



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería  
Informática**

**Comunicación con Sistemas  
Embebidos  
Documentación Técnica**



Presentado por Enrique del Olmo Domínguez  
en Universidad de Burgos — 7 de julio de 2022

Tutor: Alejandro Merino Gómez y Daniel  
Sarabia Ortiz



---

# Índice general

---

<b>Índice general</b>	<b>i</b>
<b>Índice de figuras</b>	<b>iii</b>
<b>Índice de tablas</b>	<b>v</b>
<b>Apéndice A Plan de proyecto software</b>	<b>1</b>
A.1. Introducción . . . . .	1
A.2. Planificación temporal . . . . .	1
A.3. Estudio de viabilidad . . . . .	11
<b>Apéndice B Especificación de requisitos</b>	<b>19</b>
B.1. Introducción . . . . .	19
B.2. Catálogo de requisitos . . . . .	20
B.3. Especificación de requisitos . . . . .	22
<b>Apéndice C Especificación de diseño</b>	<b>31</b>
C.1. Introducción . . . . .	31
C.2. Diseño de datos . . . . .	31
C.3. Diseño arquitectónico . . . . .	36
C.4. Diseño procedimental . . . . .	38
<b>Apéndice D Documentación técnica de programación</b>	<b>41</b>
D.1. Introducción . . . . .	41
D.2. Estructura de directorios . . . . .	41
D.3. Manual del programador . . . . .	43
D.4. Compilación, instalación y ejecución del proyecto . . . . .	50

D.5. Distribución de pines . . . . .	52
D.6. Pruebas del sistema: Packet Sender . . . . .	54
<b>Apéndice E Documentación de usuario</b>	<b>55</b>
E.1. Introducción . . . . .	55
E.2. Requisitos de usuarios . . . . .	55
E.3. Manual del usuario . . . . .	56
<b>Bibliografía</b>	<b>61</b>

---

# Índice de figuras

---

A.1. Tareas planificadas para el Sprint1 . . . . .	3
A.2. Tareas planificadas para el Sprint2 . . . . .	4
A.3. Tareas planificadas para el Sprint3 . . . . .	5
A.4. Tareas planificadas para el Sprint4 . . . . .	6
A.5. Tareas planificadas para el Sprint5 . . . . .	7
A.6. Tareas planificadas para el Sprint6 . . . . .	8
A.7. Diagrama de Gantt. Parte 1 . . . . .	9
A.8. Diagrama de Gantt. Parte 2 . . . . .	10
 B.1. Diagrama de casos de uso . . . . .	23
C.1. Establecimiento de conexión TCP ( <i>3 way Handshake</i> ) . . . . .	32
C.2. Cierre de conexión TCP . . . . .	33
C.3. Distintos algoritmos para la ejecución de un SE . . . . .	37
C.4. Diagrama de secuencia de los casos de uso . . . . .	39
 D.1. Estructura de directorios . . . . .	42
D.2. Página de descarga del IDE . . . . .	43
D.3. Elección de la ruta para guardar los proyectos en local . . . . .	44
D.4. Interfaz principal MCUXpresso . . . . .	44
D.5. Página de descarga del SDK para la placa K64F . . . . .	45
D.6. Abrir el SDK . . . . .	46
D.7. Elección de los Drivers necesarios . . . . .	46
D.8. Importación de un Proyecto . . . . .	47
D.9. Interfaz de los Pines en MCUXpresso . . . . .	48
D.10. Interfaz de la tabla de los Relojes en MCUXpresso . . . . .	49
D.11. Interfaz del diagrama de los relojes en MCUXpresso . . . . .	49
D.12. Interfaz de los Periféricos en MCUXpresso . . . . .	50

D.13. Compilación del Proyecto. . . . .	51
D.14. Definición de los pines de la placa FRDM K64F. . . . .	52
D.15. Conexión de Pines . . . . .	53
D.16. Interfaz de la herramienta Packet Sender. . . . .	54
E.1. Botones del shield de la placa maestro. . . . .	57
E.2. Botones utilizados de la placa K64F. . . . .	57
E.3. Establecer velocidad motor A. . . . .	58
E.4. Establecer velocidad motor B. . . . .	58
E.5. Se muestra la temperatura por pantalla. . . . .	59
E.6. Obtener velocidades de los motores. . . . .	59

