**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**Facultatea de Sisteme Informatice și Securitate Cibernetică**

**Departamentul de Calculatoare și Securitate Cibernetică**



***Utilizare senzor ANALOGIC SUNET***

***Platforma de dezvoltare frdm-kl25z***

Std. sg. maj. Bianca-Daniela IONASCU

Std. sg. maj. Radu-Marian Toader

Grupa C114D-C114C

**București**

**2023**

Cuprins

[1. Prezentarea senzorului DFR0027 3](#_Toc93045592)

[2. Scop proiect 4](#_Toc93045593)

[3. Conectare senzor – placă de dezvoltare 5](#_Toc93045594)

[4. Descriere program 6](#_Toc93045595)

[4.1. Funcția main 6](#_Toc93045596)

[4.2. Inițializarea modulelor 6](#_Toc93045597)

[4.2.1. Configurarea ceasului 7](#_Toc93045598)

[4.2.2. Configurarea întreruperii de ceas 8](#_Toc93045599)

[4.2.3. Inițializarea modulului UART 10](#_Toc93045600)

[4.2.4. Inițializarea modulului TPM 13](#_Toc93045601)

[4.2.5. Inițializarea modulului GPIO 15](#_Toc93045602)

[4.3. Generare PWM 15](#_Toc93045603)

[4.4. Transmitere date prin UART 17](#_Toc93045604)

[5. Rezultate MATLAB 17](#_Toc93045605)

[6. Dificultăți întâmpinate 17](#_Toc93045606)

[7. Referințe 19](#_Toc93045607)

# Prezentarea senzorului DFR0027

Senzorul DFR0027 este un senzor digital ce detectează vibrațiile cu output active LOW (valoarea 1 înseamnă că nu sunt detectate vibrații, valoarea 0 apare doar la detectarea acestora).



Figura 1. Senzorul DFR0027

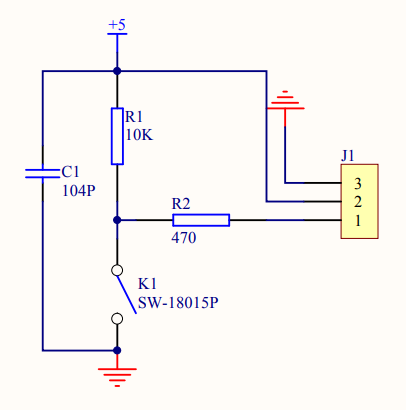


Figura 2. Circuitul senzorului DFR0027

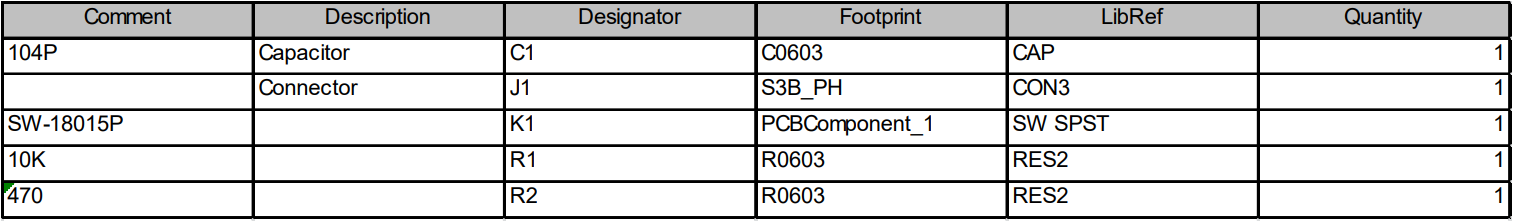


Figura 3. Componentele circuitului

Acesta este format dintr-un condensator de 104 pF și două rezistențe (una de 10KΩ, una de 470Ω. Se conectează la placa de dezvoltare prin trei conectori: unul negru (GND), unul roșu (+5V) și unul verde (OUTPUT). [1]

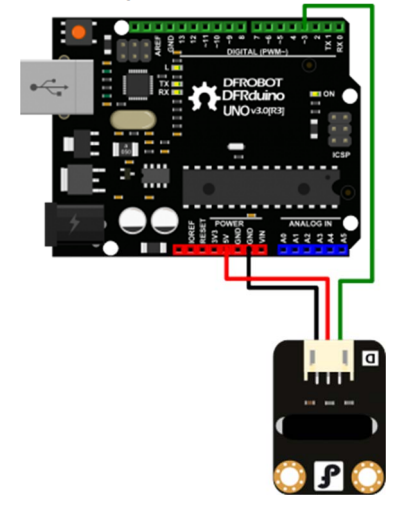


Figura 4. Exemplu conectare senzor

# Scop proiect

Scopul acestui proiect este acela de a modifica intensitatea LEDului prezent pe placa de dezvoltare pusă la dispoziție pe baza ieșirii senzorului de vibrații, printr-un semnal PWM generat.

Se va dezvolta un program care, cu frecvența de 1kHz (o dată la 1 ms), va verifica ieșirea senzorului conectat. O dată la 50 ms, în funcție de câte valori de 0 (vibrație detectată) a contorizat programul, se generează un semnal PWM cu factorul de umplere direct proporțional cu procentul de timp în care senzorul a detectat vibrații. Acest semnal PWM va fi conectat către LEDul incorporat și va modifica astfel intensitatea acestuia în funcție de factorul de umplere.

