Parking obstacle detector

Student: Catruc Ionel 332CBa

Introducere

Asistent de parcare bazat pe 4 senzori pentru fiecare parte a unei masini (Fata, spate, stanga, dreapta) care vor verifica si detecta aparitia unor obstacole pe una din parti si va semnala asta cu un sunet sonor si aprinderea unui led intr-o anumita culoare (in functie de distanta pana la obstacol). De asemenea, pe un display se va afisa starea celor 4 parti, ex. Right - Safe, Front - Medium, Behind - Critical. De asemenea va exista un buton pentru activarea modului de parcare. In caz de apropiere de obstacol, in functie de distanta, buzzerul va emite sunete de avertizare.

Descriere generală

Odata apasand pe buton pentru intrarea in modul de parcare, senzorii vor incepe sa scaneze si sa detecteze obstacolele. Cei patru senzori vor comunica distanta fata de obstacole, iar in functie de asta, “conducatorul” va fi avertizat astfel:

- led-ul RGB se va aprinde intr-o culoare, in functie de cea mai mica distanta pana la obstacole comunicata de vreun senzor, astfel ca Verde va fi distanta safe, Galben - distanta medie, Rosu, distanta critica
- ecranul LCD va afisa informatia de pe cei 4 senzori in formatul urmator:

LEFT : MEDIUM	RIGHT : SAFE
FRONT : MEDIUM	BACK : CRITICAL

- buzzerul, in functie de distanta, va scoate un sunet la o frecventa anumita in functie de distanta, cum am descris la LED.

Schema bloc:

Hardware Design

Lista piese:

- breadboard
- 4x ultrasonic senzor HC-SR04

- Arduino uno
- 1x buzzer
- 1x RGB led (anod comun)
- 3x rezistente pentru LED RGB (330 ohm)
- Fire M-T, T-T
- 1x buton
- 1x ecran LCD (I2C)

Schema electrica:

Software Design

Descrierea codului aplicaţiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării şi surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi şi structuri pe care plănuiţi să le implementaţi
- (etapa 3) surse şi funcţii implementate

Mediu de dezvoltare

Arduino IDE 2.1.0

Librarii folosite

- #include <LiquidCrystal_I2C.h> - pentru a realiza comunicarea cu display-ul LCD prin I2C

Descriere surse si functii implementate

- **customDigitalRead(pin)** - digital read implementat pe registri (GPIO)
- **customDigitalWrite(pint, value)** - digital write implementat pe registri (GPIO)
- **customPinMode(int pin, int mode)** - custom pin mode (GPIO)
- **setupUltrasonic(int TRIG_PIN, int ECHO_PIN)** - seteaza pinul trig pin pe output si echo pin pe input (pentru senzorul ultrasonic)
- **getDistance(int TRIG_PIN, int ECHO_PIN)** - calculeaza distanta obtinuta de la un senzor

ultrasonic cu pinii echo si trig primiti ca parametru
- **getRangeString(int distance)** - primeste stringul asociat distantei (CRITIC, MEDIUM, SAFE)
- **print_on_display()** - printeaza pe display starea sistemului. Daca suntem in mod de IDLE, va afisa “Press button”, daca e in mod PARKING, va afisa starea pentru fiecare senzor in functie de distanta pana la obstacole. Face update la stare pe display doar atunci cand starea curenta e diferita de starea precedenta (sa nu scriu la fiecare iteratie pe display aceleasi chestii)
- **ledWarning()** - in functie de starea sistemului, coloreaza ledul rgb in Red, Green, Orange (in functie de stare Critical, Safe, Medium) sau in culoarea de IDLE daca nu suntem in mod parcare. Folosesc LED RGB cu anod comun, deci nu voi scrie x ci 255 - x.
- **ISR(TIMER1_COMPA_vect)** - rutina timer1 pentru buzzer. Aceasta rutina va face ca buzzerul sa scoata sunet in dependenta de frecventa configurata per stare.
- **conf_critic_timer()** - configurare timer pentru distanta critica. Timer1 cu frecventa 7 hz.
- **conf_safe_timer()** - configurare timer pentru distanta safe. Mai exact, se dezactiveaza timer-ul pentru ca suntem la o distanta sigura fata de vreun obstacol
- **conf_medium_timer()** - configurarea timer1 pentru distanta medie, anume 4 hz.
- **ISR(INT0_vect)** - intrerupere care se activeaza pe Rising Edge al INT0, folosita la buton. Pentru a previne detectarile multiple, am setat un debounce de 200ms.
- **setup()** - initializez pinii necesari sistemului, cat si intreruperile necesare, display-ul.
- **loop()** - daca sistemul e in mod parcare, extrage distanta de pe toti senzorii, calculez distanta minima din cele 4, si compar cu distanta minima anterioara, daca sunt diferite, extrag starea curenta, adica, safe, medium, critical, si in functie de asta, configurez timerul necesar.

