Lie Detector

Autor: Ștefania Cristiana Olteanu
Grupa: 336CA

Introducere

Un test poligraf se bazeaza pe masurarea si inregistrarea unor parametrii fiziologici ai unui individ, cum ar fi pulsul, temperatura, respirația, perspiratia și conductivitatea pielii, in timp ce i se pun o serie de intrebari. Inca din cele mai vechi timpuri se considera ca a minti determina efecte fiziologice colaterale. In acest proiect doresc sa simulez un “detector de minciuni” care desi nu intotdeauna obtine rezultate corecte (unii oameni isi pot controla in mod constient reactia corpului lor) este un experiment interesant in a masura anumiti parametrii ai corpului si a vedea cum sau daca acestia se modifica.

Descriere generală

Prin intermediul unui senzor de masurarea a pulsului si a un senzor de temperatura voi obtine informatiile necesare unui verdict asupra valorii de adevar a raspunsului unui individ (simularea unui test poligraf). Valoriile vor fi analizate si pe baza acestora va fi declansata una din urmatoarele stari:

- **LIE** → LED-ul se face rosu, pe ecranul LCD se va afisa mesajul: “You are lying!” si buzzerul va canta Never Gonna Give You Up - Rick Astley
- **TRUTH** → LED-ul se face verde, pe ecranul LCD se va afisa mesajul: “You are telling the truth!” si buzzerul va canta 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 [1)]. 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 [2)]
  - VS Code [3)]
- pentru realizarea schemei-bloc:
  - draw.io [4)]
- pentru realizarea schemei electrice:
  - EAGLE [5)]

Biblioteci folosite:

- Wire [6)]
- LiquidCrystal_I2C [7)]
- MAX30100_PulseOximeter [8)]
- SHT21 [9)]

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)