---

# Índice de tablas

---

A.1.	Costes pertenecientes a la SS . . . . .	12
A.2.	Coste total de personal . . . . .	12
A.3.	Coste del <i>hardware</i> . . . . .	13
A.4.	Coste del <i>software</i> . . . . .	14
A.5.	Coste del Sistema Empotrado . . . . .	14
A.6.	Coste total del proyecto . . . . .	14
A.7.	Licencia Apache 2.0 . . . . .	16
A.8.	Licencia CC BY-NC-SA 4.0 . . . . .	17
B.1.	CU-1 Set Speed A . . . . .	24
B.2.	CU-2 Set Speed B . . . . .	25
B.3.	CU-3 Parada de Emergencia. . . . .	26
B.4.	CU-4 Obtener Temperatura. . . . .	27
B.5.	CU-5 Obtener Velocidad Motor A . . . . .	28
B.6.	CU-6 Obtener Velocidad Motor B. . . . .	29
C.1.	Lista de comandos . . . . .	35
D.1.	Pines FRDM K64F y pantalla LCD. . . . .	53
D.2.	Pines FRDM K64F y motores EMG30. . . . .	53



## *Apéndice A*

---

# **Plan de proyecto software**

---

## **A.1. Introducción**

Existen dos temas a tratar en este apéndice. El primero corresponde a la planificación temporal del trabajo, en la cual veremos las tareas realizadas y los tiempos requeridos para llevarlas a cabo. También veremos el tiempo total necesario para la realización de este proyecto.

El segundo tema que se va a desarrollar, es la realización de un análisis de la viabilidad de un proyecto de estas características en la vida real. Para ello nos centraremos en analizar las variables económicas y legales referentes al desarrollo de este trabajo. En cuanto a este último punto veremos los tipos de licencias que se pueden utilizar en este tipo de software

## **A.2. Planificación temporal**

Como ya hemos comentado, la planificación del proyecto se llevó a cabo siguiendo la metodología SCRUM que está basada entre otras cosas, en el uso de *sprints* para organizar la carga de tareas.

Según esta metodología, estos sprints deben durar entre 1 y 3 semanas. Puesto que se está utilizando una metodología SCRUM, algunos tiempos pueden no ajustarse lo suficiente a lo esperado sobre todo los de los primeros sprint realizados, ya que en estos se debe llevar un proceso de medición de fuerzas y dificultades del trabajo a realizar. En este caso y puesto que mi situación personal ha sido distinta a lo largo de este proceso, la duración de estos sprints no ha sido siempre la misma. Esto se debe a que mis obligaciones han ido variando a medida que avanzaba el cuatrimestre. Al principio de

este, durante los 3 primeros Sprint y comienzos del cuarto, disponía de bastante más tiempo para su realización por lo que era capaz de dedicar más horas al día. Posteriormente empecé a cursar las prácticas curriculares con un total de seis horas diarias lo que dificultó seguir con el mismo ritmo de dedicación al TFG. Se ha tratado de que los sprints estuvieran formados por aproximadamente el mismo número de horas. Estas horas están expuestas en el repositorio de GitHub y cada tarea tiene asignado un *tag*. Cada uno de estos *tags* es un número que relaciona 1 a 1 la dificultad con las horas para su realización, es decir, una dificultad de ocho equivale a aproximadamente 8 horas de trabajo. Como es lógico, cuanta mayor es la dificultad más se rompe esta relación, puesto que al ser más complicado, siempre suele alargarse el tiempo de realización de la tarea. El proyecto se desarrolló en los siguientes Sprints:

- Sprint 1. Investigación y familiarización (Software, Hardware y Herramientas)
- Sprint 2. Envío y recepción de paquetes TCP/IP
- Sprint 3. Investigación y primeros pasos con periféricos (Comunicación Uart, I2C)
- Sprint 4. Investigación sobre Wifi. Uso de potenciómetros para gestionar motores
- Sprint 5. Periféricos (sensor de temperatura y pantalla) y uso real en una empresa
- Sprint 6. Memoria, presentación y últimos retoques

En los siguientes apartados obtendremos más información sobre como fue el desarrollo de cada uno de estos Sprints.

## Sprint 1

Este Sprint está compuesto por 10 tareas que en total suman 43 puntos. Duró aproximadamente dos semanas. El comienzo fue complicado por la gran cantidad de conceptos y el funcionamiento del hardware que debí investigar, además de la dificultad de familiarizarse con dos IDEs diferentes. Las acciones principales fueron:

- Elección, descarga y familiarización del entorno de desarrollo.

- Completar las prácticas propuestas por los tutores.
- Descarga de otros programas para documentación, organización y comunicación.
- Primeros pasos con lwIP.
- Primeros pasos con RTOS.

### Sprint 1

No due date 100% complete

Primer sprint del TFG dedicado a la investigación y estudio de las herramientas y funcionamiento del hardware y software que será necesario para la realización del proyecto. Se deberán estudiar, entre otros, el funcionamiento de MCUXpresso o Kinetis, como funciona freertos y mas concretamente las tareas, elaboración de proyectos que puedan utilizar compon...