De asemenea, se va transmite prin UART către PC , o dată la 10 ms, valoarea ieșirii senzorului pentru a se putea realiza un grafic în timp real prin intermediul programului MATLAB în care să se observe detecția vibrațiilor.

# Conectare senzor – placă de dezvoltare

Vom conecta senzorul astfel:

* Firul verde – gri conectează senzorul la PTA5
* Firul roșu – alb conectează senzorul la 5V
* Firul negru – negru conectează senzorul la GND

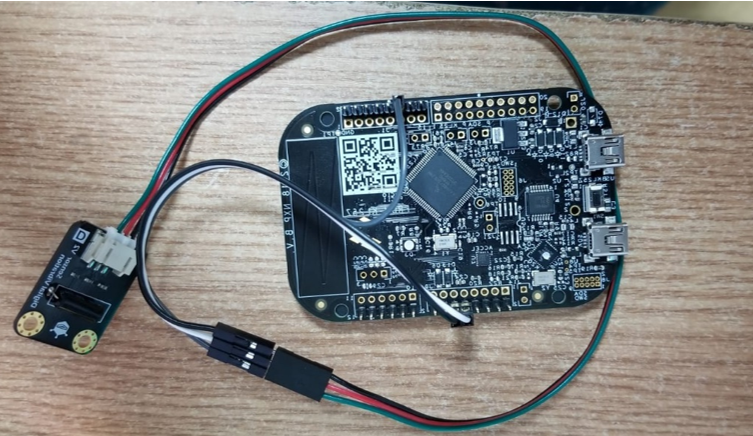


Figura 5. Conectare senzori

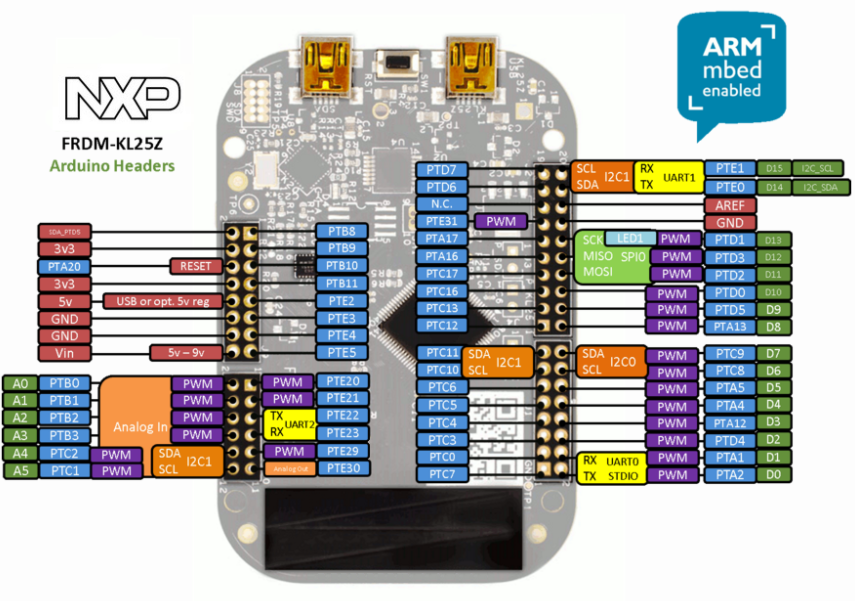


Figura 6. Pini placă dezvoltare[[1]](#footnote-1)

# Descriere program

## Funcția main

În fișierul main.c am inclus fișierele header în care sunt declarate funcții și variabile ce urmează a fi folosite: *init.h* (funcția init), *uart.h* (funcția send\_data), *interrupts.h* (variabilele sensor\_total, flag\_10msec și flag\_50msec), *controlLED.h* (funcția control\_LED) și fișierul header generat de către mediul de dezvoltare Keil, *MKL25Z4.h*, specific plăcii de dezvoltare.

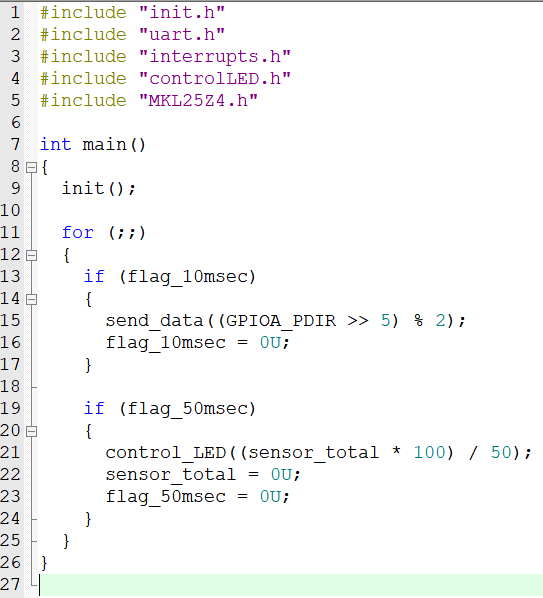


Figura 7. Funcția main

Logica principală a programului este următoarea: întâi se apelează funcția de inițializare a modulelor, apoi, într-un ciclu infinit, se verifică trecerea a 10 ms (când se va trimite valoarea de output a senzorului prin UART) și a 50 ms (când se va schimba intensitatea luminoasă, calculându-se o proporție dintre numărul total de măsurări ce au detectat vibrații și numărul maxim ce ar fi putut fi detectat, 50 – pentru că se măsoară o dată la 1 ms).

