**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**Facultatea de Sisteme Informatice și Securitate Cibernetică**

**Departamentul de Calculatoare și Securitate Cibernetică**



***UTILIZARE LED RGB INTEGRAT PENTRU AFIȘAREA CULORII ASOCIATE INTERVLULUI DE VALORI RECEPTȚIONATE DE SENZORUL DE INTENSITATE LUMINOASĂ ȘI SENZORUL DE SUNET FOLOSIND***

***PLATFORMA DE DEZVOLTARE FRDM-KL25Z***

Std. sg. maj. Denis GRIGORE (grupa C 114 D)

Std. sg. maj. Andrei CHINDA (grupa C 114 D)

Std. sg. maj. Paul CĂSĂNDROIU (grupa C 114 A)

**București**

**2022**

**Cuprins**

[1. Prezentarea senzorilor DFR0094 și DFR0026 3](#_Toc124265809)

[2. Scop proiect 4](#_Toc124265810)

[3. Conectare senzor – placă de dezvoltare 5](#_Toc124265811)

[4. Descriere program 6](#_Toc124265812)

[4.1. Funcția main 6](#_Toc124265813)

[4.2. Inițializarea modulelor 8](#_Toc124265814)

[4.2.1. Setarea bitiilor LED-urilor 10](#_Toc124265815)

[4.2.2. Inițializarea modulului UART 12](#_Toc124265816)

[4.2.3. Inițializarea modulului GPIO 14](#_Toc124265817)

[4.3. Transmitere date prin UART 14](#_Toc124265818)

[5. Rezultate Python 17](#_Toc124265819)

[6. Dificultăți întâmpinate 18](#_Toc124265820)

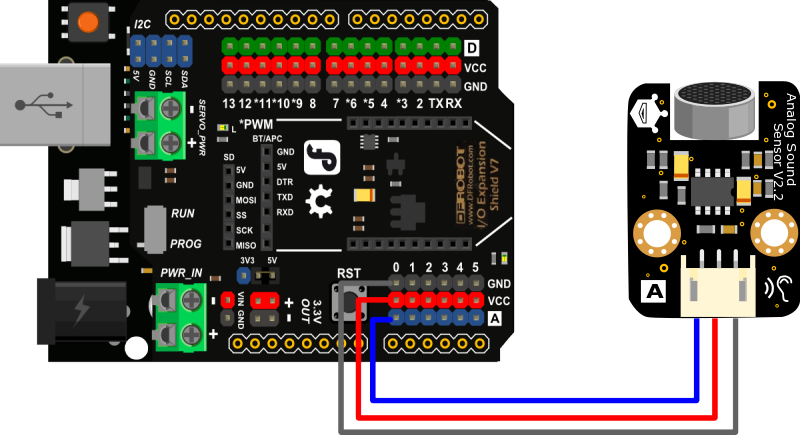
[7. Referințe 19](#_Toc124265821)

# Prezentarea senzorilor DFR0094 și DFR0026

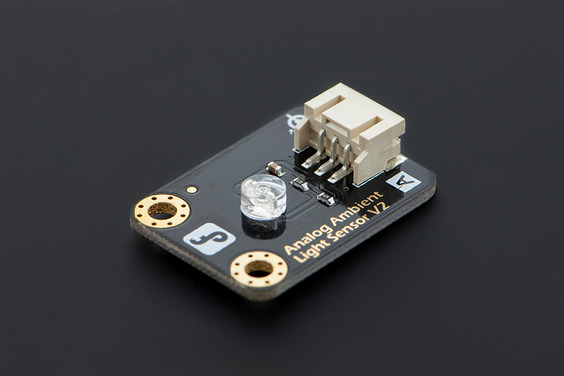
Senzorul DFR0094 este un senzor analog ce detectează intensitatea zgomotului, iar senzorul DFR0026 este un senzor analog ce detectează intensitatea luminoasă.



