**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**Facultatea de Sisteme Informatice și Securitate Cibernetică**



***FILTRAREA DATELOR PROVENITE DE LA SENZORUL DE LUMINă AMBIENTALă PRIN SETAREA MANUAlă de PRAGURI***

Std. sg. maj. Mirela URSE

Std. sg. maj. Raul VEGA

Std. sg. maj. Cosmin-Ioan ROȘU

Grupa C114E

**București**

**2023**

Cuprins

[1. Prezentarea senzorului DFR0026 (Light) 3](#_Toc123135441)

[2. Prezentarea senzorului DFR0054 (Rotation) 4](#_Toc123135442)

[3. Scop proiect 5](#_Toc123135443)

[4. Conectare senzor – placă de dezvoltare 6](#_Toc123135444)

[5. Model Tinkercad 8](#_Toc123135445)

[6. Descriere program Keil 9](#_Toc123135446)

[6.1. Funcția main 9](#_Toc123135453)

[6.2. Inițializarea modulului UART 9](#_Toc123135454)

[6.3. Inițializarea modulului ADC 11](#_Toc123135455)

[6.4. Inițializarea modulului GPIO 13](#_Toc123135456)

[6.5. Functia de intrerupere pentru modulul ADC 15](#_Toc123135457)

[6.6. Prelucrarea datelor corespunzatoare senzorilor 15](#_Toc123135458)

[6.7. Schimbarea canalului de primire a semnalului 16](#_Toc123135459)

[6.8. Transmitere de date folosind modulul UART 17](#_Toc123135460)

[7. Descriere program Python 19](#_Toc123135461)

[7.1. Initializare instanta seriala 19](#_Toc123135463)

[7.2. Citire date de la portul serial 20](#_Toc123135464)

[7.3. Initializare clasa MainWindow 21](#_Toc123135465)

[7.4. Actualizare valori grafice 22](#_Toc123135466)

[8. Rezultate PyQt 23](#_Toc123135467)

[9. Dificultăți întâmpinate 24](#_Toc123135468)

[10. Referințe 25](#_Toc123135469)

# Prezentarea senzorului DFR0026 (Light)

Senzorul DFR0026 este un senzor analogic de lumină ambientală ce ajută la detectarea densității luminii. Pe măsură ce lumina care cade pe senzor crește, ieșirea analogică (SEMNAL) va crește. Dacă lumina este foarte scăzută, atunci SIG-ul va fi de asemenea foarte scăzut.

A close-up of a computer chip

Description automatically generated with medium confidence

Figura 1. Senzorul DFR0026

**Specificatii:**

* Supply Voltage: 3.3V to 5V
* Illumination range : 1 Lux to 6000 Lux
* Responsive time : 15us
* Interface: Analog
* Size:22x30mm

Graphical user interface

Description automatically generated

Figura 2. Exemplu de conectare senzor

Se conectează la placa de dezvoltare prin trei conectori: unul gri (GND), unul roșu (+5V) și unul verde (A0).

# Prezentarea senzorului DFR0054 (Rotation)

Senzorul DFR0054 este un senzor analogic de rotație ce se bazează pe potențiometru de precizie mulți-turn. Se poate roti până la 10 ture. Dacă senzorul primește un input egal cu o tensiune de 5 V, atunci cei 5000 mV vor fi împărțiți în 3600 de porțiuni (10 ture).

A picture containing electronics, camera

Description automatically generated

Figura 1. Senzorul DFR0054

**Specificatii:**

* Arduino compatible
* Operating Voltage: +3.3-5V DC
* Signal Type: analog signal
* Size: 50x22mm
* Tnterface Mode: Gravity interface (PH2.0-3)
* Rotation Angle: 3600 degrees
* Weight: 20g

Diagram

Description automatically generatedSe conectează la placa de dezvoltare prin trei conectori: unul gri (GND), unul roșu (+5V) și unul albastru (Analog Signal Output).

Figura 2. Exemplu de conectare senzor

# Scop proiect

Scopul acestui proiect este acela de a filtra datele obținute de la senzorul de lumina ambientală prin setarea unor praguri obținute din acționarea senzorului de rotație.

Se va dezvolta un program care, cu ajutorul led-ului RGB integrat în plăcută, se va afișa plaja de valori recepționate de la cei doi senzori.( albastru – dacă sare de pragul de 6000)

De asemenea, se va trimite prin UART către PC valoarea ieșirii senzorului, iar prin intermediul framework-ului PyQt se va realiza o interfață, ce va prezenta un grafic în timp real pentru cei doi senzori asignati temei astfel:

Un grafic cu cele trei zone de culori:

* verde pentru valori mici ( între 0 si 2000 )
* galben pentru valori medii ( între 2000 si 4000)
* rosu pentru valor mari ( între 4000 si 6000)

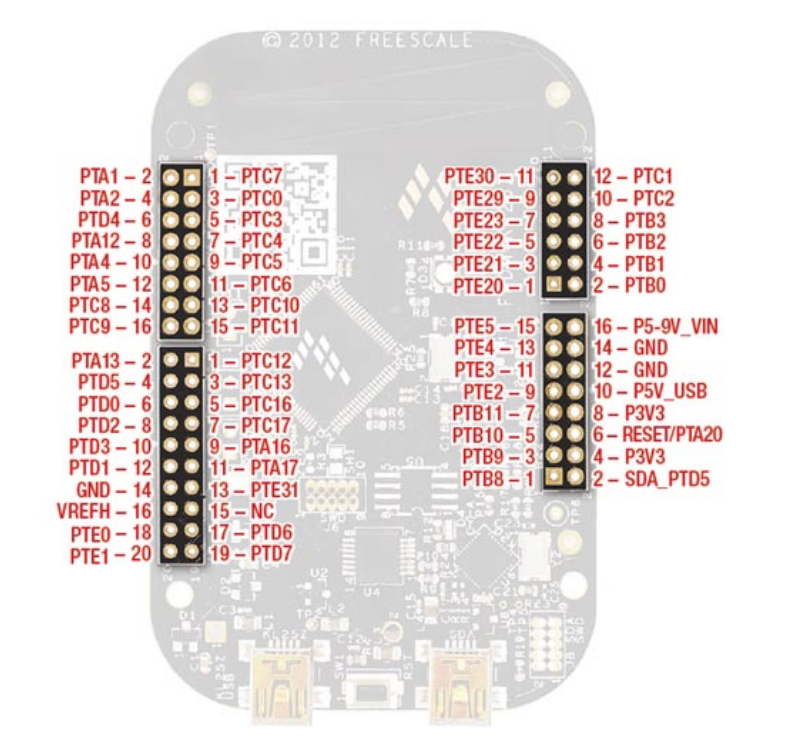
Un grafic cu valoarea de prag a senzorului de rotație.

