SmartCar

Introducere

SmartCar este o mașinuță care poate fi controlata prin bluetooth cu telefonul printr-o aplicație și dispune de un sistem pentru detectarea obstacolelor și evitarea coliziunii cu acestea. În cazul în care un obstacol este în față, mașina se va opri, iar dacă se află pe lateral (dar cu posibilitatea de a lovi mașina) mașina va încerca să reducă viteza și să vireze spre dreapta sau stânga pentru a evita obstacolul. De asemenea va avea posibilitatea de a urmări obiectul din față, în linie dreaptă, menținând constantă distanța dintre aceasta și mașinuță. În cazul în care nu exista posibilitatea de coliziuni, utilizatorul este cel care are controlul total al mașinuței. Vor exista niște leduri pentru semnalizare, stop, faruri și marșarier.

În trecut am mai construit mașinuțe de jucărie, dar erau foarte simple (mergeau doar înainte) și modificam motoarele și variam tensiunea de alimentare pentru a le face mai rapide.

Astăzi majoritatea mașinilor noi din industria auto (dacă nu chiar toate) dispun de sisteme care care să ajute șoferii în cazuri excepționale. Un acest tip de sistem poate fi folosit în aceste situații (desigur, implementarea este rudimentară în acest proiect) pentru a evita coliziunile cu alte autoturisme sau obstacole.

Descriere generală

Modulul ATmega324p este cel care conține și microcontroller-ul, comandă întreaga mașinuță și trimite date prin interfața USART către modulul bluetooth care face schimbul de informații între telefon și mașinuță. Modulul ATmega324p trimite comenzi (și un semnal PWM) către driverul de motoare pentru a controla sensul și viteza de rotație a acestora. Modulele de senzori trimit către microcontroller date despre mediu, iar acesta le procesează și decide modul în care mașinuța trebuie să reacționeze. Modulul pentru leduri primește semnale de la microcontroller, dacă LED-urile trebuie să se aprindă sau să se stingă, și de asemenea, un semnal PWM pentru semnalizare și un semnal pentru alegerea LED-urilor care trebuie să se aprindă. Pe telefon, va exista o aplicație prin care se poate controla mașinuța (LED-uri, viteza, direcție, activare/dezactivare collision detection system).