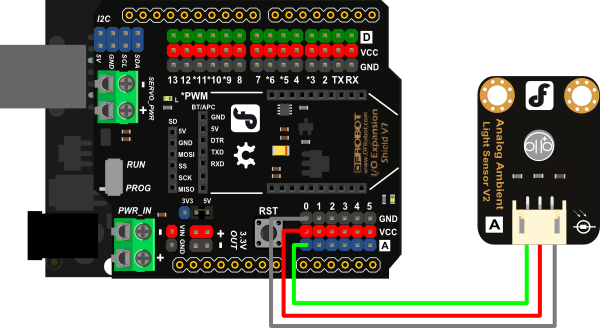
*Figura 1. Senzorul DFR0094*



*Figura 2. Exemplu conectare senzor*



*Figura 3. Senzorul DFR0026*



*Figura 4. Exemplu conectare senzor*

# Scop proiect

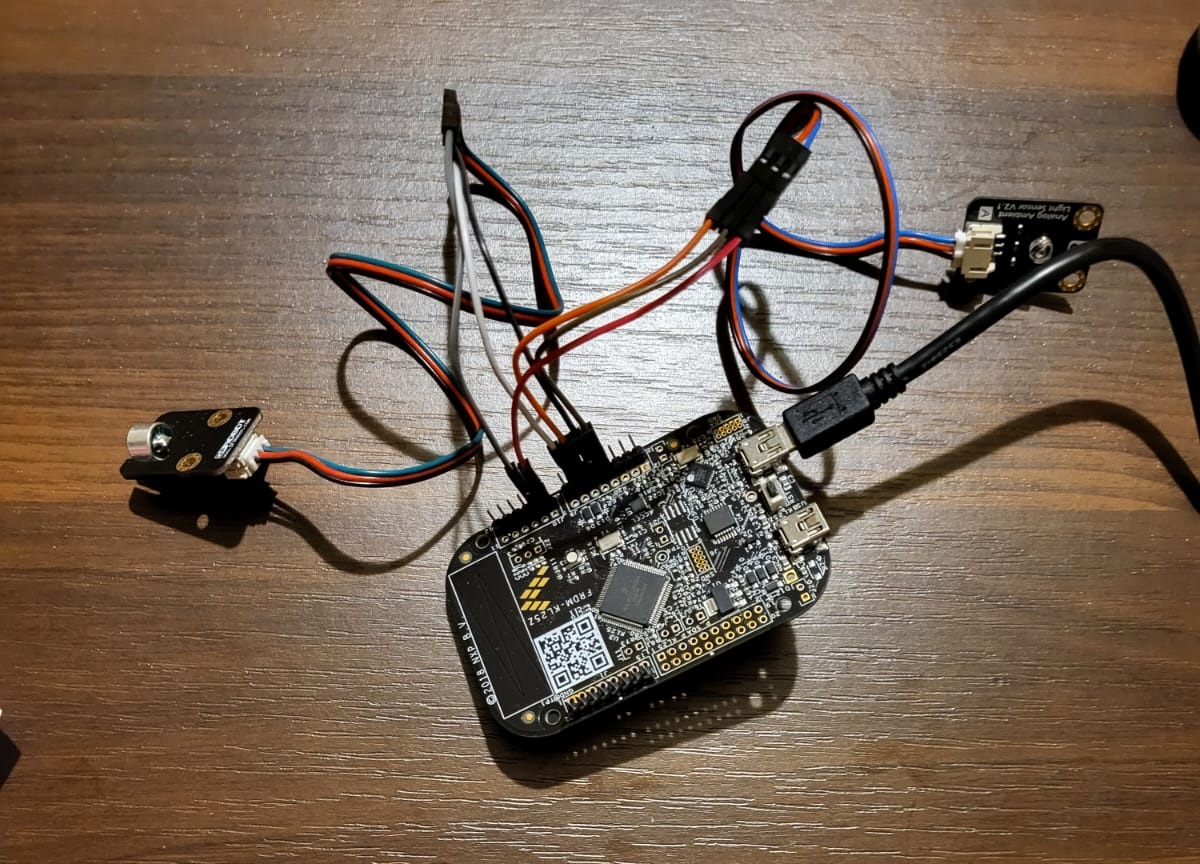
Scopul acestui proiect este acela de a modifica intensitatea LED-ului prezent pe placa de dezvoltare pusă la dispoziție pe baza ieșirii senzorului de intensitate luminoasă și celui de sunet, permițând comutarea citirii de la un zenzor la altul (la momentul T să citească de la senzorul de lumină, iar la momentul T+t, prin apăsarea unui buton, să citească valorile receptionate de la senzorul de sunet).

Se va dezvolta un program care, cu frecvența de 1kHz (o dată la 1 ms), va verifica ieșirea senzorului conectat. O dată la 1 ms se citește valoarea primită de la senzor (fie de la cel de lumina, fie de la cel de sunet) și se actualizează graficul din interfața grafică astfel încât să se observe mai bine valorile primite de la senzorul respectiv.

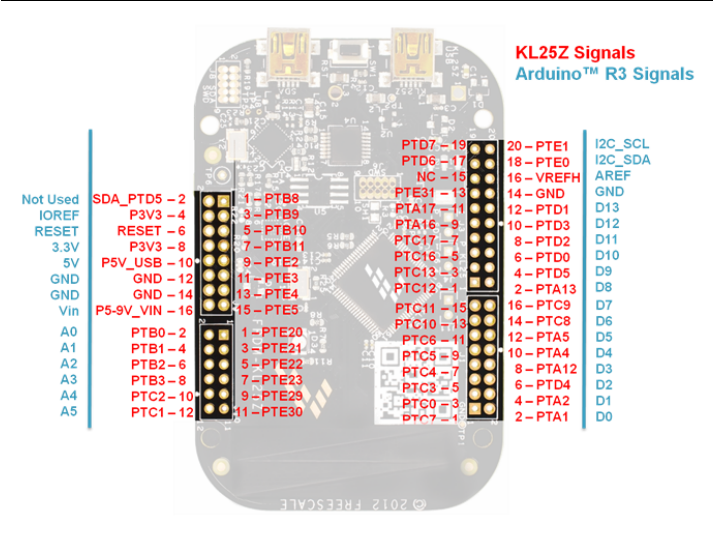
# Conectare senzor – placă de dezvoltare

Vom conecta senzorii astfel:

* Senzor lumină analog input - roșu se conecteaza la PTB0
* Senzor lumină GND - portocaliu se conecteaza la GND12
* Senzor lumină voltage - maro se conecteaza la P3V3
* Senzor sunet analog input - gri se conecteaza la PTB1
* Senzor sunet GND - alb se conecteaza la GND14
* Senzor sunet voltage - negru se conecteaza la P5V\_USB



