

/\* ========================================

\*

\*Starblaster program for Microcontroller

\*System Design CPE 3280

\*Written by Cypress Payne

\*

\* ========================================

\*/

#include "project.h"

#include <stdlib.h>

#include <stdio.h>

//registers

#define PWR\_CTRL 0x2D

#define DATA\_RG 0x31

#define DATA\_X0 0x32

#define INT\_EN 0x2E

#define INT\_MAP 0x3F

#define INT\_SRC 0x30

#define TAP\_AXES 0x2A

#define TAP\_THRESH 0x1D

#define TAP\_DUR 0x21

#define TAP\_LAT 0x22

#define TAP\_WIN 0x23

//universal variables

uint8 readbuffer[10]; //read SPI

uint8 writebuffer[10]; //write to SPI address is 0x53

int16 xval, yval, zval;

int TARGETX;

int TARGETY;

int sky[200][2];

//declaration of functions

void writeSensor(uint8 \*writebuffer, int numvals);

void readSensor(uint8 \*readbuffer, int numvals);

int16 combineNum(uint8 a, uint8 b);

void drawStars(int arr[200][2]);

void remStars(int arr[200][2]);

void drawTarget(int x, int y);

void remTarget(int x, int y);

void drawBlast(int x, int y);

void drawSuperBlast(int x, int y);

//count interrupt 20x a second

CY\_ISR(ISR\_COUNT){

Counter\_ClearInterrupt(Counter\_INTR\_MASK\_TC);

//Writes register number of x0

writebuffer[0] = DATA\_X0;

writeSensor(writebuffer, 1);

//puts x0 - z1 in readbuffer 0 - 5

readSensor(readbuffer, 6);

xval = combineNum(readbuffer[0], readbuffer[1]);

yval = combineNum(readbuffer[2], readbuffer[3]);

zval = combineNum(readbuffer[4], readbuffer[5]);

Counter\_ClearInterrupt(Counter\_INTR\_MASK\_TC);

}

//tap interrupt

CY\_ISR(ISR\_INT1){

Int1\_ISR\_ClearPending();

//write register number of tap source

writebuffer[0] = INT\_SRC;

writeSensor(writebuffer, 1);

//read tap source

readSensor(readbuffer, 1);

int16 num;

num = readbuffer[0];

//double tap

if(num == 195){

drawSuperBlast(TARGETX, TARGETY);

drawStars(sky);

drawTarget(TARGETX, TARGETY);

CyDelay(50);

}

//single tap

else if (num == 194){

drawBlast(TARGETX, TARGETY);

drawStars(sky);

drawTarget(TARGETX, TARGETY);

CyDelay(50);

}

}

int main(){

//intialize hardware

I2C\_Start();

GLCD\_Start();

GLCD\_Clear(GLCD\_BLACK);

Backlight\_Write(1);

LED\_Blue\_Write(1); LED\_Green\_Write(0); LED\_Red\_Write(1);

KhzClock\_Start();

Counter\_Start();

Count\_ISR\_ClearPending();

Count\_ISR\_StartEx(ISR\_COUNT);

Int1\_ISR\_ClearPending();

Int1\_ISR\_StartEx(ISR\_INT1);

CyGlobalIntEnable;

//turn on accelerometer

writebuffer[0] = PWR\_CTRL;

writebuffer[1] = 0b00001000;

writeSensor(writebuffer, 2);

//set data range

writebuffer[0] = DATA\_RG;

writebuffer[1] = 0b00000000;

writeSensor(writebuffer, 2);

//set interrupt enable

writebuffer[0] = INT\_EN;

writebuffer[1] = 0b01100000;

writeSensor(writebuffer, 2);

//set interrupt mapping

writebuffer[0] = INT\_MAP;

writebuffer[1] = 0b00000000;

writeSensor(writebuffer, 2);

//set tap axes

writebuffer[0] = TAP\_AXES;

writebuffer[1] = 0b00000111;

writeSensor(writebuffer, 2);

//set tap threshold

writebuffer[0] = TAP\_THRESH;

writebuffer[1] = 0x30;

writeSensor(writebuffer, 2);

//set tap duration

writebuffer[0] = TAP\_DUR;

writebuffer[1] = 0x10;

writeSensor(writebuffer, 2);

//set tap latency

writebuffer[0] = TAP\_LAT;

writebuffer[1] = 0x20;

writeSensor(writebuffer, 2);

