******

***Trabajo de Microprocesadores***

***Sistemas Electrónicos Digitales***

**Programación de un sistema de aviso LED en un sistema domótico de aire acondicionado controlado por un potenciómetro mediante el microcontrolador STM32F411E**

**Andrés Ferreres Bernardez 53744**

**Yvonne Escribano Moré 50670**

***ÍNDICE DE CONTENIDOS***

[1. OBJETIVO 3](#_Toc13172437)

[2. INTRODUCCIÓN 3](#_Toc13172438)

[3. DESARROLLO DEL PROYECTO 3](#_Toc13172439)

[3.1. PLANTEAMIENTO DEL PROBLEMA 3](#_Toc13172440)

[3.2. ELEMENTOS UTILIZADOS 4](#_Toc13172441)

[3.3. MEJORAS REALIZADAS Y PROBLEMAS RESUELTOS 5](#_Toc13172442)

[4. CÓDIGO COMENTADO 5](#_Toc13172443)

# OBJETIVO

La realización de este trabajo busca llegar a conocer el entorno de desarrollo Keil de ARM que se explica durante la práctica 0 del curso así como el planteamiento, organización y ejecución de una aplicación domótica con el microcontrolador asignado STM32, en nuestro caso STM32F411E, en la que se emplean entradas y salidas para realizar una determinada función. El proyecto debe constar de una serie de entradas (pulsadores, botones, etc.) y salidas, el uso de interrupciones, el uso de temporizadores y de convertidores D/A y/o D/A.

# INTRODUCCIÓN

Mediante el uso de microcontroladores podemos realizar una serie de aplicaciones con variedad de opciones gracias a los periféricos que realizan funciones de interfaz y control de forma concurrente. Gracias al lenguaje de programación C podemos configurar dicho microprocesador mientras que el compilador del mismo traducirá las sentencias de C a ensamblador. Tras ello, la solución de desarrollo de software se crea, se construye y se depuran las aplicaciones embebidas.

# DESARROLLO DEL PROYECTO

Este proyecto trata de realizar un sencillo sistema domótico en el que se informa al usuario de si está aumentando o disminuyendo la temperatura del termostato. Para ello, se ha utilizado un potenciómetro para representar dicho termostato y, mediante una serie de luces parpadeantes de varios colores, mostrar la situación en que se encuentra. Gracias a ellas, el usuario puede saber si el aire acondicionado se encuentra en standby o si está pudiendo modificarlo para que suba la temperatura o para que baje.

Además, se ha añadido la posibilidad de que el usuario interaccione con el sistema (además de con el potenciómetro) con dos botones: un reset y un standby.

## PLANTEAMIENTO DEL PROBLEMA

El sistema se dividirá en varios estados donde cada una de ellas determinará una acción concreta de los leds del mismo.

Así pues, tendremos los siguientes estados:

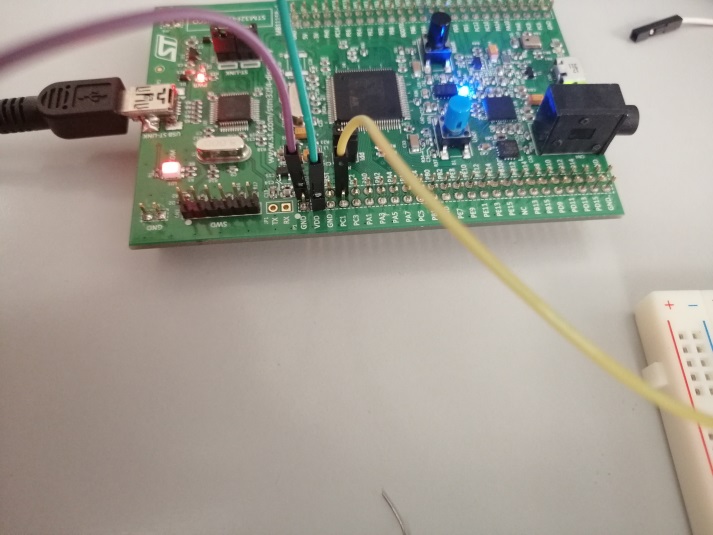
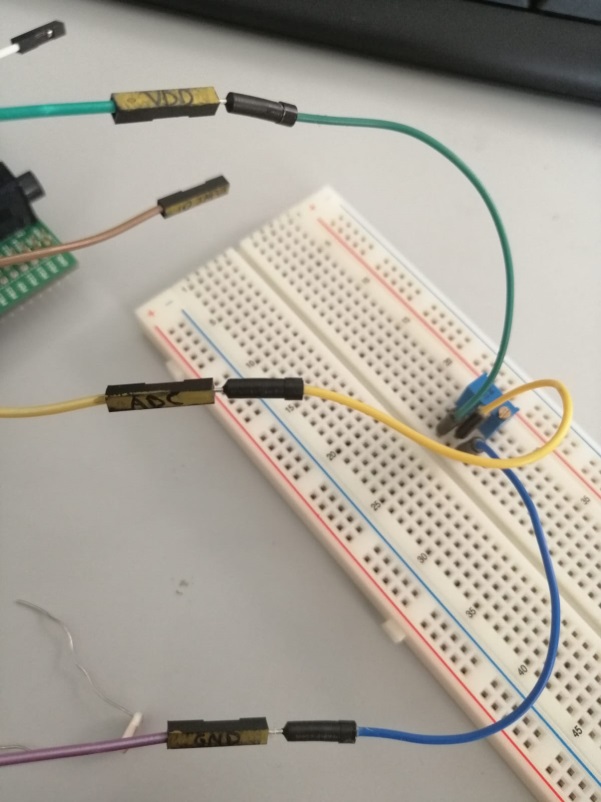
* Standby (naranja):

El LED naranja se mantiene fijo en el estado de espera hasta que el botón *User* es pulsado. Al estar implementado la marcha-paro, si se vuelve a pulsar (independientemente de si está en rojo o azul) vuelve a encenderse y apaga los demás estados.

