Creion Colorat Inteligent

Nica Alexandru Claudiu 335CB

Introducere

Cum functioneaza

Proiectul reprezinta implementarea unui creion colorat inteligent, de tipul Stylus Pen. **Culoarea creionului** este citita de un senzor de lumina, care este mai apoi afisata pe un ecran LCD I2C si de asemenea printr-un LED RGB care atunci cand se aprinde va avea culoarea respectiva. **Miscarea creionului** se va realiza folosind un joystick analogic ale carui coordonate vor fi mapate la un unghi cuprins intre 0 si 360 grade. Informatiile despre creion (culoarea si miscarea) vor fi **transmise prin Serial** catre un program Python care implementeaza **biblioteca grafica Turtle**. Acest program va misca obiectul de tip Turtle cu culoarea si miscarea indicata.

Scopul proiectului

Scopul proiectului este de a pune in practica notiunile capatate de-a lungul semestrului, de a invata sa fac un proiect hardware de la 0 si de a-mi dezvolta creativitatea si spiritul ingineresc

Ideea proiectului

Ideea proiectului a pornit de la faptul ca mereu am fost curios de cum functioneaza touch-screen-ul si stylus-ul. De asemenea, am vrut sa implementez un proiect care se imbina si cu alte limbaje de programare si care transmite informatii prin serial.

Utilitatea proiectului

Consider ca proiectul este important pentru mine deoarece este ceva ce m-a motivat si a pornit din curiozitate. Pentru colegii mei, consider ca poate fi un mod distractiv de a realiza notite sau de a se distra prin testarea de diferite culori si desene

Descriere generală

Descriere functionalitate

Creionul cunoaste implicit doar un set de culori predefinite: rosu, albastru, negru, verde, galben si alb. Culoarea creionului va fi citita de detectorul de culori TCS3200, iar in cazul in care este una cunoscuta, se va afisa pe ecranul LCD I2C un mesaj cu numele acesteia. De asemenea, am adaugat un LED RGB care va lumina culoarea citita.

Odata stiuta culoarea, creionul trebuie de asemenea sa se miste, lucru care se realizeaza folosind un joystick analogic. Acesta are 2 axe: X si Y, cu ajutorul carora se poate calcula usor unghiul facut de joystick fata de centrul sau. Unghiul rezultatul reprezinta directia de desenare a creionului.

Informatiile obtinute (culoarea si orientarea) sunt retinute intr-un mesaj de tip String care este transmis prin Serial catre un program Python care extrage datele necesare si misca cursorul de ecran. Cursorul va fi miscat folosind biblioteca Turtle si se va misca respectand culoarea si orientarea.

Creionul se misca in timp real si atata timp cat utilizatorul doreste si programul este pornit

Descriere functionare senzor TCS3200

Conform foii tehnice, TCS3200 este un convertor de lumină-frecvență color programabil, care combină fotodiodele de siliciu configurabile și un convertor curent-frecvență pe un singur circuit integrat CMOS monolitic. Ieșirea este o undă pătrată (50% ciclu de funcționare) cu frecvență direct proporțională cu

intensitatea luminii (iradiere). Frecvența de ieșire la scară completă poate fi scalată cu una din cele trei valori presetate (2%, 20%, 100%) prin intermediul a doi pini de intrare de control (S0 și S1). Voi seta atat S0 cat si S1 la 100% pentru o acuratete cat mai mare.