*Figura 5. Conectare senzori*

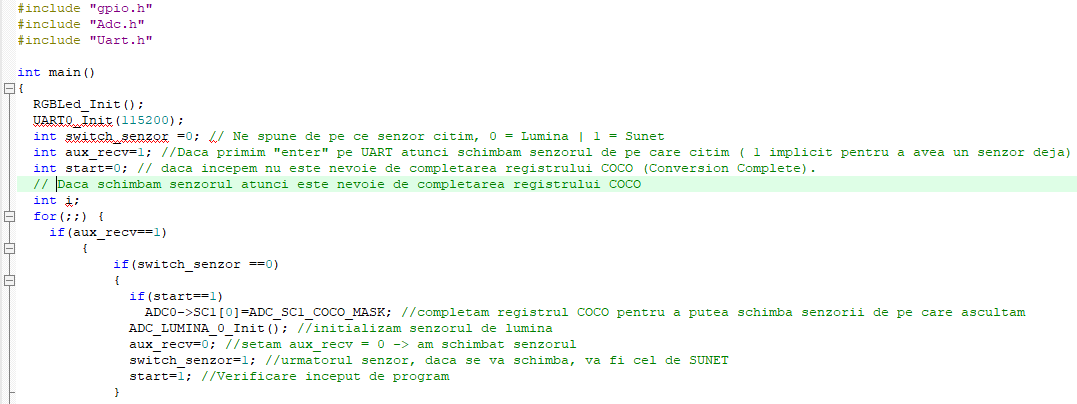


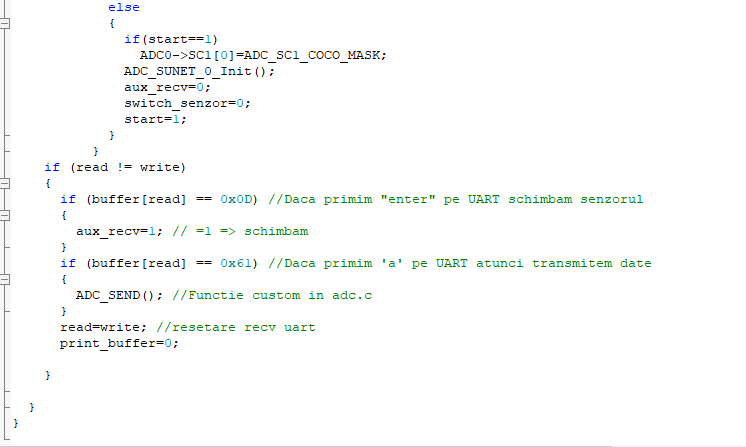
*Figura 6. Pini placă dezvoltare[[1]](#footnote-1)*

# Descriere program

## Funcția main

În fișierul main.c am inclus fișierele header în care sunt declarate funcțiile ce urmează a fi folosite: *gpio.h* (funcțiiile legate de gpio: void RGBLed\_Init(void)), *Uart.h* (funcțiile legate de uart: void UART0\_Init(uint32\_t baud\_rate)), *Adc.h* (funcțiile legate de adc: void ADC\_LUMINA\_0\_Init(void), void ADC\_SUNET\_0\_Init(void), void ADC\_SEND(void)).





*Figura 7. Funcția main*

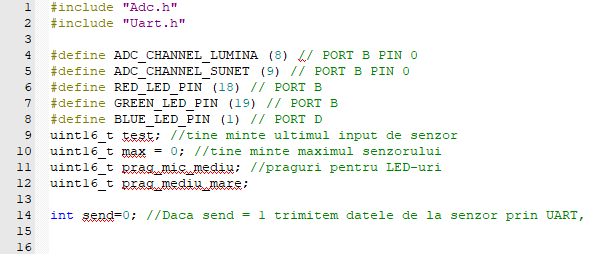
Logica principală a programului este următoarea: întâi se apelează funcția de inițializare a modulelor, apoi, într-un ciclu infinit, se verifică de la ce senzor se citesc valorile, iar în funcție de ce caracter se trimite prin uart (daca primim enter schimbam senzorul de pe care citim date, iar daca primim ‘a’, target-ul transmite datele citite de la senzorul aferent o singură dată).

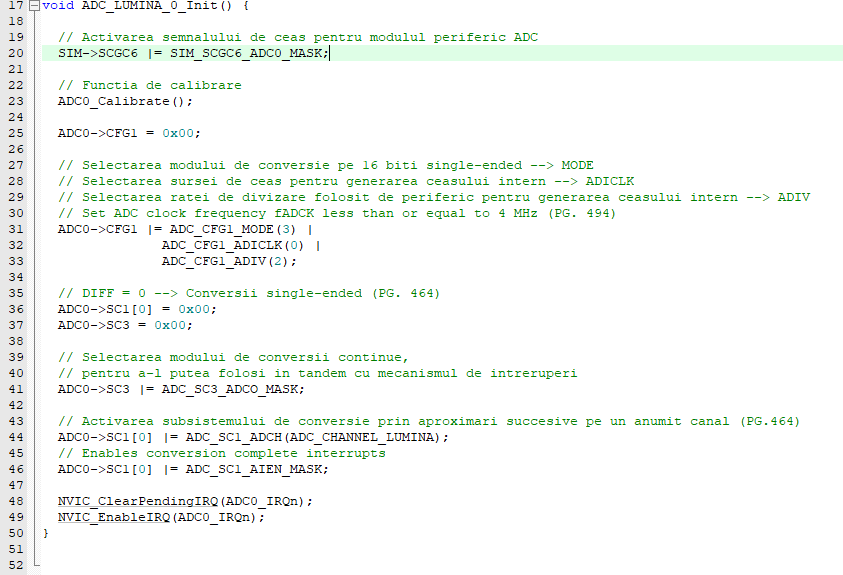
Folosim variabila ‘start’ pentru a evita completarea registrului ADC\_SC1\_COCO (Conversion Complete) înainte de inițializare, variabila “switch\_senzor’ este folosită pentru a determina ce senzor va fi inițializat la schimbarea senzorului ce citește, variabila ‘aux\_recv’ ne spune dacă s-au primit datele ce reprezinta comanda de schimbare a senzorului ce citește. Funcția ‘ADC\_SEND()’ este apelată in cadrul ‘adc.c’ iar aceasta schimba o variabila statică ce este verificata în handlerul senzorului după ce sunt trimise datele citite pe UART.

