**ROMÂNIA**

**MINISTERUL APĂRĂRII NAȚIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**Facultatea de Sisteme Informatice și Securitate Cibernetică**

**Departamentul de Calculatoare și Securitate Cibernetică**



***Utilizare senzor De ambient SI REprezentare prin LED rgb***

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

Std. sg. maj. Georgescu David-Andrei - C114A

Std. sg. maj. Tudosie Mihnea-Vlad - C114E

Std. sg. maj. Barbulescu Daniel-Alexandru - C114D

**București**

**2022**

Contents

[**1.** **Prezentarea senzorului DFR0026(Light)** 3](#_Toc124340428)

[**2.** **Scop proiect** 4](#_Toc124340429)

[**3.** **Conectare senzor – placa de dezvoltare** 5](#_Toc124340430)

[**4.** **Model Tinkercad** 7](#_Toc124340431)

[**5.** **Descriere program** 8](#_Toc124340432)

[**5.1.** **Functia main** 8](#_Toc124340433)

[**5.2.** **Initializarea modulului UART** 8](#_Toc124340434)

[**5.3.** **Initializarea modulului ADC** 10](#_Toc124340435)

[**5.4.** **Initializarea LED-ului RGB** 14](#_Toc124340436)

[**5.5.** **Functia de intrerupere pentru modulul ADC** 16](#_Toc124340437)

[**6.** **Descriere cod Python** 18](#_Toc124340438)

[**6.1.** **Initializare port** 18](#_Toc124340439)

[**6.2.** **Initializare Main Window** 19](#_Toc124340440)

[**6.3.** **Actualizare date grafic** 20](#_Toc124340441)

[**6.4.** **Colectarea datelor** 21](#_Toc124340442)

[**7.** **Exemplificare rezultate PyQt** 22](#_Toc124340443)

[**8.** **Bibliografie** 25](#_Toc124340444)

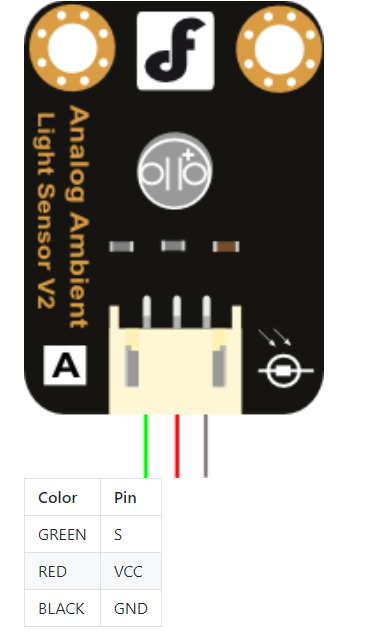
# **Prezentarea senzorului DFR0026(Light)**

Senzorul DFR0026 detecteaza densitatea luminii si reflecta valoarea ca un semnal analogic pentru controller.



**Specificatii:**

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

****

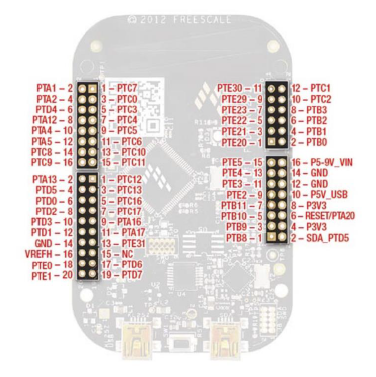
# **Scop proiect**

Scopul acestui proiect este acela de a reprezenta datele, in timp real, obtinute de la senzorul de lumina ambientala avand pragurile stabilite in prealabil. Led-ul RGB integrat pe placuta se va aprinde in functie de gradul de iluminare captat de senzorul ambiental:

* verde pentru valori intre 0 si 2000 Lux
* galben pentru valori intre 2000 si 4000 Lux
* rosu pentru valori intre 4000 si 6000 Lux

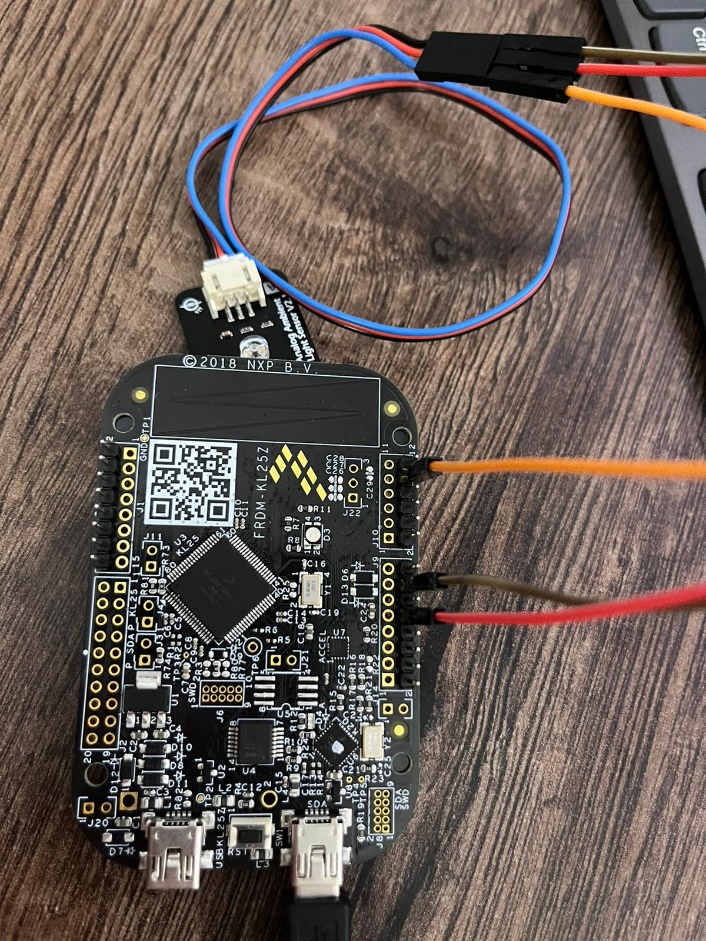
Totodata, prin UART vom transmite catre host valoarea receptata de senzor pentru a fi interpretata printr-un grafic generat cu ajutorul PyQt.