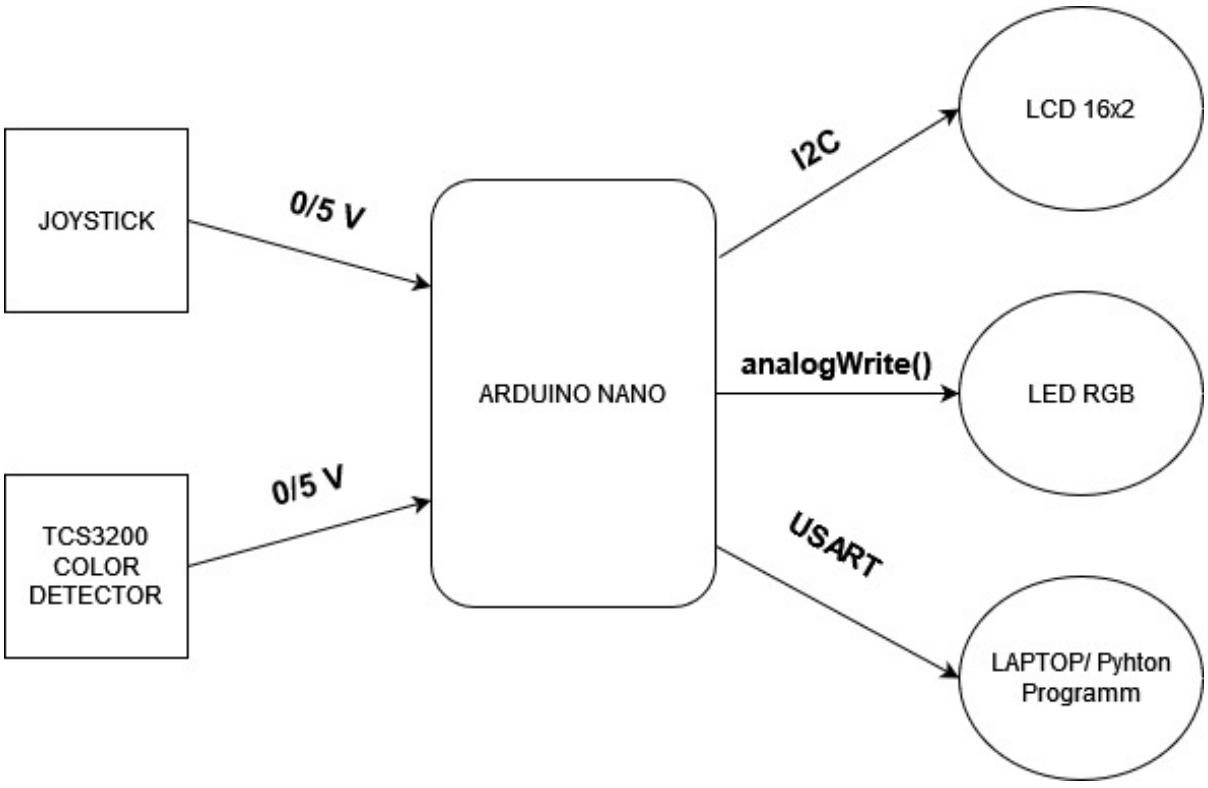
In TCS3200, convertorul lumină-frecvență citește o matrice de 8 x 8 de fotodiode: 16 fotodiode au filtre verzi, 16 rosii, 16 albastre si 16 fara filtre.

Pinii S2 și S3 sunt utilizați pentru a selecta ce grup de fotodiode (roșu, verde, albastru, fara filtru) sunt active, deci teoretic sunt folositi pentru a seta culorile ce pot fi citite. Combinatiile LOW-HIGH, definite si in datasheet-ul senzorului de culori, permit selectarea a diferite fotodiode, deci permit citirea de componente R, G, B (rosu, verde, albastru). Pentru o acuratete mai mare, pentru fiecare verificare de culoare, voi masura fiecare componenta de 10 ori, iar in final voi retine media. Analizand valorile fiecarei componente pot deduce ce culoare a fost analizata de senzorul TCS

S0	S1	OUTPUT FREQUENCY SCALING (f_o)
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	PHOTODIODE TYPE
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Schema bloc



