Mihai-Adrian SPĂTARU (78428) - LineFollower "Caramida"

Autorul poate fi contactat la adresa: **Login pentru adresa**

Introducere

Prezentarea pe scurt a proiectului vostru:

- ce face
- care este scopul lui
- care a fost ideea de la care aţi pornit
- de ce credeţi că este util pentru alţii şi pentru voi

Robotul de tip LineFollower este un robot clasic, care urmareste o linie neagra pe un fundal alb (sau invers, depinzand de variantii). Scopul acestuia este de a termina circuitul intr-un timp cat mai scurt. Am ales acest proiect, intrucat am mai facut roboti asemanatori, insa niciodata programati direct in C pe registrii (AVR) si ma astept ca acest lucru sa fie un “challenge”. In urma acestui proiect sper sa imi extind intelegerea asupra procesarii datelor de catre un robot de LineFollower intrucat acest limbaj este mai apropiat de procesor decat limbajul folosit inainte (Arduino).

Descriere generală

O schemă bloc cu toate modulele proiectului vostru, atât software cât şi hardware însoţită de o descriere a acestora precum şi a modului în care interacţionează.

Hardware Design

Aici puneţi tot ce ţine de hardware design:

- listă de piese
- scheme electrice
- diagrame de semnal
- rezultatele simulării

Microcontroller ATMEGA 328

Modul cu Driver de Motor Dual (L298N)

Bară de Senzori Infraroșu Reflectivi (QTR-8A)

7805 (la care am renuntat pana la urma, intrucat era inclus in driver)

2 motoare (10:1)

2 roti

1 baterie 12V

1 sasiu (pcb taiat)

