

# Lie Detector

Autor: [Ștefania Cristiana Olteanu](#)

Grupa: 336CA

## Introducere

Un test poligraf se bazează pe măsurarea și înregistrarea unor parametri fiziologici ai unui individ, cum ar fi pulsul, temperatura, respirația, perspirația și conductivitatea pielii, în timp ce i se pun o serie de întrebări. Încă din cele mai vechi timpuri se considera că a minți determină efecte fiziologice colaterale. În acest proiect doresc să simulez un "detector de minciuni" care deși nu întotdeauna obține rezultate corecte (unii oameni își pot controla în mod conștient reacția corpului lor) este un experiment interesant în a măsura anumiți parametri ai corpului și a vedea cum sau dacă aceștia se modifică.

## Descriere generală

Prin intermediul unui senzor de măsurarea a pulsului și a un senzor de temperatura voi obține informațiile necesare unui verdict asupra valorii de adevăr a răspunsului unui individ (simularea unui test poligraf). Valorile vor fi analizate și pe baza acestora va fi declansată una din următoarele stări:

- **LIE** → LED-ul se face roșu, pe ecranul LCD se va afișa mesajul: "You are lying!" și buzzerul va cânta Never Gonna Give You Up - Rick Astley
- **TRUTH** → LED-ul se face verde, pe ecranul LCD se va afișa mesajul: "You are telling the truth!" și buzzerul va cânta Hedwig's theme from the Harry Potter Movies

## Schemă bloc



## Hardware Design

### Lista componente

| Nume           | Număr Piese |
|----------------|-------------|
| Arduino UNO R3 | 1           |

|                                       |   |
|---------------------------------------|---|
| Breadboard                            | 1 |
| Ecran LCD 16×02 (I2C)                 | 1 |
| Senzor puls MAX30100                  | 1 |
| Senzor temperatura si umiditate SHT21 | 1 |
| LED rosu                              | 1 |
| LED RGB                               | 1 |
| Buzzer                                | 1 |
| Rezistor 330kΩ                        | 5 |

## Schemă electrică



Placuta comandata care include senzorul are o problema de design, respectiv liniile de SDA si SCL sunt HIGH la tensiunea de 1.8V, si nu 3.3V, astfel incat Arduino UNO nu poate comunica cu senzorul <sup>1)</sup>. Am modificat placuta printr-un jumper astfel incat SDA si SCL sunt alimentate de la 3.3V.

## Software Design

Mediul de dezvoltare:

- software:
  - Arduino IDE <sup>2)</sup>
  - VS Code <sup>3)</sup>
- pentru realizarea schemei-bloc:
  - draw.io <sup>4)</sup>
- pentru realizarea schemei electrice:
  - EAGLE <sup>5)</sup>

Biblioteci folosite:

- Wire <sup>6)</sup>
- LiquidCrystal\_I2C <sup>7)</sup>
- MAX30100\_PulseOximeter <sup>8)</sup>
- SHT21 <sup>9)</sup>

## Structura codului

**Initializari** - inainte de orice functie, initializam urmatoarele variabile:

- *pox* → clasa care defineste senzorul de puls-oximetru
- *sht* → clasa care defineste senzorul de temperatura si umiditate
- *tsLastReport* → initializat cu 0, folosita pentru a actualiza display-ul la intervale determinate
- *is\_lying* → initializat cu -1, ne aflam intr-o stare in care nu se stie daca se minte sau se spune

adevarul

- *count* → necesar pentru a obtine media valorilor ce trebuie comparata cu threshold-ul
- *melody, tempo, notes, wholenote, divider, noteDuration* → variabilele necesare buzzerului pentru a canta melodiile de mintit si spus adevarul (cate o variabila diferita pentru fiecare)