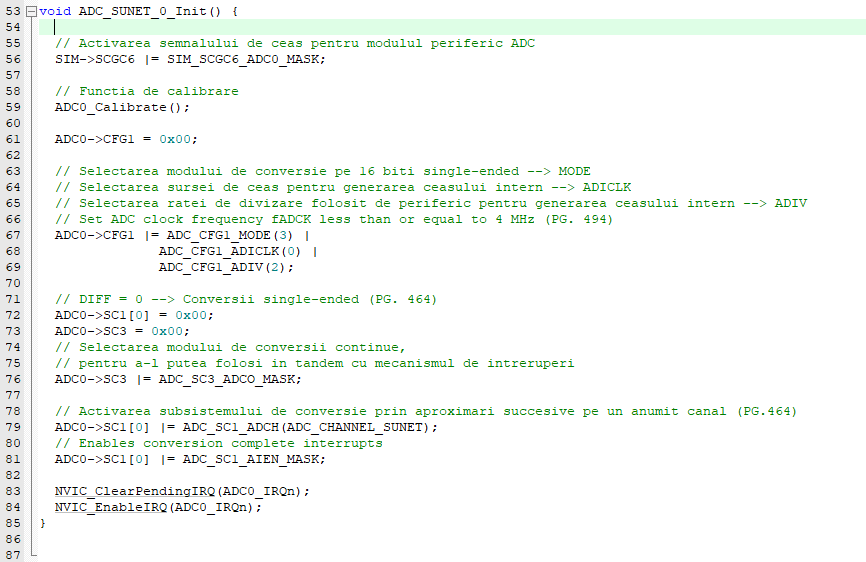
## Inițializarea modulelor

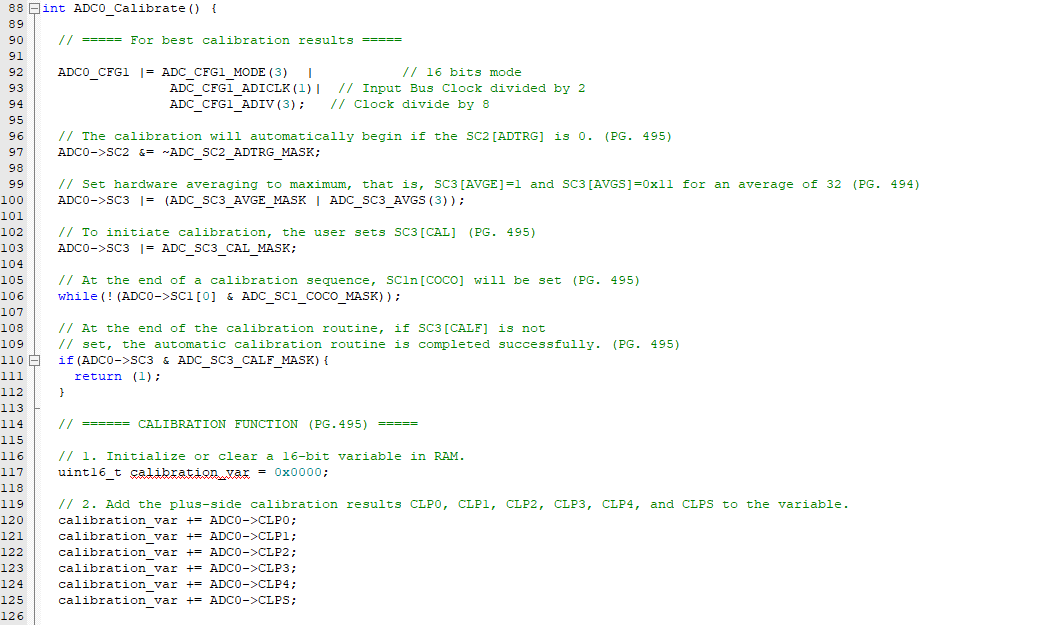
Prin apelul funcției RGBLed\_Init(), sunt apelate funcții ce configurează anumite module necesare, ce vor fi utilizate. În cadrul fișierului init.c sunt create mai multe macrodefiniții:

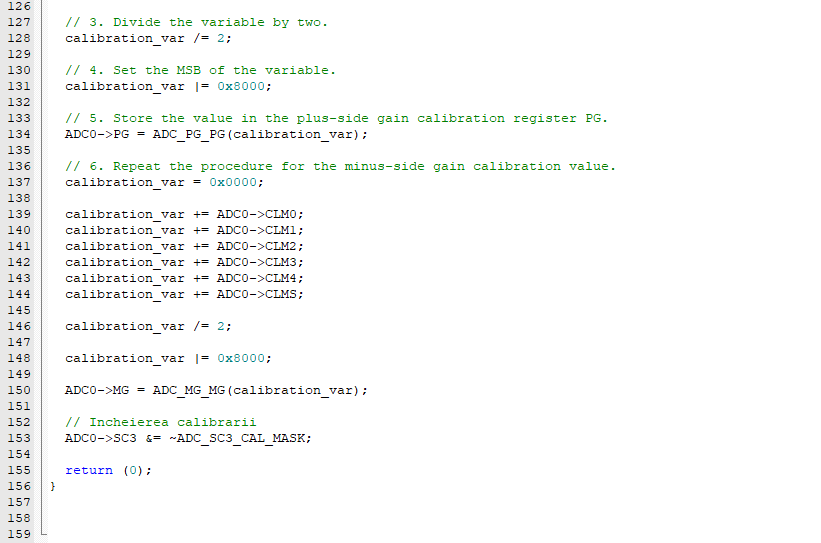
* SENZOR\_LUMINA– pinul 8 al portului B
* SENZOR\_LUMINA– pinul 9 al portului B
* RED\_LED\_PIN – pinul 18 al portului B
* GREEN\_LED\_PIN– pinul 19 al portului B
* BLUE\_LED\_PIN – pinul 1 al portului D