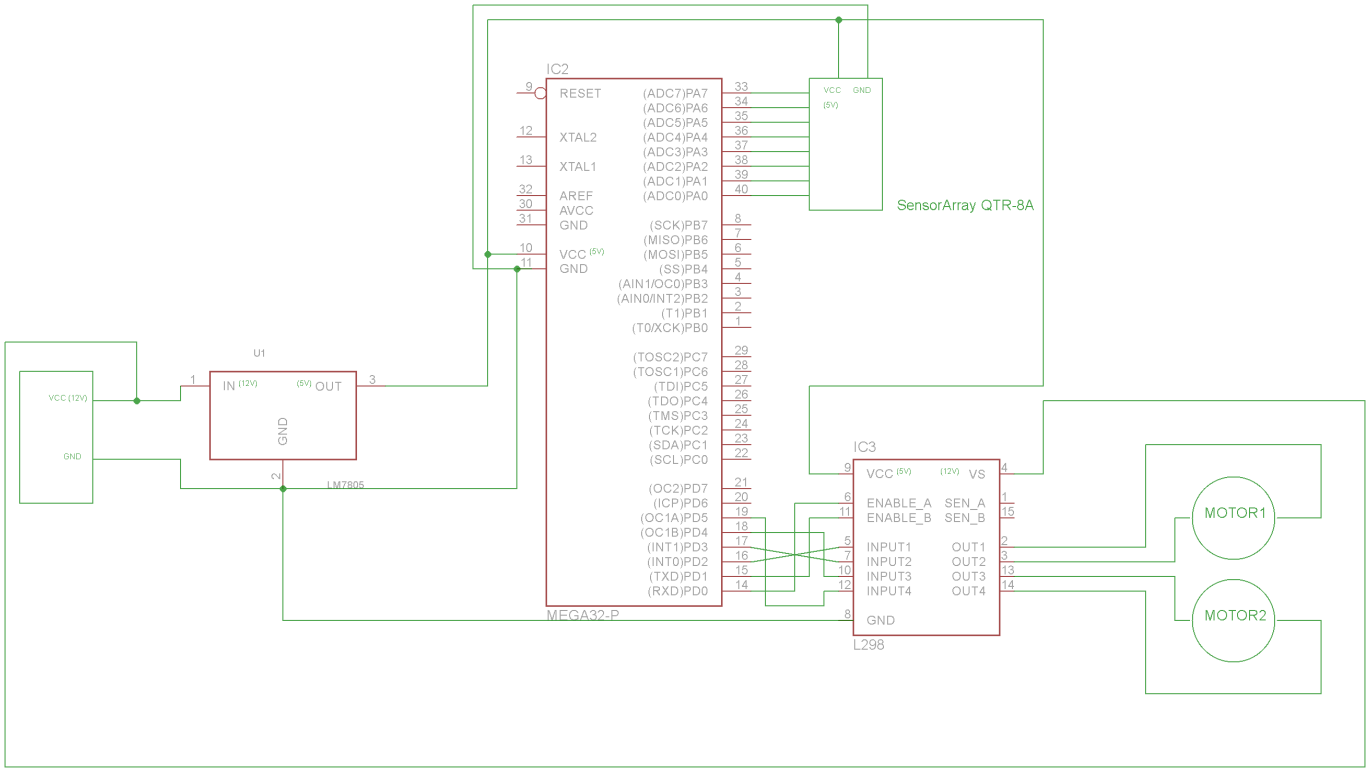
1 Ball Caster

pini tata-tata

fire mama-mama si fire simple

1 dioda

suruburi + piulite



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

Software-ul robotului a fost creat in Notepad++, compilat in cmd folosind WinAVR si uploadat pe placuta folosind HIDBootFlash.

Algoritmul programului este unul destul de simplu. Se citesc senzorii, se calculeaza eroarea si se dau pe motoare valorile necesare pentru a corecta eroarea. Initial am vrut sa implementez si algoritmul PID, insa intrucat viteza roborului este relativ mica, iar oscilatiile mici, am renuntat la idee.

Pentru calculul erorii, am creat un vector cu ponderi (factor[8] = {-3*k, -2*k, -1*k, -0, 0, 1*k, 2*k, 3*k}), unde k=7 (ales dupa cateva incercari). Programul va citi inputul de pe pini, iar daca pe pinul i vede “negru”, va adauga la eroare (initial 0) factor[i].

Am creat, de asemenea, o functie de PWM de mana (am scris mai jos de ce nu am folosit PWM clasic) in care, un interval de timp de 20 ms l-am impartit in p ms is 20-p ms. Pentru p ms activez pinul de output specific motorului si sensului pe care il vreau, iar pentru 20-p ms, il sting. Am scris aceasta functie de 4 ori, pentru fiecare pin. Programul apeleaza functia specifica folosind motorRightForward(7 - err) si motorLeftForward(7 + err). Astfel, daca linia este in dreptul senzorilor din dreapta, atunci err>0, iar motorul din stanga va da mai multa viteza, iar cel din dreapta mai

putina; daca linia este in dreptul senzorilor din stanga, atunci err<0, iar motorul din stanga va da mai putina viteza, iar cel din dreapta mai multa. Cu cat err este mai mare in modul, cu atat diferenta va fi mai mare.

Am abordat de asemenea 2 cazuri speciale:

→ toti senzorii citesc “negru”: exista deci o portiune de traseu unde 2 linii se suprapun, asa ca robotul va merge inainte, cu aceeasi viteza pe ambele motoare

→ toti senzorii citesc “alb”: robotul a pierdut linia. Pentru a reajunge pe linie, verificam valoarea lui err (inainte sa o suprascriem) si avem 2 cazuri:

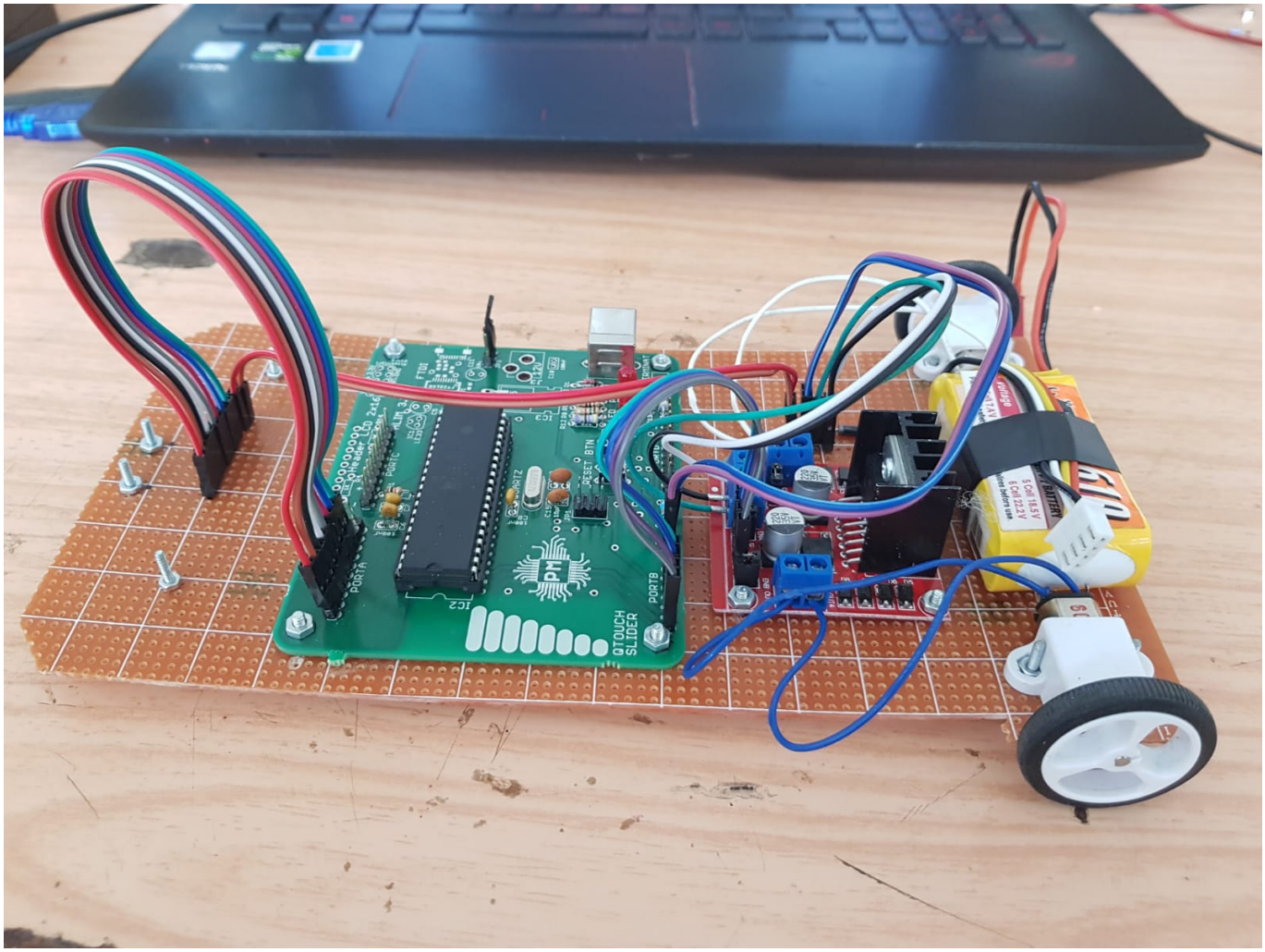
1. err>0, deci linia a fost vazuta ultima data de senzorii din dreapta, asa ca vom merge inainte cu motorul din stanga si cu spatele cu motorul din dreapta
2. err<0, deci linia a fost vazuta ultima data de senzorii din stanga, asa ca vom merge inainte cu motorul din dreapta si cu spatele cu motorul din stanga