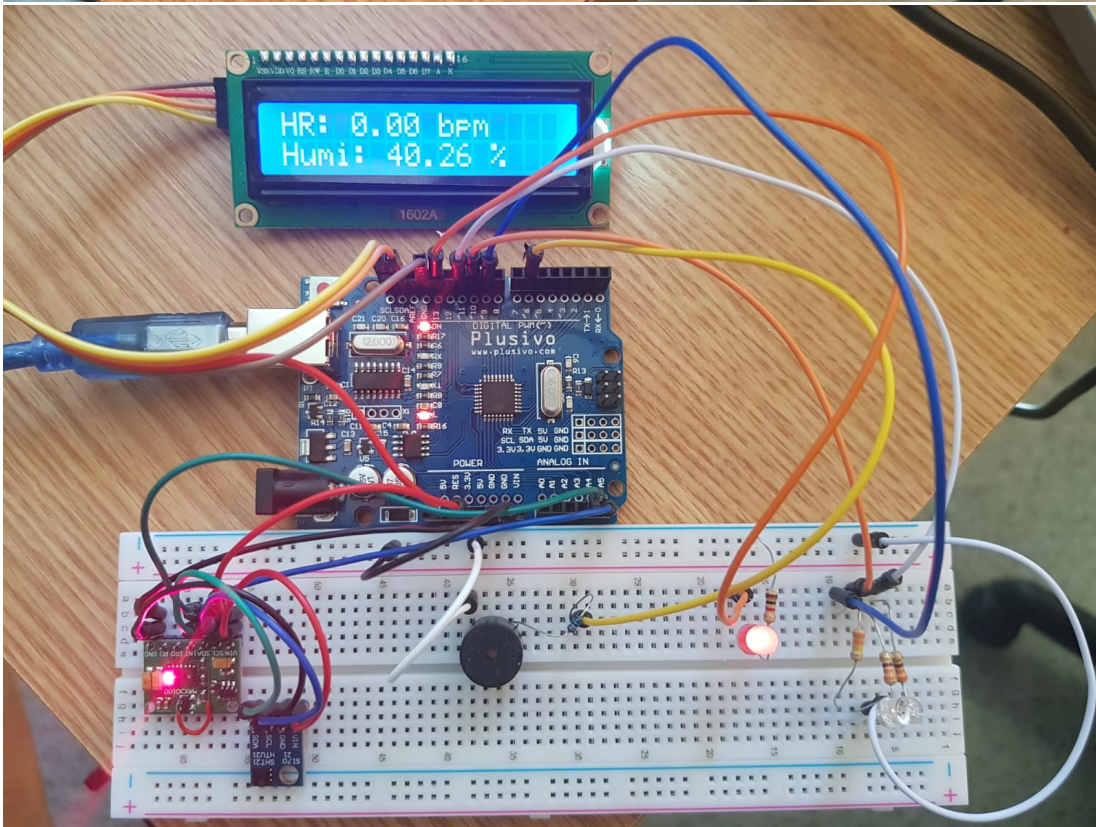
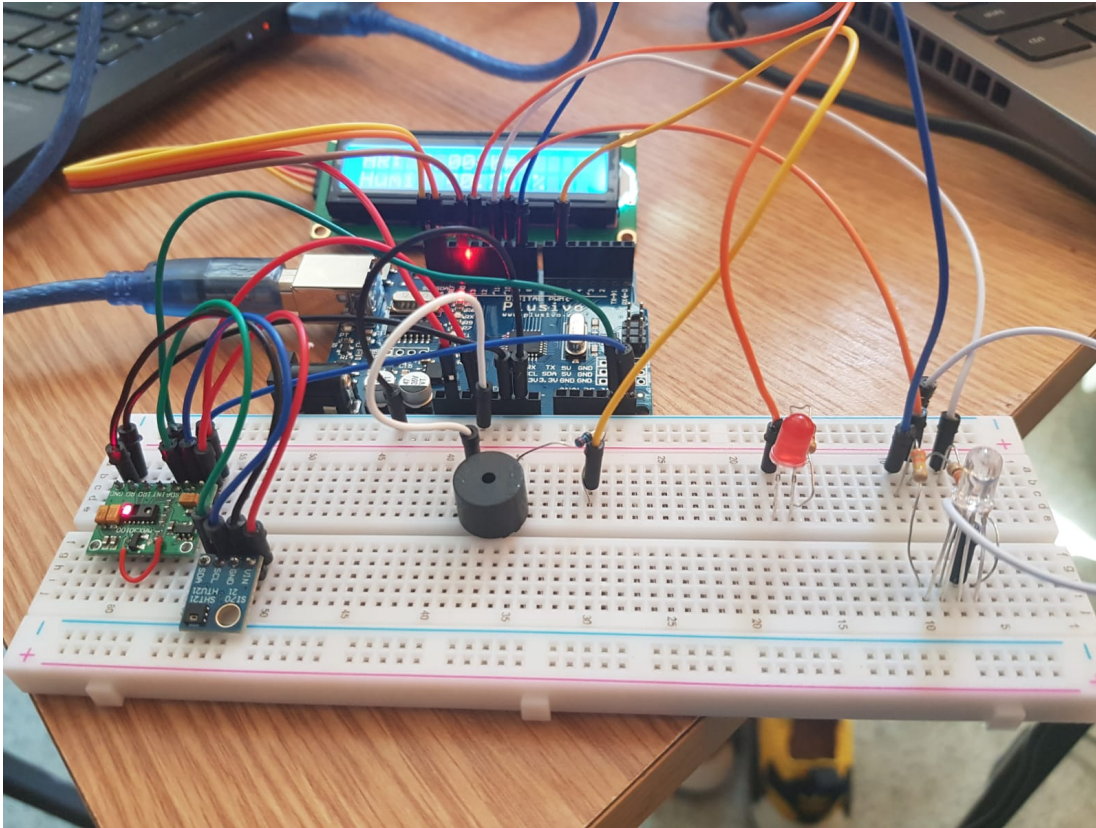
## Funcții