Show more ▾

	Open	Closed
□ ○ 0 Open	✓ 10 Closed	
□ ○ Descarga y aprendizaje de GitKraken	Point: 2 Tools	
#9 by eod1001 was closed on 23 Mar		
□ ○ Repaso General del Sprint 1	Investigation Point: 13	
#10 by eod1001 was closed on 23 Mar		
□ ○ Actualización de repos de GitHub	documentation Point: 1	
#8 by eod1001 was closed on 23 Mar		
□ ○ Realización de las Prácticas propuestas por los tutores	Point: 8 work	
#7 by eod1001 was closed on 23 Mar		
□ ○ Descarga de Termite & Docklight	Point: 1 Tools	
#4 by eod1001 was closed on 23 Mar		
□ ○ Descarga de IDE Kinetis + sdk	Point: 2 Tools	
#5 by eod1001 was closed on 23 Mar		
□ ○ Investigación sobre freeRTOS y el control de tareas.	Investigation Point: 3	
#6 by eod1001 was closed on 23 Mar		
□ ○ Investigación sobre la Librería lwIP	Investigation Point: 3	
#2 by eod1001 was closed on 23 Mar		
□ ○ Descarga de las herramientas MCUXpresso + sdk correspondiente.	Point: 2 Tools	
#3 by eod1001 was closed on 23 Mar		
□ ○ Lectura y Comprensión del TFM creado por RPC	Investigation Point: 8	
#1 by eod1001 was closed on 23 Mar		

Figura A.1: Tareas planificadas para el Sprint1.

Como podemos ver en la figura anterior, este primer Sprint se basó en la creación del entorno de trabajo y primeros pasos con las herramientas que iba a usar. Estos primeros pasos incluían ya la utilización de botones, leds y configuración de algún periférico.

### Sprint 2

El sprint 2 recoge 9 tareas que suman 35 puntos de dificultad. Tuvo una duración de dos semanas y media. Este sprint estuvo enfocado a comenzar con la investigación del funcionamiento de lwIP. Cómo funcionaba y para qué servía, fueron los dos puntos más importantes. Tras entender estos dos puntos, busqué información sobre cómo implementarlo y comencé por la obtención de una IP según la mac de la placa y la creación de una tarea

que pudiera recibir paquetes con información a través de este servicio. Para enviar los primeros paquetes utilicé Packet Sender.

- Investigación Protocolo TCP/IP y su implementación con lwIP.
- Primeras partes del software.
- Pruebas con la herramienta Packet Sender.

### Sprint 2

No due date 100% complete	
Desarrollo de los primeros programas mas complejos en los que se empiecen a tratar temas mas enfocados al objetivo del TFG.	
<input type="checkbox"/> <input checked="" type="radio"/> 0 Open <input checked="" type="checkbox"/> 9 Closed	
<input type="checkbox"/>	<input checked="" type="radio"/> Creación de un Listener y un Writer que permitan a la Placa intercambiar Mensajes via TCP/IP <a href="#">Investigation</a> Point: 13 work #17 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Repaso de todo lo Hecho durante el Sprint 2 Point: 5 work #18 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Descarga y Utilización de Wireshark y Packet Sender Point: 3 Tools work #19 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Creación/Adaptación de un Código que permita a la Placa conectarse por Ethernet <a href="#">Investigation</a> Point: 5 work #16 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Actualización de la Documentación creada en este Sprint en GitHub <a href="#">documentation</a> Point: 2 #15 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Primeros pasos con laTeX y comienzo de la Memoria <a href="#">documentation</a> Point: 3 work #13 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Búsqueda de Información sobre el Protocolo TCP/IP <a href="#">Investigation</a> Point: 2 #12 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Descarga de LaTeX y las herramientas necesarias para comenzar con la memoria. Point: 1 Tools #14 by eod1001 was closed on 29 Mar
<input type="checkbox"/>	<input checked="" type="radio"/> Búsqueda de Información sobre Protocolo DHCP <a href="#">Investigation</a> Point: 1 #11 by eod1001 was closed on 29 Mar

Figura A.2: Tareas planificadas para el Sprint2.

Otra parte a destacar fue el comienzo de la memoria con la herramienta LáTEX. Una herramienta que puede ser compleja de utilizar puesto que se debe aprender su sintaxis y acostumbrarse a un nuevo entorno para su uso.

### Sprint 3

En este Sprint se buscó conseguir la comunicación de tres placas mediante el protocolo TCP/IP. En este punto, ya se disponía de conexión en red y de un ‘Listener’ que recibía paquetes de datos. El objetivo ahora era crear además, un ‘Writer’ capaz de enviarlos. Además, también se decidió la manera en que se transmitiría la información y sería en forma de comandos, por lo tanto también se implementó una especie de filtro que gestionaba el tipo de comando recibido en caso de existir. Por otro lado, también se comenzó a investigar e implementar las comunicaciones con los periféricos,

concretamente con los motores y la pantalla. En este caso el sprint está constituido por 8 tareas que suman 36 puntos. Tareas principales:

- Creación del ‘Writer’.
- Creación del filtro de comandos.
- Primeros pasos con UART y comunicación con los motores.
- Primeros pasos con I2C y la pantalla LCD.