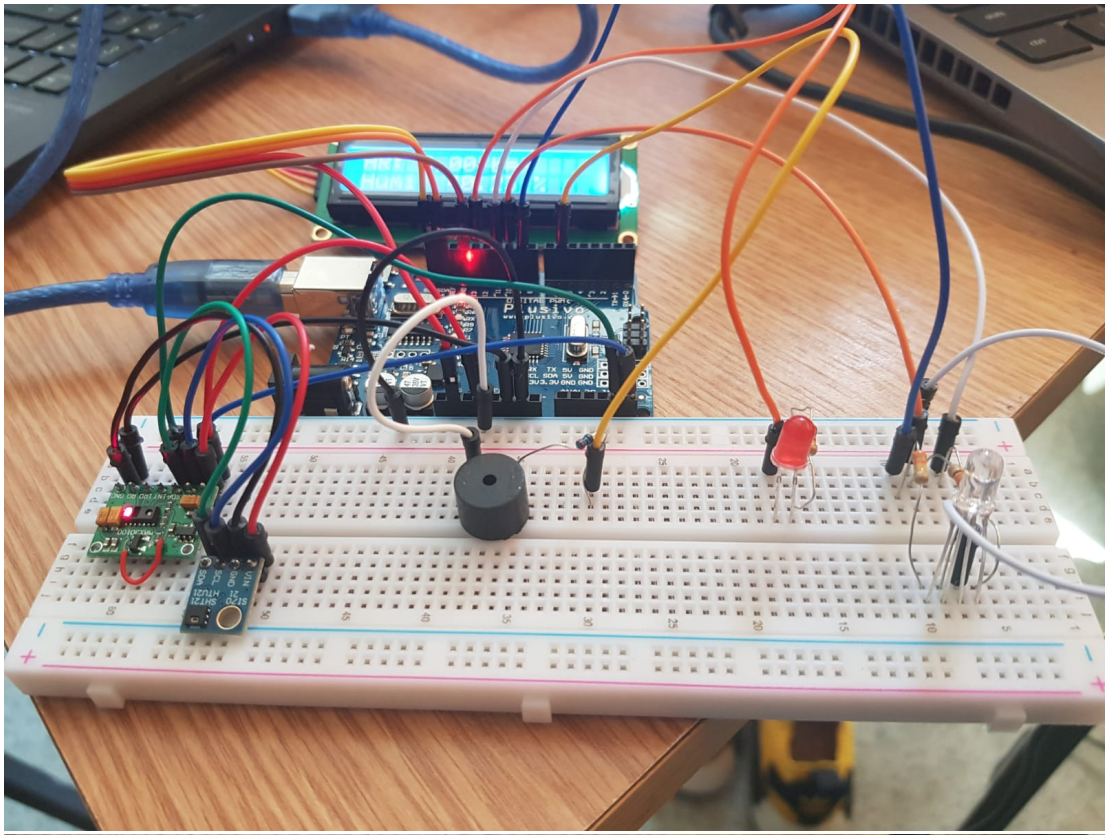
Functii

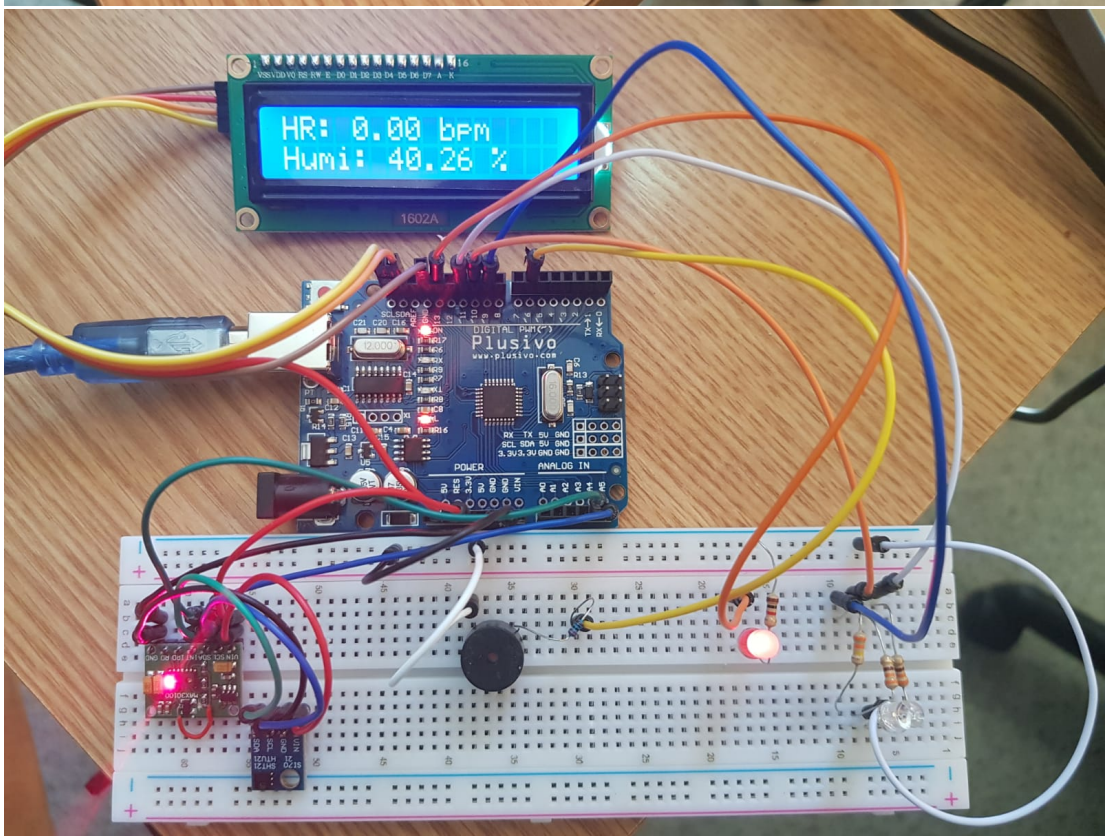
- *setup()* → initializarea pinilor pentru buzzer si LED-uri, initializarea ecran LCD, initializarea senzorului de puls, inregistrarea functie de callback pentru fiecare puls
- *loop()* → logica principala a programului: apeleaza *pox.update()* la fiecare 3 secunde calculeaza pulsul si umiditatea si afiseaza aceste infromatii 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 thershold 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 Obţinute

Circuit final





Demo

Accesand acest link puteți găsi un demo pe YouTube in care prezint funcționalitățile proiectului.

Concluzii

A fost o experienta intersanta sa lucrez la acest proiect. De la gaisrea unei idei pana la cumpararea si asamblarea componetelor 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

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