Group Member:

Alex, Liming Hu, Jeff Deng.

Abstract

This lab is mainly about two-player game, we first made a one board game. We used red board and an UNO to implement the program. The second one is tow Board implementation.

Introduction

In this lab, we used Interboard communicating with each other (through pin 4 and pin 5). The game can let players set up the initial mine field for each other. Player can’t move after reaching the edge. There will be four mines for the other player to set up, players will die after they reach any of those mines. If player get to the final diagonal point then he or she wins the game.

Discussion of the lab

1.Brief Design Specification

1.Requirements: One Board / Game Implementation (45) 4 by 4 grid, User entry and move selection, boundaries Handled Win or Lose Annunciation.

Input and Output: Serial Monitor.

2.Requirements: Two Board / Game Implementation, 4 by 4 grid

Board 1: Contains user1 entry and move selection for player2.

Board2: Contains user2 entry and move selection for player1.

Input and Output: Serial Monitor.

2.Hardware Implementation

Serial Monitor

Serial Monitor

Arduino UNO1

Arduino UN21

3.Software Implementation

Function of the program is to complete an interactive game within two boards. 4 by 4 grid is printed by a loop. MasterPlayer1 is going to enter mine field position firstly. SlavePlayer2 will be able to enter the mine position for player1 once player had finished setting up. Then MasterPlayer1 will request data from player2. Game will initialize after palyer2 finished setting up.

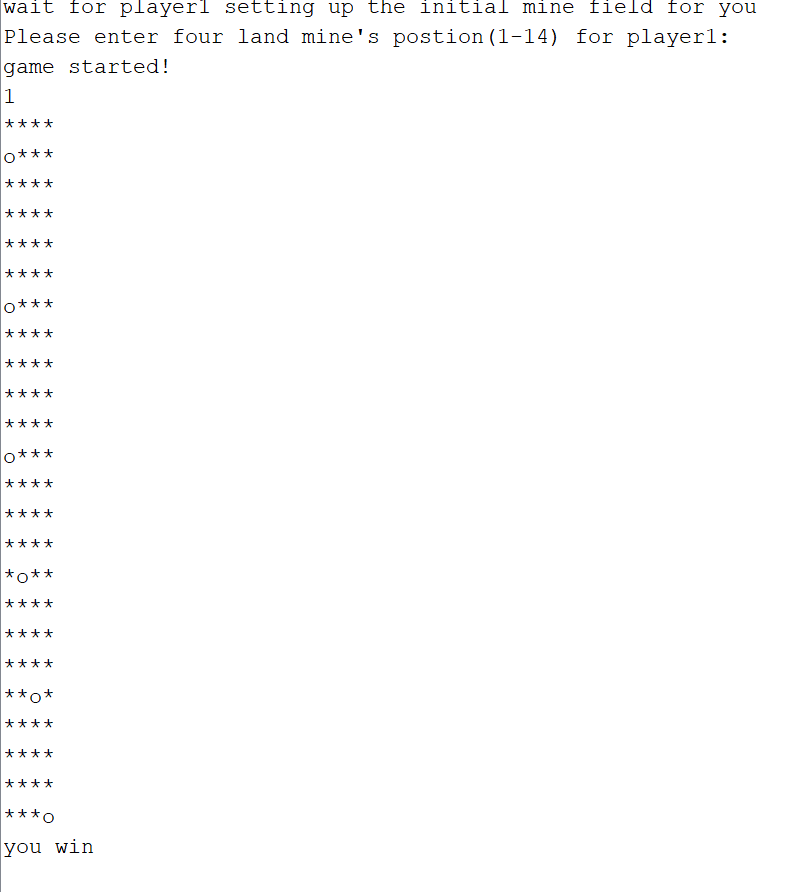
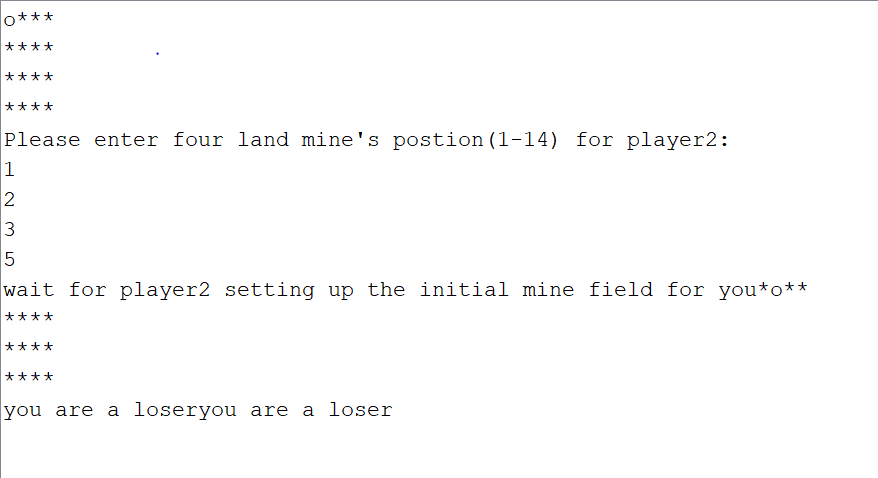
The picture of game is displayed by void print(char a[]), while there is a input, we will read the letter(a , w , s , d) and dealing with it as a direction. We use int i(index) recording the position of the player, and compare it with the position of mine and destination(i == 15) to determine whether the player has won or lost the game.

SerialFlush() is used to clear monitor buffer.