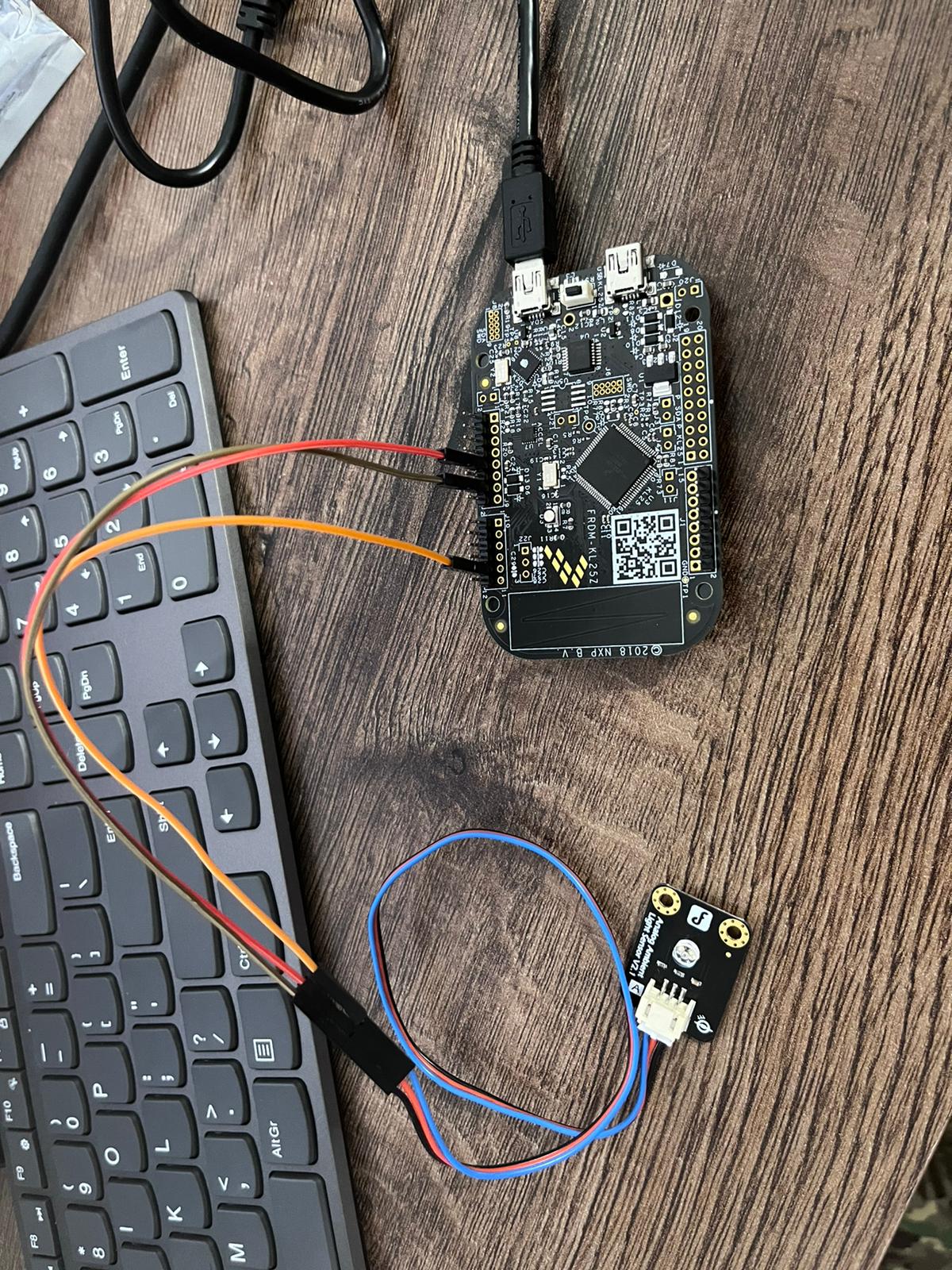
# **Conectare senzor – placa de dezvoltare**



Vom conecta senzorul astfel:

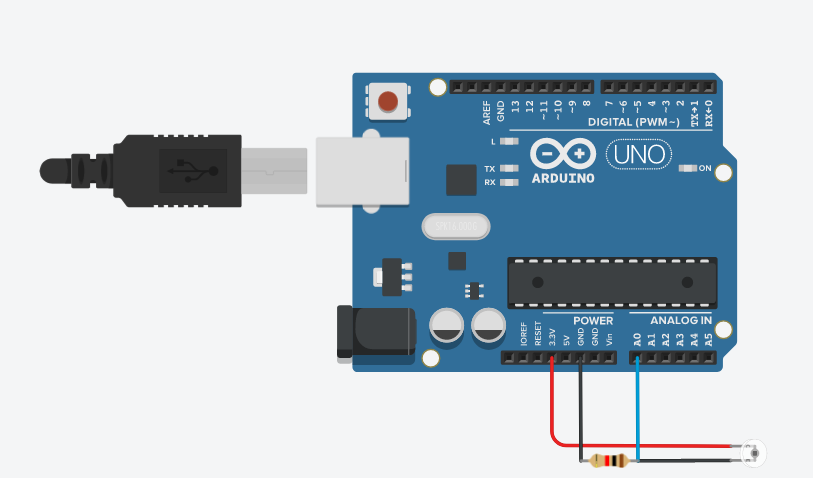
* La portul PTC2 am conectat firul albastru al senzorului, corespunzator firului de prelungire portocaliu.
* La 3.3V am conectat firul rosu al senzorului, corespunzator firului de prelungire rosu
* La portul GND am conectat firul negru al senzorului, corespunzator firului de prelungire kaki





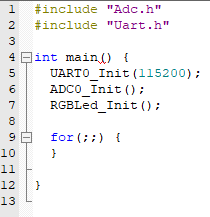
# **Model Tinkercad**

Am simulat modelul real, inlocuind target-ul nostru cu o placa Arduino.



