TRABAJO DE SISTEMAS ELÉCTRÓNICOS DIGITALES

Curso 2021-2022

Domótica de una casa: Persiana

López de Meneses Calvo, Laura 54952

Ortega Monge, María 54777

Villajos Mateo, Laura 54904

ÍNDICE:

1. Introducción

El objetivo de este proyecto es diseñar una funcionalidad de la domótica de una casa con un microcontrolador. En nuestro caso es una persiana con la placa STM32F407. Hemos implementado el proyecto con las funciones y conocimientos aprendidos en las clases de micros.

Se ha diseñado una persiana que cuenta con dos modos: manual y automático. Su funcionamiento será descrito más adelante.

2. Componentes

- STM32: microcontrolador base sobre el cual se sustenta el proyecto.
- Botones: utilizados como entradas. Contamos con dos botones auxiliares y el botón conectado al PA0 de la propia placa. Dos de los botones son utilizados para el control manual de la persiana (uno para subir y otro para bajar) y el botón restante sirve para el modo de funcionamiento automático.
- Leds: utilizados como salidas. Hemos utilizado los de la propia placa. Son indicadores de los procesos subir y bajar y de la subida o bajada total de la persiana.
- Sensor de luminosidad: utilizado como ADC. El sensor entra en funcionamiento en el modo automático.
- Servo motor: utilizado para generar el movimiento de subida y bajada.
- Pantalla LCD: utilizada como salida para describir el movimiento que se está llevando a cabo.

3. Funcionamiento

Como se ha mencionado anteriormente, la persiana cuenta con dos modos de funcionamiento: manual y automático.

- El modo manual cuenta con dos botones conectados al PA0 y al PA1 de la placa. Este modo es muy sencillo, cada botón tiene una funcionalidad: subir o bajar la persiana.
- El modo automático empieza a funcionar al pulsar el botón conectado al PA3. Su funcionamiento se basa en un sensor de luminosidad (ADC); la persiana se subirá automáticamente cuando la variable luz (lo que recibimos del sensor) sea menor que un umbral establecido. En caso contrario la persiana bajará automáticamente.

En ambos modos podemos saber en que momento del proceso se encuentra la persiana por medio de los leds de la placa y el siguiente código de colores:

Rojo-> bajando, azul-> subiendo, verde-> persiana bajada y naranja->persiana subida.

En la maqueta hemos conseguido el movimiento de la persiana por medio de un servo motor conectado al PA15 que funciona con un temporizador PWM. El servo está unido a un soporte que, al girar enrolla o desenrolla a la tela que actúa como persiana.

Además, contamos con una pantalla lcd cuya función es mostrar los siguientes mensajes:

- Al inicio: “Persiana 2022” “LAURA MARIA LAURA”.
- Durante la subida: “Subiendo”.
- Durante la bajada: “Bajando”.

Por último, hemos añadido un código antirebotes para una correcta implementación de las funcionalidades, sobre todo para los botones.

4.Descripción de código

```
#include "main.h"
#include "main.h"
#include<stdlib.h>
#include <LCD1602.h>
volatile int arriba=0,abajo=1,subir=0,bajar=0,botonaux=0;
int row=0;
int col=0;
__IO uint16_t  luz=0;
```

Bibliotecas y variables necesarias para la implementación del resto del código.

```
if (GPIO_Pin==GPIO_PIN_0)  // boton azul placa baja
{
if(abajo==1) // si ya esta subida procedemos a bajarla
subir=1;
}

if (GPIO_Pin==GPIO_PIN_1)
{
if(arriba==1) // si ya esta subida procedemos a bajarla
bajar=1;
}

if (GPIO_Pin==GPIO_PIN_3)
{
botonaux=1;
}
}
```

Interrupción creada para la pulsación del botón de la placa (PA0) para subir y bajar y el botón auxiliar para activar el modo automático.

```
int antirrebotes(volatile int* button_int, GPIO_TypeDef* GPIO_Port, uint16_t
GPIO_number)
{
static uint8_t button_count = 0;
static int counter = 0;
if (*button_int == 1)
{
if (button_count == 0)
{
counter = HAL_GetTick();
button_count ++;
}
if (HAL_GetTick() - counter >= 20)
{
counter = HAL_GetTick();
if (HAL_GPIO_ReadPin(GPIO_Port, GPIO_number) != 1)
{
button_count = 1;
}
else
button_count ++;
if (button_count == 4)
{ //Periodo Antirrebotes
button_count = 0;
*button_int = 0;
return 1;
}
}
```

Código de la función **anti rebotes**.

```
int main(void)
{
MX_GPIO_Init();
  MX_ADC1_Init();
  MX_TIM2_Init();
  MX_I2C1_Init();
  HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_1);
  lcd_init ();
  lcd_put_cur(0, 0);
  lcd_send_string("PERSIANA  ");
  lcd_send_string("2022");
  lcd_put_cur(1, 0);
  lcd_send_string("LAURA MARIA LAURA");
  HAL_Delay(3000);
  lcd_clear();
```