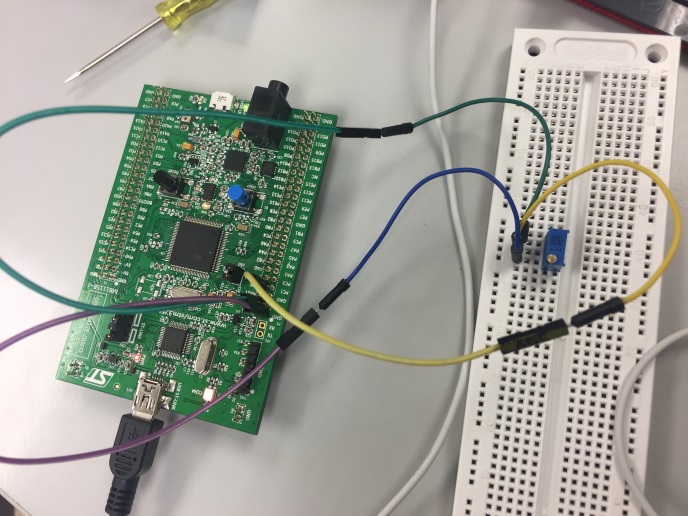
* Subida de temperatura (rojo):

El LED rojo parpadea si la entrada ADC está por debajo de 200.

* Bajada de temperatura (azul):

En este estado el LED azul parpadeará si la entrada ADC está por encima de 250.

## ELEMENTOS UTILIZADOS

* Microcontrolador STM32F411E-DISCO y cable de conexión a PC.
* Protoboard para realizar las conexiones
* Potenciómetro de 10 k Ω
* Destornillador para usar el potenciómetro
* 3 cables macho
* 3 cables hembra

## MEJORAS REALIZADAS Y PROBLEMAS RESUELTOS

* La función Callback:

Esta función tiene como uno de sus términos un puntero a un temporizador indeterminado. Para elegir cuál de los dos temporizadores de distintas frecuencias de interrupción usa (TIM2 o TIM3), se emplea el atributo “*Instance*” del puntero llamado y, por ello, podemos tratar las variables *flickr* en función del temporizador al cuál se refiere.

* Variables *flickr*:

Estas variables se tratan de bits que se pasan directamente al pin de control de cada LED para controlar su parpadeo. Funcionan negando su valor anterior al final de cada ciclo completo de su temporizador correspondiente.

* Cambio de estado:

El sistema consta de dos entradas diferentes (botón *User*, de color azul) y una entrada ADC del sistema (el pin PC0). *User* se emplea con el fin de entrar y salir del estado *Standby* (LED naranja) mientras que la entrada analógica nos dice si calentamos o enfriamos el sistema (LED rojo o azul, respectivamente).

* Reset:

El botón *Reset* (color negro) resetea el programa entero y lo deja en Standby preparado para ser iniciado cuando el usuario considere.

* Marcha-paro:

Por motivos de seguridad, se ha implementado un sistema Marcha-Paro. En caso de que, por algún motivo, se deba parar de golpe el sistema de aire acondicionado se debe pulsar el botón User. Si se activa en mitad de un ciclo, automáticamente se vuelve al estado Standby. Si se moviera el potenciómetro mientras nos encontramos en paro no ocurriría nada aunque se almacenaría el cambio y, al salir de Paro y volver a Marcha, nos encontraríamos en la situación almacenada.

# CÓDIGO COMENTADO

Todo el código al que se hace referencia en este apartado se encuentra subido a Github en la dirección <https://github.com/YvonneEM/Proyecto-Microprocesadores-SED>.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteSe instancian las funciones que necesitaremos emplear así como se usan los *flickr*.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteSe inicializan funciones y temporizadores.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteSe configura el conversor ADC.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteImagen que contiene captura de pantalla

Descripción generada automáticamenteSe eligen las frecuencias de TM2 y TM3.

Imagen que contiene captura de pantalla

Descripción generada automáticamenteSe configuran las entradas y las salidas:

Imagen que contiene captura de pantalla

Descripción generada automáticamenteImagen que contiene captura de pantalla

Descripción generada automáticamenteSe configura la función que nos permite realizar el cambio entre estados y la Marcha-Paro:

Imagen que contiene captura de pantalla

Descripción generada automáticamenteSe adjudican las salidas en función de cada estado:

Imagen que contiene captura de pantalla

Descripción generada automáticamente

# CONCLUSIONES

La realización de este trabajo nos ha permitido observar cómo nuestros conocimientos sobre microprocesadores se han visto ampliados y complementados. Así mismo, nos ha permitido comprender la dificultad intrínseca de esta programación lo que nos hace creer en la necesidad de formarnos más y mejorar nuestro nivel y control.

Hemos comprendido la necesidad de microprocesadores en el mundo que nos rodea y sus grandes posibilidades de aplicación para cualquier aspecto de nuestras vidas no solo en cuanto a domótica sino a sanidad, transporte, alimentación… Viendo el futuro estamos convencidos de la necesidad de desarrollar estos instrumentos que nos permitirán muchos y muy interesantes desarrollos.