Departamentul de Calculatoare
Universitatea Tehnică din Cluj Napoca

Proiectare cu Microprocesoare

Flight Simulator

Nume: Bogdan Maxim, Stefan Pamparau
Grupa:30235

Indrumator: Prof. dr. eng. Radu Danescu

Cuprins

Capitolul 1

Introducere

Proiectul prezentat in continuare este un flight simulator 2D realizat cu Arduino la care a fost conectat un accelerometru. Arduino este programat să recunoască datele trimise de la accelerometru și să le trimită mai departe pe portul serial. Portul serial este in continuare citit in mediul Processing, care reprezintă și interfața grafică a proiectului. În mediul Processing a fost codat un joc de evitare de obstacole, constituit dintr-o nava ce este controlată de utilizator și din niște asteroizi ce trebuiesc a fi evitați.

Placa Arduino este un Arduino Mega 2560.

Capitolul 2

Implementare Arduino

2.1 Date tehnice

Ca si accelerometru am folosit MMA8452Q, un accelerometru cu mod de functionare atat pe 8 cat si pe 12 biti. El are o scala setabila intre valorile 2g/4g/8g(atat pozitiv, cat si negativ).

Datele de iesire pot fi trimise la frecvente care variaza intre 1.56Hz si 800Hz.

Modul de comunicare intre Arduino si accelerometru se realizeaza prin interfata serial I2C. Avem doi pini de intrerupere, la care vom atasa pe intreruperea 1 diferenta de pozitie, iar pe intreruperea 2, diverse date tehnice, cum ar fi cele despre orientare, daca s-a facut vreun tap pe axa z, etc. Tensiunea de alimentare este de 3.3V.

2.2 Mod de conectare la placa

Pinul de GND al accelerometrului se va conecta la pinul GND al placii. Pinul de Vcc al accelerometrului se va conecta la pinul de 3.3V. Pinul de SCL al accelerometrului se va conecta la pinul SCL al placii. Pinul SDL al placii se va conecta la pinul SDL al placii. Pinul INT1 se conecteaza la pinul D2. Pinul INT2 se conecteaza la pinul D3. In acest moment, montajul este gata.

2.3 Implementare cod microcontroller

In functia setup, initializam modul de comunicare Serial cu PC-ul la un baud rate de 9600, dupa care setam pinii de intrerupere. Apoi realizam un test de control pentru a fi siguri ca accelerometrul este conectat la controller. Daca da, atunci initializam accelerometrul cu date pentru a fi citite. Daca nu, atunci punem intr-o bucla inifinita.

In functia de loop, aplicam tehnica de polling : verificam incontinuu daca avem pe intreruperea 1 sau 2 valoarea 1(acest lucru se poate face deoarece frecventa microcontrollerului este mai mare decat frecventa accelerometrului). Daca da, trimitem pe Serial datele citite. Nu putem folosi intreruperi deoarece acestea sunt dezactivate in momentul in care se intra pe o intrerupere.

Toata comunicarea intre placa si accelerometru se face prin interfata I2C. Accelerometrul are un set de registrii care pot fi modificati. Arduino-ul va transmite prin interfata ce registru sa scrie si cu ce valoare sa scrie, iar accelerometrul va executa. Pentru acest lucru, am implementat trei functii : writeRegister, readRegister si readRegisters. Comunicarea se realizeaza cu Wire. Pentru writeRegister, incepem transmisiunea, scriem adresa si apoi data care o dorim la adresa data. Pentru readRegister, scriem adresa dupa care depunem o cerere de citire. Daca acea adresa este valabila, initializam citirea.

Functia readRegisters este apelata o singura data, in momentul in care ne vin date despre pozitia actuala al accelerometrului in spatiu. Spre deosebire de readRegister, aceasta functie, cere 6 adrese si itereaza prin ele.

In initializarea accelerometrului, setam mai intai bitii de scalare, bitii de frecventa(data rate), dupa care setam sa recunoasca portrait/landscape, tapping-ul si intreruperile. Pentru fiecare dintre aceste componente, setam cate o functie de handling. In functia de handling pentru Portrait/Landscape – citim valoarea registrului si apoi in functie de aceasta, ne dam seama de orientare. Acelasi lucru pentru tapping si valorile inregistrate.

Capitolul 3

Implementare Processing

Pentru proiectul de simulare zbor am implementat in processing un joc de evitare de obstacole. Jucătorul are la dispoziție o nava spațială, iar sarcina acestuia este de a evita asteroizii care îi sunt puși în cale. Jocul a fost implementat in limbajul de programare java.

Descrisă pe larg, aplicația se conectează la portul serial de unde citește datele transmise de arduino prin intermediul accelerometrului. Acesta parsează inclinația pe cele trei axe de coordonate și deplasează nava corespunzător. De asemenea, jocul oferă posibilitatea măririi sau scăderii vitezei de deplasare a navei dacă jucătorul efectuează un tap, respectiv double tap. O descriere in detaliu a claselor programului precum și a pașilor efectuați de acesta o voi realiza in continuare.

Aplicația definește trei clase: Ship, Asteroid și Block. Aceste clase conțin informațiile specifice pentru cele trei obiecte modelate. Pentru Ship, acestea sunt poziția, viteza precum și hit blocurile acesteia. Pentru Asteroid, acestea sunt poziția și viteza. Pentru Block informațiile conținute sunt marginile acestuia.

În pasul de inițializare, aplicația instanțiază obiectele necesare și se conectează la portul serial pentru a citi datele de la arduino. În faza de citire, jocul determină tipul input-ului (poate fi de tip accelerare, tap, etc.) și i-a acțiunea corespunzătoare acesteia. Dacă acțiunea este de tip accelerare, se determină in ce direcție trebuie mutată nava, urmând ca poziția acesteia să fie updatată corespunzător. Dacă acțiunea este de tip tap sau double tap, viteza de deplasare a navei crește, respectiv descrește. La fiecare mișcare a navei marginile hit block-urilor acesteia sunt recalculate. De asemenea, se asigură că nava nu părăsește ecranul.

Determinarea direcției de deplasare a navei se realizează cu ajutorul unor praguri predefinite de sensibilitate. Dacă una din coordonatele trimise de arduino depășește un prag înseamnă că utilizatorul a inclinat placa în direcția respectivă și dorește deplasarea pe acea rută.

Pe lângă updatarea navei, aplicația se ocupă și de generarea asteroizilor. Numarul maxim de asteroizi prezenți pe ecran este predefinit, in acest moment la 5, urmând ca in momentul in care un asteroid parasește marginile ecranului, un alt asteroid să îi ieie locul din partea superioară a jocului. Un alt aspect care este tot procesat la nivel de processing este detectarea coliziunii. Atât timp cât jocul este încă în desfășurare, la fiecare pas, marginile hit block-urilor navei sunt comparate cu asteroizii prezenți pe ecran. Dacă este prezentă o coliziune, atunci jocul se sfârșeste.

Capitolul 4

Concluzii

În concluzie, proiectul a constat în realizarea unui flight simulator 2D. Au fost utilizate placa Arduino la care a fost conectat un accelerometru, precum și mediul de programare Processing, in care a fost realizat un joc de evitare obstacole.