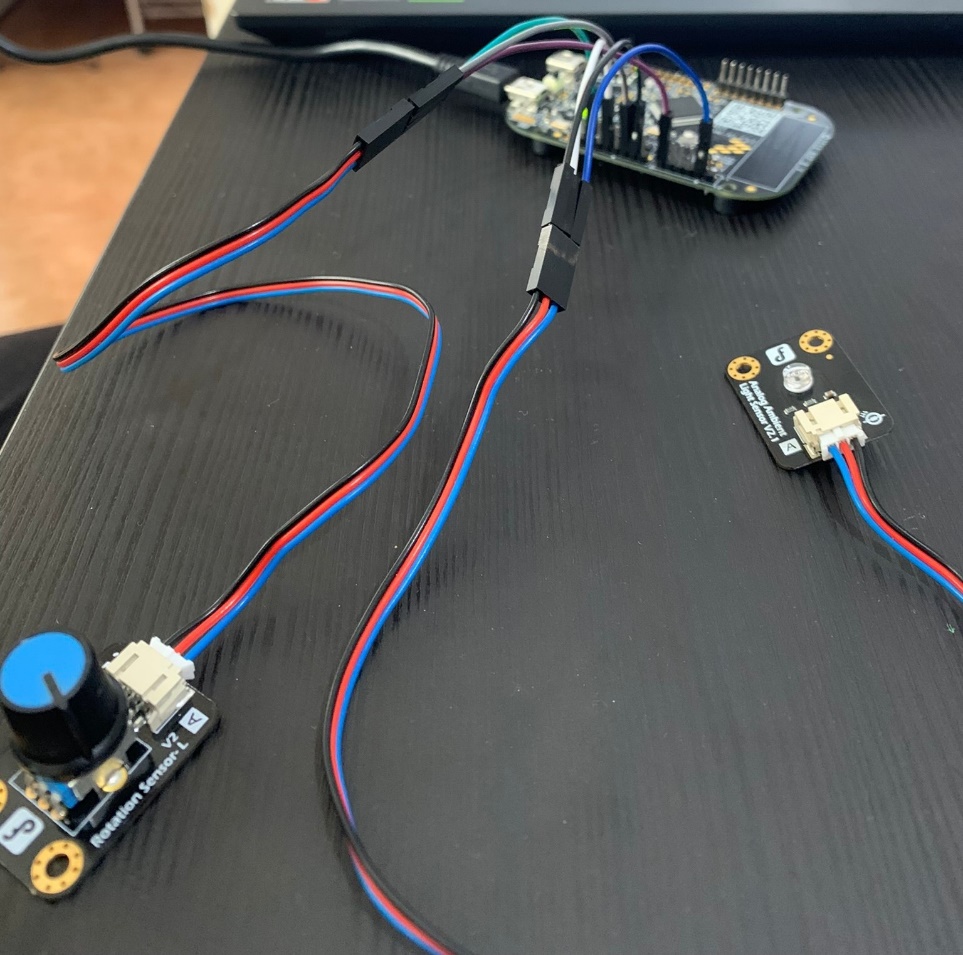
# Conectare senzor – placă de dezvoltare

Vom conecta cei doi senzori astfel:

* Firul albastru – albastru conecteaza senzorul de lumina la PTC2
* Firul rosu – alb conecteaza senzorul de lumina la 3.3 V
* Firul negru – negru conecteaza senzorul de lumina la GND
* Firul albastru – mov conecteaza senzorul de rotatie la PTB0
* Firul rosu – verde conecteaza senzorul de rotatie la 3.3 V
* Firul negru – gri conecteaza senzorul de rotatie la GND

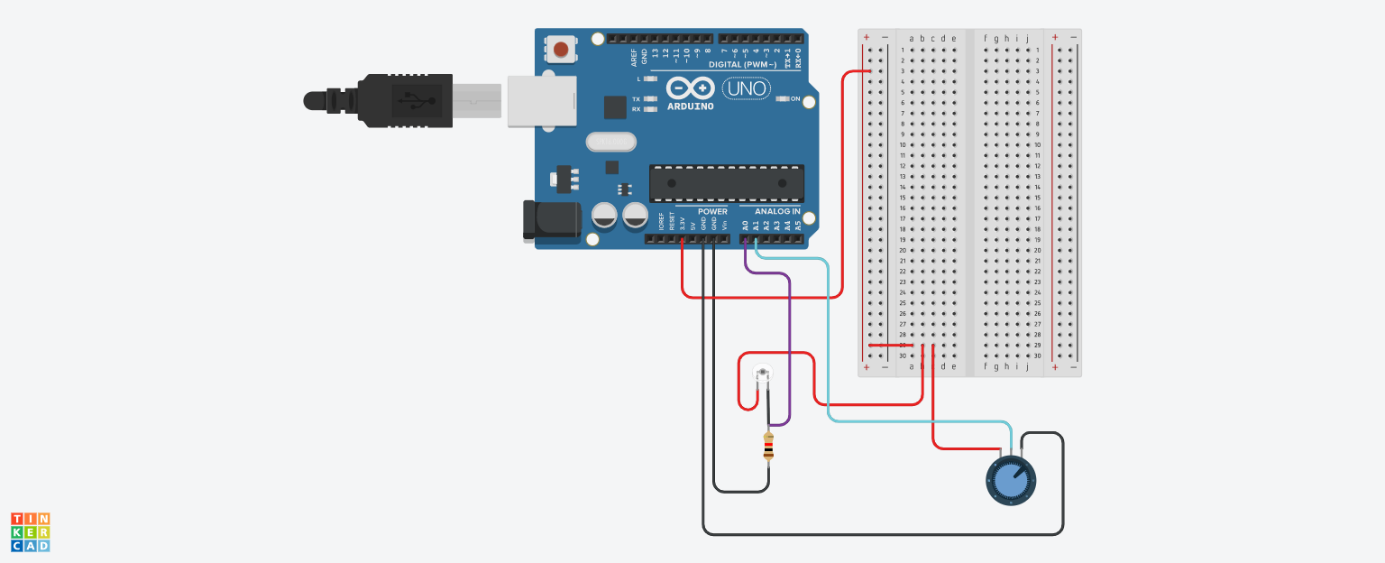






# Model Tinkercad

Am realizat un model asemanator celui real, inlocuind microcontroller-ul nostru cu o placa Arduino. Restul componentelor au ramas la fel.