### Sprint 3

No due date 100% complete

Se deberá conseguir la comunicación de tres placas, una maestro y dos esclavos, mediante paquetes TCP/IP y que además las placas esclavo se comuniquen con dos motores, uno cada una, mediante UART. La placa maestro deberá ser capaz de enviar órdenes y recibir información sobre los motores.

<input type="checkbox"/>	<input checked="" type="radio"/> 0 Open	<input checked="" type="checkbox"/> 8 Closed
<input type="checkbox"/>	<input checked="" type="radio"/>	Realización de pruebas finales <small>bug</small> Point: 5 <small>work</small>
		#27 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Código Final hasta el Sprint 3 Point: 8 <small>work</small>
		#26 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Actualización del repositorio de GitHub hasta el Momento <small>documentation</small> Point: 2
		#25 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Prueba de que la comunicación TCP/IP desde una placa Maestro a las placas Esclavo y viceversa Point: 5 <small>work</small>
		#23 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Actualización de la Documentación hasta el Momento <small>documentation</small> Point: 3
		#24 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Establecer la comunicación UART para el control de los motores desde las placas esclavo <small>Investigation</small> Point: 8 <small>work</small>
		#22 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Búsqueda de Información sobre comunicaciones UART, USART y I2C <small>Investigation</small> Point: 2
		#21 by eod1001 was closed on 10 Apr
<input type="checkbox"/>	<input checked="" type="radio"/>	Lectura y Comprensión de la Documentación de los motores <small>documentation</small> <small>Investigation</small> Point: 3
		#20 by eod1001 was closed on 10 Apr

Figura A.3: Tareas planificadas para el Sprint3.

Este sprint tuvo una duración de dos semanas.

### Sprint 4

En este Sprint se terminan, la comunicación de los motores y la de la pantalla, quedando operativas a falta de algunas mejoras. Además se produce uno de los hitos más importantes del proyecto, que es la investigación y abandono del procedimiento para la instalación del módulo wifi por falta de tiempo. En ese momento estaba comenzando las prácticas y se decide dejar apartada la parte de conexión inalámbrica para centrarme en la obtención de más funcionalidades. Esto viene dado porque desde la empresa en la que cursé las prácticas me ofrecieron la posibilidad de hacer algo que pudiera usarse en la fábrica. Debido a esto, también me pongo a investigar sobre la

posibilidad de introducir potenciómetros y sensores de temperatura para poder medir la temperatura de su CPD. El sprint suma una dificultad de 42 puntos en 8 tareas y se realizó en una semana y media. Tareas principales:

- Investigación sobre la implementación de un módulo WIFI, ESP8266.
- Mejoras del código referente a la comunicación con los motores y la pantalla LCD.
- Investigación e implementación de la lectura de los potenciómetros.

#### Sprint 4

No due date 100% complete			
En este Sprint nos centraremos en la inclusión de potenciómetros desde los cuales controlaremos las velocidades de los motores. Además también recrearemos el proyecto para que la obtención de la ip y modo de envío de paquetes del protocolo TCP/IP sea mediante wifi.			
<input type="checkbox"/> 0 Open <input checked="" type="checkbox"/> 8 Closed			
<input type="checkbox"/> <input checked="" type="radio"/> Mejoras en los códigos del Sprint 4 <span style="background-color: red; border-radius: 5px; padding: 2px 5px;">bug</span> Point: 5 <span style="background-color: green; border-radius: 5px; padding: 2px 5px;">work</span> #35 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Aumento de Pruebas del Sprint 4 <span style="background-color: red; border-radius: 5px; padding: 2px 5px;">bug</span> Point: 5 <span style="background-color: green; border-radius: 5px; padding: 2px 5px;">work</span> #34 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Actualización de la memoria y repositorio en GitHub <span style="background-color: blue; border-radius: 5px; padding: 2px 5px;">documentation</span> Point: 3 #32 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Integración del modulo ESP8266 en el proyecto Point: 8 <span style="background-color: green; border-radius: 5px; padding: 2px 5px;">work</span> #31 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Pruebas y Mejoras sobre el final del Código del Sprint 4 <span style="background-color: red; border-radius: 5px; padding: 2px 5px;">bug</span> Point: 3 <span style="background-color: green; border-radius: 5px; padding: 2px 5px;">work</span> #33 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Integración del uso del potenciómetro en el proyecto Investigation Point: 8 <span style="background-color: green; border-radius: 5px; padding: 2px 5px;">work</span> #30 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Comprensión del funcionamiento de la antena wifi ESP8266 Investigation Point: 5 #29 by eod1001 was closed 26 days ago			
<input type="checkbox"/> <input checked="" type="radio"/> Investigación y Comprensión del funcionamiento de un Potenciómetro Investigation Point: 5 #28 by eod1001 was closed 26 days ago			

Figura A.4: Tareas planificadas para el Sprint4.

