#include <LedControl.h>

//Define The Snake as a Struct

typedef struct Snake Snake;

struct Snake{

int head[2]; // the (row, column) of the snake head

int body[40][2]; //An array that contains the (row, column) coordinates

int len; //The length of the snake

int dir[2]; //A direction to move the snake along

};

//Define The Apple as a Struct

typedef struct Apple Apple;

struct Apple{

int rPos; //The row index of the apple

int cPos; //The column index of the apple

};

//MAX72XX led Matrix

const int DIN =12;

const int CS =11;

const int CLK = 10;

LedControl lc = LedControl(DIN, CLK, CS,1);

const int varXPin = A3;//X Value from Joystick

const int varYPin = A4;//Y Value from Joystick

byte pic[8] = {0,0,0,0,0,0,0,0};//The 8 rows of the LED Matrix

Snake snake = {{1,5},{{0,5}, {1,5}}, 2, {1,0}};//Initialize a snake object

Apple apple = {(int)random(0,8),(int)random(0,8)};//Initialize an apple object

//Variables To Handle The Game Time

float oldTime = 0;

float timer = 0;

float updateRate = 3;

int i,j;//Counters

void setup() {

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

/\*

The MAX72XX is in power-saving mode on startup,

we have to do a wakeup call

\*/

lc.shutdown(0,false);

/\* Set the brightness to a medium values \*/

lc.setIntensity(0,4);

/\* and clear the display \*/

lc.clearDisplay(0);

//Set Joystick Pins as INPUTs

pinMode(varXPin, INPUT);

pinMode(varYPin, INPUT);

}

void loop() {

// put your main code here, to run repeatedly:

float deltaTime = calculateDeltaTime();

timer += deltaTime;

//Check For Inputs

int xVal = analogRead(varXPin);

int yVal = analogRead(varYPin);

if(xVal<100 && snake.dir[1]==0){

snake.dir[0] = 0;

snake.dir[1] = -1;

}else if(xVal >920 && snake.dir[1]==0){

snake.dir[0] = 0;

snake.dir[1] = 1;

}else if(yVal<100 && snake.dir[0]==0){

snake.dir[0] = -1;

snake.dir[1] = 0;

}else if(yVal >920 && snake.dir[0]==0){

snake.dir[0] = 1;

snake.dir[1] = 0;

}

//Update

if(timer > 1000/updateRate){

timer = 0;

Update();

}

//Render

Render();

}

float calculateDeltaTime(){

float currentTime = millis();

float dt = currentTime - oldTime;

oldTime = currentTime;

return dt;

}

void reset(){

for(int j=0;j<8;j++){

pic[j] = 0;

}

}

void Update(){

reset();//Reset (Clear) the 8x8 LED matrix

int newHead[2] = {snake.head[0]+snake.dir[0], snake.head[1]+snake.dir[1]};

//Handle Borders

if(newHead[0]==8){

newHead[0]=0;

}else if(newHead[0]==-1){

newHead[0] = 7;

}else if(newHead[1]==8){

newHead[1]=0;

}else if(newHead[1]==-1){

newHead[1]=7;

}

//Check If The Snake hits itself

for(j=0;j<snake.len;j++){

if(snake.body[j][0] == newHead[0] && snake.body[j][1] == newHead[1]){

//Pause the game for 1 sec then Reset it

delay(1000);

snake = {{1,5},{{0,5}, {1,5}}, 2, {1,0}};//Reinitialize the snake object

apple = {(int)random(0,8),(int)random(0,8)};//Reinitialize an apple object

return;

}

}

//Check if The snake ate the apple

if(newHead[0] == apple.rPos && newHead[1] ==apple.cPos){

snake.len = snake.len+1;

apple.rPos = (int)random(0,8);

apple.cPos = (int)random(0,8);

}else{

removeFirst();//Shifting the array to the left

}

snake.body[snake.len-1][0]= newHead[0];

snake.body[snake.len-1][1]= newHead[1];

snake.head[0] = newHead[0];

snake.head[1] = newHead[1];

//Update the pic Array to Display(snake and apple)

for(j=0;j<snake.len;j++){

pic[snake.body[j][0]] |= 128 >> snake.body[j][1];

}

pic[apple.rPos] |= 128 >> apple.cPos;

}

void Render(){

for(i=0;i<8;i++){

lc.setRow(0,i,pic[i]);

}

}

void removeFirst(){

for(j=1;j<snake.len;j++){

snake.body[j-1][0] = snake.body[j][0];

snake.body[j-1][1] = snake.body[j][1];

}

}