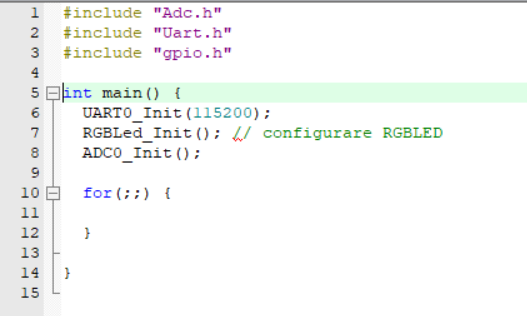


# Descriere program Keil



## Funcția main

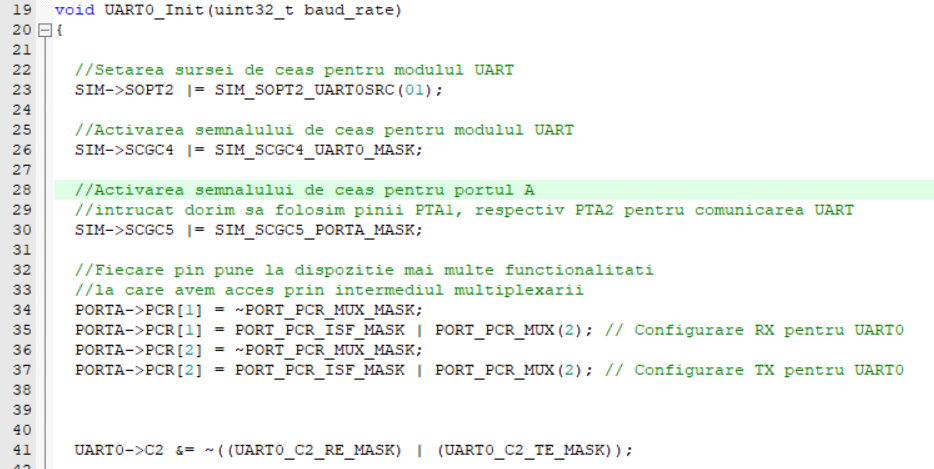
In fisierul main.c am inclus fisierele header cu declaratiile functiilor pentru initializarea modulelor specifice. De asemenea am apelat functiile de initializare (UART, LED, ADC):



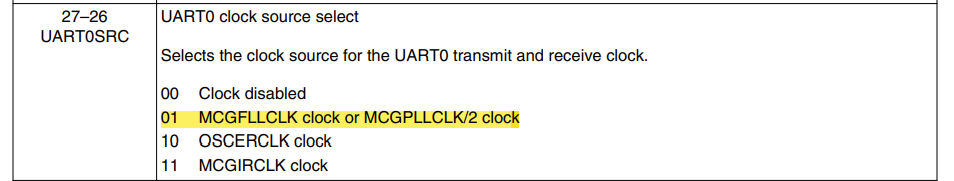
Nu este necesar apelul unei functii de citire date deoarece strategia utilizata presupune folosirea intreruperilor: in momentul in care senzorul emite semnal, este generata o intrerupere, vom trata intreruperea respectiva si apoi vom schimba canalul de pe care citim (ADC channel)

## Inițializarea modulului UART

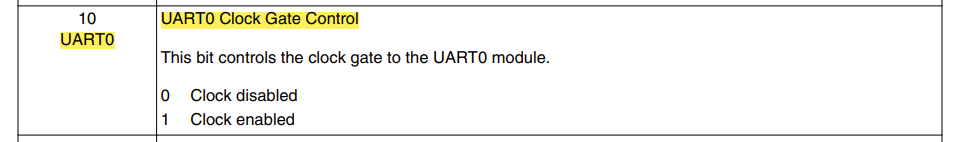
In fisierul uart.c am initializat modulul uart astfel:



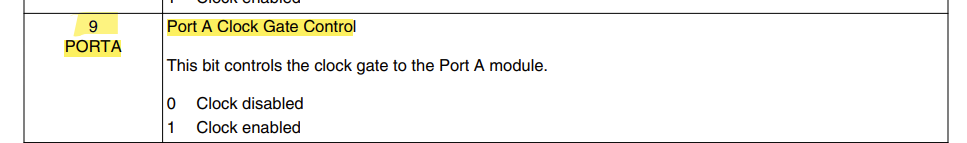
* In registrul SIM\_SOPT2 setam pe ‘01’ valoarea bitilor 26-27 pentru selectarea ca ceas al modulului MCGFLLCLK



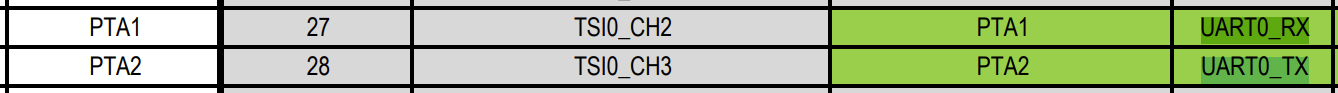
* In registrul SIM\_SCGC4 (system clock gating control) activam semnalul de ceas pentru modulul UART

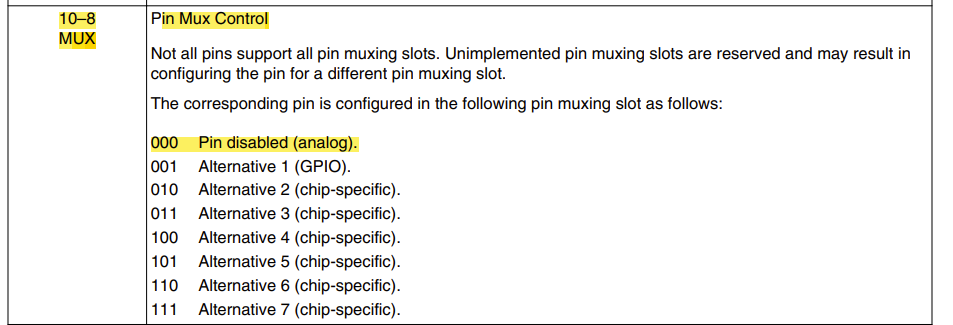


* In registrul SIM\_SCG5 activez semnalul de ceas pentru portul A deoarece doresc sa utilizez pinii PTA1 si PTA2 pentru comunicarea UART

