Bionic Hand

Autor: Cinjau Constantin Iulian
Grupa: 336CB

Introducere

- Proiectul pe care mi-am propus sa il implementez reprezinta o mana robotica minimala ce simuleaza doua tipuri de miscari: miscarea simultana a celor 5 degete si rotatia mainii. Ambele tipuri de miscari vor fi realizate cu ajutorul unor motorase electrice, iar acestea vor fi primite ca si comenzi de pe un telefon ce se conecteaza prin bluetooth la placuta arduino.
- Scopul acestui proiect este de a dezvolta cat mai multe cunostinte legate de cum putem folosi un microcontroller pentru a conecta mai multe componente si a creea ceva ce se poate dovedi util atat pentru noi cat si pentru ceilalti.
- Ca si utilitate practica, nu pot spune ca are o utilitate anume, e mai mult un fel de “jucarie” pe care o poti controla din telefon si o poti arata prietenilor.
- Ideea de la care am plecat consta mai mult intr-o curiozitate pe care am avut-o in legatura cu controlul de pe telefon al unui dispozitiv creat de mine folosind un microcontroller, mi se parea destul de interesant ca se pot da comenzi prin bluetooth catre microcontroller ca mai apoi acesta sa controleze niste motorase pentru a misca mana in functie de comanda.

Descriere generală

Descriere generala:

Prin intermediul unei aplicatii de telefon minimaliste(facuta in MIT App Inventor) vom putea trimite comenzi de miscare a mainii: miscarea degetelor sau rotirea acesteia, catre placuta Arduino cu ajutorul modulului de bluetooth cu care microcontroller-ul comunica prin protocolul I2C, urmand ca placuta sa comunice cu LCD-ul(care va afisa comanda data de utilizator pe telefon) prin SPI si sa trimita semnale catre unul dintre cele doua motorase pentru a se realiza miscarea solicitata, de asemenea vor fi implementate anumite verificari astfel incat miscarea degetelor sa fie posibila doar intre niste unghiuri limita(de la 0 grade, in pozitia normala, la maxim 90 de grade) si rotirea mainii sa fie posibila intre 0 grade(pozitia normala) pana la 180 de grade(asa cum este si in realitate), in aceste cazuri limita se vor afisa si niste mesaje specifice pe LCD pentru a anunta user-ul ca limita miscarii respective a fost atinsa si sa incerce sa faca miscarea inversa.

Hardware Design

Lista piese:

- Arduino UNO(ATMega328p)
- Modul LCD cu I2C.
- Modul Bluetooth Master Slave HC-05 cu Adaptor.
- Fire de legatura.
- Breadboard.
- 2 x Servo-Motor Electric MicroServo 9g SG90.
- 6 x Baterie 1.5V AA + 2 x Suport baterii 3S.
- Rezistente: 1 x 1k, 1 x 2.2k, 1 x 4.7k.

Prin intermediul laptopului conectat la Arduino Uno(ATMega328p) prin USB vom furniza acesteia tensiunea necesara pentru a functiona. La randul sau, prin intermediul portului de 5V, placuta va alimenta:

- Modulul de Bluetooth HC-05, care functioneaza cu o tensiune de alimentare cuprinsa intre 3.6V si 6V. Nivelul logic al celor doi pini de date al modulului este de 3.3V, deci pe legatura dintre pinul de TX de placuta(in cazul nostru pinul 8) si pinul RX de pe modul trebuie sa formam un divizor de tensiune din doua reziste(una de 2.2k dupa care vom lua semnalul si il vom duce catre pinul modulului si una de 5.7k = 1 x 4.7k + 1 x 1k care va continua catre GND), deoarece output-ul pinilor de pe placuta este de 5V, ceea ce ar inseamna ca modulul nostru s-ar putea arde. Pe legatura dintre pinul de RX de pe placuta(in cazul nostru pinul 9) si pinul TX de pe modul nu trebuie sa adaugam nimic, deoarece nivelul dat ca OUTPUT de catre modul(3.3V) este suficient pentru ca placuta sa il interpreteze ca HIGH logic.
- LCD-ul cu controller-ul sau I2C, pentru care vom conecta cei 4 pini ai controller-ului in felul urmator: SCL la pinul Analog 5 depe placuta, SDA la pinul Analog 4, Vcc la pinul de 5V impreuna cu modulul de bluetooth si GND-ul la ground-ul comun.

In ceea ce priveste cele 2 servo-motoare, nu le vom putea conecta tot la pinul de 5V al placutei, deoarece nu ar mai fi destula tensiune de alimentare si pentru ele, ceea ce inseamna ca va trebui sa avem cate o sursa de tensiune suplimentara pentru fiecare in parte. Servo-motorul Micro Servo 9g SG90, poate fi alimentat cu o tensiune cuprinsa intre 3V si 7.2V, asa ca cele 2 surse auxiliare de tensiune pe care le voi folosi sunt formate fiecare in parte din 3 baterii de 1.5V AA legate in serie pentru a avea o tensiune de alimentare per motor de 4.5V. Pentru linia de semnal catre cele 2 motoare am folosit pinii 10 si 11 de pe placuta.

