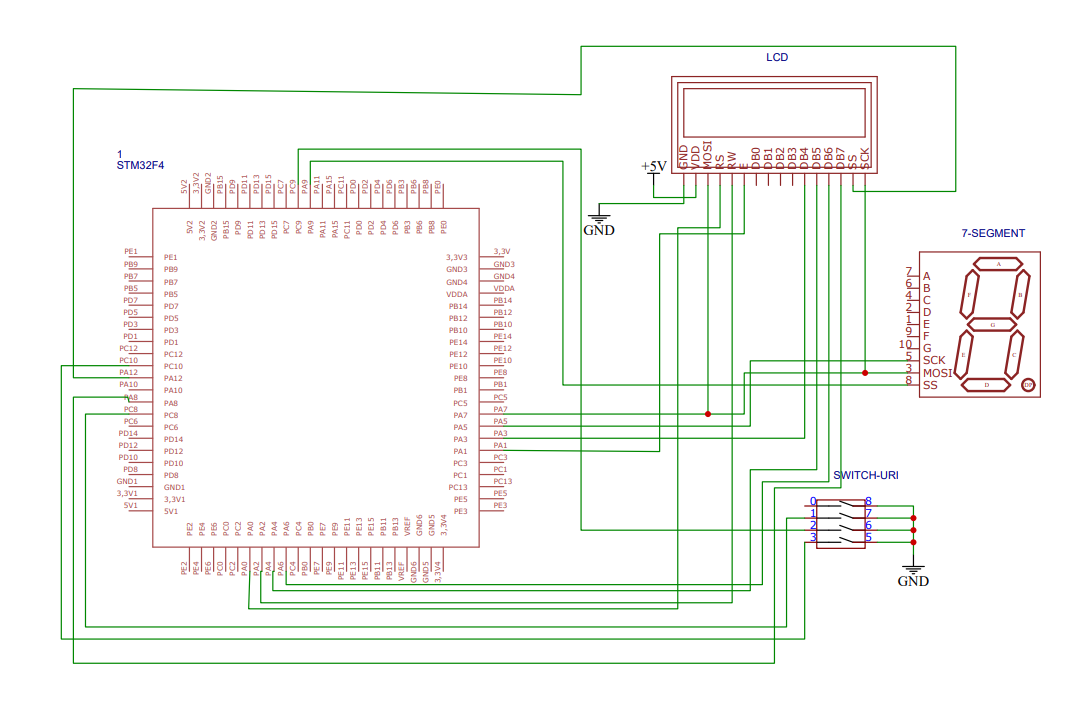
### Butoane de intrare

* Tip: Momentary push-buttons (butoane temporare)
* Funcție: Generarea unui semnal digital (HIGH/LOW) la apăsare
* Interfață: Conectate la pinii GPIO ai microcontrolerului
* Tensiune de operare: 3.3V sau 5V (în funcție de schema de conexiuni)
* Pini principali:
  + SW1, SW2, SW3: Conectați la pinii GPIO pentru detectarea stării (HIGH/LOW)

## Configurația Pinilor

* Interfața SPI pentru 7- segmente:
  + SCLK: PA5
  + MOSI: PA7
  + SS: PA9
* GPIO și interfața SPI pentru Control LCD:
  + RS: PA0
  + RW: PA2
  + E: PA1
  + D4: PA3
  + D5: PA4
  + D6: PA6
  + D7: PA8
  + VSS: GND
  + VDD: +5V
  + SCK: PA7
  + MOSI: PA5
  + SS: PA12
* GPIO pentru Butoane (Switch-uri):
  + SW1 (PC8): Buton pentru navigarea la stânga în selectarea literelor
  + SW2 (PC9): Buton pentru navigarea la dreapta în selectarea literelor
  + SW3 (PC10): Buton pentru confirmarea selecției literei