Los **init** necesarios y la inicialización de los mensajes que emite la pantalla lcd.

```
while((subir==1)&&(arriba==0))
{ //subir persiana // 15 subiendo 13 subido del todo
 lcd_init ();
 lcd_put_cur(0, 0);
 lcd_send_string("SUBIENDO");
 HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,0);
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15,1);
for (int i=600;i>0;i=i-30){
__HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,i);
 HAL_Delay(200);}
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,1);
```

Código para subir la persiana con los leds, temporizadores y flags correspondientes.

```
arriba=1;
abajo=0;
subir=0;
lcd_clear();
}
```

Código para subir la persiana con los leds, temporizadores y flags correspondientes.

```
while((bajar==1)&&(abajo==0))
{ //bajar persiana // 14 bajando 12 abajo
lcd_init ();
lcd_put_cur(0, 0);
lcd_send_string("BAJANDO");
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14,1);//bajando
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,0);
for (int i=0;i<600;i=i+30){
__HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,i);
HAL_Delay(200);}
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_14,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12,1);
abajo=1;
arriba=0;
bajar=0;
lcd_clear();
}
```

Código para bajar la persiana con los leds, temporizadores y flags correspondientes.

```
while(botonaux){
if(luz<50 && abajo==1)
{  //subir persiana // 15 subiendo 13 subido del todo
lcd_init ();
lcd_put_cur(0, 0);
lcd_send_string("SUBIENDO");
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15,1);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,0);
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,0);
for (int i=600;i>0;i=i-30){
__HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,i);
HAL_Delay(200);}
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_15,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,1);
arriba=1;
abajo=0;
subir=0;
lcd_clear();
}
HAL_ADC_Start(&hadc1);
if (HAL_ADC_PollForConversion(&hadc1,1000000)==HAL_OK)
{
luz=HAL_ADC_GetValue(&hadc1);
}
```

Entra en funcionamiento el modo automático al presionar el botón auxiliar.

Si luz es menor que el umbral, se subirá la persiana.

```
if(luz>50 && arriba==1 )
{ //bajar persiana // 14 bajando 12 abajo
lcd_init ();
lcd_put_cur(0, 0);
lcd_send_string("BAJANDO");
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_13,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14,1);//bajando
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,0);
for (int i=0;i<600;i=i+30){
__HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,i);
HAL_Delay(200);}
HAL_GPIO_WritePin(GPIOD,GPIO_PIN_14,0);
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12,1);
abajo=1;
arriba=0;
bajar=0;
lcd_clear();
}
HAL_ADC_Start(&hadc1);
if (HAL_ADC_PollForConversion(&hadc1,1000000)==HAL_OK)
{
luz=HAL_ADC_GetValue(&hadc1);
}
}
```

Entra en funcionamiento el modo automático al presionar el botón auxiliar.

Si luz es menor que el umbral, se bajará la persiana.

5. Problemas y soluciones adoptadas

- Código anti-rebotes: decidimos añadir este código porque al principio, no detectaba la pulsación de los botones o la detectaba doble y no funcionaba correctamente por ello.

- La pantalla lcd: fue necesario crear dos archivos para su correcto funcionamiento que fueron una biblioteca LCD1602.h y un archivo .c llamado LCD1602.c con el código de funcionamiento de la pantalla.

- El contraste de la pantalla lcd: para regular el contraste de la pantalla y que se reflejaran bien los mensajes fue necesario añadir un potenciómetro. Esto lo aprendimos con el proyecto de informática del primer curso de la carrera.

6. Conclusiones

Al haber finalizado el trabajo hemos obtenido las siguientes conclusiones:

La herramienta que más hemos utilizado del programa STM Cube IDE es el debug para ir paso a paso sobre nuestro código y encontrar errores (lógicos y sintácticos) de una manera más rápida y eficiente.

El código anti-rebotes pensábamos que era prescindible, pero al repasar el código con debug nos dimos cuenta de que las pulsaciones de los botones generaban mucho rebote y tuvimos que incorporarlo para mejorar el funcionamiento.

Asistir a las clases de micros nos ha servido para saber implementar el servo, la pantalla lcd y la última práctica de laboratorio para implementar el sensor de luminosidad.

Realizar estos trabajos nos permite poner en práctica lo que aprendemos en clase y materializar la teoría. No son tarea fácil, pero conseguirlo te da la satisfacción de ser consciente del poder del conocimiento aprendido en clase.

7. Enlace de github y video demostrativo

Se adjunta enlace de nuestro github en el que tenemos el control de versiones, la versión definitiva, este documento como memoria del proyecto y un video demostrativo.

https://www.youtube.com/watch?v=wFbwDSKUR3w

Enlace al control de versiones de github:

https://github.com/mariaortegamonge/trabajoSED-MICROS