

Cătălin OLARU (66873) - Robotic Arm

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

Introducere

Proiectul are ca scop realizarea unui brat robotic, capabil sa apuce obiecte. Ideea mi-a fost propusa de asistentul de laborator, iar mie mi s-a parut interesanta intrucat nu presupune doar scris cod, ci si ceva mecanica. In ceea ce priveste utilitatea proiectului, aceasta nu prea exista, fiind mai mult un proiect din care sper sa invat lucruri.

Descriere generală

Microcontrollerul va primi input de la un set de butoane, asezate pe o telecomanda. La randul lui, aceste va actiona 4 servomotoare, care vor roti baza mobila a bratului, vor misca bratele, respectiv vor actiona gripperul.

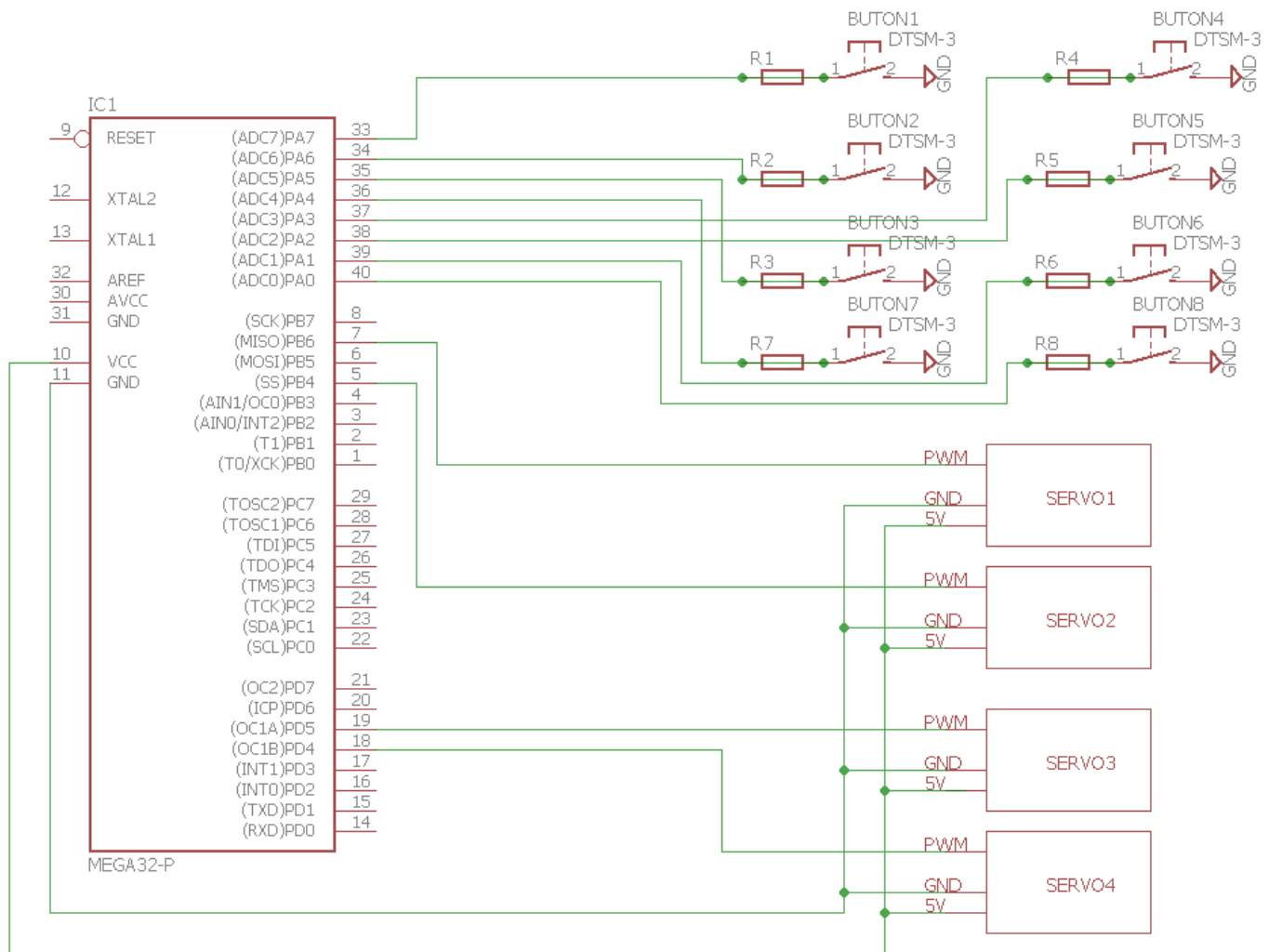
Schema bloc



Hardware Design

Nume componenta	Cantitate	Distribuitor
Placa de baza PM	1	Echipa PM
Butoane	6	Optimus Digital
Servomotoare MG996	4	Optimus Digital
Placi plexiglass	1	Optimus Digital
Fire mama-mama	10	Optimus Digital
Fire mama-tata	20	Optimus Digital

Schema Electrica



Software Design

Mediu de dezvoltare

Dezvoltarea a fost facuta in Programmer's Notepad iar compilarea cu avr-gcc. Nu am folosit nicio librarie 3-rd party.