Rezultate Obţinute

Sistemul functioneaza perfect si detecteaza obstacolele.

Avem doua stari, IDLE si PARKING.

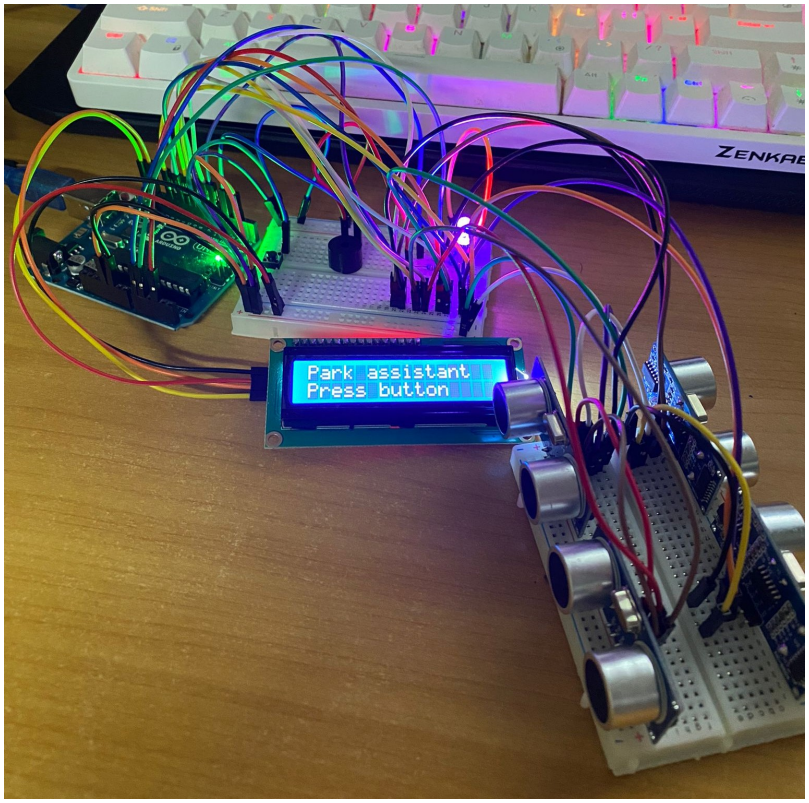
In starea idle nu se verifica prezenta obstacolelor (putem asocia asta cu faptul ca nu avem nevoie de asta, nu ne parcam).

Starea de parking va incepe verificarea obstacolelor si calcularea distantei pana la ele.