## Inițializarea modulelor

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

* SYSTEM\_CLOCK\_FREQ – frecvența de ceas a sistemului, folosită în calcule; valoarea acesteia este de 48 MHz [2]
* SYSTICK\_TIME\_INTERVAL – valoarea la care vom seta întreruperea de ceas, 1 ms
* SYSTICK\_TIME\_FREQ – frecvența asociată intervalului de întrerupere (1/1ms = 1 \* 1000 = KHz
* SYSTICK\_TMR\_RELOAD\_VAL – valoarea de resetare a timerului, raportul dintre cele două frecvențe – 1 (numărarea începe de la 0)

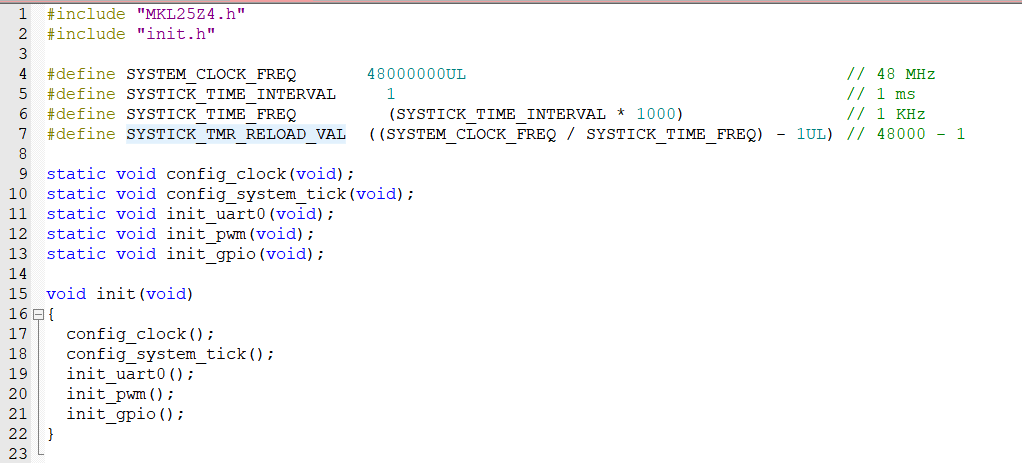
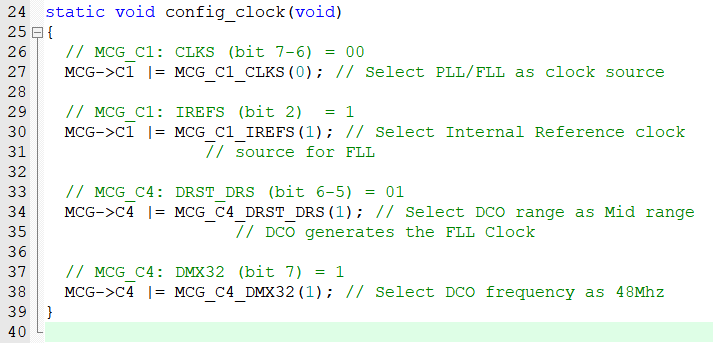


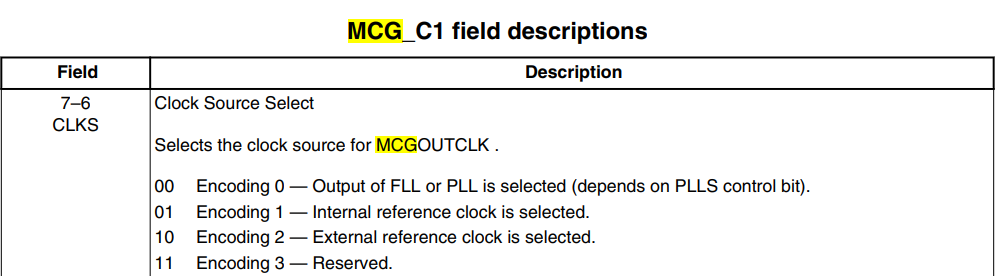
Figura . Funcția init

### Configurarea ceasului

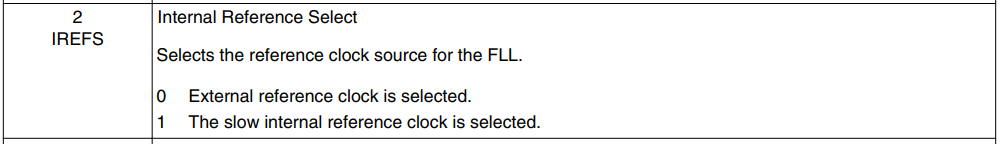
Modulul MCG (Multipurpose Clock Generator) oferă semnale de ceas pentru unitatea centrală. Acesta conține un FLL (frequency-locked loop), controlabil de către un ceas de referință intern.