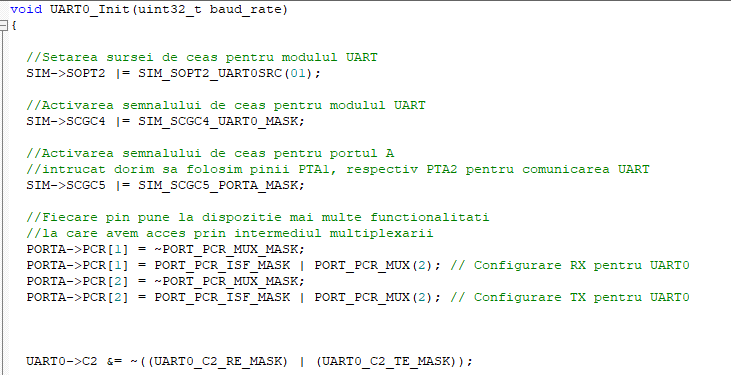
# **Descriere program**

# **Functia main**

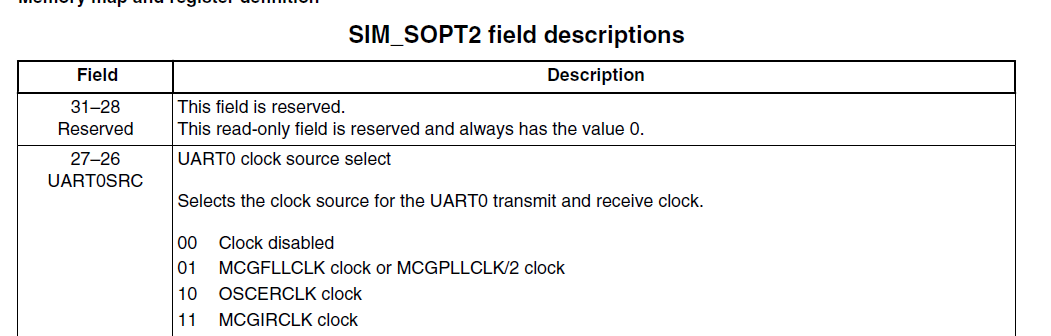


In sursa main.c am am apelat metodele de initializare ale modulelor UART, ADC, LED si am introdus o bucla infinita pentru a tine programul pornit. Nu am introdus alt cod in functia main, deoarece logica programului se bazeaza pe intreruperi. In momnetul in care senzorul emite un se declanseaza o intrerupere.

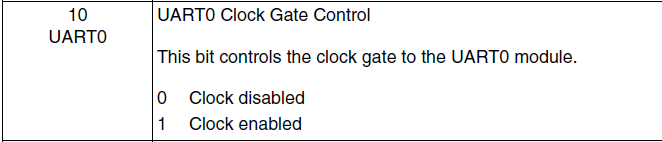
# **Initializarea modulului UART**



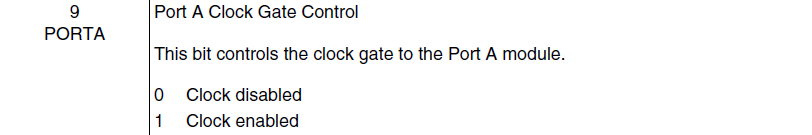
In registrul SIM\_SOPT2 setam pe 01 valoarea bitilor 26 si 27 pentru selectarea ca ceas a modulului MCGFLLCLK.



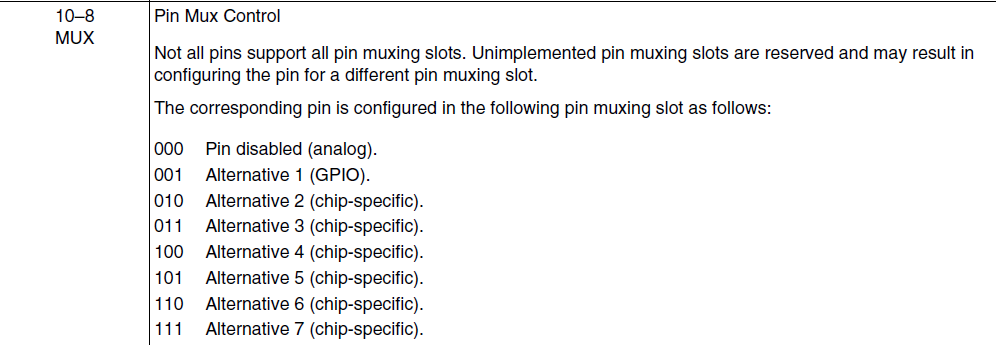
Pentru activarea semnalului de ceas pentru modulul UART setam bitul din registrul SIM\_SCGC4.



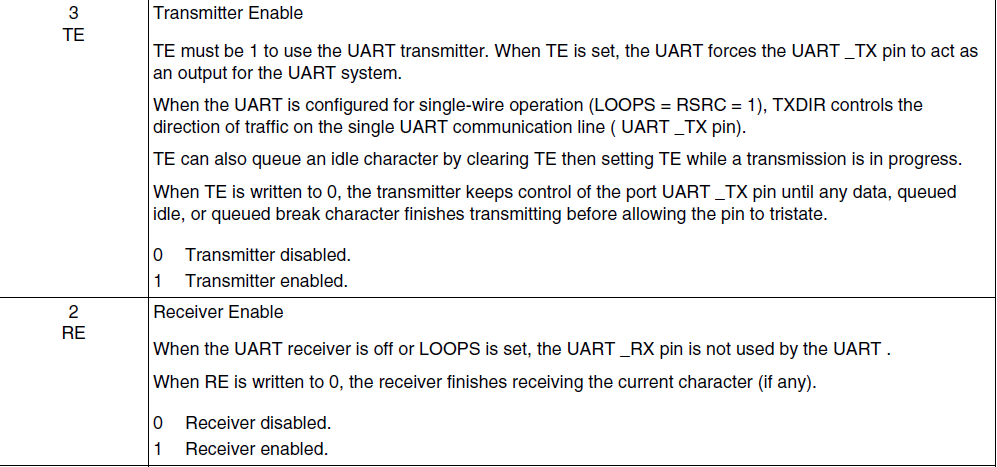
Setam bitul din registrul din SIM\_SCGC5 pentru activarea semnalului de ceas pentru portul A, intrucat dorim sa folosim pinii PTA1, respectiv PTA2 pentru comunicarea UART.