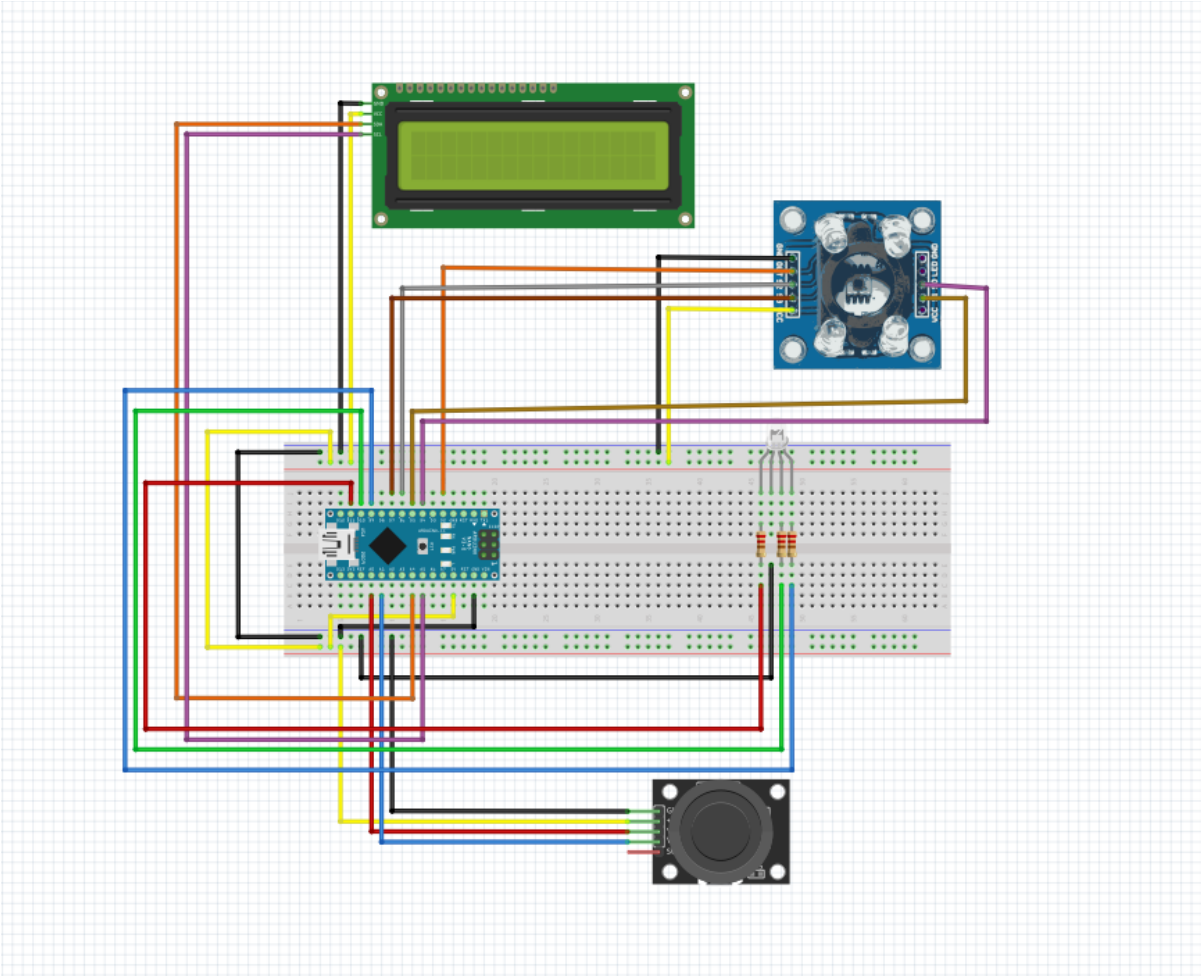
Hardware Design

Elemente folosite

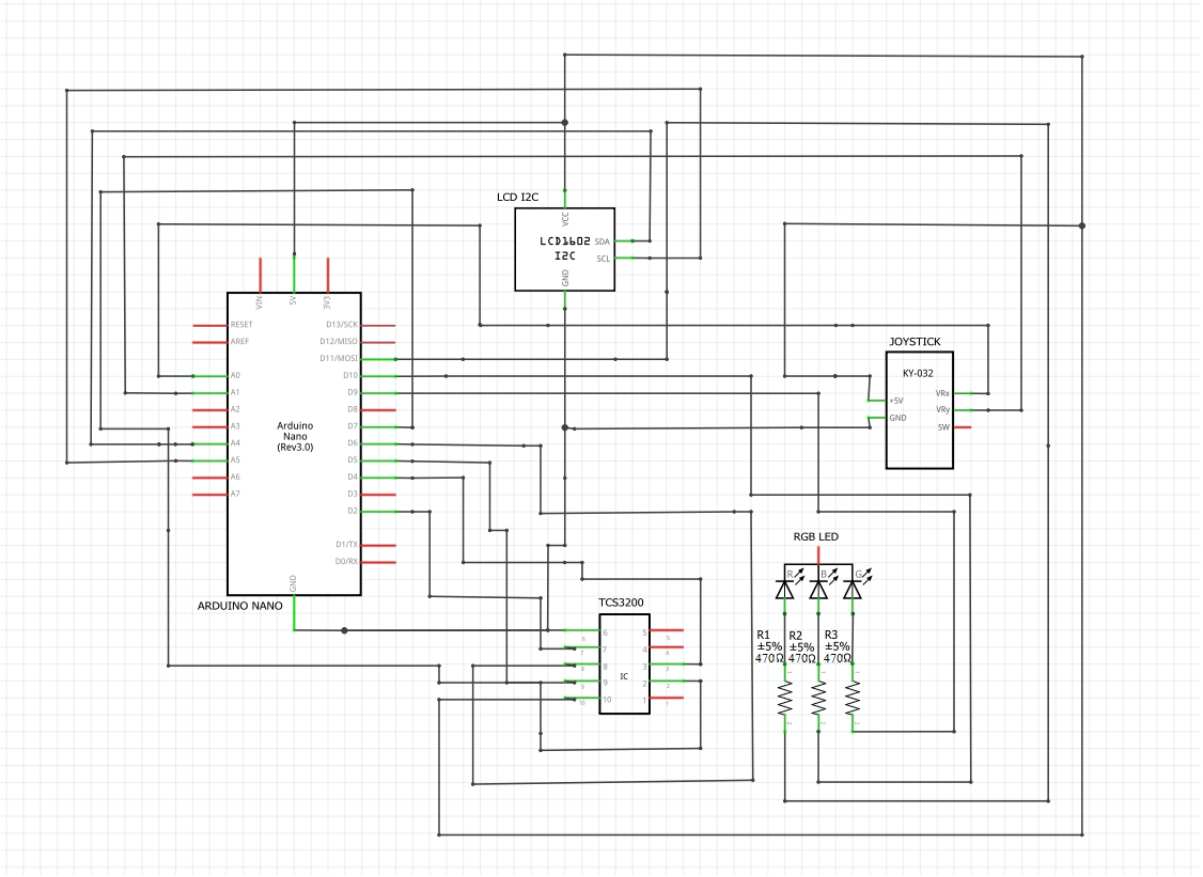
- Arduino Nano: x1
- TCS3200: x1
- Joystick analog: x1