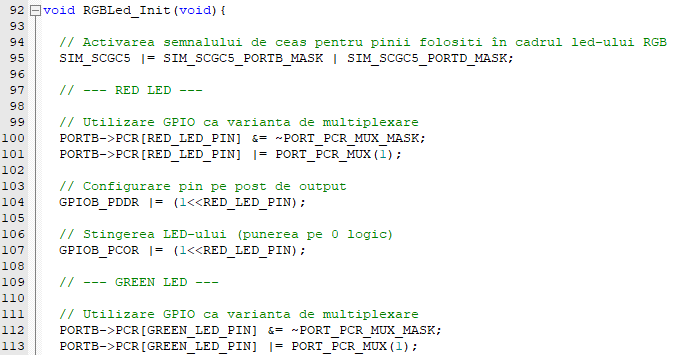










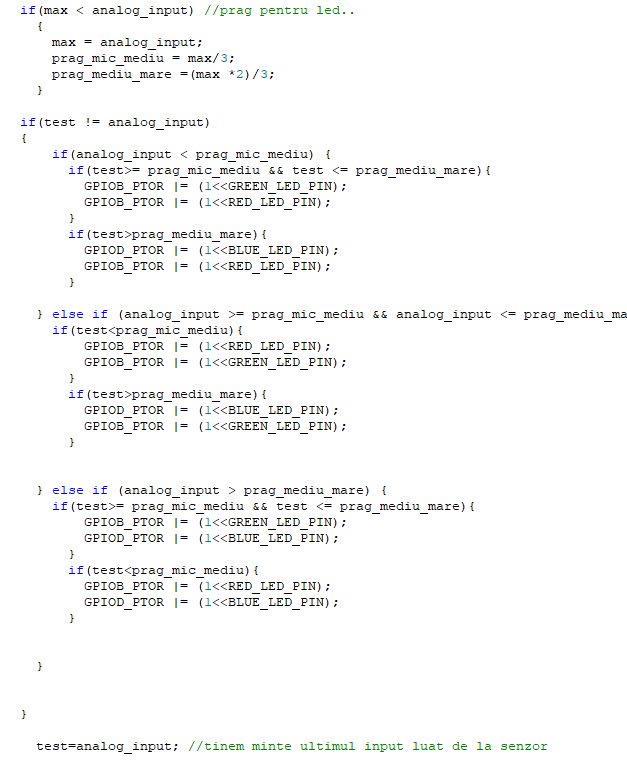


*Figura 8. Funcțiile de inițializare folosite pentru LED-ul RGB*

### Setarea bitiilor LED-urilor

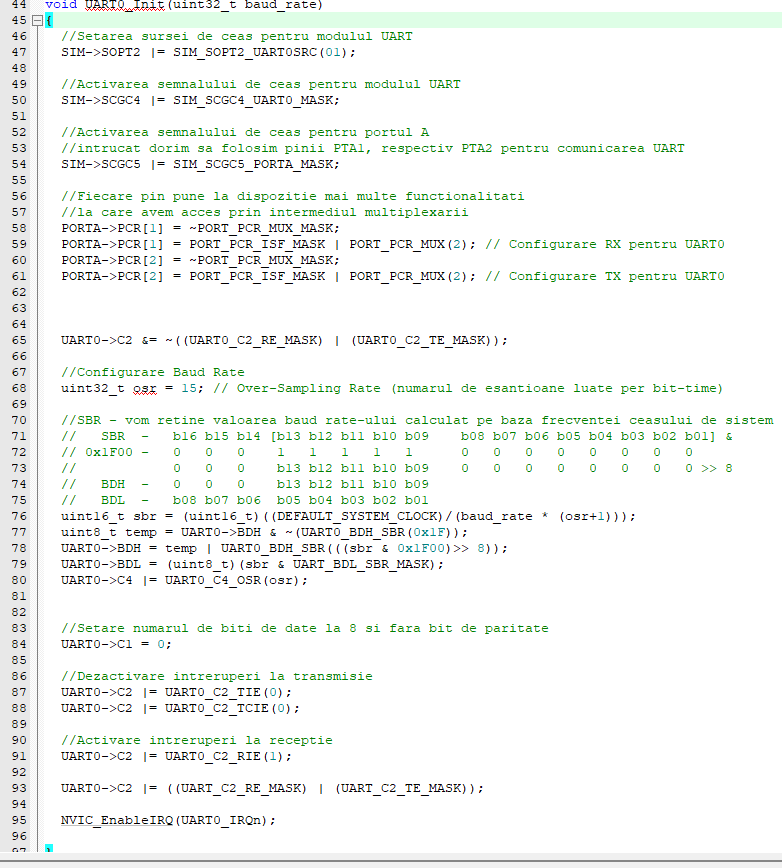
Se ia în considerare trecerea la un alt prag în funcție de ‘analog\_input’. Avem 2 praguri intermediare. Aceste praguri sunt calculate în funcție de maximul ‘analog\_input’ fiecare reprezentând: pragul mic este o treime din maximul ‘analog\_input’ iar pragul mare este două treimi din maximul ‘analog\_input’.