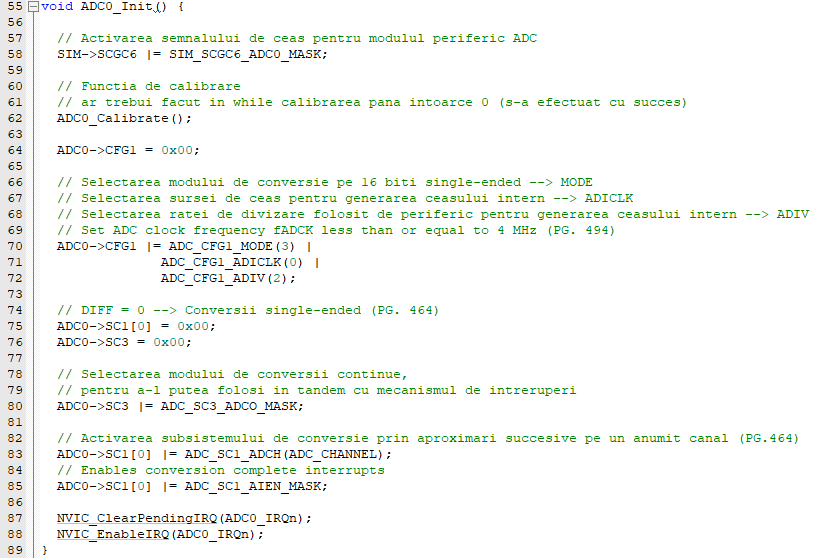
Am configurat pinii pentru a utiliza optiunea dorita, respectiv ALT2, deoarece dorim sa folosim porturile PTA1 si PTA2.



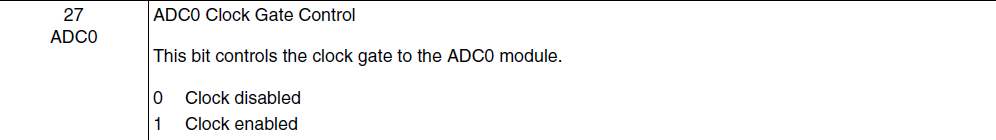
Registrul UART0\_C2 va fi dezactivat pe perioada configurarii baud rate-ului.



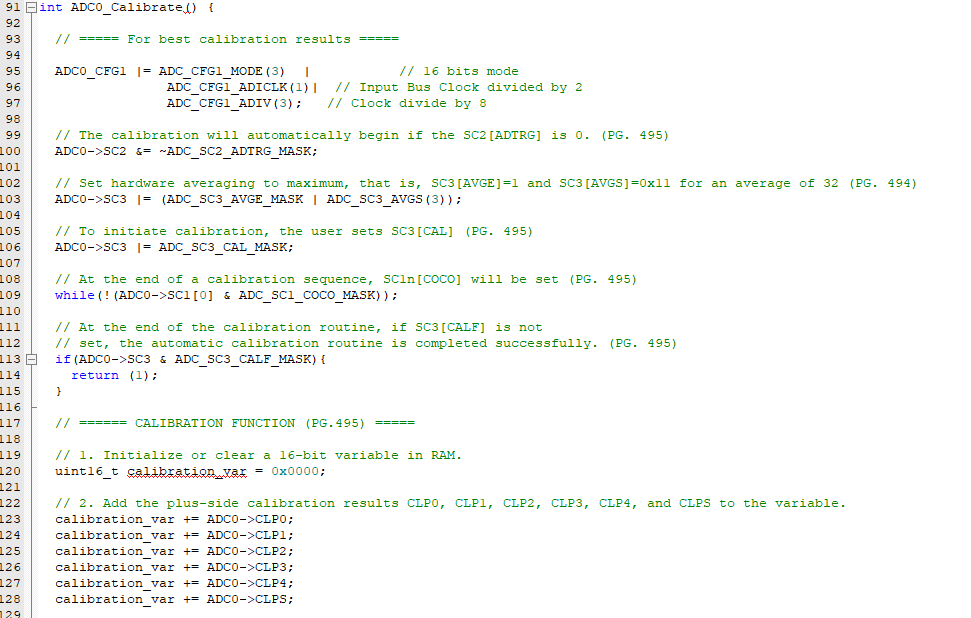
# **Initializarea modulului ADC**



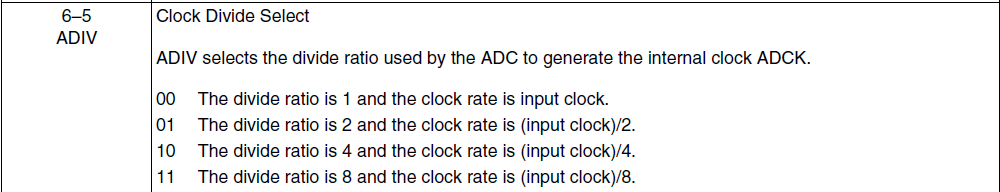
Se seteaza bitul de pe registrul din SIM\_SCGC6 pentru a activa semnalul de ceas pentru modulul periferic ADC.

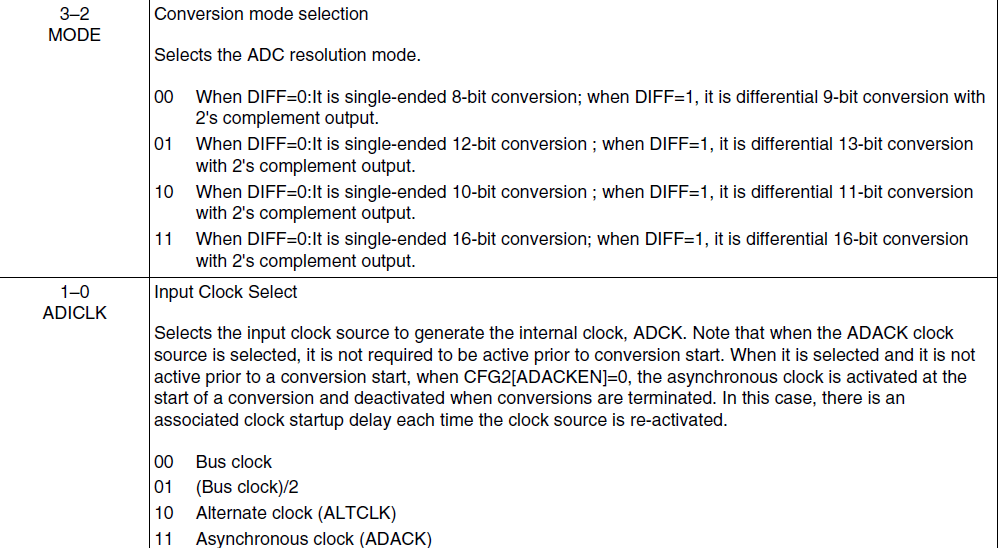


Se apeleaza functia de calibrare.