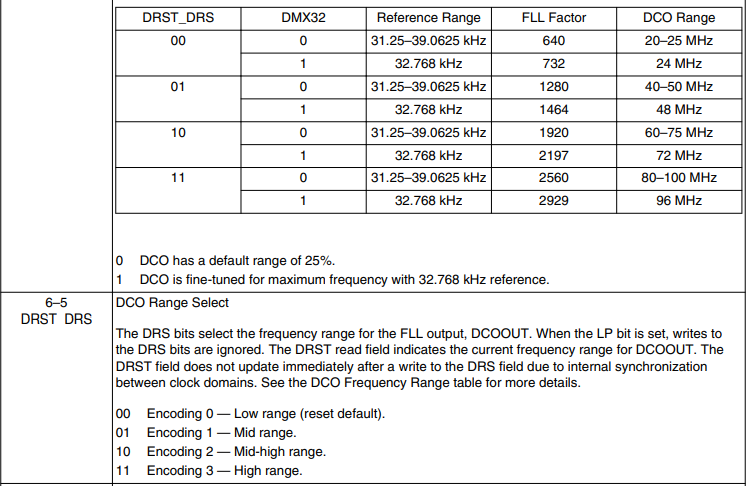
În MCG\_C1 (MCG Control 1 Register) am setat câmpul CLKS (Clock Source Select, biții 7-6) pe valoarea 0b00 pentru selectarea FLL/PLL ca output.



Cel folosit va fi selectat prin câmpul IREFS (Internal Reference Select, bitul 2) pe valoarea 1 – se selectează ceasul intern de referință, așadar se folosește FLL (PLL – phase-locked loop este configurabil doar extern).

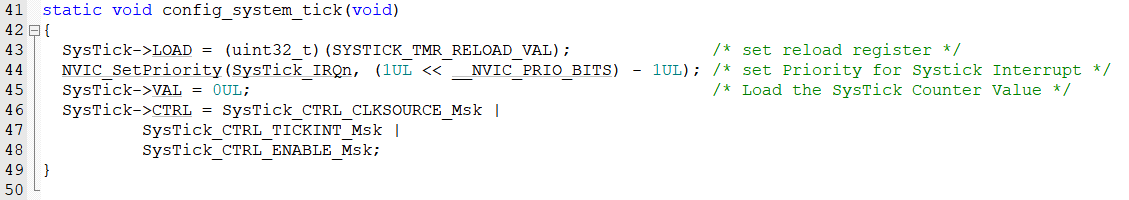


În MCG\_C4 (MCG Control 4 Register), setăm câmpul DRST DRS (biții 6-5) pe valoarea 0b01 – se selectează frecvența DCO (Digitally-controlled oscillator) ca fiind una medie (40 – 50 MHz).

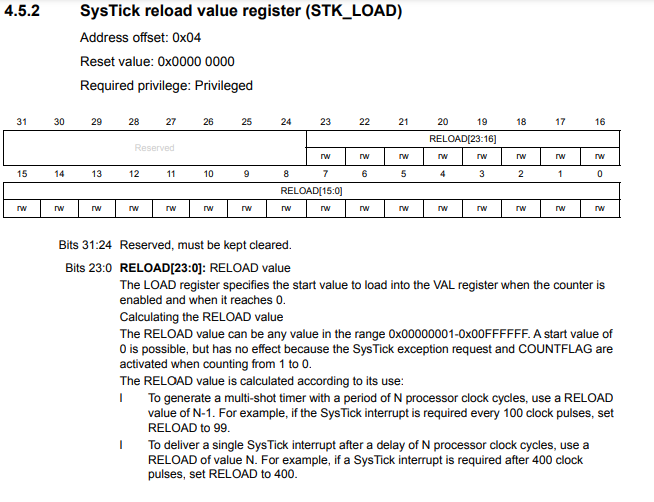


În același registru setăm câmpul DMX32 (DCO Maximum Frequency, bitul 7) pe valoarea 1 – DCO va avea frecvența de 48 MHz. [2]

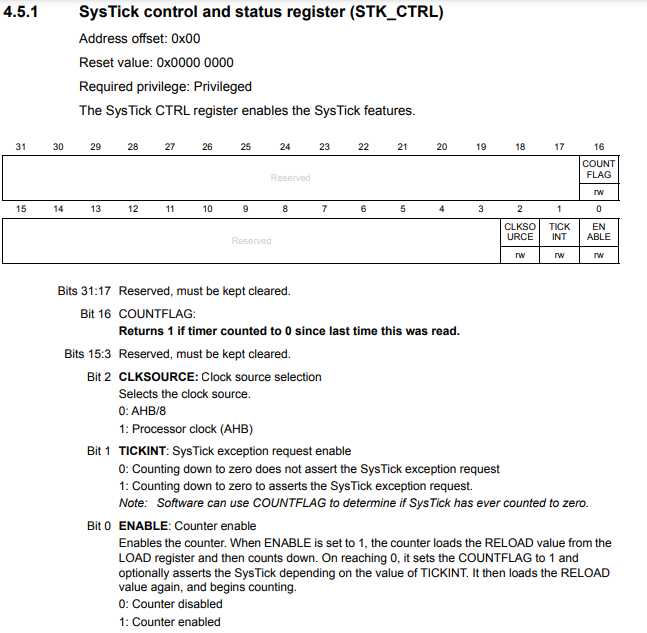
### Configurarea întreruperii de ceas



Setăm valoarea registrului de reload (STK\_LOAD) cu valoarea calculată în macro, care specifică intervalul de numărare al counterului.