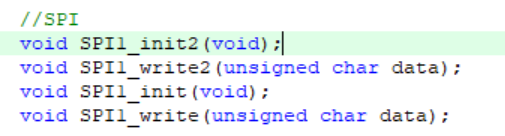
# Schemă Hardware



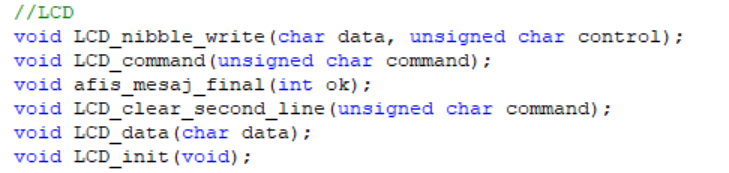
# Implementarea software

## Inițializare componente și variabile globale

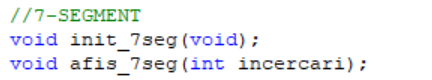
* + 1. Inițializarea SPI:
* Se configurează pinii GPIOA pentru funcționalitatea SPI (SCK și MOSI);
* Configurarea SPI1 pentru comunicare la 8-bit, Baud Rate configurabil;
* Activarea ceasului pentru SPI și GPIOA;
* Se utilizează funcțiile:



* + 1. Inițializarea LCD:
* Se trimite secvența de inițializare pentru a seta modul de funcționare 4-bit;
* Configurarea pinii RS, EN, și RW pentru controlul LCD-ului;
* Trimiterea comenzilor pentru setarea modului de afișare (clear screen, cursor blinking);
* Funcții utilizate:



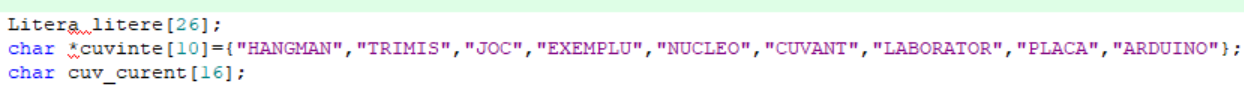
* + 1. Inițializarea 7- segmente:
* Se configurează pinii GPIOA pentru controlul segmentelor prin SPI;
* Trimiterea datelor pentru afișarea inițială (de exemplu, afișarea numărului 6 la început);
* Funcții utilizate:



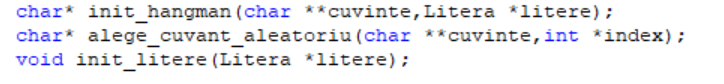
* + 1. Inițializarea Butoanelor (Switch-uri):
* Configurarea pinii GPIOC pentru input cu rezistențe de pull-up activate;
* Citirea stării butoanelor pentru a detecta apăsarea acestora;
* Funcții utilizate:



* + 1. Inițializarea cuvintelor, cuvânt curent, alfabet:
* În main avem variabilele globale:



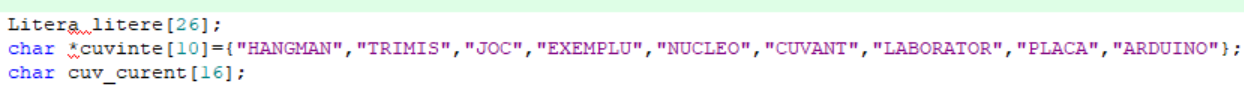
* Inițializarea acestor variabile se face cu ajutorul următoarelor funcții:



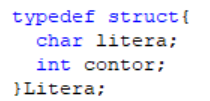
* init\_litere()- inițializează alfabetul începând de la caracterul ‘A’ până la ‘Z’;
* init\_hangman()- inițializează și returnează cuvântul curent ales aleator prin intermediul funcției alege\_cuvant\_aleatoriu();
* alege\_cuvant\_aleatoriu()- returnează cuvântul de la indexul ales aleator.