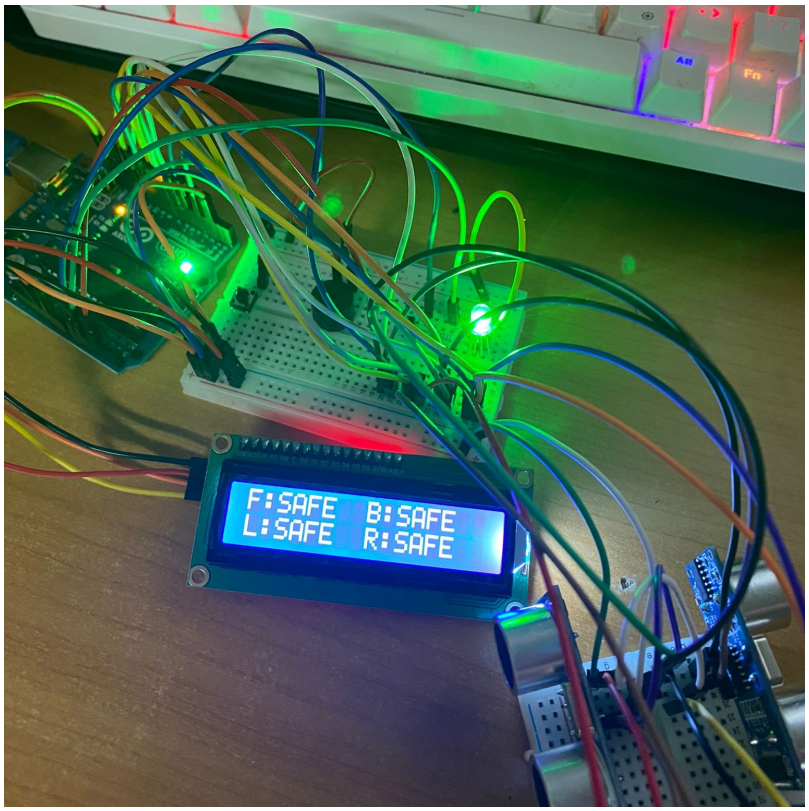
Am 3 nivele: Critic (obstacol intre 2-10 cm), Medium (obstacol intre 10-20 cm) si Safe, cand nu este nici un obstacol sau se afla peste limita de 20 cm. Am configurat limita la 20 pentru ca mereu detecta obiecte in jur la testare pe masa. In functie de distanta, aprinde ledul in rosu, portocaliu sau verde. Buzerrul va semnala daca avem vreun obstacol prea aproape, iar cu cat mai aproape se afla obstacolul, cu atat mai des va buzzui.