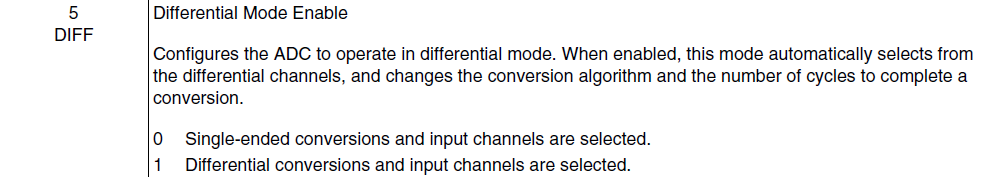


Se selecteaza modul de conversie pe 16 biti single-ened, se selecteaza sursa de ceas pentru generarea ceasului intern si se selecteaza rata de divizare folosita de periferic pentru generarea ceasului intern.

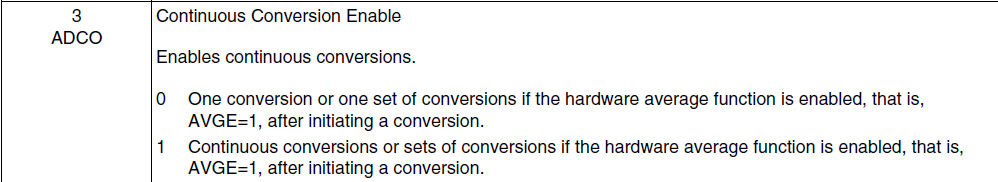




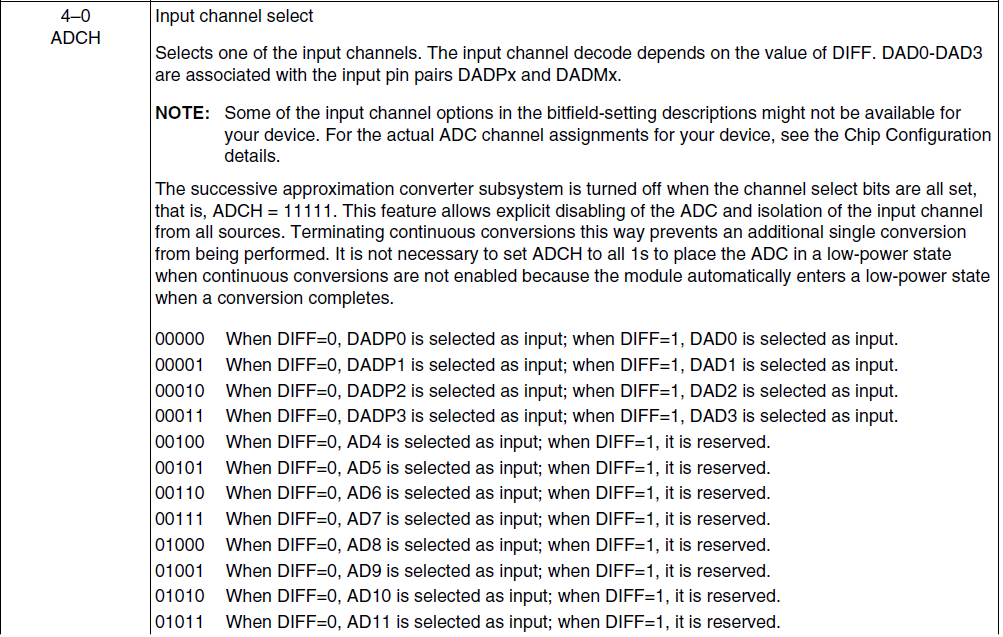
Pentru a selecta modul de conversie single-ended am setat registrul ADC0\_SC1 cu valoarea 0.



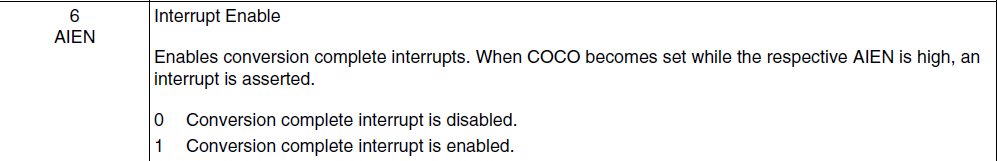
Selectam modulul de conversii continue pentru a-l putea folosi in tandem cu mecanismul de intreruperi



Activam subsistemul de conversie prim aproximari succesive pe un anumit canal

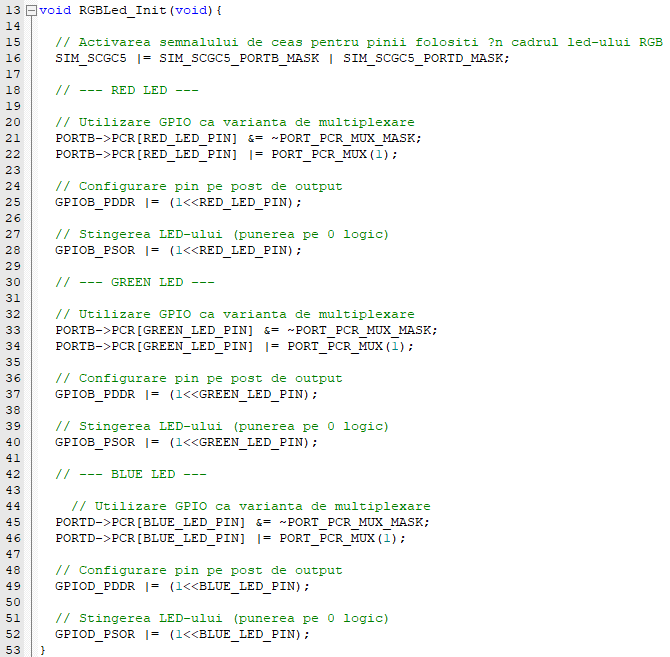


Activam intreruperile declansate la finalizarea conversiei



# **Initializarea LED-ului RGB**

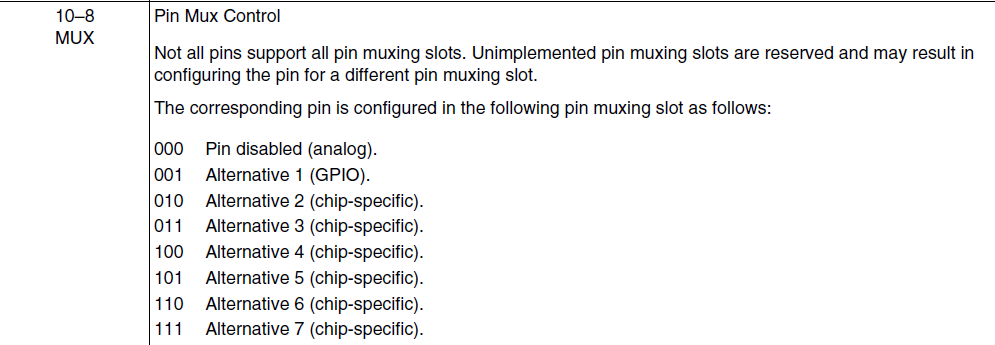
Modulul ne este necesar pentru schimbarea culorii led-ului in urma receptarii semnalului de catre senzorul ambinetal.