În variabila ‘test’ este ținut minte ultimul ‘analog\_input’ pentru a ne asigura că LED-urile primesc biții corespunzători. În caz contrar LED-urile ar fi aprinse greșit. Acest lucru este realizat folosind condițiile din codul de mai jos ce include variabilele numite intuitiv ‘prag\_mic\_mediu’ și ‘prag\_mediu\_mare’.

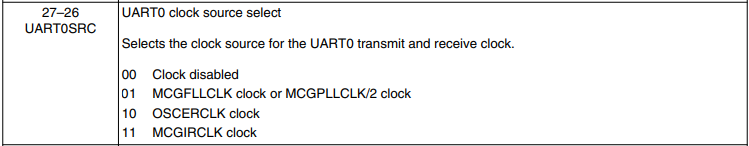


### Inițializarea modulului UART

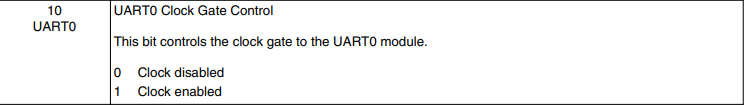
Vom folosi modulul UART0 pentru comunicația serială cu PC prin cablul USB.



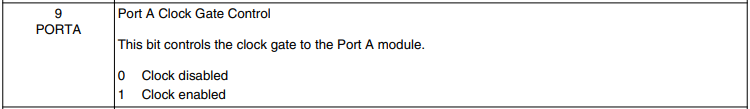
În registrul SIM\_SOPT2 (System Options Register 2) setăm pe 0b01 câmpul UART0SRC (biții 27-26) pentru selectarea ca ceas al modulului MCGFLLCLK anterior configurat.



În registrul SIM\_SCGC4 (System Clock Gating Control Register 4) setăm pe 1 câmpul UART0 (bitul 10) pentru activarea ceasului pentru acest modul.



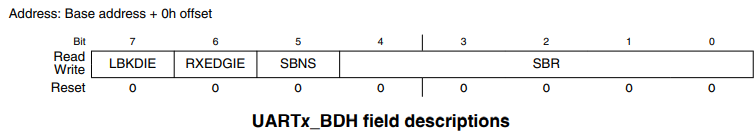
În registrul SIM\_SCGC5 (System Clock Gating Control Register 5) setăm pe 1 câmpul PORTA (bitul 9) pentru activarea ceasului acestui port.

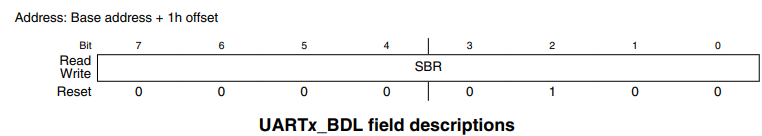


În regiștrii de control ai pinilor 1 și 2 din portul A (PORTA\_PCR1/2), setăm câmpul MUX (biții 10-8) pe valoarea 0b010, care înseamnă folosirea acestora în modulul de UART0 (RX/TX).

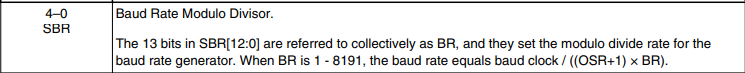


Configurăm baud rate la 9600 astfel: punem în câmpul UART0\_BDH valoarea 0x1, iar în UART0\_BDL valoarea 0x38. Primii 13 biți din cei doi octeți concatenați reprezintă valoarea SBR – baud rate setting = 0x138 = 312.

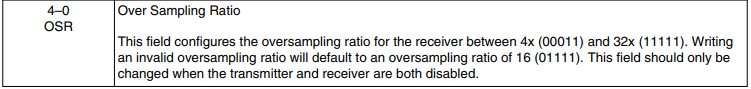




Formula după care se calculează baud rate este: baud clock / ((OSR+1) × BR) (pentru SBR între 1 – 8191).

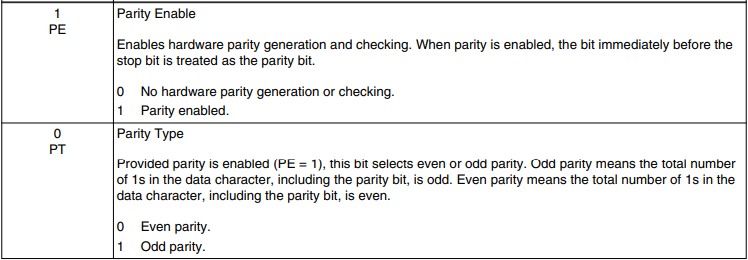


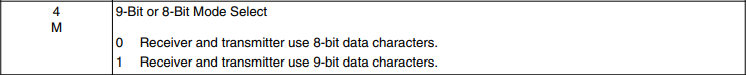
OSR (Over Sampling Ratio) are valoarea implicită 0b01111 adică 15, iar baud clock a fost setat la cel de 48 MHz.

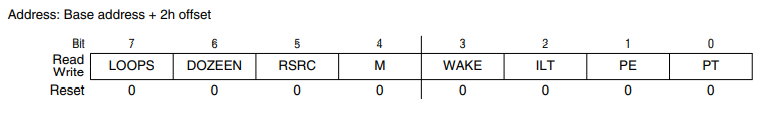


Valoarea baud rate este așadar: 48000000 / (16 × 312) aproximativ 9600.

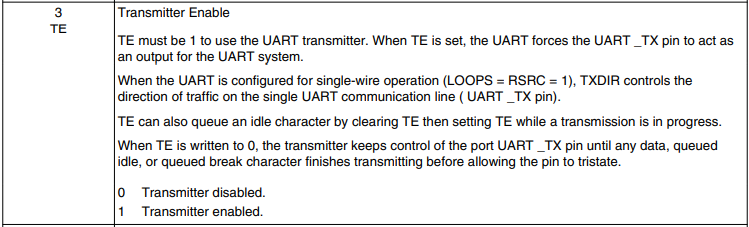
Portul serial este setat implicit cu 8 biți date, niciun bit de paritate și un bit de stop.