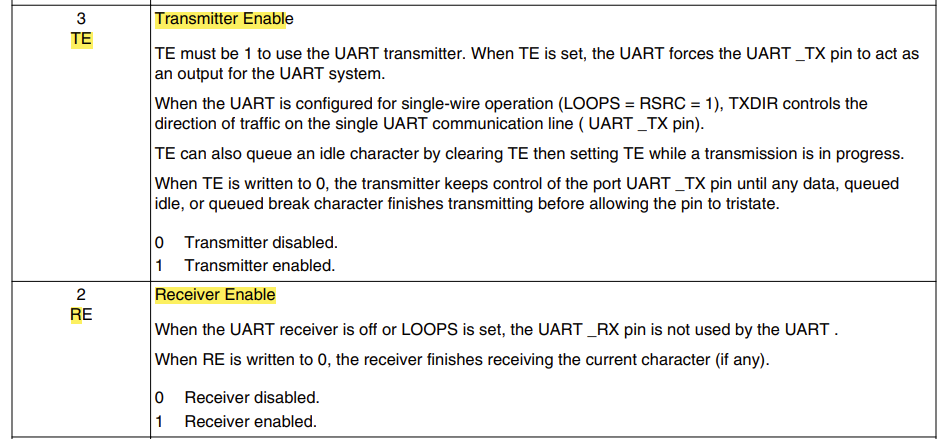


* In registrul PORTx\_PCRn mai intai dezactivam pinul (setam bitii MUX pe 0) si apoi ii vom configura pentru optiunea dorita (ALT2 in cazul PTA1 si PTA2 reprezinta UART0\_RX respectiv UART0\_TX)



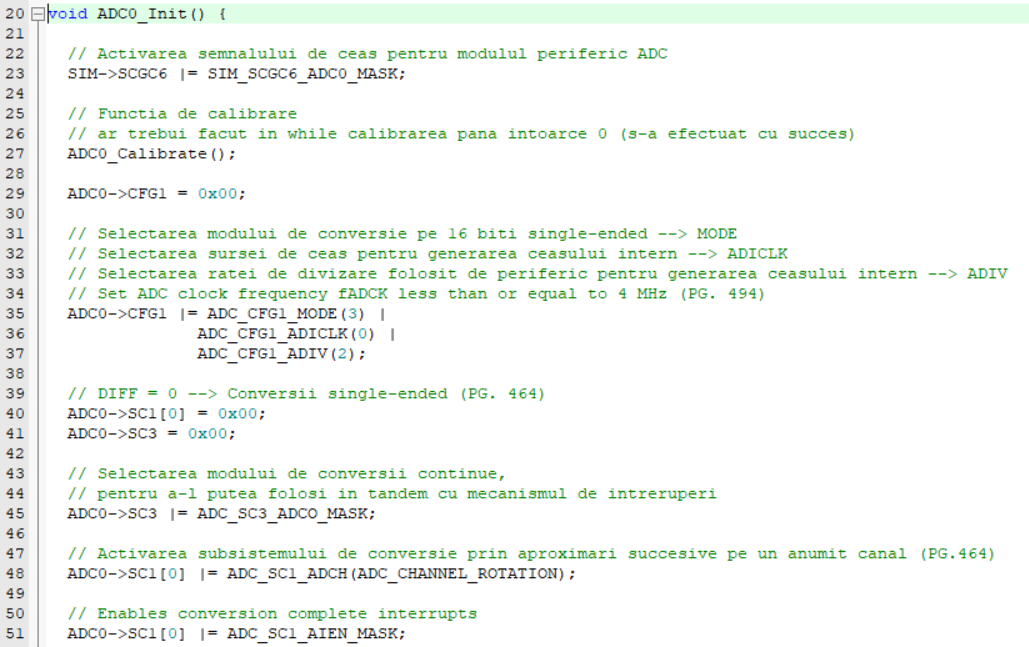


* Ulterior dezactivam registrul UART0\_C2 pentru a configura baud rate-ul si apoi il vom reactiva la final



## Inițializarea modulului ADC

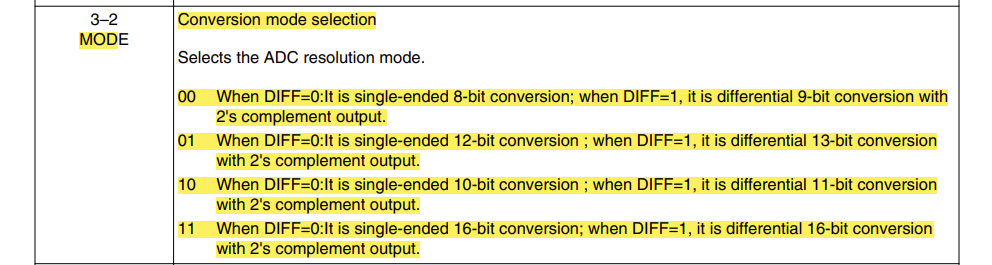
Avand in vedere folosirea celor doi senzori analogici, trebuie sa citim datele transmise de acestia. Pentru a putea face asta vom initializa modulul ADC (analog to digital).

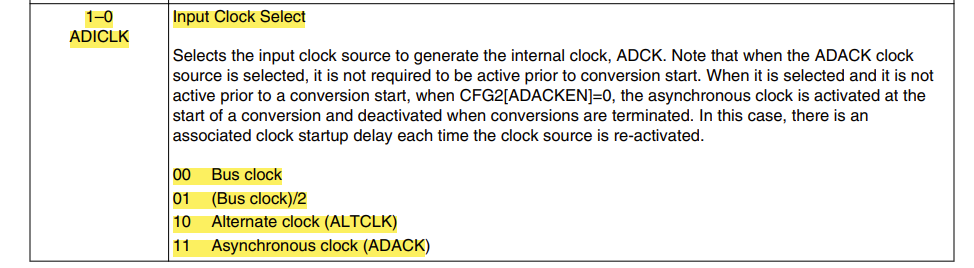


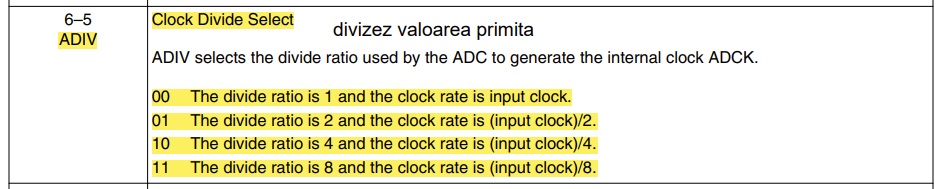
* Mai intai activam semnalul de ceas pentru modulul periferic ADC



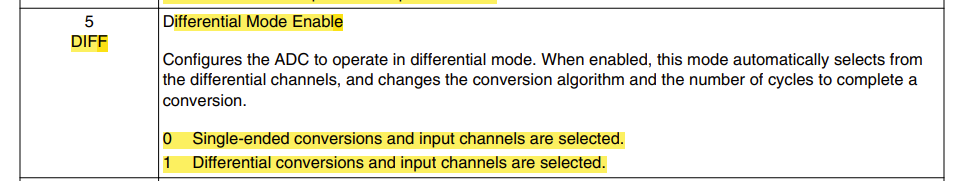
* Apoi apelam functia de calibrare pentru a avea o reproducere cat mai fidela a inputului generat (fara offset)
* Selectam modul de conversie (16 biti SE), sursa de ceas pentru ceasul intern si a ratei de divizare