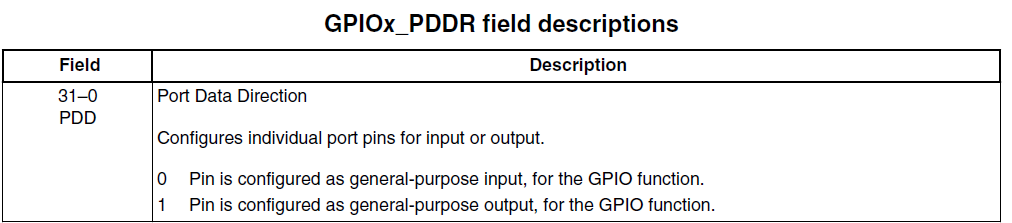
Activam semnalul de ceas pentru pinii folositi in cadrul led-ului RGB



Alegem alternativa 1 pentru PORTx\_PCRn pentru activarea modulului GPIO.

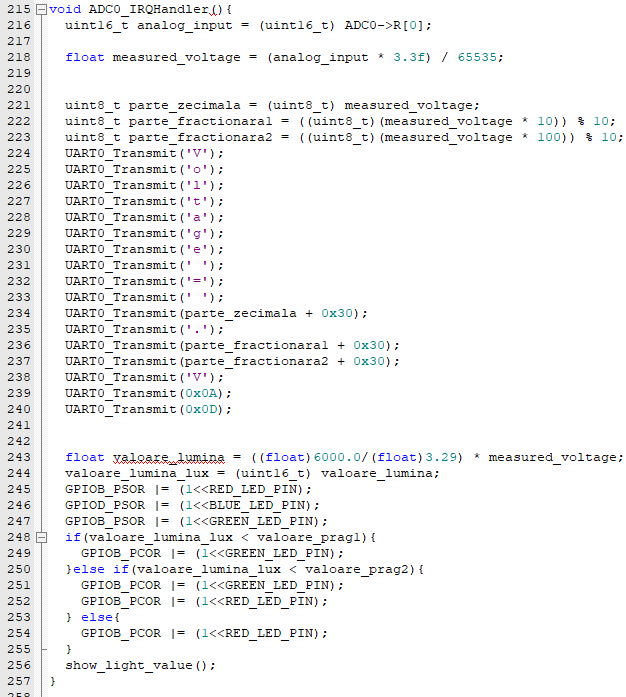


Este configurat registrul GPIOx\_PDDR pentru output.

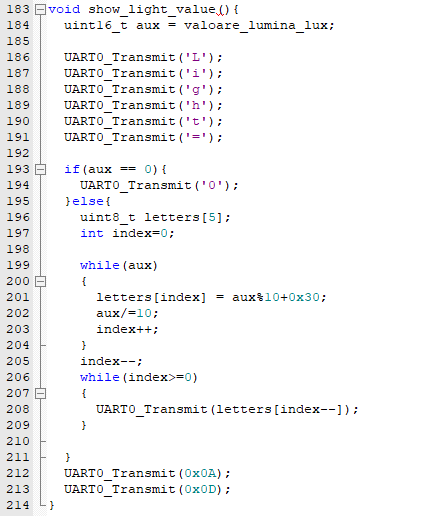


# **Functia de intrerupere pentru modulul ADC**

Pentru fiecare intrerupere declansata, valoarea semnalul transmis de catre senzorul de ambient (o valoare aflata in intervalul 0 si 65535) va fi convertita in volti (0 – 3.3V), folosind regula de trei simpla.



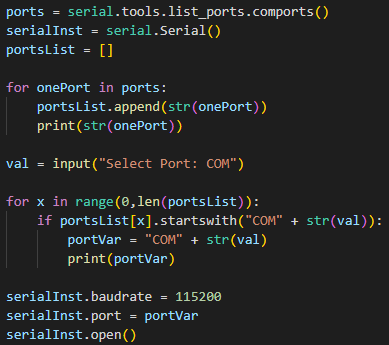
Dupa prelucrarea datelor, acestea vor fi trimise cu ajutorul modulului UART pentru a fi afisate la consola si utilizate de codul python pentru a fi reprezentate in grafic. Folosind regula de trei simpla, s-a calculat valoarea in Lux corespunzatoare voltajului. Valorile in Lux sunt intre 0 si 6000. Acest interval a fost impartit intre esantioane egale astfel: 0-2000 – verde; 2001-4000 – galben; 4001-6000 – rosu. Se apeleaza functia „show light value” care foloseste modulul UART pentru transmiterea valorii in Lux.



# **Descriere cod Python**

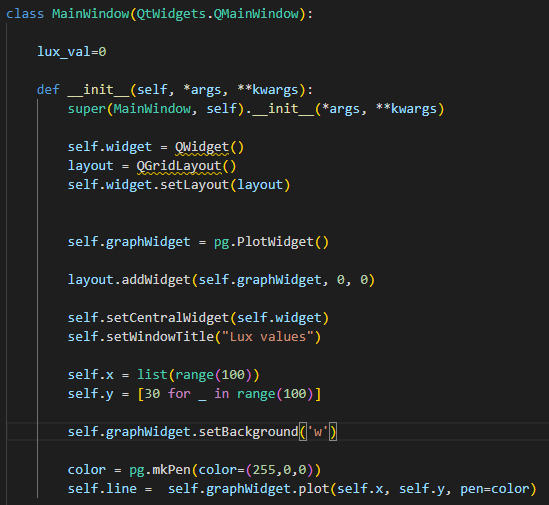
Procesarea si afisarea datelor receptate de senzorul ambiental se realizeaza cu ajutorul bibliotecii PyQt. Astfel, vom desena un grafic care se actualizeaza in timp real cu valoarea senzorului de lumina. De asemena, graficul isi va schimba culoarea in functie de pragul pe care il intrece valoarea luminii.