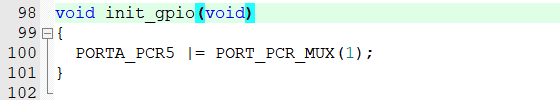


Activăm transmiterea prin UART prin setarea câmpului TE (bitul 3, Transmitter Enable) din UART0\_C2 pe valoarea 1.



### Inițializarea modulului GPIO

Pentru că ceasul portului A a fost deja activat, nu ne rămâne decât să multiplexăm pinul de input de la senzor pe utilizare GPIO (ALT1). Acesta este implicit în mod input.

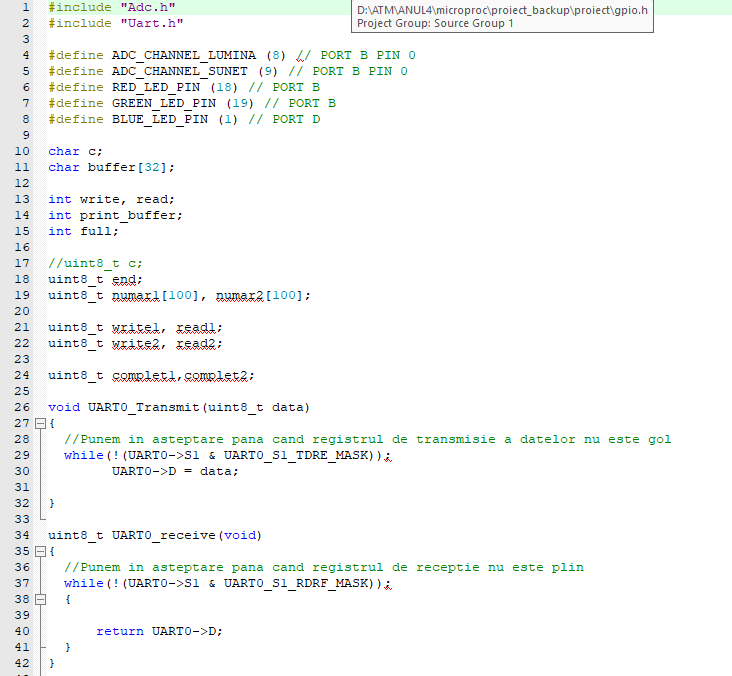


Pentru setarea multiplexării, folosim tot câmpul MUX din registrul PORTx\_PCRy.

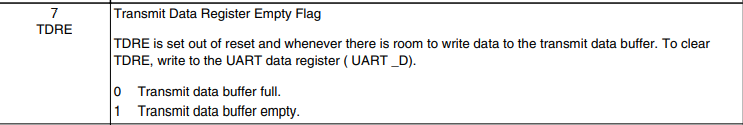


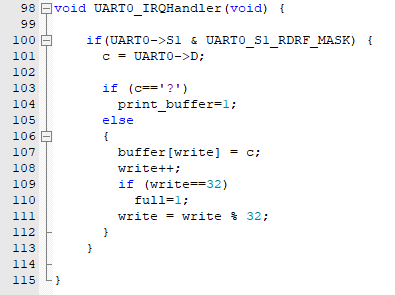
## Transmitere date prin UART

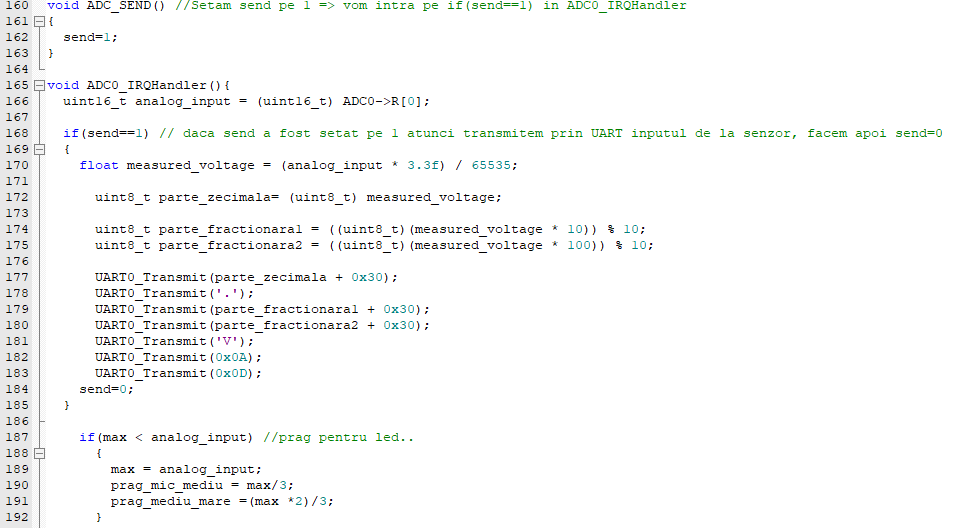
La recepția caracterului ‘a’ funcția ‘ADC\_SEND’ este apelată, aceasta setează variabila statică ‘send’ pe 1. Acest lucru determină ca handler-ul senzorului activ sa intre in rutina noastra de trimitere a datelor unde inputul analog este convertit la ‘float’, urmând să fie transmit folosind uart, iar variabila ‘send’ să fie setata pe zero.