Hardware Design

Piese obligatorii

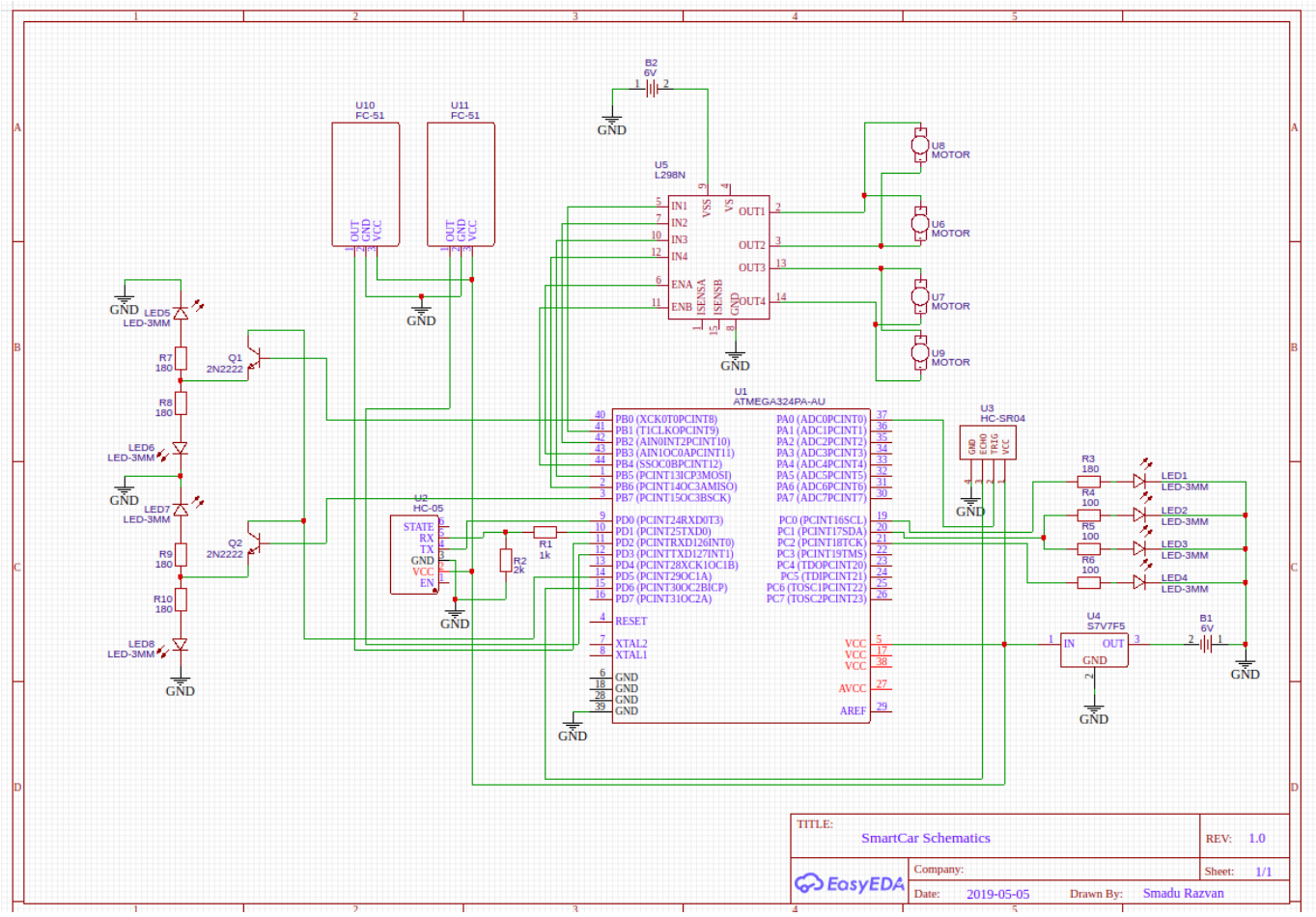
Nume piesă	Număr piese	Preț

ATMEGA324A_PU	1	
USB-B	1	
16MHZ QMIM016	1	
ZENER	2	
LED EL333	2	
K1X10 WSL040	2	
K2X10	1	
PUSHBUTTON	2	
100R	2	
470R	3	
1k5	1	
10k	1	
100nF	3	
15pF	2	
Soclu DIP40	1	
100uF	1	

Piese adiționale

Nume piesă	Număr piese	Preț
HC-05	1	22
Driver Motoare Dual L298N	1	9.99
Senzor ultrasonic HC-SR04	1	4.89
LED Roșu	1	0.49
LED Alb	3	1.47
LED Galben	4	1.96
Rezistor 100	10	1
Rezistor 220	10	1
Rezistor 180	10	1
Rezistor 1k	10	1
Kit Robot cu 4 Motoare	1	79.99
Senzor Infrarosu de Obstacole FC-51	2	8.98
Tranzistor NPN	3	1.47
Breadboard	2	4.78
Sursa Step-Up/Down S7VF5	1	23.99
Fire de conexiune mama-mama	20	6.98
Fire de conexiune mama-tata	20	8.90
Fire de conexiune tata-mama	10	3.49
Suport baterii 4 x R6	1	4.86

Schema electrică



Software design

Scrierea codului s-a facut in Visual Studio Code, iar dezvoltarea s-a facut in totalitate pe Windows. Am folosit pentru USART, scheletul de la laboratorul 1.

Controlarea motoarelor se face printr-un PWM emulat software. Emulare se poate face utilizând un timer de 10kHz care incrementeaza un contor. Contorul este verificat cu o valoare de prag și când este mai mare decât acea valoare se setează semnalul pe 1, altfel pe 0. Aceasta este o implementare software pentru un phase correct PWM. In funcție de comanda data prin Bluetooth, se controlează deplasarea înainte, înapoi, la dreapta sau la stânga. Tot prin transmisia/recepția prin Bluetooth se face prin USART. Senzorul de distanta este folosit astfel: se trimite un impuls de 10us, se așteaptă ca pe echo sa se facă 1 și apoi se măsoară timpul pana se face 0. Aceasta valoare este proporțională cu distanta. Folosind timer-ul, se emuleaza și alte timere. Unul pentru milisecunde și altul pentru secunde. Acest lucru se realizează utilizând timerul de 10kHz și se incrementeaza un contor. În funcție de valoarea la care se ajunge, se realizează o cuanta de timp pentru fiecare timer. Practic este un divizor de ceas.

Deplasarea masinii se face folosind PWM-ul software care e setat in functie de directia de deplasare (pentru inainte se seteaza doar 2 pini cu out de PWM si ceilalti sunt pusi la '0', se seteaza turatia prin modificare Duty Cycle-ului). De asemenea, este un implementat un mecanism de 'frecare' care duce la oprirea masinii dupa un timp scurt in care are semnalul de inainte. Astfel, pt a controla turatia, se trimit comenzi regulat, in functie de turatia dorita.