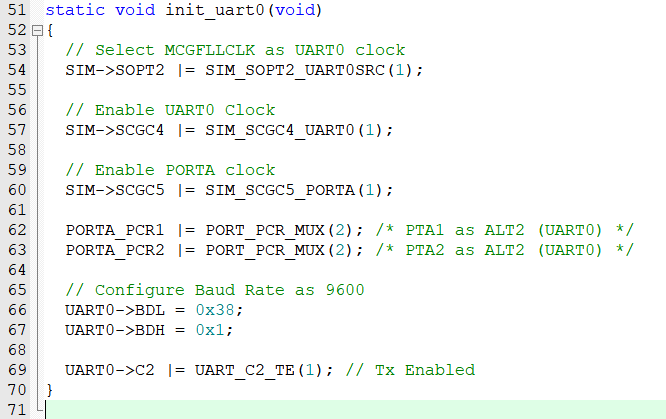


Setăm prioritatea întreruperii SysTick pe valoarea maximă și valoarea inițială pe 0. În registrul STK\_CTRL (registrul de control și stare SysTick) setăm biții 2 (CLKSOURCE – selectează ca sursă ceasul procesorului), 1 (TICKINT – activează întreruperea la valoarea 0) și 0 (ENABLE – pornește timerul). [3]

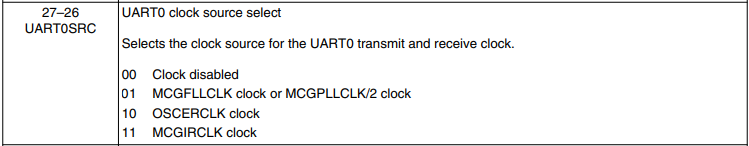


### Inițializarea modulului UART

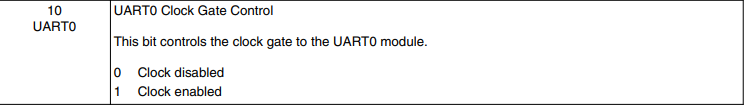
Vom folosi modulul UART0 pentru comunicația serială cu PC prin cablul USB.



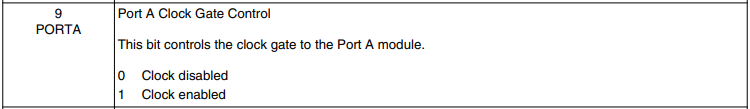
În registrul SIM\_SOPT2 (System Options Register 2) setăm pe 0b01 câmpul UART0SRC (biții 27-26) pentru selectarea ca ceas al modulului MCGFLLCLK anterior configurat.



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



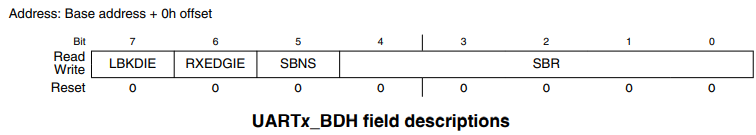
În registrul SIM\_SCGC5 (System Clock Gating Control Register 5) setăm pe 1 câmpul PORTA (bitul 9) pentru activarea ceasului acestui port.

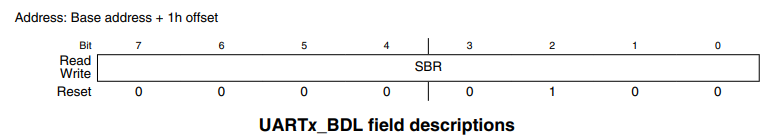


În regiștrii de control ai pinilor 1 și 2 din portul A (PORTA\_PCR1/2), setăm câmpul MUX (biții 10-8) pe valoarea 0b010, care înseamnă folosirea acestora în modulul de UART0 (RX/TX).

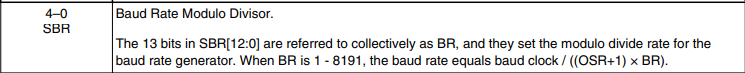


Configurăm baud rate la 9600 astfel: punem în câmpul UART0\_BDH valoarea 0x1, iar în UART0\_BDL valoarea 0x38. Primii 13 biți din cei doi octeți concatenați reprezintă valoarea SBR – baud rate setting = 0x138 = 312.

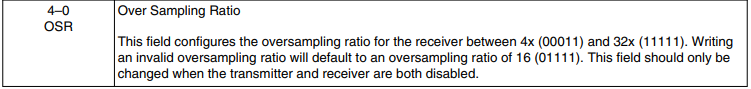




Formula după care se calculează baud rate este: baud clock / ((OSR+1) × BR) (pentru SBR între 1 – 8191).

