//este programa contiene el ejercicio, 5, 6, 7, 8,

//includes

#include <stdio.h>

#include "sapi.h"

//--DEFINES--

#define COMMON\_CATHODE

//#define COMMON\_ANODE

#ifdef COMMON\_CATHODE

#define DIGIT\_ON 1 //con transistor va en 1 y sin transistor va en 0

#define DIGIT\_OFF 0//con trnsistor va en 0 y sin transistor va en 1

#define PIN\_VALUE\_ON 1

#define PIN\_VALUE\_OFF 0

#endif

#ifdef COMMON\_ANODE

#define DIGIT\_ON 1

#define DIGIT\_OFF 0

#define PIN\_VALUE\_ON 0

#define PIN\_VALUE\_OFF 1

#endif

#define TIEMPO\_ON 500

/\* Configuracion de pines para el display 7 segmentos \*/

/\*

--------------------------+------------+-----------+----------------

Segmento encendido | Valor BIN | Valor HEX | GPIO resultado

--------------------------+------------+-----------+----------------

Enciende el segmento 'a' | 0b00000001 | 0x01 | GPIO0

Enciende el segmento 'b' | 0b00000010 | 0x02 | GPIO4

Enciende el segmento 'c' | 0b00000100 | 0x04 | GPIO3

Enciende el segmento 'd' | 0b00001000 | 0x08 | GPIO7

Enciende el segmento 'e' | 0b00010000 | 0x10 | GPIO6

Enciende el segmento 'f' | 0b00100000 | 0x20 | GPIO2

Enciende el segmento 'g' | 0b01000000 | 0x40 | GPIO1

Enciende el segmento 'h' | 0b10000000 | 0x80 | GPIO5

--------------------------+------------+-----------+----------------

---------------------------------+------------+-----------+----------------

Digito encendido | Valor BIN | Valor HEX | GPIO resultado

---------------------------------+------------+-----------+----------------

Enciende el digito unidad | 0b00000001 | 0x01 | LCD3

Enciende el digito decena | 0b00000010 | 0x02 | LCD2

Enciende el digito centena | 0b00000100 | 0x04 | LCD1

Enciende el digito unidad de mil | 0b00001000 | 0x08 | LCD4

---------------------------------+------------+-----------+----------------

a

-----

f / / b

/ g /

-----

e / / c

/ d /

----- O h = dp (decimal point).

\*/

// Symbols formed by segmens

/\*

------------+------+---------

Segmentos | HEX | Simbolo

------------+------+---------

hgfedcba | |

0b00111111 | 0x0F | 0

0b00000110 | 0x00 | 1

0b01011011 | 0x00 | 2

0b01001111 | 0x00 | 3

0b01100110 | 0x00 | 4

0b01101101 | 0x00 | 5

0b01111101 | 0x00 | 6

0b00000111 | 0x00 | 7

0b01111111 | 0x00 | 8

0b01101111 | 0x00 | 9

0b01011111 | 0x00 | a 10

0b01111100 | 0x00 | b 11

0b01011000 | 0x00 | c 12

0b01011110 | 0x00 | d 13

0b01111011 | 0x00 | e 14

0b01110001 | 0x00 | F 15

0b01110111 | 0x00 | A 16

0b00111001 | 0x00 | C 17

0b01111001 | 0x00 | E 18

0b01110110 | 0x00 | H

0b00011110 | 0x00 | J

0b00111000 | 0x00 | L

0b01110011 | 0x00 | P

0b00111110 | 0x00 | U

0b10000000 | 0x00 | .

a

-----

f / / b

/ g /

-----

e / / c

/ d /

----- O h = dp (decimal point).

\*/

const uint8\_t display7SegmentOut[26] = {

0b00111111, // 0

0b00000110, // 1

0b01011011, // 2

0b01001111, // 3

0b01100110, // 4

0b01101101, // 5

0b01111101, // 6

0b00000111, // 7

0b01111111, // 8

0b01101111, // 9

0b01011111, // a

0b01111100, // b

0b01011000, // c

0b01011110, // d

0b01111011, // e

0b01110001, // f

0b01110111, // A

0b00111001, // C

0b01111001, // E

0b01110110, // H

0b00011110, // J

0b00111000, // L

0b01110011, // P

0b00111110, // U

0b10000000, // .