- *setup()* → initializarea pinilor pentru buzzer si LED-uri, initializarea ecran LCD, initializarea senzorului de puls, inregistrarea functie de callback pentru fiecare puls
- *loop()* → logica principala a programului: apeleaza *pox.update()* la fiecare 3 secunde calculeaza pulsul si umiditatea si afiseaza aceste informatii pe ecran. Daca avem o valoare a pulsului valida (considerata de mine peste 60 bpm) atunci v-om adauga valoarea plusului si a umiditatii la cate o suma si vom incrementa counterul. Se apeleaza si functia *make\_choice()*
- *make\_choice()* → Cand un counter declarat global ajunge la o valoare predefinita se va calcula media valorilor pentru puls si umiditate si se vor compara fiecare cu un threshold in functie de care se da verdictul de minciuna sau adevar si se apeleaza functia corespunzatoare.
- *sing\_lying\_song()* → se afiseaza pe ecran mesajul "You are lying!" si se porneste buzzer-ul care canta melodia corespunzatoare, apoi se reinitializeaza counterul, *tsLastReport* si *pox*;
- *sing\_telling\_the\_truth\_song()* → se afiseaza pe ecran mesajul "You are telling the truth!" si se porneste buzzer-ul care canta melodia corespunzatoare, apoi se reinitializeaza counterul, *tsLastReport* si *pox*;
- *onBeatDetected()* → functie callback care face un LED rosu si buzzerul sa se porneasca de fiecare data cand detecteaza un puls
- *RGB\_color(red, green, blue)* → seteaza pinii LED-ului RGB cu valorile date ca parametri

Senzorul puls-oximetru trebuie interogat cat mai rapid, altfel datele din buffer-ul sau se pierd. Conform bibliotecii pe care am folosit-o, actualizarea ar trebui facuta la aproximativ 100Hz. Astfel, in cod trebuie apelat cat mai des *pox.update()*

## Rezultate Obținute

### Circuit final



## Demo

Accesand acest [link](#) puteti gasi un demo pe YouTube in care prezint functionalitatile proiectului.

## Concluzii

A fost o experienta interesanta sa lucrez la acest proiect. De la gaisrea unei idei pana la cumpararea si asamblarea componentelor si scrierea codului si a documentatiei. Am avut cateva probleme cu senzorul de puls care mi-au aratat importanta unui senzor de calitate si necesitatea uneori de a modifica anumite aspecte ale componentelor cumparate pentru a functiona in concordanta cu anumite valori ale microcontrollerului. (adaugarea unui jumper pentru ca SDA si SCL sa fie alimentate de la 3.3V la senzorul de puls) si mai ales nevoia de a citi datasheet-ul si documentatia pentru bibliotecile folosite pentru a sti exact cu ce ai deaface.

## Download

[lie\\_detector.zip](#)

## Jurnal

- 2.05.2022 → alegerea proiectului
- 9.05.2022 → schema bloc
- 10.05.2022 → pagina wiki - milestone 1
- 20.05.2022 → finalizare componenta hardware si software a proiectului
- 27.05.2022 → schema electrica in Eagle
- 30.05.2022 → finalizare documentatie

## Bibliografie/Resurse

[Export to PDF](#)

- <sup>1)</sup> <https://how2electronics.com/interfacing-max30100-pulse-oximeter-sensor-arduino/>
- <sup>2)</sup> <https://www.arduino.cc/en/software>
- <sup>3)</sup> <https://code.visualstudio.com>
- <sup>4)</sup> <https://app.diagrams.net>
- <sup>5)</sup> <https://www.autodesk.com/products/eagle/overview?term=1-YEAR&tab=subscription>
- <sup>6)</sup> <https://www.arduino.cc/reference/en/language/functions/communication/wire/>
- <sup>7)</sup> <https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/>
- <sup>8)</sup> <https://github.com/oxullo/Arduino-MAX30100>
- <sup>9)</sup> <https://github.com/e-radionicom/SHT21-Arduino-Library>

From:

<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:

[http://ocw.cs.pub.ro/courses/pm/prj2022/avaduva/lie\\_detector](http://ocw.cs.pub.ro/courses/pm/prj2022/avaduva/lie_detector)



Last update: **2022/06/01 12:28**