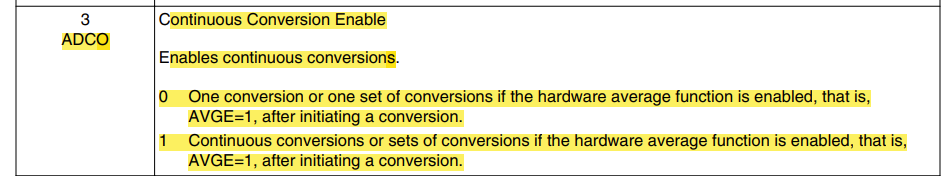




* Setam registrul ADC0\_SC1n cu valoarea 0 pe bitii corespunzatori lui DIFF pentru a selecta modul de conversie single-ended

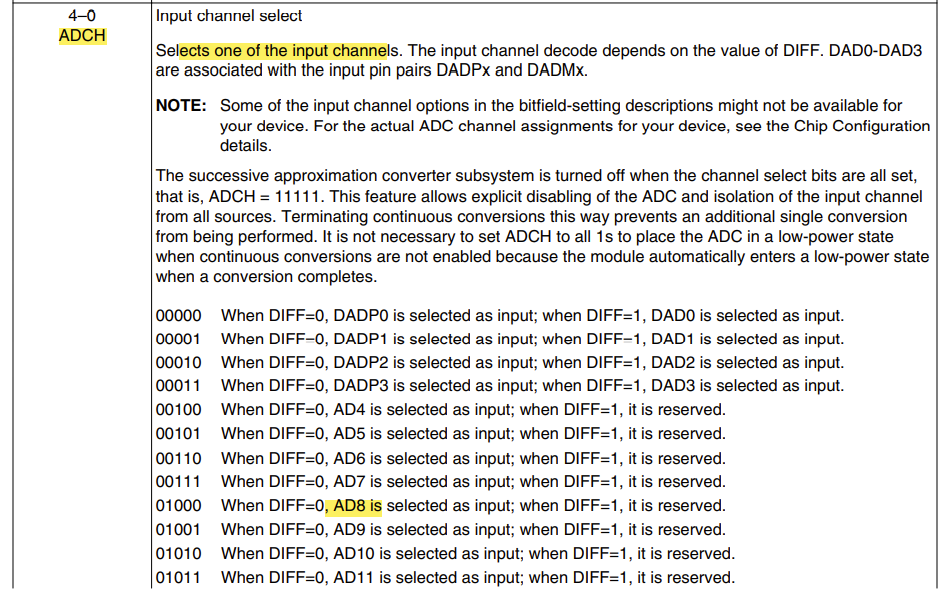


* Selectam modul de conversii continue pentru a-l putea folosi in acelasi timp cu mecanismul de intreruperi -> setam pe 1 valoarea bitului 3 din registrul ADC0\_SC3

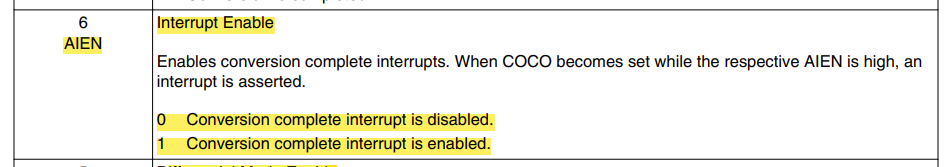


* Vom activa subsistemul de conversii pe un anumit canal: initial vom activa pe canalul corespunzator senzorului de rotatie deoarece la prima intrerupere dorim sa aflam valoarea pragului setat prin senzorul de rotatie



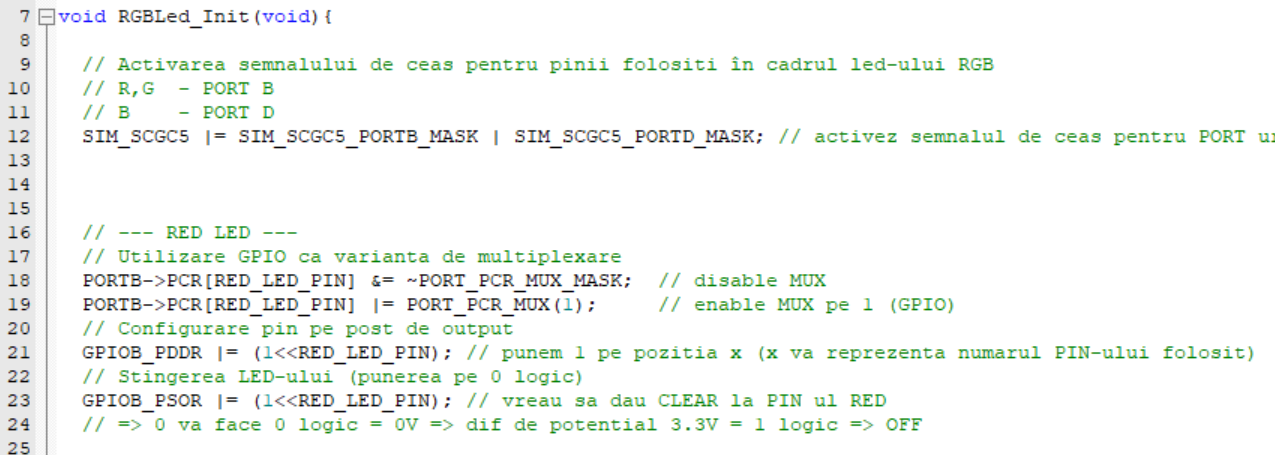


* In final vom activa intreruperile generate la finalizarea unei conversii

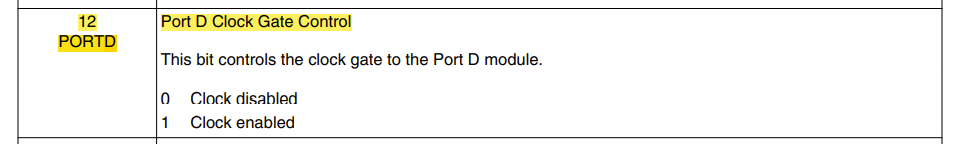


## Inițializarea modulului GPIO

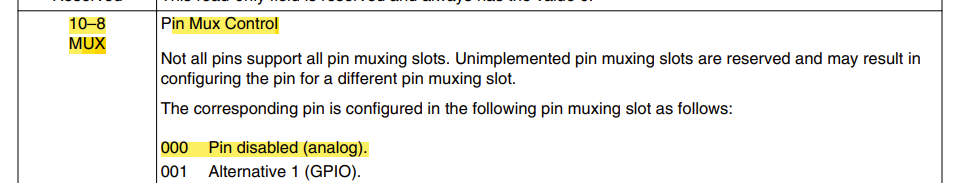
Vom initializa modulul GPIO pentru a putea folosi led-ul RGB integrat din placuta. Ne este necesar pentru a afisa culorile dorite conform spectrului de valori reprezentat de inputul de la senzorul de lumina