Test Plan

Enter four integers in player1 first, then enter four integers in player2. Game picture will display and ‘a’ ‘s’ ‘w’ ‘d’ can control the direction of o( represents the player). After testing the game we make sure that Mine field is transferred through board.

Presentation, discussion, and analysis of the results



Analysis of any errors

During the program, we’ve met plenty of problems. For example, when add serial’s function or other functions within receiveEvent(int howMany) and requestEvent(), it could cause the program go wrong while could be complied and uploaded, So after a lot of tests and adjustments we finally found and solved the problems.

Summary and conclusion

The program included communicating within two boards. We can make two boards send data to each other, and receive data at the same time by using receiveEvent and requestEvent.

Appendices

Masterplayer1

#include<Wire.h>

char a[16],action;

int i=0;

//initializing Serial and Wire

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

for(int i=0;i<16;i++)

a[i]='\*';

a[0]='o';

print(a);

Serial.print("Please enter four land mine's postion(1-14) for player2:\n");

Wire.begin();

}

//It’s player1’s code

void loop(){

//transfer and receive the mine position

if(Serial.available()>0){

int tran\_e = Serial.parseInt();

Serial.println(tran\_e);

int tran\_b = Serial.parseInt();

Serial.println(tran\_b);

int tran\_c = Serial.parseInt();

Serial.println(tran\_c);

int tran\_d = Serial.parseInt();

Serial.println(tran\_d);

Wire.beginTransmission(4);

Wire.write(tran\_e);

Wire.write(tran\_b);

Wire.write(tran\_c);

Wire.write(tran\_d);

Wire.endTransmission();

Serial.print("wait for player2 setting up the initial mine field for you");

delay(8000); //waits for 8 seconds

Wire.requestFrom(4,8);

//initializing the game

while(true){

if(Wire.available())

{

int e= Wire.read();

int b=Wire.read();

int c=Wire.read();

int d=Wire.read();

serialFlush();

while(true){

if(Serial.available()>0){

action=Serial.read();

if(action=='w'&&i>=4){

a[i]='\*';

i-=4;

a[i]='o';

print(a);

}

if(action=='s'&&i<=11){

a[i]='\*';

i+=4;

a[i]='o';

print(a);

}

if(action=='a'&&i!=0&&i!=4&&i!=8&&i!=12){

a[i]='\*';

i-=1;

a[i]='o';

print(a);

}

if(action=='d'&&i!=3&&i!=7&&i!=11&&i!=15){

a[i]='\*';

i+=1;

a[i]='o';

print(a);}

if(i==e||i==b||i==c||i==d){

Serial.print("you are a loser");

delay(10000);}

if(i==15){

Serial.print("you win");

delay(10000);}

}

}

}

}

}

}

//refresh player’s position

void print(char a[])

{

for(int i=0;i<16;i++){

if((i+1)%4==0)

Serial.println(a[i]);

else

Serial.print(a[i]);

}

}

//clean the serial monitor buffer

void serialFlush(){

while(Serial.available() > 0) {

char t = Serial.read();

}

}

slavePlayer2:

#include<Wire.h>

char a[16],action;

int i=0;

int e = 0;

int b = 0;

int c = 0;

int d = 0;

int tran\_e = 0;

int tran\_b = 0;

int tran\_c = 0;

int tran\_d = 0;

int firstTime = true;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

for(int i=0;i<16;i++)

a[i]='\*';

a[0]='o';

print(a);

Serial.print("wait for player1 setting up the initial mine field for you\n");

Wire.begin(4);

Wire.onReceive(receiveEvent);

Wire.onRequest(requestEvent);

}

//player2

void loop(){

if(firstTime == true && e!= 0)

{

Serial.print("Please enter four land mine's postion(1-14) for player1:\n");

serialFlush();

while(true){

if(Serial.available()>0){

tran\_e = Serial.parseInt();

tran\_b = Serial.parseInt();

tran\_c = Serial.parseInt();

tran\_d = Serial.parseInt();

break;

}

}

firstTime = false;

}

if(firstTime == false)

{

playGame();

}

}

//receive data

void receiveEvent(int howMany) {

e= Wire.read();

b=Wire.read();

c=Wire.read();

d=Wire.read();

}

void print(char a[])

{

for(int i=0;i<16;i++){

if((i+1)%4==0)

Serial.println(a[i]);

else

Serial.print(a[i]);

}

}

//Transfer data

void requestEvent()

{

Serial.println(tran\_e);//test

Wire.write(tran\_e);

Wire.write(tran\_b);

Wire.write(tran\_c);

Wire.write(tran\_d);

}

//initializing game

void playGame()

{

serialFlush();

Serial.print("game started!\n");

while(e != 0){

if(Serial.available()>0){

action=Serial.read();

if(action=='w'&&i>=4){

a[i]='\*';

i-=4;

a[i]='o';

print(a);

}

if(action=='s'&&i<=11){

a[i]='\*';

i+=4;

a[i]='o';

print(a);

}

if(action=='a'&&i!=0&&i!=4&&i!=8&&i!=12){

a[i]='\*';

i-=1;

a[i]='o';

print(a);

}

if(action=='d'&&i!=3&&i!=7&&i!=11&&i!=15){

a[i]='\*';

i+=1;

a[i]='o';

print(a);}

if(i==e||i==b||i==c||i==d){

Serial.print("you are a loser\n\n");

delay(1000000);}

if(i==15){

Serial.print("you win\n\n");

delay(1000000);}

}}

}

void serialFlush(){

while(Serial.available() > 0) {

char t = Serial.read();

}

}