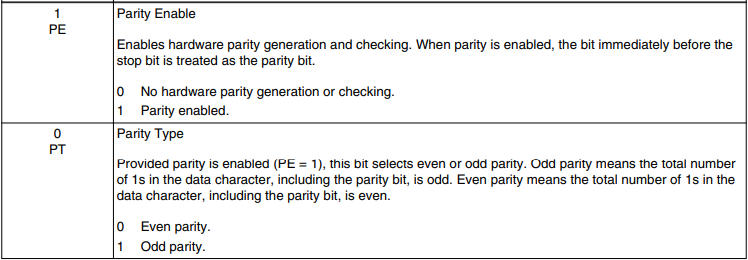


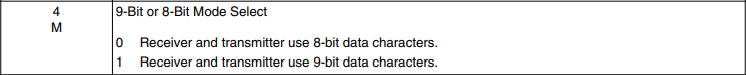
OSR (Over Sampling Ratio) are valoarea implicită 0b01111 adică 15, iar baud clock a fost setat la cel de 48 MHz.

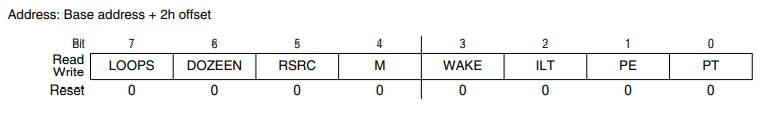


Valoarea baud rate este așadar: 48000000 / (16 × 312) aproximativ 9600.

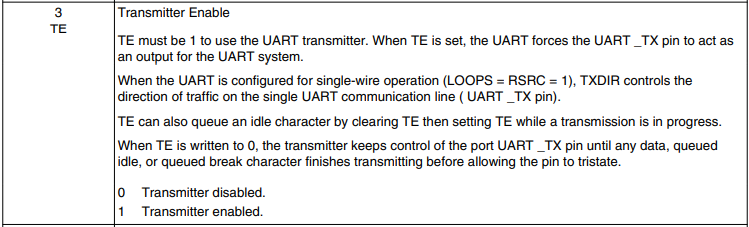
Portul serial este setat implicit cu 8 biți date, niciun bit de paritate și un bit de stop.





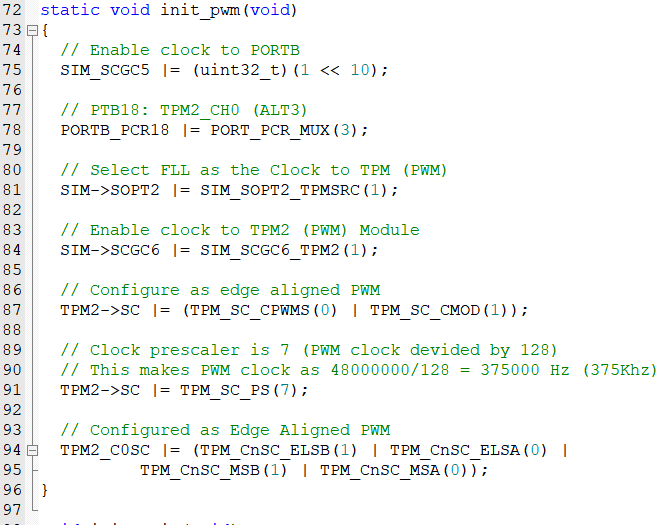


Activăm transmiterea prin UART prin setarea câmpului TE (bitul 3, Transmitter Enable) din UART0\_C2 pe valoarea 1.



### Inițializarea modulului TPM

Vom folosi modulul TPM (Timer/PWM Module) pentru generarea semnalului PWM.



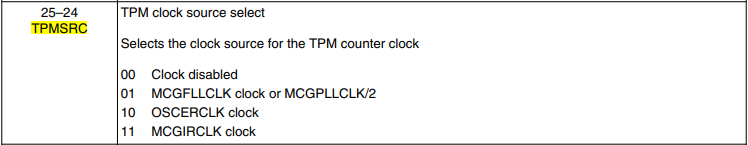
Vom porni ceasul portului B, unde se află LEDul incorporat (culoarea roșie), prin setarea bitului 10 al registrului SIM\_SCGC5 pe valoarea 1.



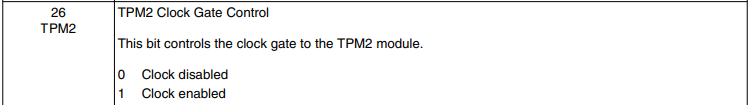
Setăm multiplexarea pinului 18 din portul B pe ALT3, adică canalul 0 al modulului 2 TPM, prin setarea câmpului MUX pe valoarea 0b011.



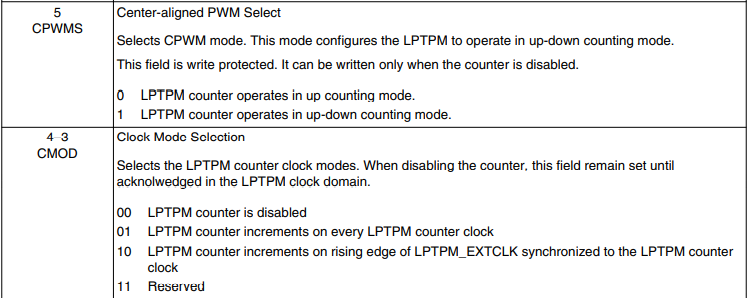
Setăm FLL ca sursă de ceas prin setarea câmpului TPMSRC (biții 25-24, TPM clock source select) din registrul System Options Register 2 (SIM\_SOPT2) pe valoarea 0b01.