#### Sprint 5

Este sprint contiene 9 tareas llegando a un total de 32 puntos de dificultad. Es en este sprint donde se implementa la lectura del sensor de temperatura. Además se centran muchos esfuerzos en tratar de conseguir que todas las tareas funcionen adecuadamente. Se realizan multitud de pruebas y se trata de conseguir que la interacción con el usuario sea simple e intuitiva. Las tareas principales se resumen en:

- Mejora de usabilidad por el usuario.
- Implementación del sensor de temperatura.
- Mejora del código y funcionalidades de los periféricos.

### Sprint 5



No due date 100% complete

Este Sprint se va a centrar en la mejora de la interacción con el usuario y la implantación de otro pequeño proyecto para la empresa en la que curso las prácticas. El proyecto consistirá en instalar en una de las placas un sensor de temperatura que se pondrá en el CPD de la empresa y que esta placa se comunique con otra para alertar en caso de que la temp...  
[Show more ▾](#)

<input type="checkbox"/>	<input checked="" type="radio"/> 0 Open	<input checked="" type="checkbox"/> 9 Closed
<input type="checkbox"/>	<input checked="" type="radio"/> Actualización del repositorio	Point: 1 work
	#42 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Actualización de la memoria	Point: 8 work
	#43 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Pruebas Funcionamiento Completo	bug Point: 5 work
	#44 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Pruebas configuraciones Sensor de Temperatura y receptor de datos	bug Point: 3 work
	#41 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Configuracion de la placa que recibe las temperaturas	Point: 2 work
	#39 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Pruebas de que las configuraciones de los motores funcionan adecuadamente	Point: 2 work
	#40 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Instalación y programación de la placa que tiene el sensor	Point: 3 work
	#37 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Mejora y limpieza del código	Point: 5 work
	#38 by eod1001 was closed 15 days ago	
<input type="checkbox"/>	<input checked="" type="radio"/> Investigación sobre el funcionamiento del Sensor de Temperatura	Investigation Point: 3
	#36 by eod1001 was closed 15 days ago	

Figura A.5: Tareas planificadas para el Sprint5.

La duración del sprint fue de una semana y media.

### Sprint 6

Este Sprint se centra en dar los últimos retoques y completar la documentación a presentar para superar el TFG. Es el sprint más largo con 76 puntos de dificultad. En este caso se deberían haber separado en dos Sprints, pero puesto que todas las tareas se trataban sobre la documentación, se decidió dejar en solo uno. La duración de este sprint fue de dos semanas y media.

- Últimos retoques al código.
- Generar la documentación de todo el trabajo.

### Sprint 6

No due date 0% complete			
Retoques finales. Realización de la memoria y anexos y creación de la presentación.			
<input type="checkbox"/>	<input checked="" type="radio"/> 8 Open	<input checked="" type="checkbox"/> 0 Closed	
<input type="checkbox"/>	<input checked="" type="radio"/> Realización de la presentación <a href="#">documentation</a>	Point: 8	
	#45 opened 15 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Documentación de código y últimos retoques <a href="#">documentation</a>	Point: 8	1
	#46 opened 15 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Realización de la memoria <a href="#">documentation</a>	Point: 13	
	#47 opened 15 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Realización de los anexos <a href="#">documentation</a>	Point: 13	
	#48 opened 15 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Realización de la memoria V2 <a href="#">documentation</a>	Point: 13	
	#49 opened 12 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Bug encontrado en el código <a href="#">bug</a>	Point: 5	
	#50 opened 4 days ago by eod1001		
<input type="checkbox"/>	<input checked="" type="radio"/> Memoria version 2 <a href="#">documentation</a>	Point: 8	
	#51 opened 4 days ago by eod1001		
<input type="checkbox"/> :	<input checked="" type="radio"/> Anexos V2 <a href="#">documentation</a>	Point: 8	
	#52 opened yesterday by eod1001		

Figura A.6: Tareas planificadas para el Sprint6.

### Diagrama de Gantt

Para terminar con esta sección, se muestra el diagrama de Gantt seguido según la planificación temporal para la empresa, teniendo en cuenta festivos y fines de semana.