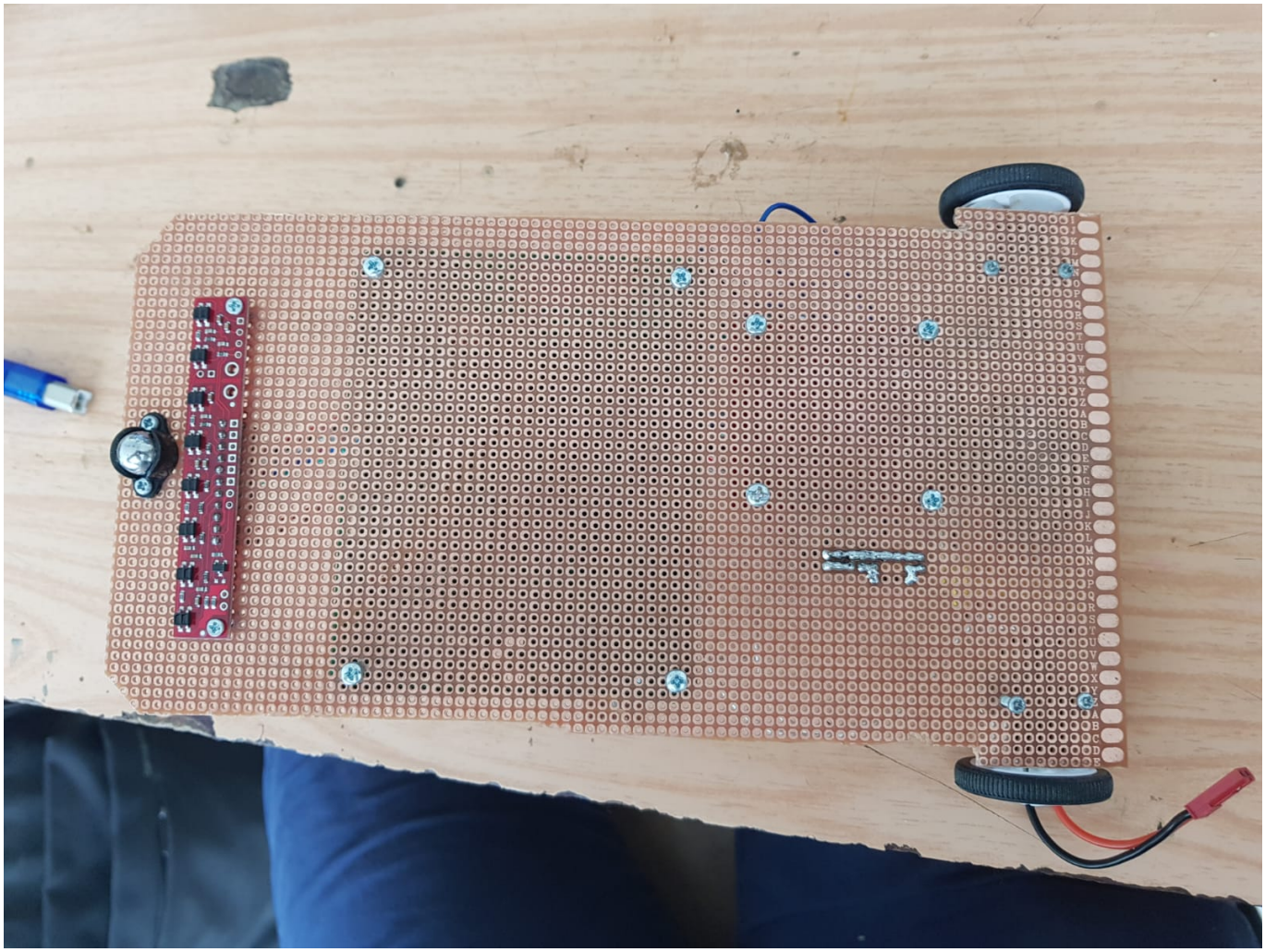
Nota: Am incercat 1 zi intreaga sa fac PWM-ul sa mearga asa cum este prezentat in laborator, insa nu a mers de niciun fel (ori motorul nu mergea deloc, ori la putere maxima). Am cautat alte tutoriale pe net, am discutat cu colegi ce foloseau acelasi driver de motoare, insa nimic. Mi-am prezentat problema catre 2 laboranti diferiti, insa nu am reusit sa identificam problema impreuna. Unul dintre ei mi-a recomandat insa sa scriu PWM-ul de mana, ceea ce am si facut pana la urma (chiar daca este o solutie mai “urata” dpdv al implementarii).