Software Design

Pentru a putea controla cele 2 motorase intr-un mod mai interactiv am dezvoltat in prima faza o aplicatie de Android minimalista, folosind site-ul MIT App Inventor si un tutorial pe Youtube dupa care sa ma ghidez. Aplicatia contine un:

- **buton** ce redirectioneaza catre o lista cu toate dispozitivele asociate cu telefonul prin bluetooth si din cadrul careia putem selecta modulul nostru HC-05, dupa ce conexiunea se realizeaza avem un mesaj specific in dreapta butonului.
- **2 slidere** prin intermediul carora putem modifica unghiul pentru degetele si incheietura mainii(cel pentru degete are ca limita 90 de grade, iar cel pentru incheietura 180 de grade).
- Un **checkbox** care este bifat la un interval de o secunda, pentru ca utilizatorul sa isi dea seama ca telefonul are o **conexiune stabila** cu placuta prin intermediul bluetooth-ului si primeste date constant de la aceasta.
- Un buton prin intermediul caruia poate face transmisia manual(in mod normal aceasta ar trebui sa se declanseze de fiecare data cand unul dintre slidere este actionat, dar m-am gandit ca ar fi util ca utilizatorul sa poata trimita si el manual in cazul in care i se pare ca nu s-a transmis ce a vrut).

Pentru comunicarea realizata prin Bluetooth, a trebuit sa folosesc un fel de “protocol” de comunicatie, in sensul ca aplicatia de pe telefon am definit-o astfel incat atunci cand doreste sa trimita un mesaj(cand se schimba pozitia unui slider sau se apasa butonul), marcheaza acest lucru prin trimiterea caracterului '?', pentru a delimita 2 valori diferite, se foloseste caracterul '&', iar pentru a marca finalul unui mesaj se foloseste caracterul ';', deci mesajul pe care il primim pe placuta va fi in formatul: ?fingers=value1&wrist=value2;. Pentru a comunica cu **modulul Bluetooth HC-05** am folosit biblioteca SoftwareSerial.h prin intermediul careia am creat un obiect de tipul SoftwareSerial pe care am atasat pinii: 9(pentru RX) si 8(pentru TX). Din cadrul acestei clase am utilizat 3 functii importante: available(), write() si read(). Aceste functii le-am folosit pentru a implementa alte 3 functii:

- **verify_bt_serial** = functie care este apelata in loop, si care are urmatoarea logica: asteapta sa se primeasca token-ul de inceput al mesajului(?), dupa ce primeste token-ul de inceput, citeste datele de pe seriala byte cu byte si le pune intr-un buffer pana cand primeste token-ul de final al mesajului(;), caz in care trimite mesajul catre functia de parsare si reseteaza buffer-ul si flag-ul de asteptare al inceputului unui nou mesaj. De asemenea, aceasta face si o verificare, in cazul in care mesajul pe care il primeste depaseste 20 de caractere(aplicatia nu va trimite atatea caractere in niciun caz), va ignora mesajul, pentru ca este posibil ca acesta este posibil sa contina date incorecte datorita conexiunii.
- **parse_message** = functie in care parsam valorile primite intr-un mesaj, aceasta verifica in primul rand daca mesajul este in formatul in care il asteptam(fingers=xx&wrist=yyy), si daca salveaza cele doua valori in format string(xx si yyy) pentru ca ulterior sa fie prelucrate de functia de validare a datelor si executarea comenzii pe servomotoare.
- **write_values_to_servo** = functie care valideaza cele doua string-uri de mai sus in 2 moduri:daca valoarea convertita de la string folosind functia toInt() este 0, si ultima valoarea scrisa pe motoras este ceva mult mai mare decat valori apropiate lui 0, este posibil ca valorile xx sau yyy sa fi continut caractere ciudate si de aceea returnul conversiei era 0, caz in care vom ignora valorile si vom astepta altele noim, iar, al doilea mod de validare consta in verificarea celor 2 valori sa fie in limitele permise: pentru degete [0, 90] grade si pentru incheietura [0, 180] grade. De asemenea functia verifica si daca noua valoare pentru un motor este egala cu cea veche, caz in care nu mai face write pentru ca ar fi inutil. Daca toate validarile merg ok, atunci noile valori sunt transmise catre servomotoare.

Pentru **controlul servomotoarelor** am folosit libraria Servo.h cu care am creat 2 obiecte de tipul Servo: pe primul dintre ele(cel pentru incheietura) l-am atasat la pinul 10 si pe celalalt(cel pentru degete) la pinul 11 prin intermediul metodei attach din cadrul clasei. Pentru a transmite semnal catre servomotoare am folosit metoda write.