## A.2. Planificación temporal

9

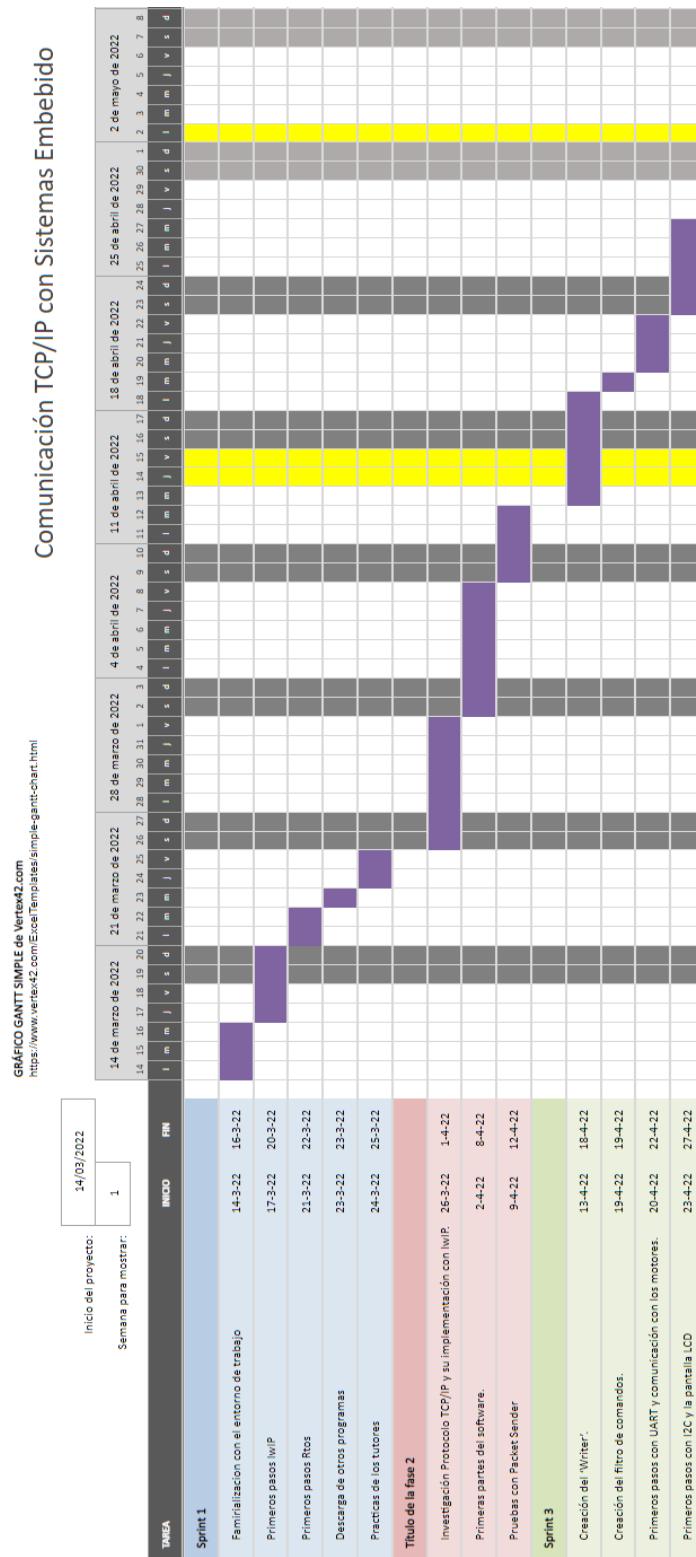


Figura A.7: Diagrama de Gantt. Parte 1.