## Modul de funcționare a jocului:

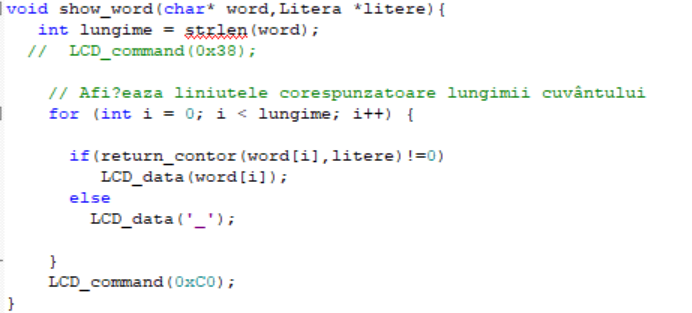
* + 1. Variabilele utilizate:



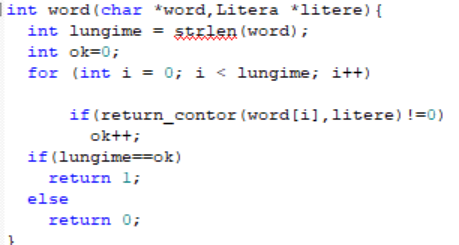
* Vectorul „litere” sunt de tipul struct Litera:



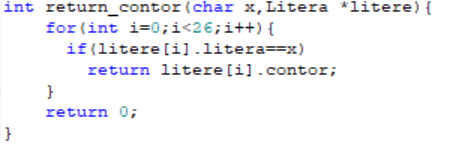
* Pointerul de șiruri „cuvinte” cuprinde toate cuvintele care pot să fie alese pentru joc;
* Șirul „cuv\_curent” reprezintă cuvântul ales în urma funcției init\_hangman().
  + 1. Funcțiile utilizate:
* show\_word()- afișează „\_” dacă litera nu este ghicită încă, și litera corespunzătoare dacă a fost ghicită:



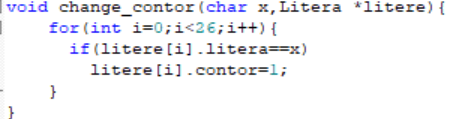
* int word()- verifică dacă toate literele din cuvânt au fost ghicite:



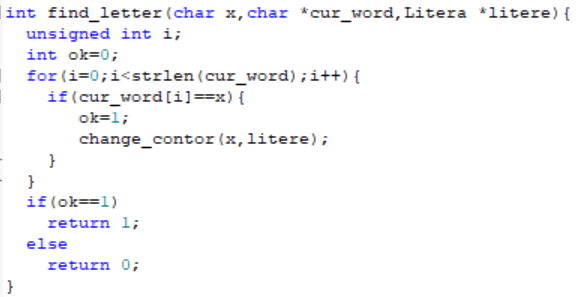
* int return\_contor()- caută litera curentă „x” și îi returnează contorul, care se regăsește în alfabet la poziția literei:



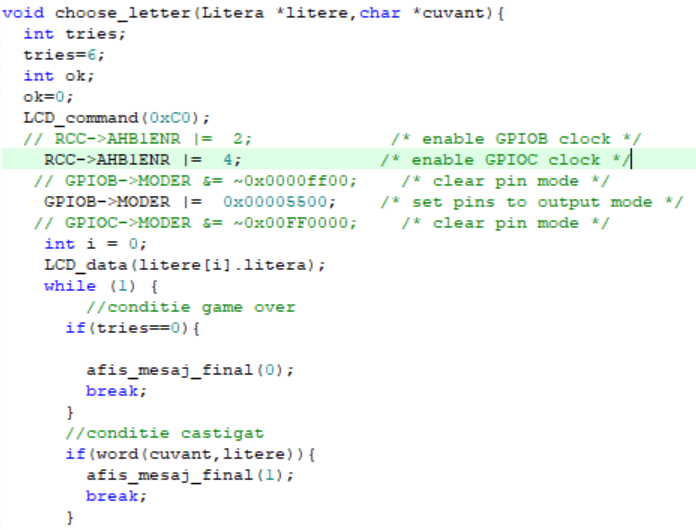
* void change\_contor(c)- modifică contorul literei „x” în alfabet la poziția literei respective:

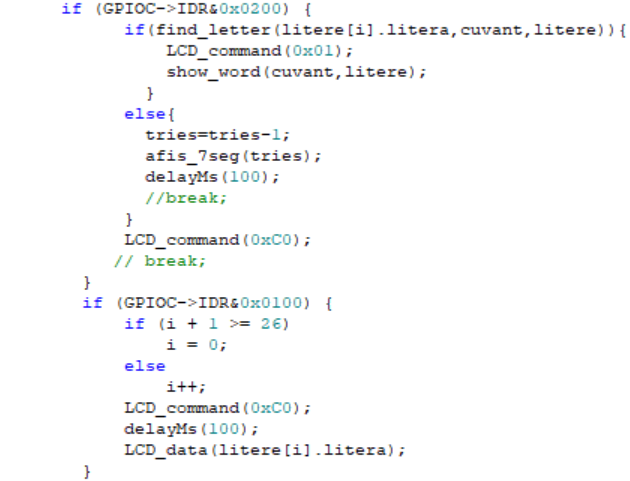


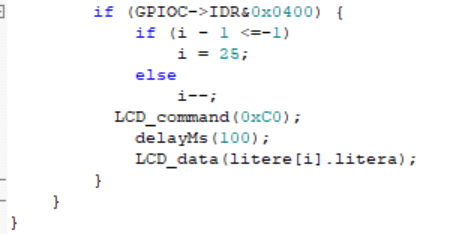
* int find\_letter()- verifică dacă litera transmisă ca parametru „x” se regăsește în cuvântul curent:



* void choose\_letter()
  + Aici are loc inițializarea butoanelor;
  + La început se inițializează variabilele „tries” (ce reprezintă numărul de încercări) și „i”(poziția în alfabet, litera curentă).
  + În while se verifică mai întâi dacă tries este egal cu 0 (cazul în care pierzi jocul) sau dacă word(cuvant,litere) returnează 1(cazul în care ai ghicit toate literele și ai câștigat);
  + La fiecare buton apăsat avem diferite funcționalități:
    - SW1: verificăm dacă cursorul „i” a ajuns la prima literă din alfabet(A) dacă nu a ajuns mutăm cursorul la poziția anterioară din alfabet;
    - SW2: acest buton are rol de ENTER în jocul nostru, dacă este apăsat verificăm dacă litera curentă se află sau nu în cuvânt. În cazul în care funcția find\_letter() returnează 1 dăm ștergere la ecran pentru a încarca varianta actualizată a cuvantului ce trebuie ghicit. În cazul în care litera nu se regăsește în cuvânt numărul de încercări scade și încărcăm afișajul actualizat cu numărul de încercări;
    - SW3: verificăm dacă cursorul „i” a ajuns la ultima literă din alfabet(Z) dacă nu a ajuns mutăm cursorul la următoarea poziție din alfabet;







# Bilbiografie

1. <https://web.archive.org/web/20220122054809/http://www.microdigitaled.com/ARM/STM_ARM_books.html>
2. Stm32 Arm Programming for Embedded Systems-Muhammad Ali Mazidi, Shujen Chen, Eshragh Ghaemi
3. <https://cv.upt.ro/pluginfile.php/800065/mod_folder/content/0/STM32F446RE_Datasheet.pdf?forcedownload=1>
4. <https://cv.upt.ro/pluginfile.php/800065/mod_folder/content/0/STM32F446RE_ReferenceManual_partial.pdf.zip?forcedownload=1>
5. <https://cv.upt.ro/pluginfile.php/800065/mod_folder/content/0/Nucelo-F44RE-UserManual.pdf?forcedownload=1>