0b00000000 // display off

};

gpioMap\_t valor[] = {

GPIO0, // a

GPIO4, // b

GPIO3, // c

GPIO7, // d

GPIO6, // e

GPIO2, // f

GPIO1, // g

GPIO5 // h = dp

};

gpioMap\_t digito[] = {

//LCD4, // D4

LCD3, // D3

LCD2, // D2

LCD1 // D1

};

void displayMostrarDigito( uint8\_t val, uint8\_t dig );

void displayMostrar( uint16\_t val, uint32\_t tiempo );

struct joystick {

int x, y;

};

//prototipo de funciones

struct joystick leerJoystick ();

int imprimirValores(struct joystick);

int mover\_servo (struct joystick valores);

int angulo\_servo (int angulo);

//---------MAIN--------------

int main ()

{

boardInit();

uint32\_t i = 0;

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

gpioInit( valor[i], GPIO\_OUTPUT );

}

for( i=0; i<3; i++ ) {

gpioInit( digito[i], GPIO\_OUTPUT );

}

gpioWrite( digito[0], DIGIT\_OFF );

gpioWrite( digito[1], DIGIT\_OFF );

gpioWrite( digito[2], DIGIT\_OFF );

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

gpioWrite( valor[i], PIN\_VALUE\_OFF );

}

delay\_t tiempo;

delayConfig( &tiempo, 1000 );

// Habilitar pwm de servos

servoConfig( 0, SERVO\_ENABLE );

// Habilitar salida SERVOn (n puede ser 0...8)

servoConfig( SERVO0, SERVO\_ENABLE\_OUTPUT );

while(TRUE) {

leerJoystick ();

angulo\_servo(mover\_servo(leerJoystick ()));

if(delayRead(&tiempo)) {

imprimirValores(leerJoystick ());

mover\_servo (leerJoystick ());

}

}

return 0;

}

struct joystick leerJoystick ()

{

struct joystick valores;

valores.x = adcRead(CH1);

valores.y = adcRead(CH2);

return valores;

}

/\* ejercicio 6 \*/

int imprimirValores(struct joystick valores)

{

printf("[ %d,%d ]\r\n", valores.x, valores.y);

return 0;

}

/\* ejercicio 7 \*/

int mover\_servo (struct joystick valores)

{

int valor\_pote\_x=0;

int grados\_servo =0;

valor\_pote\_x=valores.x;

grados\_servo = ((valor\_pote\_x/1023.0) \*180);

// Escribir el servo con el angulo

servoWrite( SERVO0, grados\_servo );

return grados\_servo;

}

/\* ejercicio 8\*/

void displayMostrarDigito( uint8\_t val, uint8\_t dig )

{

uint8\_t j = 0;

gpioWrite( digito[dig], DIGIT\_ON );

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

if( display7SegmentOut[val] & (1<<j) ) {

gpioWrite( valor[j], PIN\_VALUE\_ON );

} else {

gpioWrite( valor[j], PIN\_VALUE\_OFF );

}

}

delay(5);

gpioWrite( digito[dig], DIGIT\_OFF );

}

void displayMostrar( uint16\_t val, uint32\_t tiempo )

{

uint8\_t i = 0;

uint8\_t unidad = 0;

uint8\_t decena = 0;

uint8\_t centena = 0;

for( i=0; i<3; i++ ) {

gpioWrite( digito[i], DIGIT\_OFF );

}

if( val <= 999 ) {

centena = val/100;

decena = (val%100)/10;

unidad = val%10;

for( i=0; i<tiempo/15; i++ ) {

displayMostrarDigito( unidad, 0 );

//if(val>9)

displayMostrarDigito( decena, 1 );

//if(val>99)

displayMostrarDigito( centena, 2 );

}

}

}

/\* ejercicio 8\*/

int angulo\_servo (int angulo)

{

displayMostrar( angulo, 100 );

}