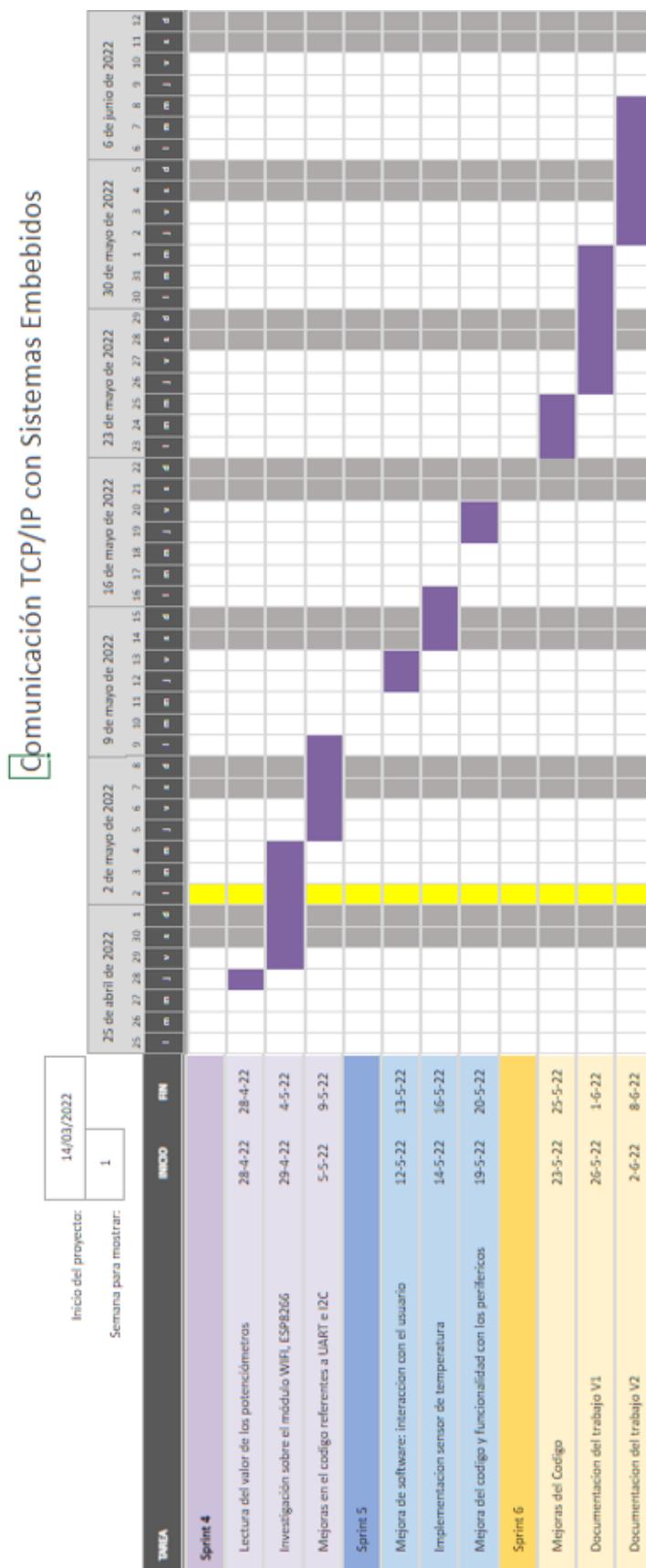


Figura A.8: Diagrama de Gantt. Parte 2.

## A.3. Estudio de viabilidad

### Viabilidad económica

En este apartado se expone la viabilidad económica del desarrollo de este proyecto. Se tendrán en cuenta los gastos previstos durante el desarrollo y los beneficios, si los hubiera. Para calcularlo supondremos que el trabajo se hubiera desarrollado en el entorno real de una empresa.

#### Coste de personal.

Para el desarrollo del producto final, vamos a suponer que se hubiera encargado una sola persona, durante un tiempo aproximado de 3 meses y 25 horas semanales. Supongamos que este trabajador junior tenga un salario bruto de 1500 euros al mes. Para conocer el salario neto lo haremos con la siguiente formula [21]:

$$\text{salario bruto} - \text{IRPF} - \text{SS} = \text{salario neto} \quad (\text{A.1})$$

Por lo que el trabajador recibiría el siguiente salario:

$$1500\text{€} - 10,49\% - 6,35\% = 1247,4\text{€} \quad (\text{A.2})$$

Para la realización de la fórmula anterior se han utilizado los tramos mas altos de la autonomía de Castilla y León [6] y la escala a nivel nacional que podemos encontrar en [22]. El cálculo está simplificado al no tener en cuenta la progresividad de la escala de tramos.

#### Costes pertenecientes a la SS.

Sobre el salario bruto, se producen retenciones y pagos a la Seguridad Social. De estos impuestos algunos los paga la empresa y otros el trabajador. En este apartado se van a calcular los importes de estas retenciones, para ello, tomaremos como referencia la tabla de bases de cotización de la Seguridad Social. En esta tabla, deberemos buscar el grupo que nos corresponda, en este caso “Ingenieros y Licenciados. Personal de alta dirección no incluido en el artículo 1.3. c) del Estatuto de los Trabajadores”. Este grupo tiene unas bases mínimas de 1466,40€/mes. Y unas máximas de 4070,10€/mes.

Concepto	Empresa	Trabajador
Contingencias comunes	23,60 %	4,70 %
Desempleo	5,50 %	1,55 %
FOGASA	0,20 %	0,00 %
Formación	0,60 %	0,10 %
<b>Total</b>	<b>29,9 %</b>	<b>6,35 %</b>

Tabla A.1: Costes pertenecientes a la SS

Como podemos observar en la figura anterior los costes que la empresa debe asumir son el 29,9 % del salario del trabajador y el trabajador sufriría una retención del 6,35 %

### Coste total de personal

Para hallar el coste total del desarrollo del proyecto calcularemos la siguiente formula.

$$(salario mensual + costes ss) \times n^o \text{ meses} = \text{coste total} \quad (\text{A.3})$$

De modo que el coste total se obtiene así:

$$(1500\text{€} + 448,5\text{€}) * 3 = 5845,5\text{€} \quad (\text{A.4})$$

La siguiente tabla recoge diferentes costes asociados al personal y su coste total.