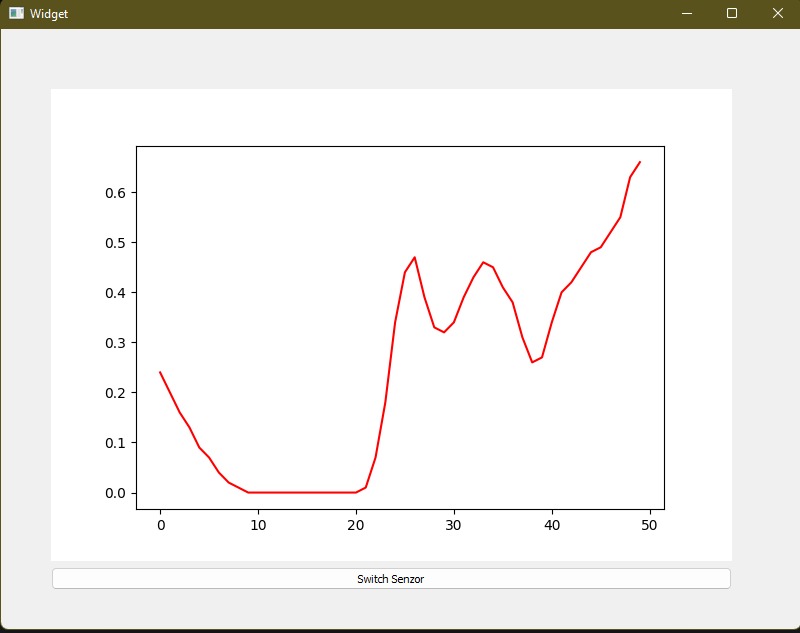
Valoarea octetului dat ca parametru este pus în registrul UART0\_D (UART Data Register). Nu se părăsește funcția până când câmpul TDRE (Transmit Data Register Empty Flag) din registrul UART0\_S1 (UART Status Register 1) este 1, adică bufferul de trimitere s-a golit.

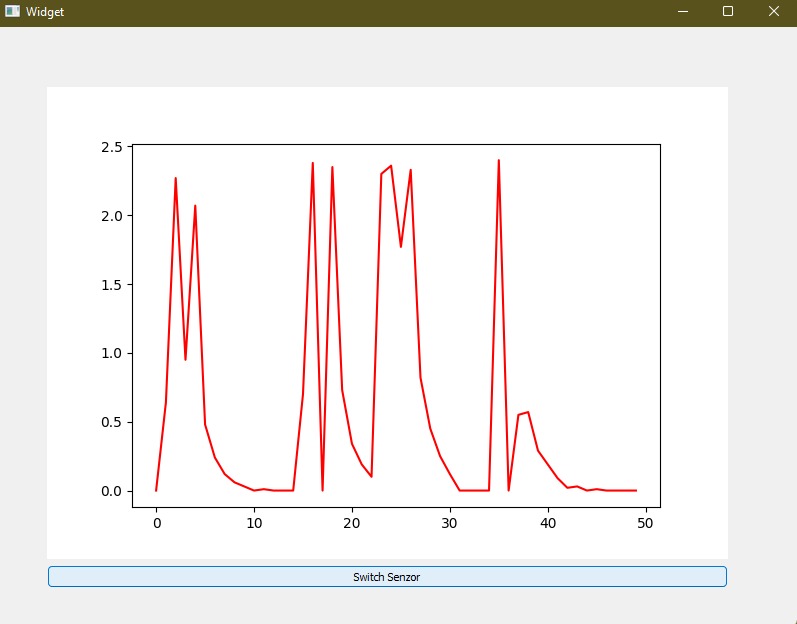






# Rezultate Python

*Figura 9. Exemplu grafic Python (senzorul de lumină)*



*Figura 9. Exemplu grafic Python (senzorul de sunet)*

# Dificultăți întâmpinate

În timpul realizării acestui proiect, am întâmpinat mai multe dificultăți, precum:

* Sincronizarea ratei de citire a senzorului cu rata de transmitere (Soluție: transmitem pe Uart pentru a sincroniza)
* Modalitatea de a schimba senzorul de pe care citim valorile (Soluție: la pornirea aplicației citim de pe senzorul de lumină, urmând ca prin apăsarea butonului “Switch senzor” să trecem la citirea valorilor primite de la senzorul de sunet)
* Creare pragurilor de input și aprinderea LED-urilor corespunzătoare (Soluție: am definit formulele și condițiile respective, fiind prezente în cod)
* Transiterea incorectă de la target la host pe Uart, anume date inutile (Soluție: transmitem date folosind ADC0\_IRQHandler, iar în Adc.h există o variabilă send care, la apelarea funcției ADC\_Send valoarea acesteia devine 1, astfel că ne așteptăm să trimitem date)
* Reinițializarea senzorilor în momentul comutării senzorului de pe care citim valorile (Soluție: folosirea registrului ADC\_SC1\_COCO (conversion complete) ce ne lasă să reinițializăm ADC)

# Referințe

|  |  |
| --- | --- |
| [1] | „DFR0027 Datasheet,” [Interactiv]. Available: https://www.farnell.com/datasheets/2700130.pdf. |
| [2] | https://wiki.dfrobot.com/DFRobot\_Ambient\_Light\_Sensor\_SKU\_DFR0026 |
| [3] | STM, STM32F10xxx/20xxx/21xxx/L1xxxx Cortex®-M3 programming manual, 2017. |
| [4] | https://wiki.dfrobot.com/Analog\_Sound\_Sensor\_SKU\_\_DFR0034 |

1. https://os.mbed.com/platforms/KL25Z/ [↑](#footnote-ref-1)