Pentru a folosi **LCD-ul** am folosit libraria LiquidCrystal_I2C.h, creand un obiect de tipul

LiquidCrystal_I2C pentru care am aflat Slave Address-ul necesar protocolului I2C. Ca si functii utilizate din aceasta biblioteca sunt: init, backlight, setCursor si print. Datele de pe ecran sunt reprezentate de cele doua unghiuri, respectiv niste avertismente atunci cand se ating cele 2 limite(90/180 de grade) si le-am actualizat la un interval de o secunda prin intermediul unui timer pe care il voi descrie mai jos.

Dupa cum am specificat si in descrierea aplicatiei de Android, placuta trimite un semnal periodic(1 sec) telefonului, pentru ca utilizatorul sa stie ca, conexiunea este inca stabila si toate comenzile pe care le trimite ajung la placuta. Am realizat acest lucru periodic, tot cu ajutorului unui timer configurat prin intermediul **Timer0**, astfel incat sa genereze 250 de intreruperi pe secunda, si la cea de-a 250-a sa realizeze atat actualizarea datelor de pe LCD, cat si transmiterea semnalului catre telefon. Pentru a-l configura am definit functia **init_timer0_one_sec**, in care am pus modul de functionare(CTC), valoarea prescaler-ului(256) si valoarea pragului de numarare(250), respectiv activarea intreruperii atunci cand counter-ul ajunge la un anumit prag, iar in rutina de tratare a intreruperii **ISR(TIMER0_COMPA_vect)** doar verificam unde am ajuns cu celalalt counter, si daca s-a ajuns la 250(adica a trecut o secunda), avem un flag pe care il facem true si prin care anuntam functia loop ca poate face transmisia catre telefon si afisarea pe LCD.

In functia de **setup** apelam functia de initializare a timer-ului, configuram baudrate-ul serialei de comunicare cu modulul de bluetooth, initializam LCD-ul, respectiv cele 2 servomotoare.

In functia de **loop**, apelam functia de verificare a serialei cu modulul(verify_bt_serial), care la randul ei le apela pe celelalte cand era nevoie, adica cand s-a primit vreo comanda, respectiv, verifica flag-ul pentru a vedea daca trebuie actualizat LCD-ul sau trimis semnal catre telefon.

Rezultate obtinute

Rezulatul este o mana controlabila din telefon. Pentru cei care se uita va rog sa nu radeti, numai eu stiu cate straturi de superglue am pe degete :(. Multumesc.

Scurt demo cu proiectul: Demo Bionic Hand

Ca si observatie, motorasele se invart corect, dar scheletul mainii esti putin fragil si de aceea nu se observa chiar cum as fi vrut.

Concluzii

Realizarea proiectului a constat intr-o munca constanta pe parcursul mai multor zile, ca si probleme am avut in special la parsarea mesajelor primite prin Bluetooth, pentru ca uneori primeam mesaje incorecte datorita conexiunii si a durat ceva pana cand mi-am dat seama ca trebuie sa verific corectitudinea acestora. O alta problema intalnita a fost legata de timere, pentru ca incercam sa folosesc Timer1 pentru implementarea mea fara sa-mi dau seama ca acesta este folosit in spate de biblioteca Servo.h. Partea de design al proiectului, a fost destul de grea, dar in acelasi timp interesanta, sa iti imaginezi cum poti face un lucru din diferite materiale sa arate a ceva ce ti-ai propus si sa functioneze cum iti imaginai.

Download

Codul sursa pentru placuta si pentru aplicatia de telefon se regasesc in urmatoarea arhiva:

Bionic_Hand_Project.zip

Jurnal

- 3.05.2023: Crearea paginii proiectului si descrierea sumara a functionalitatilor acestuia, motivatai din spate si scopul pe care il are.
- 18.05.2023: Completarea paginii cu partea de Hardware Design si conectarea fizica a componentelor. Testarea componentelor pe rand pentru a verifica daca functioneaza corect, actualizarea schemei de hardware design cu mici diferente in ceea ce priveste piesele folosite si actualizarea si listei in care le-am specificat.
- 25.05.2023: Adaugarea de surse de alimentare suplimentare pentru fiecare servomotor.
- 27.05.2023: Modificari finale asupra circuitului si a codului.
- 28.05.2023: Crearea de elemente pentru design-ul mainii si actualizarea paginii proiectului.

Bibliografie/Resurse

- Lab PM 3
- Lab PM 5
- Lab PM 6
- Creare aplicatie Android
- Reglare contrast LCD

Export to PDF

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

Permanent link:
http://ocw.cs.pub.ro/courses/pm/prj2023/iotelea/bionic_hand

Last update: **2023/05/29 17:09**