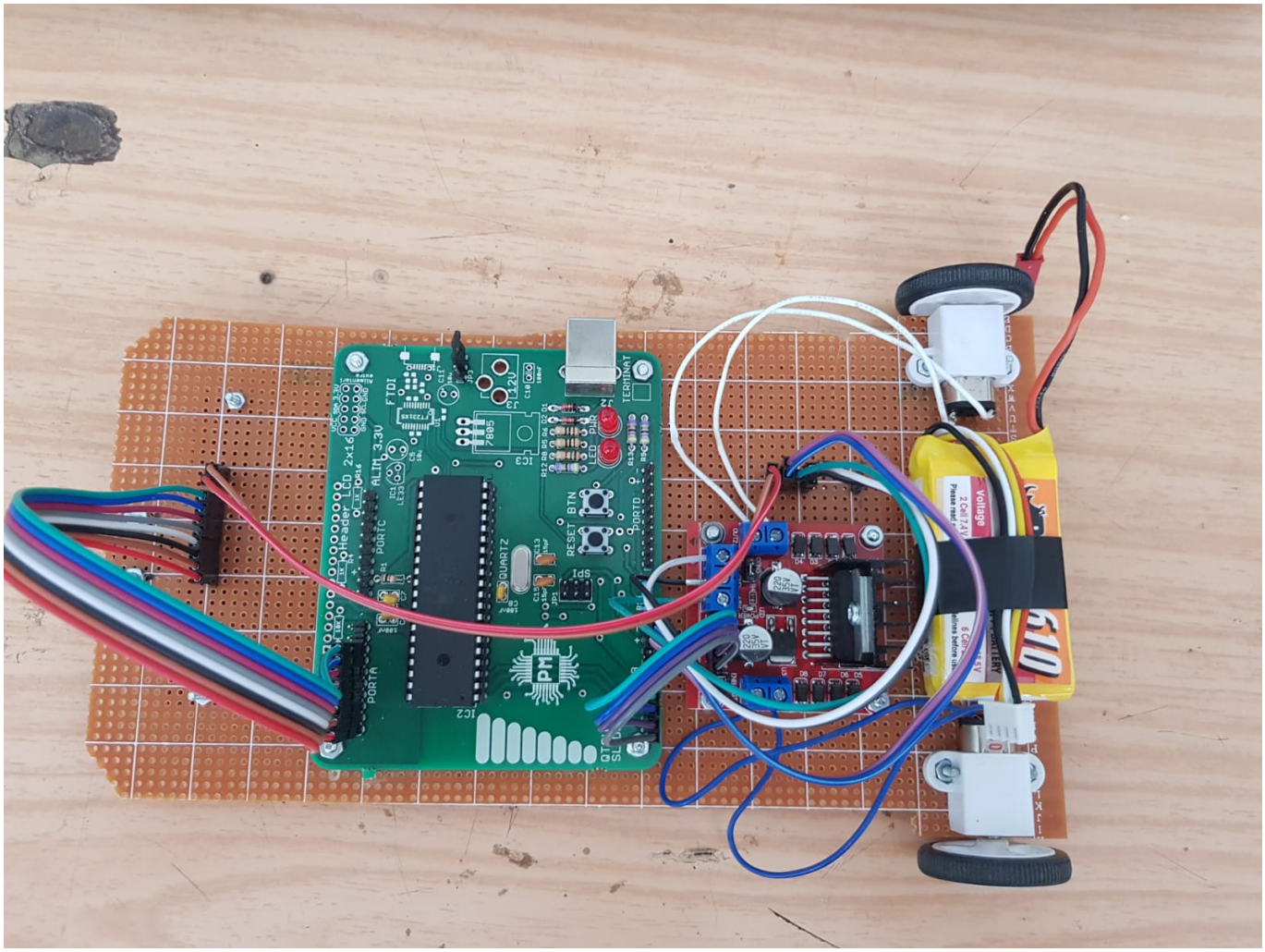
Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Robotul final este unul functional, chiar daca arata ca o caramida (de aici si numele).


PORTA
PORTB
QTOUCH
SLIDER
RESET BTN





Concluzii

Am mai facut 2 LineFollower-uri pana acum (folosind Arduino ProMini insa), asa ca nu prea am avut probleme la partea de hardware, ba chiar am incercat sa o simplific cat mai mult posibil. Problemele de la partea de software au constat in principal in a face PWM-ul sa mearga, care a esuat. Nici acum nu stiu de ce nu functioneaza. Per total, proiectul a fost unul interesant, care m-a fortat sa invat sa programez un microcontroller, chiar si atunci cand nu am “wrapper-ul” Arduino care sa ma ajute.

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2017:avoinescu:dumitru_alin**.

proiectpm.zip

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Laboratorul de PM cu PWM: http://cs.curs.pub.ro/wiki/pm/lab/lab3

Tutoriale de PWM gasite pe internet, dintre care, mai importante:

→ http://robotika.yweb.sk/skola/AVR/visionrobo%20com/PWM%20in%20AVR%20v1.0.pdf

→ http://maxembedded.com/2012/01/avr-timers-pwm-mode-part-ii/

Documentatie L298N:
https://tronixlabs.com.au/robotics/motor-controllers/l298n-dual-motor-controller-module-2a-australia/

- Documentația în format PDF