- LCD16x2 I2C: x1
- LED RGB catod comun: x1
- Rezistente 470 Ohmi: x3
- Breadboard: x1
- Fire de legatura mama-tata
- Fire de legatura tata-tata

Schema Hardware



Schema Electrica



Conectare componente hardware

Senzorul TCS3200:

- GND ⇒ GND Arduino
- OUT ⇒ pinul D2 Arduino
- S2 ⇒ pinul D6 Arduino
- S3 ⇒ pinul D7 Arduino
- VCC ⇒ pinul de +5V Arduino
- S0 ⇒ pinul D4 Arduino
- S1 ⇒ pinul D5 Arduino

LCD16x2 I2C:

- VCC ⇒ pinul de +5V Arduino
- GND ⇒ GND Arduino
- SDA ⇒ pinul A4 Arduino
- SCL ⇒ pinul A5 Arduino

Joystick analogic:

- GND ⇒ pinul GND Arduino
- +5V ⇒ pinul +5V Arduino
- VRX ⇒ pinul A0 Arduino
- VRY ⇒ pinul A1 Arduino

RGB LED

- componenta R ⇒ pinul D11 Arduino
- componenta G ⇒ pinul D10 Arduino
- componenta B ⇒ pinul D9 Arduino
- GND ⇒ GND Arduino

Am optat pentru folosirea Arduino Nano in loc de Arduino Uno deoarece:

- Are dimensiuni mult mai mici, fiind mai ergonomic si usor de transportat, intrucat poate fi montat direct pe breadboard
- Are costul mai redus, ceea ce mi-a permis sa adaug alte functionalitati (LED RGB)

Componentele R,G,B din LED sunt inseriate cu rezistente de 470 Ohmi pentru a nu arde LED-ul

Software Design

Mediu dezvoltare

- Schema bloc: Draw.io
- Schema hardware/ Schema electrica: Fritzing
- Cod arduino: Arduino IDE
- Cod python: IDLE

Biblioteci folosite

1. Arduino:
    - *Wire.h*: biblioteca folosita pentru a scrie pe LCD
    - *LiquidCrystal_I2C.h*: biblioteca folosita pentru comunicarea cu ecranul LCD prin intermediul I2C
2. Python:
    - *turtle*: biblioteca grafica folosita pentru a desena pe ecran
    - *serial*: biblioteca folosita pentru a comunica prin serial cu placuta Arduino
    - *math*: biblioteca folosita pentru a calcula unghiul rezultat in urma miscarii joystick-ului