Pentru a asigura ca masina raspunde rapid la comenzi, s-au folosit intreruperile. Busy-waiting se face

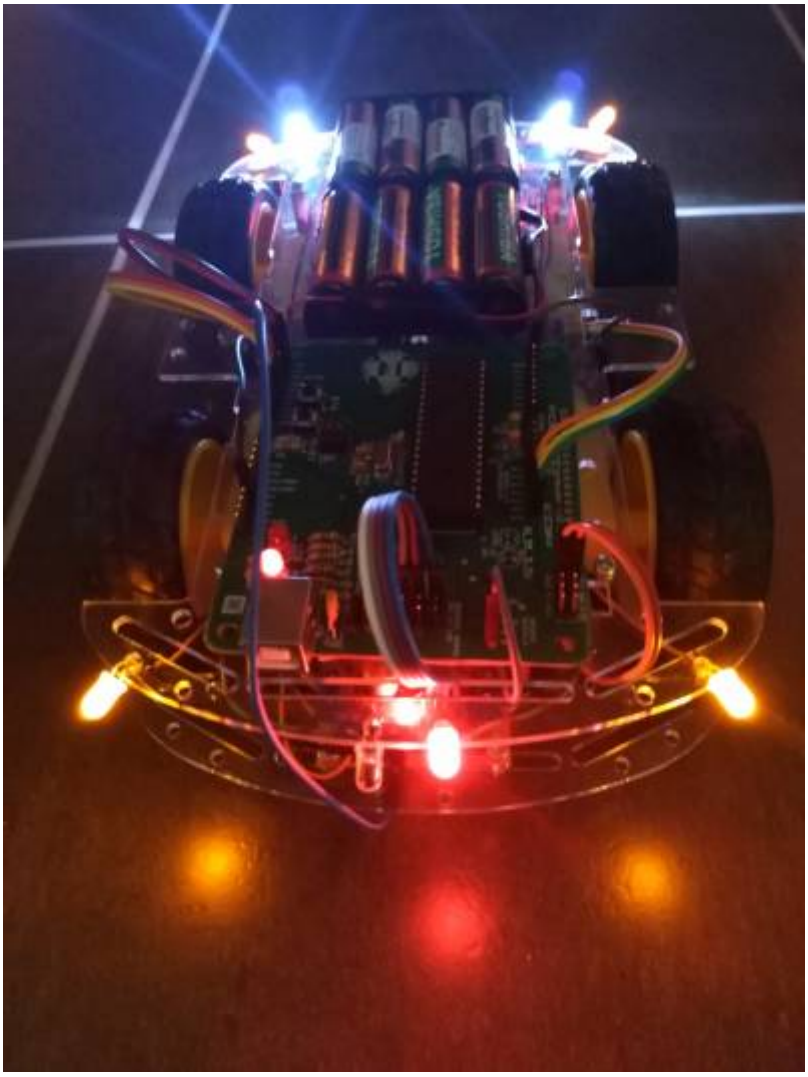
doar atunci cand se genereaza impulsuri la senzorul de distanta, dar acesta oricum ruleaza la mereu.

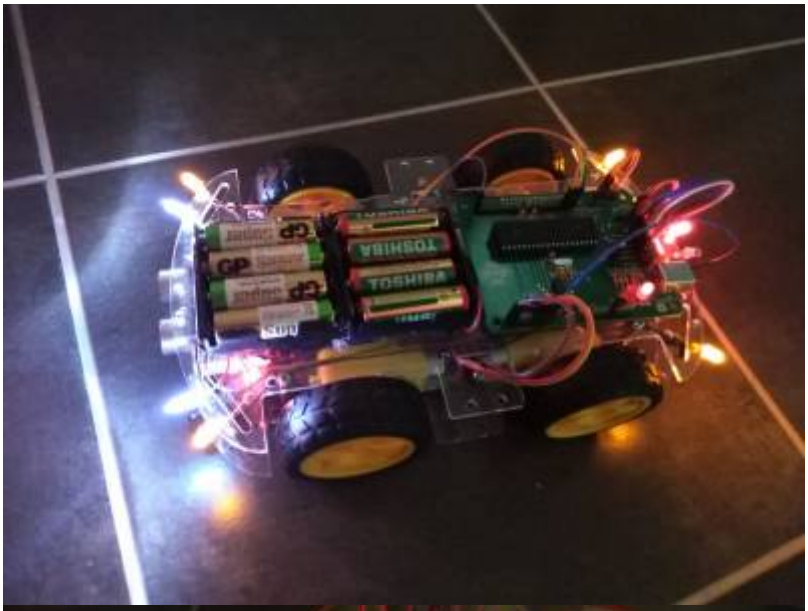
Functiile pe care le poate face masinuta sunt: semnalizare/avarii, aprintere/stingere faruri, deplasare fata, spate cu aprindere led, virare stanga/dreapta, Simulare franare, detectie coloziuni laterale (cu incercarea de a evita obstacolul, virand dreapta/stanga) si frontal (oprind complet masina), deplasarea cu o viteza proportionala cu frecventa semnalului de deplasare.