# **Initializare port**



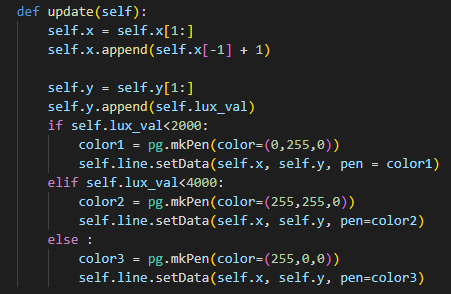
* Se insereaza in portsList toate porturile deschise, obtinue cu ajutorul bibliotecii serial tools
* Se afiseaza la consola portul deschis si se selecteaza acesta
* Se initializeaza portul si i se atribuie un baud rate
* In final, se deschide portul pentru a asculta date de pe placuta

# **Initializare Main Window**



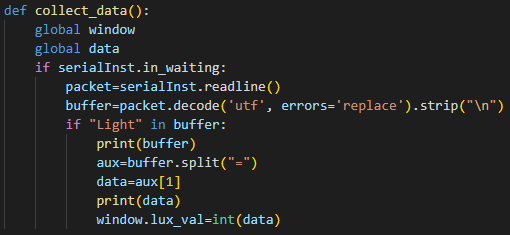
* In primul rand, declaram o variabila corespunzatoare cu valoarea intensitatii luminii masurata in lux si o initializam cu 0
* Se creeaza un layout in care se adauga un widget de tip grafic pentru plotarea valorilor receptate de senzorul de ambient
* Setam preferintele pentru acest widget (pozitie centrala, titlul ferestrei, culoarea backgroundului)
* Se alege o culoare initiala si se ploteaza graficul

# **Actualizare date grafic**



* Pentru a updata datele, eliminam primul element din vector si supraascriem ultimul element
* Verificam intervalul in care se afla valoarea receptata de senzor si schimbam culoarea graficului dupa cum urmeaza:
  + Verde pentru valori mai mici de 2000 Luxi;
  + Galben pentru valori din intervalul 2000 – 4000;
  + Rosu pentru valori mai mari de 4000 Luxi.

# **Colectarea datelor**



* Declaram variabilele globale windows si data prin intermediul carora updatam graficul afisat
* Functia in\_waiting apelata de portul deschis se ocupa de ascultarea acestui port si de citirea datelor la fiecare input nou
* Se citeste linia transmisa de placuta catre host si se parseaza
* In urma parsarii, se verifica componenta cuvantului „Light” si daca acesta exista, se parseaza bufferul
* In final, se printeaza valoarea in consola si se updateaza lux\_val pentru a fi afisat in grafic

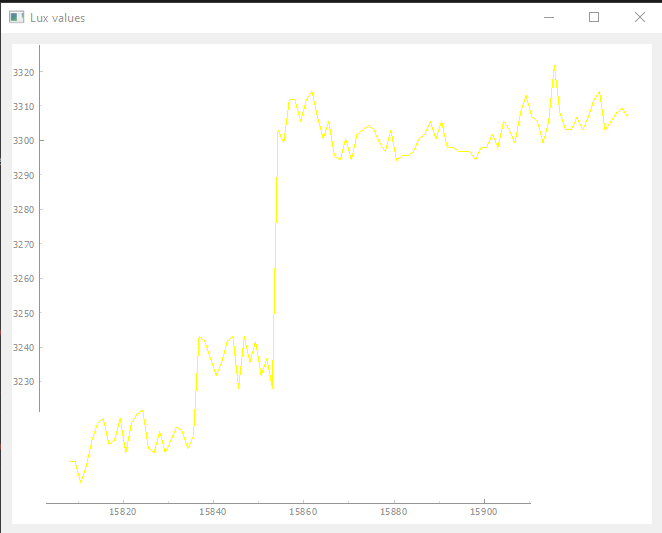
# **Exemplificare rezultate PyQt**

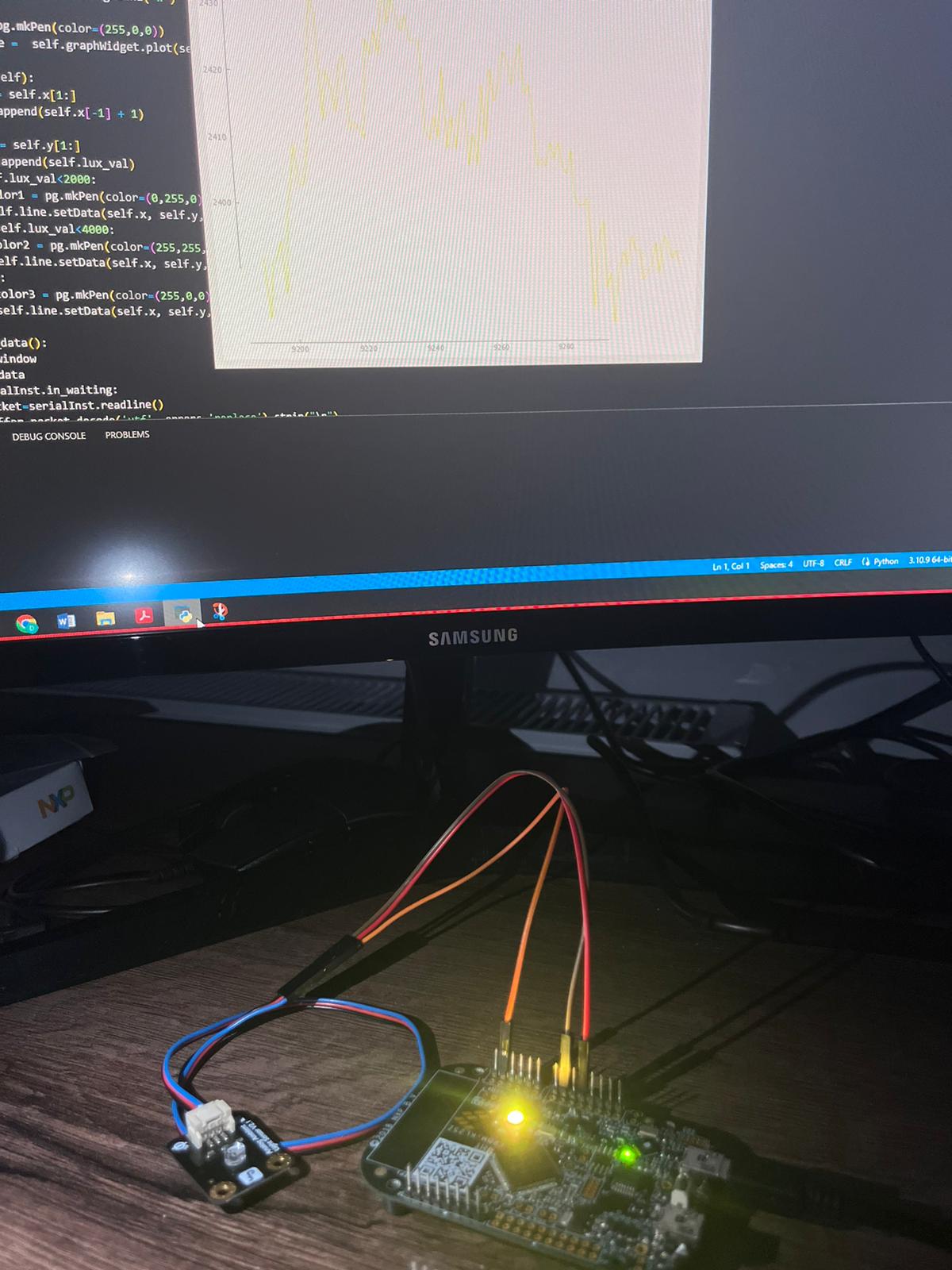
Pentru valori intre 0-2000 Lux, graficul are culoarea verde.