Descriere functii

1. Codul Arduino, impartit in 3 fisiere principale:
    a. Detector_Culori.ino:
        - *char computeColorToSend(String LAST_KNOWN_COLOR)*: calculeaza ce informatie trebuie transmisa in functie de culoare prin serial si returneaza un caracter ce indica culoarea
        - *void setup()*: se seteaza pinii, variabilele initiale, serialul si se initializeaza ecranul LCD
        - *void loop()*: contine implementarea propriu-zisa a programului, care se va apela la infinit
    b. Functii_Printare.ino:
        - *void showDataLCD(void)*: printeaza pe ecranul LCD detalii despre fiecare componenta R,G,B, dar si culoarea identificata, daca se stie
    c. Functii_RGB.ino:
        - *void getColor()*: determina culoarea curenta in functie de componentele R, G, B
        - *readRGB():* citeste componentele R, G, B prin intermediul senzorului TCS3200
        - *setRGB(int red, int green, int blue)*: seteaza componentele R,G,B ale LED-ul la

valorile primite
- *turnLEDOn(String LED_COLOR)*: aprinde LED-ul cu culoarea specificata

2. Codul Python, aflat in fisierul penScript.py:
   - *getPenColor(color)*: returneaza culoarea corecta in functie de inputul primit
   - *printInfo(color, angle)*: afiseaza informatii de folos despre program
   - *programul principal Python*: primeste informatiile de la serial si misca cursorul pe ecran corespunzator

Mai multe detalii despre implementarea codului se regasesc in comentariile din cod si in README

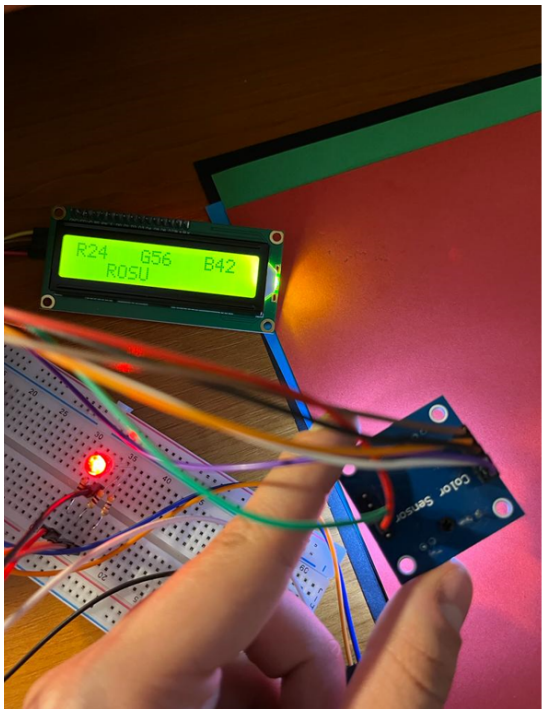
Pentru o functionare corecta, trebuie modificat in programul python numele portului folosit pentru comunicarea prin serial