Istoric implementare

Am pornit implementarea prin a incerca sa controlez un singur servomotor din cod. Intrucat servomotoarele folosite de mine nu aveau absolut nimic legat de duty-cycle-ul necesar pentru a le controla in datasheet, aceasta etapa a fost mai mult trial and error, intrucat a trebuit sa determin

empiric factorii de umplere.

Dupa ce am stat aproximativ o zi si jumătate ca sa determin duty-cycle-ul (mi-a fost si frica sa nu ard servomotorul, asa ca testarea a mers destul de greu), am determinat ca pentru a efectua o rotatie full stanga sunt necesare 0.688ms (dintr-o perioada de 20ms) iar pentru full dreapta 2.36ms. Merita mentionat ca motoarele functioneaza la o frecventa de 50Hz.

Dupa ce am gasi aceste date, pasul urmator a fost sa incerc sa controlez un servo-motor din butoane. Nu am folosit butoane efective, ci am folosit o tastatura matriceala 4x4. Evident ca m-am complicat destul de mult deoarece citirea tastaturii a fost mult mai complicata decat ma asteptam (si pentru ca nu am citit cu atentie datasheet-ul tastaturii).

Pentru a citi tastatura folosita de mine, pe primele 4 linii trebuie sa se de curent (output HIGH) iar pe celelalte 4 se citeste. Astfel, pentru a citi prima linie din tastatura, primul port trebuie pus pe LOW, toti ceilalti pe HIGH, dupa care se verifica ultimele 4 linii. Daca vreuna din valori este 0, inseamna ca butonul de la intersectia dintre linia care se verifica si coloana aferenta portului a fost apasat. Din nou, si acest lucru mi-a luat destul de mult.

Dupa ce am putut controla un servomotor din butoane, urmatorul pas logic a fost sa incerc sa controlez toate 4 servomotoarele. Acesta a fost cel mai dificil pas.

Pentru a controla primul servomotor, eu foloseam portul OCR1A al timer1 (16-bit timer). Cu toate acestea, imi trebuiau 4 porturi de PWM. Partea proasta este ca timer0 si timer2 (ambele de 8 biti) nu puteau sa genereze semnal de 50hz, motiv pentru care a trebuit sa schimb implementarea.

Ce am facut a fost sa schimb modul ceasului si sa il fac sa genereze o intrerupere la fiecare 0.01ms. Astfel, pot sa monitorizez timpul destul de usor si pot simula un semnal PWM cu un anumit factor de umplere prin scriere de 1 respectiv 0 pe un port oarecare.

Astfel, am mutat porturile de comanda ale motoarelor pe PB0-PB3. Dupa ce am reusit sa fac si aceste modificari, practic am terminat partea software a proiectului meu.

Detalii Implementare

Pentru detalii specifice de implementare, puteti consulta sursa proiectului (tot codul este comentat si se descrie foarte in detaliu implementarea).

Rezultate Obținute

Pana in momentul de fata inca nu am reusit sa fac bratele efective ale robotului, voi uploada imagini de la PM Fair.

Concluzii

Un proiect interesant, din care am invatat destul de mult, dar prea complicat pentru putinul timp disponibil pe care il avem in semestrul 2.

Cu toate acestea, ca sa faci un brat robotic in adevaratul sens al cuvantului, proiectul se complica infinit de mult, mai ales daca vrei feature-uri precum self collision detection sau sa evite singularitatile.

Deși încă nu am terminat bratele efective, consider că pe partea de hardware și software proiectul și-a atins scopul.

Dacă vor fi studenți în anii următori care vor vrea să facă un proiect asemănător, următoarele sfaturi cred că vor fi utile: * nu vă luați cele mai puternice servomotoare, eu mi-am cumpărat MG996R (11kg/f) și sunt monstruoase, nu aveți ce să faceți cu ele * alimentați servomotoarele extern, întrucât mie cu 4 servomotoare mi se încălzeste regulatorul imediat * dacă folosiți mai mult de 2 servomotoare, categoric vă recomand să consultați codul meu, întrucât debugging-ul e destul de greu de făcut pe placuță * un multimetru ar fi mega util

Download

Sursele, documentația pentru ATmega324 și schema electrică a proiectului se găsesc aici [olaru_catalin_proiect_pm.zip](#)

Bibliografie/Resurse

1. Laboratoarele de PM: <http://cs.curs.pub.ro/wiki/pm>
 2. Datasheet ATMEGA324: http://cs.curs.pub.ro/wiki/pm/_media/doc8272.pdf
 3. WinAVR: <https://sourceforge.net/projects/winavr/files/>
 4. Multumiri speciale asistentilor de PM.
- Documentația în format [PDF](#)

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

Permanent link:
<http://ocw.cs.pub.ro/courses/pm/prj2017/anitu/shukarime>

Last update: **2021/04/14 15:07**