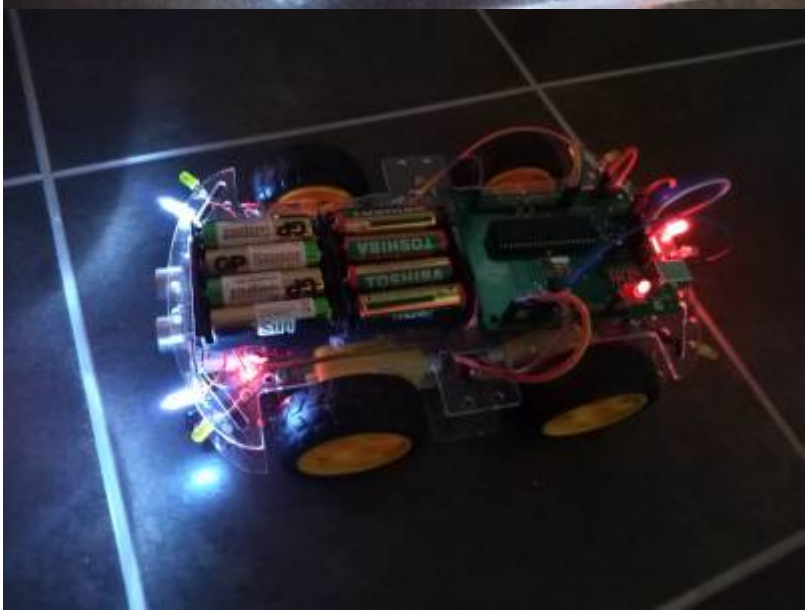
Rezultate Obţinute

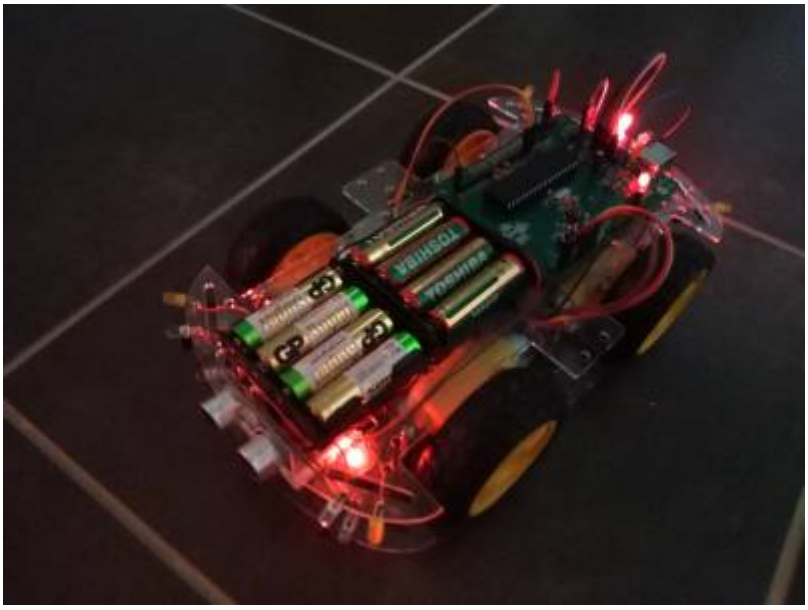
Masinuta se deplasează bine și reușește sa detecteze obstacolele care au o suprafață dreapta. De asemenea, evita obsacolele laterale si poate sa urmeze obiectul din fata, daca acesta merge inainte.

Poze SmartCar:




TOSHIBA
TOSHIBA


TOSHIBA
TOSHIBA



Concluzii

Proiectul a fost unul interesant, din care am avut de învățat multe lucruri. În primul rand, senzorii pot fi foarte ușor influențati de condițiile mediului exterior (raze IR generate se soare) care influențează în mod negativ “percepția” mașinii asupra mediului. Un alt lucru a fost pe partea de programare. Instructiunile au un timp relativ mare de execuție, și dacă folosești un timer prea mic practic la o întrerupere durează mai mult sa se execute instructiunile decât timpul în sine. Și în ultimul rand, trebuie sa fii atent cum configurezi pinii, altfel poți sta foarte mult pe debugging.

Bibliografie

[1] http://cs.curs.pub.ro/wiki/pm

[2] https://www.electronicwings.com/avr-atmega/ultrasonic-module-hc-sr04-interfacing-with-atmega1632

Resurse

smadu_razvan_smartcar.pdf

smartcar.zip