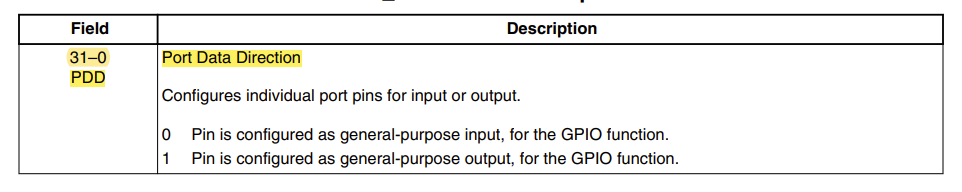
* Activam semnalul de ceas pe porturile B si D deoarece cele 3 led-uri sunt pe urmatorii pini: Red – PTB18, Green – PTB19, Blue – PTD1



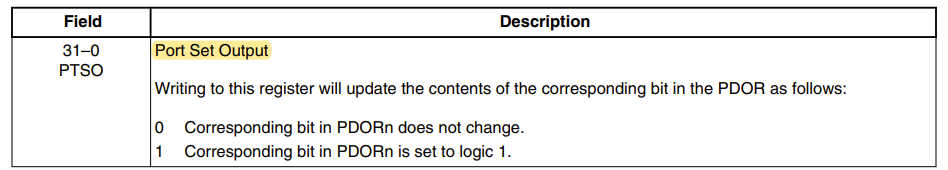
* Setam alternativa 1 pentru cei 3 registri PORTx\_PCRn deoarece dorim sa activam modulul GPIO

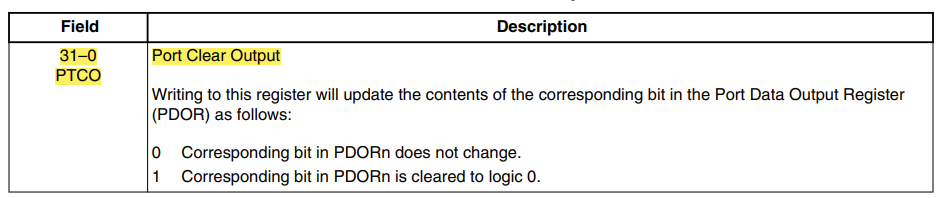


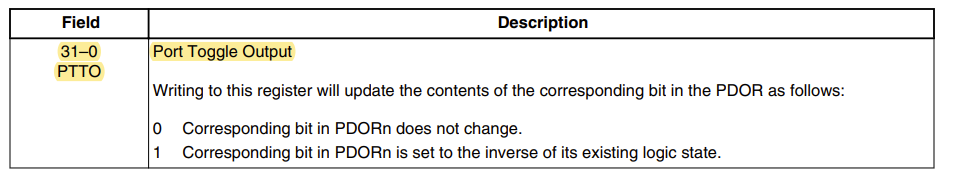
* Configuram registrul GPIOx\_PDDR in asafel incat sa setam valoarea 1 (directia de OUTPUT) pe bitul cu numarul pinului respectiv



* Pentru a modifica starea led-urilor, ne folosim de registri GPIOx\_PSOR/ GPIOx\_PCOR/ GPIOx\_PTOR ce corespund actiunilor de SET, CLEAR si TOGGLE. Asadar, in functie de valoarea bit-ului pus pe pozitia corespunzatoare led-ului, vom aprinde respectiv stinge luminile dorite



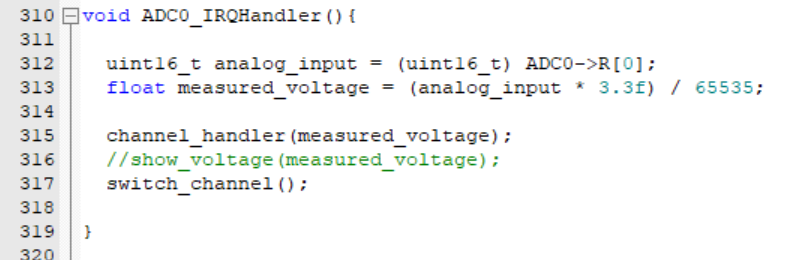




* Pentru inceput stingem toate led-urile urmand sa le aprindem in functie de valorile primite

## Functia de intrerupere pentru modulul ADC

In momentul in care este generata o intrerupere, de unul din semnalele transmise de cei doi senzori, vom prelua valoarea primita (o valoare intre 0 si INT\_MAX ~ 65535) si o convertim la o valoare corespunzatoare intre 0 si 3.3V, astfel ne asiguram ca lucram cu valori in volti. Mai departe, flow-ul este simplu: tratam inputul primit in functie de ce senzor a trimis date si la final schimbam canalul de pe care vom prelua date la urmatoarea citire: alternam intre cei 2 senzori pentru a citi odata o valoare de prag apoi o valoare de lumina.



## Prelucrarea datelor corespunzatoare senzorilor

Programul se ramifica in doua variante corespunzatoare celor doi senzori in functie de o variabila globala setata pentru a fi folosita drept flag (ne comunica de la ce canal a fost realizata ultima citire):

* Daca am primit date de la senzorul de rotatie, mapam valoarea primita la unul din pragurile setate (din 500 in 500 de luxi) si il salvam in variabila globala „valoare\_prag”
* Daca am primit date de la senzorul de lumina ambientala, transformam valoarea din volti in Lux si apoi o comparam cu ultima valoare de prag inregistrata.
  + Daca acesta nu trece de prag, vom „ignora” datele primite astfel: am putea sa lasam led-ul stins dar pentru o mai buna verificare a aplicatiei am ales sa semnalizam prin aprinderea led-ului cu o culoare de albstru (Blue).
  + Daca valoarea primita este mai mare decat pragul, facem verificarile corespunzatoare pentru a stabili spectrul de valori in care se situeaza (mici, medii, mari). Ulterior incadrarii intr-o zona de valori, vom aprinde led-ul/led-urile corespunzatoare: VERDE (Green), GALBEN (Red+Green), Rosu (Red).