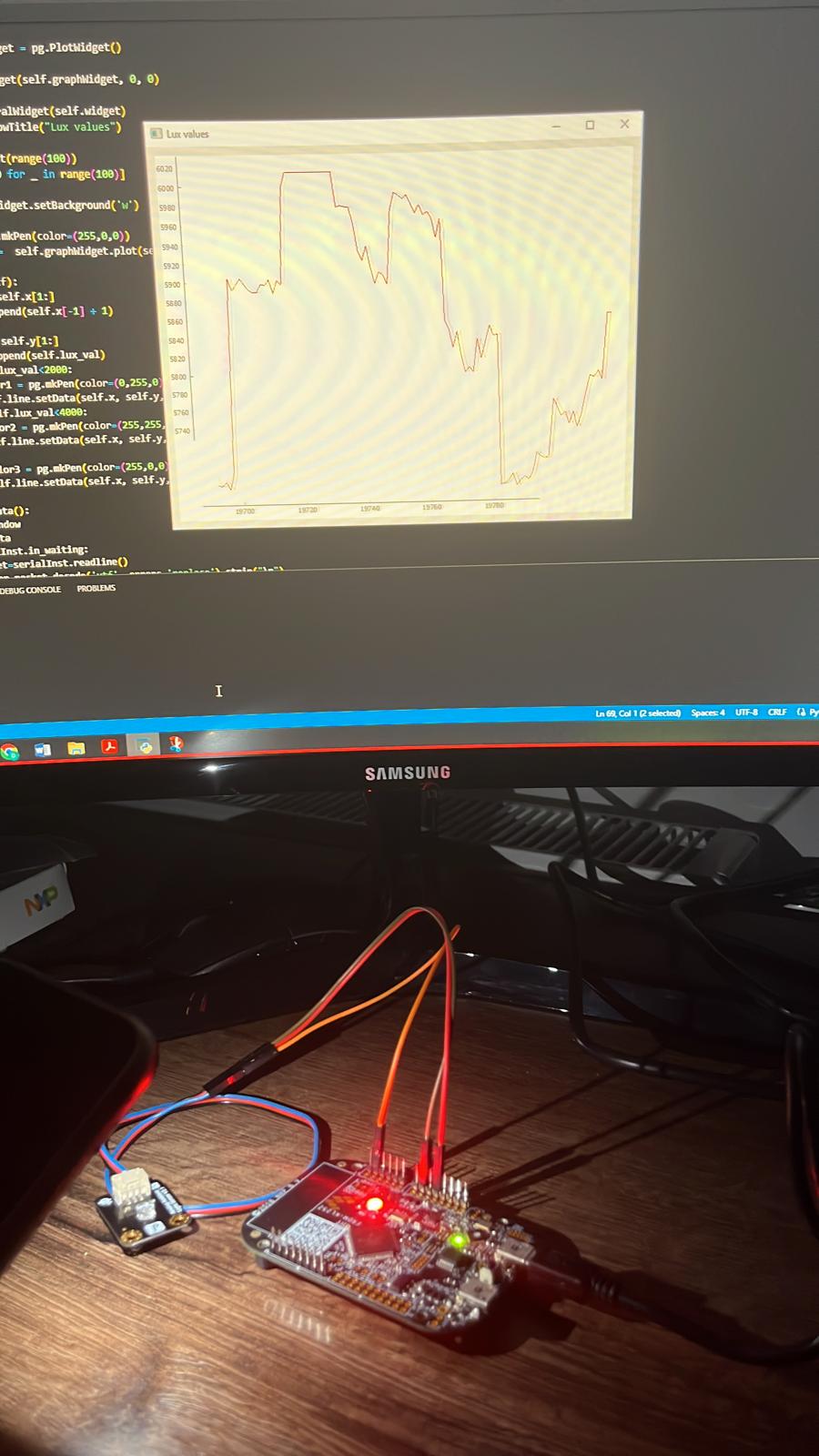
Pentru valori intre 2000-4000 Lux, graficul are culoarea galben.





Pentru valori intre 4000-6000 Lux, graficul are culoarea rosu.





# **Bibliografie**

* <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_Schematics.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_Datasheet.pdf>
* <https://github.com/undacmic/MCULabs/blob/main/Resurse/FRDM-KL25Z_Pinouts.pdf>
* <https://www.pythonguis.com/faq/pyqtgraph-plotting-over-time/>
* <https://www.tinkerassist.com/blog/arduino-serial-port-read/>
* <https://wiki.dfrobot.com/DFRobot_Ambient_Light_Sensor_SKU_DFR0026>
* <https://www.tinkercad.com/>