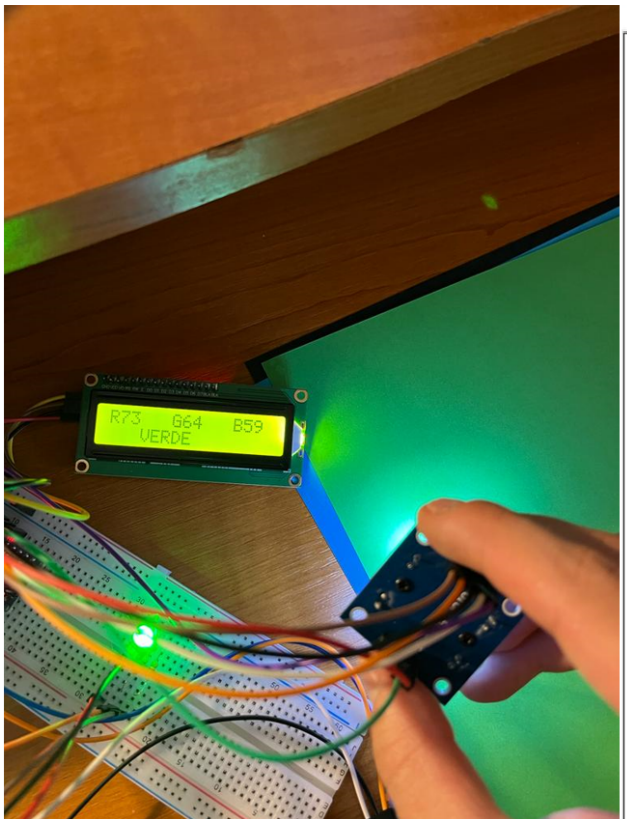
Rezultate Obţinute

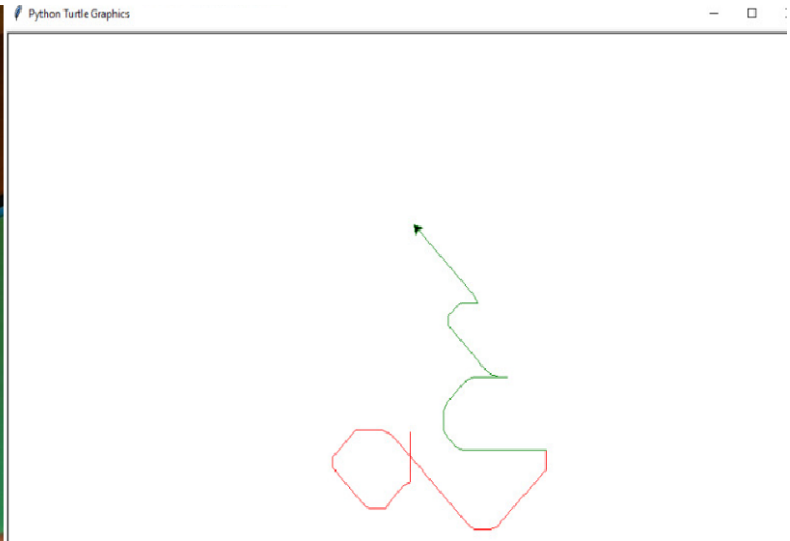
Dupa cum se observa si in imaginile atasate, detectorul de culori reuseste de cele mai multe ori sa detecteze culoarea. Nu este insa cel mai performant si citirea corecta depinde de lumina si cum este detectorul tinut fata de obiectul dorit. Pentru acest proiect insa, reuseste sa si faca treaba suficient de bine.

Becul se aprinde la culoarea citita, mai putin in cazul in care se citeste negru, intrucat nu se poate reda cu un LED RGB.

Informatiile de la joystick si detectorul de culori sunt cu succes transmise prin serial si pe ecran apare desenul de culoarea corecta si deplasat corect.







Am realizat de asemenea un **demo** care surprinde principalele functionalitati ale proiectului:

https://www.youtube.com/watch?v=s9GKNvTCYUk [https://www.youtube.com/watch?v=s9GKNvTCYUk]

Concluzii

In concluzie, proiectul a atins, in opinia mea, toate cerintele pentru a demonstra notiunile invatate pe parcusul semestrului. De asemenea, consider ca mi-a dezvoltat mult abilitatile de a realiza un proiect hardware.

Download

Arhiva cu cod ce contine:

- Folderul Detector_Culori cu toate fisierele de cod Arduino
- penScript.py ce contine codul in Python
- README
- nicaalexandru335cb_codpm.zip

PDF cu toata documentatia:

Bibliografie/Resurse

1. Datasheet TCS3200:
    - https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiy5c-8jIj4AhUD26QKHTU-CI8QFnoECAMQAQ&url=https%3A%2F%2Fwww.mouser.com%2Fcatalog%2Fspecsheets%2Ftcs3200-e11.pdf&usg=AOvVaw2p-

DOtMCjty77AqLGN9f6E [https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiy5c-8jIj4AhUD26QKHTU-CI8QFnoECAMQAQ&url=https%3A%2F%2Fwww.mouser.com%2Fcatalog%2Fspecsheets%2Ftcs3200-e11.pdf&usg=AOvVaw2p-DOtMCjty77AqLGN9f6E]

2. Transmiterea prin serial catre Pyhton:
   - https://create.arduino.cc/projecthub/ansh2919/serial-communication-between-python-and-arduino-e7cce0 [https://create.arduino.cc/projecthub/ansh2919/serial-communication-between-python-and-arduino-e7cce0]
3. Biblioteca turtle si functile sale:
   - https://docs.python.org/3/library/turtle.html [https://docs.python.org/3/library/turtle.html]
4. Laborator 1: USART, LCD
   - https://ocw.cs.pub.ro/courses/pm/lab/lab1 [https://ocw.cs.pub.ro/courses/pm/lab/lab1]
5. Laborator 6: I2C
   - https://ocw.cs.pub.ro/courses/pm/lab/lab6-2022 [https://ocw.cs.pub.ro/courses/pm/lab/lab6-2022]