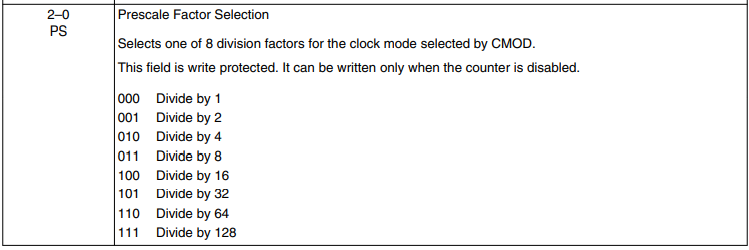
Activăm ceasul modului 2 TPM prin setarea câmpului TPM2 (bitul 26) din SIM\_SCGC6 pe 1.



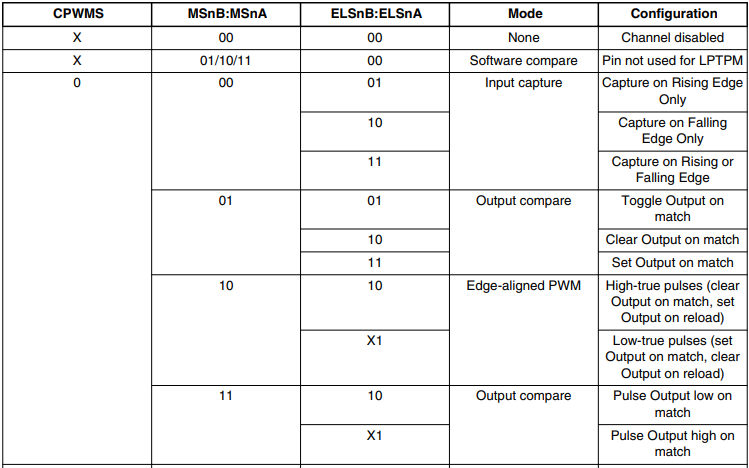
Setăm semnalul PWM generat ca fiind edge-aligned, setând counterul în mod up counting prin valoarea 0 în câmpul CPWMS (bitul 5, Center-aligned PWM Select) și valoarea 0b01 în câmpul CMOD (biții 4-3, Clock Mode Selection) pentru incrementarea numărătorului LPTPM.



Prin setarea câmpului PS (biții 2-0, Prescale Factor Selection) pe valoarea 0b111, ceasul sistemului va fi divizat cu 128, deci ceasul PWM va avea frecvența de 375 KHz.

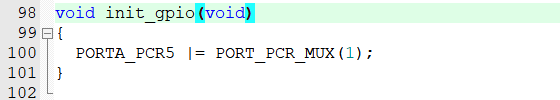


În registrul TPM2\_C0SC (Channel 2 Status and Control) setăm următoarele câmpuri: ELSB și MSB pe 1, ELSA și MSA pe 1. Modulul va fi configurat edge-aligned PWM.



### 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.

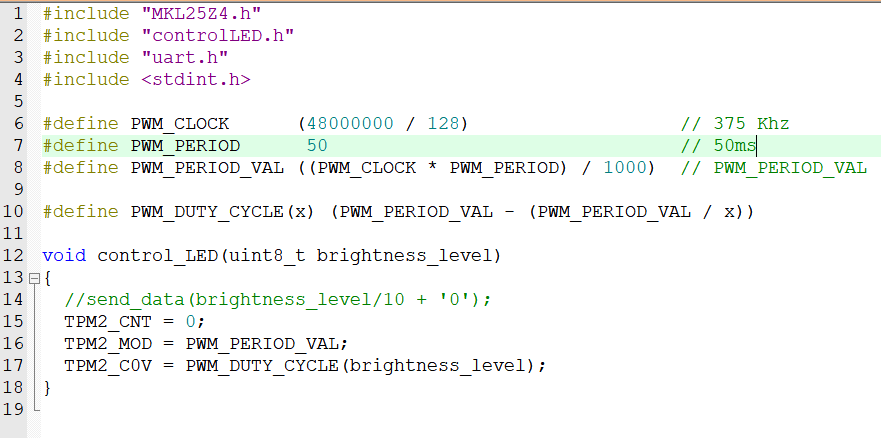


Pentru setarea multiplexării, folosim tot câmpul MUX din registrul PORTx\_PCRy.



## Generare PWM

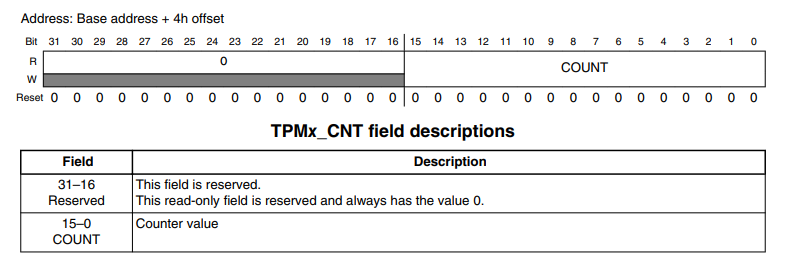
După cum am menționat, în funcția main, atunci când flagul ce marchează trecerea a 50 ms devine 1 după incrementarea acestuia în funcția ce se ocupă de întreruperea de ceas, SysTick\_Handler, este apelată funcția controlLED, ce primește ca parametru o valoare între 0 și 99, direct proporțională cu raportul de detectare vibrații/măsurări totale în interval.