//set tap window

writebuffer[0] = TAP\_WIN;

writebuffer[1] = 0x40;

writeSensor(writebuffer, 2);

// generate stars

int starNum = 100 + rand()%100;

int16 i;

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

int starX = rand()%131;

int starY = rand()%131;

sky[i][0] = starX;

sky[i][1] = starY;

GLCD\_Pixel(starX, starY, GLCD\_WHITE);

}

//generate target

TARGETX = 63;

TARGETY = 63;

drawTarget(TARGETX,TARGETY);

//infinite loop

for(;;)

{

while(xval < -50) {

if(TARGETX > 15){

remTarget(TARGETX,TARGETY);

TARGETX -= 1;

drawTarget(TARGETX,TARGETY);

}

}

while(yval > 50) {

if(TARGETY < 120){

remTarget(TARGETX,TARGETY);

TARGETY += 1;

drawTarget(TARGETX,TARGETY);

}

}

while(xval > 50) {

if (TARGETX < 120){

remTarget(TARGETX,TARGETY);

TARGETX += 1;

drawTarget(TARGETX,TARGETY);

}

}

while(yval < -50) {

if(TARGETY > 15){

remTarget(TARGETX,TARGETY);

TARGETY -= 1;

drawTarget(TARGETX,TARGETY);

}

}

drawStars(sky);

}

}

/\*function definitions\*/

//write to buffer

void writeSensor(uint8 \*writebuffer, int numvals){

I2C\_I2CMasterClearStatus();

I2C\_I2CMasterWriteBuf(0x53, writebuffer, numvals, I2C\_I2C\_MODE\_COMPLETE\_XFER);

while(!(I2C\_I2CMasterStatus() & I2C\_I2C\_MSTAT\_WR\_CMPLT));

}

//read from buffer

void readSensor(uint8 \*readbuffer, int numvals){

I2C\_I2CMasterClearStatus();

I2C\_I2CMasterReadBuf(0x53, readbuffer, numvals, I2C\_I2C\_MODE\_COMPLETE\_XFER);

while(!(I2C\_I2CMasterStatus() & I2C\_I2C\_MSTAT\_RD\_CMPLT));

}

//finalize values from buffer

int16 combineNum(uint8 a, uint8 b){

int16 final;

final = b << 8;

final = final | a;

return final;

}

//draws stars for background

void drawStars(int arr[200][2]){

for(int i = 0; i < 200; i++){

int x,y;

x = arr[i][0];

y = arr[i][1];

GLCD\_Pixel(x, y, GLCD\_WHITE);

}

}

//removes stars in background leaving black sky

void remStars(int arr[200][2]){

for(int i = 0; i < 200; i++){

int x,y;

x = arr[i][0];

y = arr[i][1];

GLCD\_Pixel(x, y, GLCD\_BLACK);

}

}

//draws target

void drawTarget(int x, int y){

GLCD\_DrawCircle(x, y,5,GLCD\_YELLOW);

GLCD\_DrawLine(x,y-10,x,y+10,GLCD\_YELLOW);

GLCD\_DrawLine(x-10,y,x+10,y,GLCD\_YELLOW);

}

//removes target

void remTarget(int x, int y){

GLCD\_DrawCircle(x, y,5,GLCD\_BLACK);

GLCD\_DrawLine(x,y-10,x,y+10,GLCD\_BLACK);

GLCD\_DrawLine(x-10,y,x+10,y,GLCD\_BLACK);

}

//draws blast

void drawBlast(int x, int y){

int rad;

for(rad = 0; rad < 20; rad++){

CyDelay(15);

GLCD\_DrawCircle(x,y,rad,GLCD\_GREEN);

}

for(rad = 0; rad < 20; rad++){

CyDelay(15);

GLCD\_DrawCircle(x,y,rad,GLCD\_BLACK);

}

}

//draws superblast for when blast has charged

void drawSuperBlast(int x, int y){

int rad;

for(rad = 0; rad < 20; rad++){

CyDelay(15);

GLCD\_DrawCircle(x,y,rad,GLCD\_GREEN);

}

for(rad = 20; rad < 30; rad++){

CyDelay(15);

GLCD\_DrawCircle(x,y,rad,GLCD\_BLUE);

}

for(rad = 0; rad < 30; rad++){

CyDelay(15);

GLCD\_DrawCircle(x,y,rad,GLCD\_BLACK);

}

}

/\* [] END OF FILE \*/