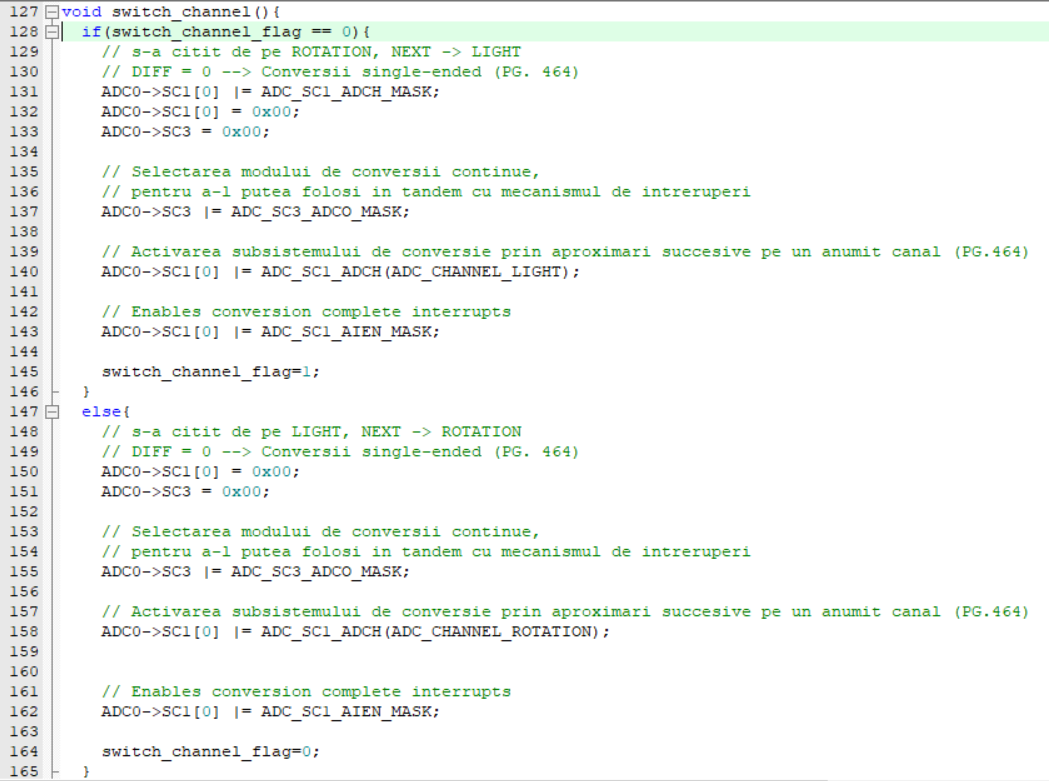


## Schimbarea canalului de primire a semnalului

La finalul actiunii corespunzatoare input-ului primit, ne asiguram ca schimbam canalul setat in registrul ADC0\_SC1n pentru a face toggle intre senzori. In functie de variabila globala switch\_channel\_flag setam:

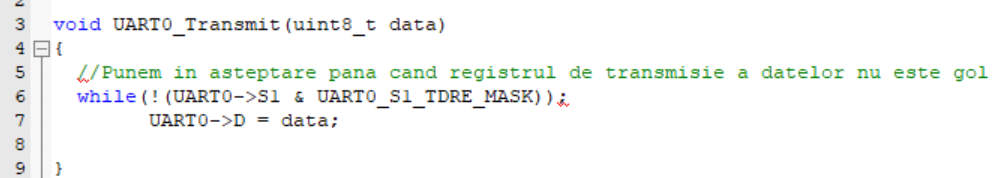
* Pentru valoarea 0 vom seta canalul 11 – ADC\_CHANNEL\_LIGHT
* Pentru valoarea 1 vom seta canalul 8 – ADC\_CHANNEL\_ROTATION

Putem observa faptul ca a fost necesara o resetare totala a registrilor folositi deoarece daca incercam sa schimbam doar valoarea ultimilor 5 biti (corespunzatori bitilor de ADCH), actiunea ar fi fost ignorata si nu s-ar fi realizat modificarea valorii. Asadar am optat pentru o resetare de tip „hard” a valorilor pentru a ne asigura de buna functionare.

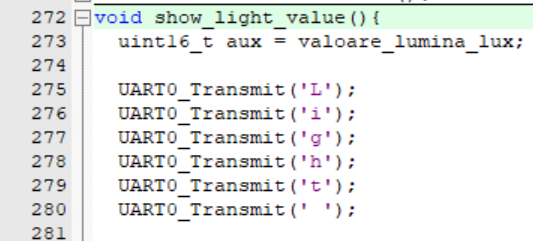


## Transmitere de date folosind modulul UART

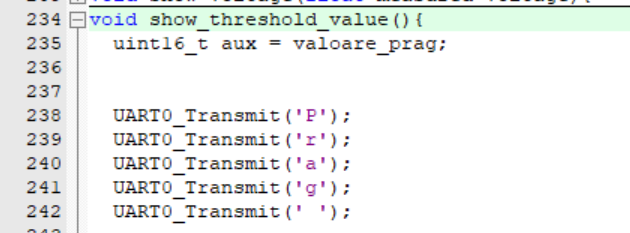
La finalul prelucrarii datelor primite, va trebui sa efectuam o actiune corespunzatoare: vom trimite prin modulul UART valoarea inregistrata de senzorul de lumina sau valoarea setata de senzorul de rotatie.



* Afisarea valorii inregistrate se va face sub forma: „Light xxxx Lux” deoarece marimea generata de senzorul de lumina ambientala repreztina de fapt valoarea in luxi a razelor de lumina ce cad direct pe senzor. Acesta poate inregistra valori intre 0 si 6000 Lux



* In ceea ce priveste pragul setat, de asemenea, senzorul de rotatie trimite o valoare inregistrata pe care noi o mapam intr-un interval specific astfel: din moment ce lumina inregistrata poate fi intre 0 si 6000 de luxi, pragurile vor fi tot in acest interval si pentru a evita o granularitate prea mare a pragului, am ales sa mapam din 500 in 500 de luxi valoarea pragului. Cu alte cuvinte, orice valoare intre 0 si 499 Lux (convertind semnalul primit de la senzorul de rotatie – Volti in valoare echivalenta in Lux) va seta pragul la 0, de la 500 la 999 pragul va fi 500 si astfel multimea de valori disponibila va fi: 0, 500, 1000, 1500, 2000, etc. De asemenea, rezultatul transmis va aparea sub forma „Prag xxxx Lux”.

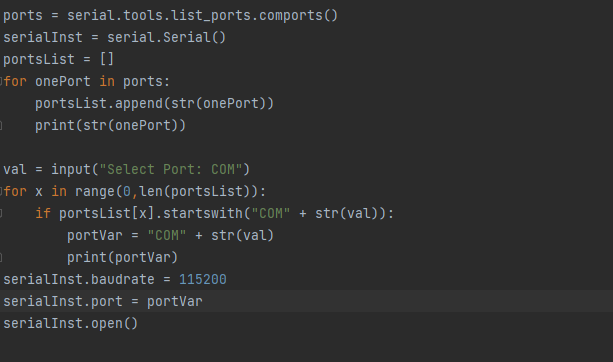


# Descriere program Python

Programul se foloseste de libraria PyQt pentru a putea desena grafice si a vedea in timp real datele procesate de la cei 2 senzori analogici. Primul grafic ne va arata valoarea senzorului de lumina iar cel de-al doilea grafic ne va arata valoarea perceputa de la senzorul de rotatie, stabilind astfel un prag pentru intervalul de valori in care senzorul de lumina va capta date si totodata tipul culorii care va fi afisat pe grafic in functie de acest prag.