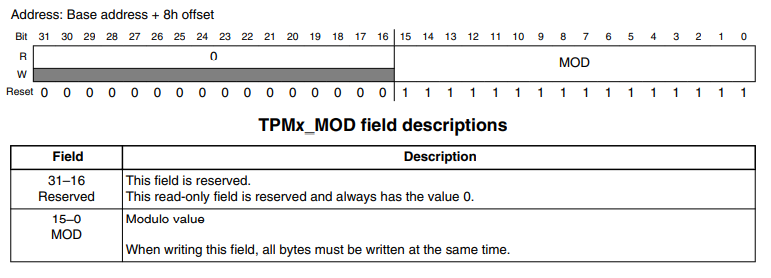


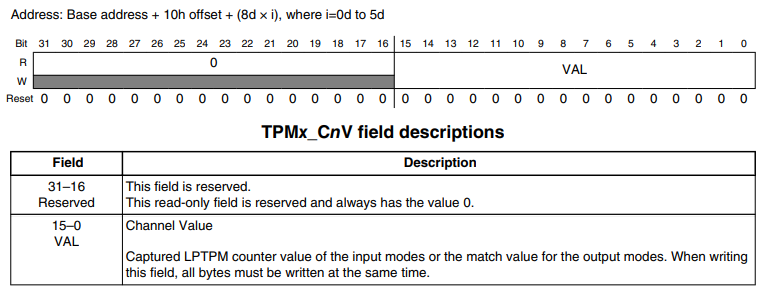
În controlLED.c avem următoarele macrodefiniții:

* PWM\_CLOCK – frecvența ceasului PWM (frecvența de ceas / 128 – prescaler)
* PWM\_PERIOD – perioada dorită a semnalului PWM – 50 ms
* PWM\_PERIOD\_VAL – valoarea până la care trebuie să numere pentru a termina o perioadă – frecvența ceasului \* perioada semnalului / 1000 (pentru conversia din ms)
* PWM\_DUTY\_CYCLE(X) – de la cât să înceapă numărătoarea pentru modificarea factorului de umplere (1-X/X)

La fiecare 50 ms, în funcție de raportul calculat, modificăm regiștrii: TPM2\_CNT (resetăm numărătorul), TPM2\_MOD (setăm valoarea până la care numărăm), TMP2\_C0V (setăm valoarea de la care pornim). [4]

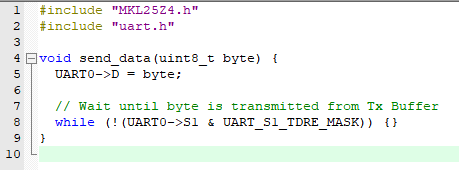




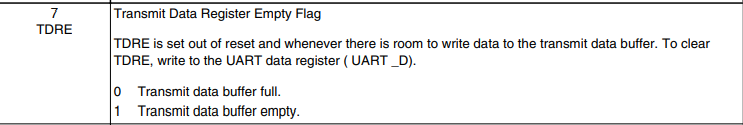


## Transmitere date prin UART

O dată la 10 ms, când flagul corespunzător devine, este apelată funcția send\_data ce primește ca parametru valoarea primită de la senzor.



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 MATLAB

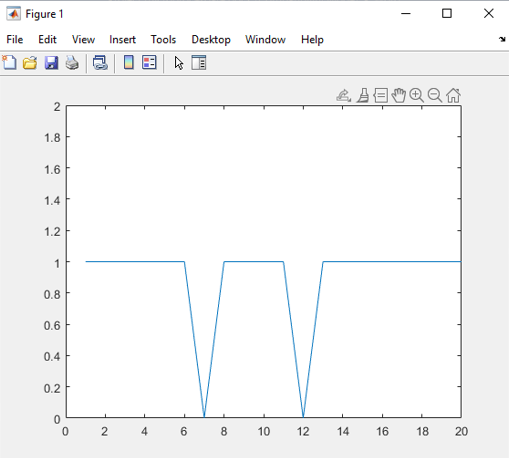


Figura . Exemplu grafic MATLAB

# Dificultăți întâmpinate

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

* Folosirea inițială a pinului PTA20 (din considerente de a minimiza porturile folosite și de a grupa cablurile) pentru primirea inputului de la senzor, dar nu am reușit să citim de pe acesta întrucât nu am putut să îi dezactivăm funcționalitatea implicită de reset
* Omiterea activării ceasului portului B pentru funcționarea LEDului

# Referințe

|  |  |
| --- | --- |
| [1] | „DFR0027 Datasheet,” [Interactiv]. Available: https://www.farnell.com/datasheets/2700130.pdf. |
| [2] | Freescale Semiconductor, Inc., KL25 Sub-Family Reference Manual, 2012. |
| [3] | STM, STM32F10xxx/20xxx/21xxx/L1xxxx Cortex®-M3 programming manual, 2017. |
| [4] | „Using PWM module of KL25Z Series MCU to Control LED Brightness,” [Interactiv]. Available: https://learningmicro.wordpress.com/controlling-led-brightness-using-pwm/. [Accesat 05 01 2022]. |

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