Demo: https://www.youtube.com/shorts/B_JiHE_W5l0

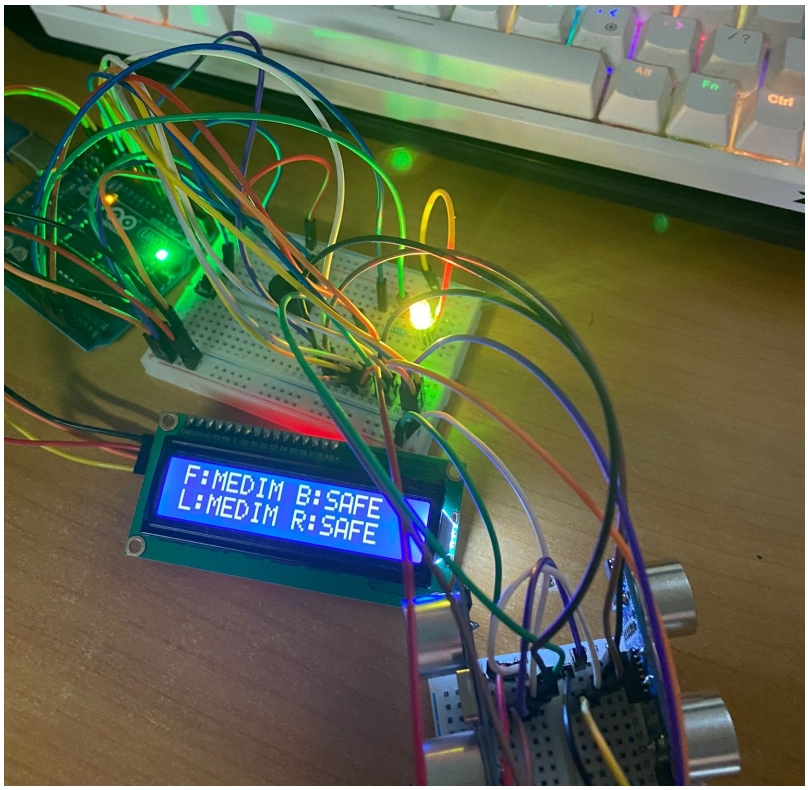
Starea de start



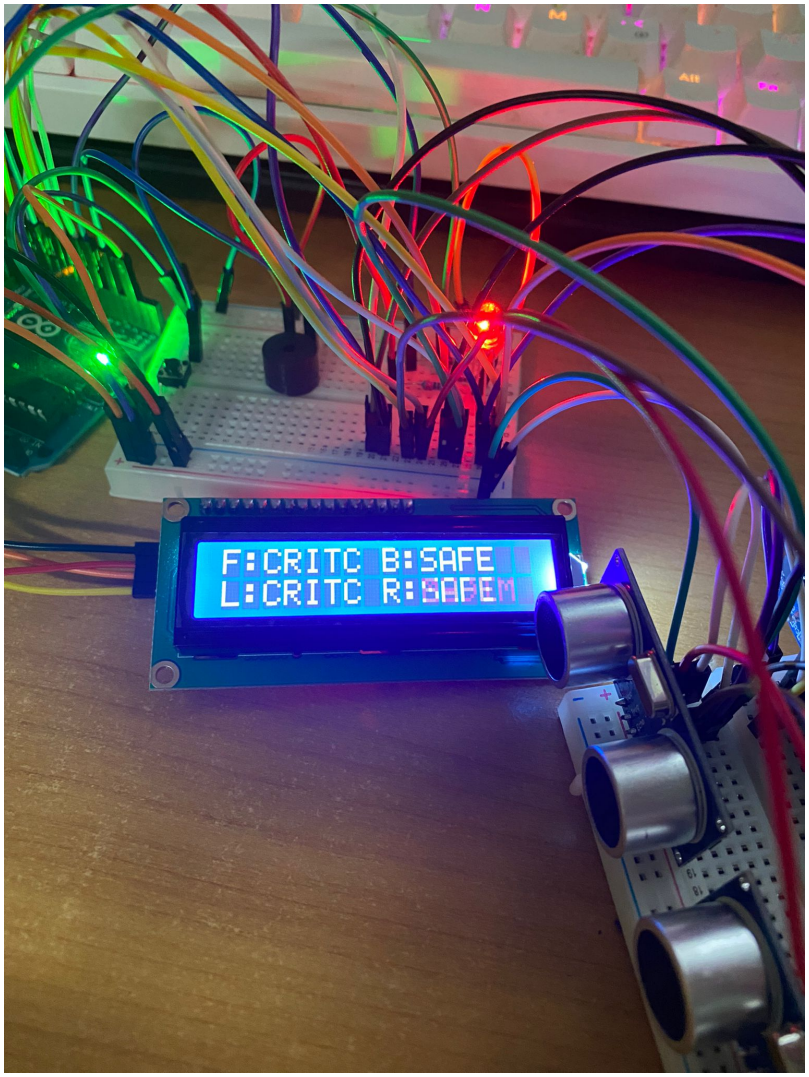
Nici un obstacol



Distanta medie



Distanta critica



Concluzii

- Proiect foarte interesant
- Mai usor sa faci totul pe un tool online (TinkerCad, CircuitoIO, Wokwi) si dupa sa treci totul pe placa
- Mi-am luat destul de multe piese de rezerva, dar nu mi-au trebuit
- Cabluri, cabluri, cabluri... Are cineva cabluri?
- Destul de complicat sa bagi totul intr-o cutie si sa fac sa arate prezentabil...
- Uneori simularea nu reflecta realitatea.

Download

Codul sursa : proiect_catruc_ionel_332cba.zip

Jurnal

- 26.04.2023 - Alegere tema proiect
- 04.05.2023 - Primire piese necesare proiectului
- 16.05.2023 - Testare piese, asamblare hardware (software - arduino library)
- 20.05.2023 - Trecere de la arduino framework la implementare pe registri. Folosire timere, intrerupere, GPIO etc.
- 28.05.2023 - Documentatia finala.

Bibliografie/Resurse

- https://arduinogetstarted.com/tutorials/arduino-lcd-i2c
- https://www.hackster.io/techmirtz/using-common-cathode-and-common-anode-rgb-led-with-arduino-7f3aa9
- Laboratoarele OCW
- https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Export to PDF