## Initializare instanta seriala



* importam ,folosindu-ne de libraria serial tools, toate porturile deschise
* initializam instanta care se va ocupa de citirea de pe port
* inseram intr-un vector de porturi, toate porturile seriale deschise
* se va afisa in consola ce port este deschis si apoi se va selecta acesta
* setam baudrate
* initializam instanta seriala cu baudrate si portul aferent si o deschidem pentru a primi date de la placuta

## Citire date de la portul serial



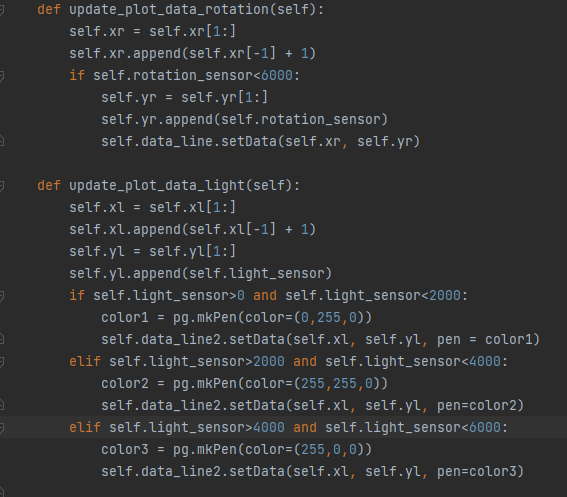
* definim o functie de unde vom citi datele primite de la placuta
* declaram datele globale care vor fi folosite pentru a face update in grafice
* ne folosim de functia in\_waiting care va citi datele primite de la placute intr-un ciclu infinit
* citim linia si o vom parsa
* datele de la placuta vor ajunge in formatul: "Light {{value}} Lux" si atunci vom parsa aceasta linie pentru a lua doar valoarea ce ne intereseaza
* in cazul in care programul citeste de la placuta gresit datele, acestea vor veni sub formatul "Light {{value}} LuxLight" iar pe a doua linie ne va aparea "{{value}} Lux" iar atunci programul va incerca sa citeasca valoarea cuvantului "Lux" care e un string si prin urmare va arunca o eroare
* verificam daca pe pozitia a 2-a din vector avem valoare numerica
* daca pe prima pozitie apare cuvantul Light inseamna ca vom citi date de la senzorul de lumina, altfel de la cel de rotatie
* actualizam datele din grafic

## Initializare clasa MainWindow



* initializam valorile de la senzorii de rotatie si lumina cu 0 initial
* adaugam un layout pentru grafic
* layout-ul va avea 2 grafice-> unul ce va arata valorile senzorului de lumina si celalalt valorile senzorului de rotatie
* le adaugam la fereastra principala
* setam culoarea backgroundului
* initial asignam culoarea alb pentru linia graficului de rotatie
* initial asignam culoarea rosu pentru linia graficului de lumina

## Actualizare valori grafice



* Scoatem primul element din lista
* Adaugam o valoare peste ultima
* Daca senzorul de rotatie primeste o valoare mai mica decat 6000 care este pragul maxim atunci se va face update
* La fel se procedeaza si pentru senzorul de lumina cu o singura diferenta
* Daca e valoare mica facem graficul sa afiseze verde (0,255,0)
* Daca e valoare medie facem graficul sa afiseze galben (255,255,0)
* Daca e valoare mare facem graficul sa afiseze rosu (255,0,0)

# Rezultate PyQt

# Dificultăți întâmpinate

In timpul realizarii acestui proiect am intampinat urmatoarele dificultati:

* Initial am ales folosirea portului PTC1 pentru unul din senzori dar s-a dovedit ca acesta nu functiona normal deoarece avea o tensiune reziduala, astfel singurele valori care puteau fi citite de pe el erau intre 3.26 – 3.29V, in schimb noi doream sa obtinem toata gama de valori: 0 – 3.3V
* Nu am reusit sa citim de pe cei doi senzori analogici doar schimband canalul de pe care se realiza citirea in modulul ADC. In schimb a trebuit sa resetam toti registri acestui modul si sa ii initializam din nou la fiecare schimbare de canal.
* In momentul in care am implementat in python o interfata grafica ce urma sa afiseze valorile de la cei doi senzori transpuse in grafice specifice, am citit de la portul serial ceea ce trimite placuta (ex „Light 1300 Lux” sau „Prag 500 Lux”) dar cateodata transmiterea suprapunea multiple caractere/secvente din ceea ce ar fi trebuit sa fie linii diferite. Asadar programul realizat in python arunca erori deoarece nu mai recunostea inputul primit (am setat un sablon pentru cum ar trebui sa arate inputul primit de la serial port). Drept urmare am ales sa tratam si acest caz, sanitizand inputul (ne asiguram ca respecta sablonul setat – nr cuvinte, formatare,etc; altfel ignoram linia malformata)
* Deoarece am ales sa folosim metoda de citire folosind intreruperi, aceasta avea un timp extrem de scurt de functionare. Astfel trecerea de la un canal la altul se realiza foarte rapid si practic nu avea timp sa schimbe in cod canalul si rezultatul era faptul ca se inversau valorile citite de la pini. In timpul testarii prin debug citirea se realiza conform pinilor conectati dar la o rulare normala deoarece intreruperea (precum si transmiterea datelor) se realiza mult prea rapid, practic se confundau cele doua canale. O posibila idee de realizare a acestei probleme ar fi schimbarea strategiei de citire (apelam functie de citire din main, nu folosim intreruperi pt ADC) si introducerea unui modul de PIT.

# Referințe

* <https://wiki.dfrobot.com/DFRobot_Ambient_Light_Sensor_SKU_DFR0026>
* <https://www.robotics.org.za/DFR0054>
* <https://wiki.dfrobot.com/Analog_Rotation_Sensor_V1__SKU__DFR0054_>
* <https://www.pythonguis.com/faq/pyqtgraph-plotting-over-time/>
* <https://www.tinkerassist.com/blog/arduino-serial-port-read/>
* <https://www.tinkercad.com/dashboard>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_ReferenceManual.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_Schematics.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_UserManual.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_Pinouts.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_Datasheet.pdf>