Concepto	Coste
Salarios	1800 €
Seguridad Social	448,5 €
Meses	3 meses
<b>Total</b>	<b>5845,5 €</b>

Tabla A.2: Coste total de personal

### Costes del hardware.

Como ya vimos en la memoria para el desarrollo de este proyecto, se han necesitado varios dispositivos. Aquí surge un inconveniente puesto que,

según se recoge en la ley de impuestos sobre sociedades [4] se necesitan 8 años para amortizar los equipos relacionados con la rama de la información, pero la renovación de este tipo de hardware se debe realizar en un periodo mucho menor, de 4 años. En este caso he optado por realizar los cálculos con un periodo de 4 años. Quedaría de la siguiente manera:

El coste mensual de un dispositivo se calcula así:

$$\text{Coste del dispositivo} / \text{Periodo de amortización (en meses)} \quad (\text{A.5})$$

Por lo tanto, el coste de un dispositivo es el siguiente:

$$\text{Coste mensual amortizado} \times n^o \text{ meses} \quad (\text{A.6})$$

Para desarrollar cómodamente este software se ha utilizado una estación de trabajo portátil de 1000€ y un monitor de 24 pulgadas que tiene un coste de aproximadamente 150€.

<i>Hardware</i>	Coste	Coste amortizado
Entorno de trabajo	1150€	71,87€
<b>Total</b>	<b>1150€</b>	<b>71,87€</b>

Tabla A.3: Coste del *hardware*

### Costes del software.

Para el desarrollo del proyecto, se han utilizado varios programas y aplicaciones. Según la ley del impuesto sobre sociedades [8], el máximo de años para realizar la amortización de sistemas y programas informáticos es de 6 años. Sin embargo, al igual que hicimos en el apartado de los costes hardware, se calculará en un periodo de 4 años. Es importante tener en cuenta que la mayoría de programas utilizados se encuentran bajo licencias que permiten su uso sin coste, por lo que no aparecerán en la tabla de costes de software. Por otro lado, gran parte del software utilizado se usa mediante suscripción, por lo que su coste aparecerá computado bajo su suscripción determinada.

<i>Software</i>	Coste	Coste amortizado
Windows 10 Pro[14]	259€	21,58€
Office 365[13]	126€	31,5€
<b>Total</b>	<b>385€</b>	<b>53,08€</b>

Tabla A.4: Coste del *software*

### Coste del sistema empotrado.

Veamos los costes de los componentes utilizados para el laboratorio.

<i>Hardware</i>	Coste	Unidades	Coste Total
Placas FRDM-K64F[24]	42,36€	3	127,08€
Shield Arduino[9]	21,50€	1	21,50€
Pantalla LCD[25]	14,58€	1	14,58€
Motores[10]	70,57€	2	141,14€
Placa Motores[20]	93,46€	1	93,46€
Switch[19]	10,95€	1	10,95€
Cables de Red[23]	2,55€	3	7,65€
Cables [26]	5,12€	1	5,12€
<b>Total</b>	-	-	<b>421,48€</b>

Tabla A.5: Coste del Sistema Empotrado

### Coste total del proyecto

A continuación se muestra el coste total asignado a todo el proyecto:

Tipo de coste	Coste
Personal	5845,5€
<i>Hardware</i>	71,87€
<i>Software</i>	53,08€
Componentes del SE	421,48€
<b>Total</b>	<b>6391.93€</b>

Tabla A.6: Coste total del proyecto

### Beneficios del proyecto

Es importante mencionar que el SE no tiene una función comercial específica y por lo tanto no se tiene un beneficio económico. El proyecto se ha realizado como investigación para que en el futuro se puedan crear sistemas de SE semejantes, rentabilizando los costes producidos por este proyecto. Por lo tanto, no se consideran los gastos como pérdidas sino como una inversión generada con el objetivo de la adquisición de competencias.

### Viabilidad legal

En cuanto a la viabilidad legal, encontramos que la parte más importante sería el conocimiento y utilización de licencias. Las licencias establecen una serie de términos y condiciones en los que se pueden utilizar los softwares de terceros. En el próximo apartado, veremos cuales son las licencias que se han usado durante el desarrollo de este proyecto.

#### Licencias utilizadas en el desarrollo del SE

En esta subsección, se muestran las licencias que tiene el software de terceros que se ha utilizado para la creación del programa presentado:

- “The 3-Clause BSD License” (BSD-3): Bajo este tipo de licencia estaría el software generado por MCUXpresso y lwIP. Esta licencia permite crear y distribuir nuevo software a partir de los programas que estén adscritos a este tipo de licencia.
- Apache 2.0: Hay una parte del código del procesador de la placa que pertenece a ARM Limited y se encuentra bajo la licencia de Apache 2.0. Esta licencia permite el uso, copia, modificación y redistribución del código.
- El código perteneciente a FreeRTOS es propiedad de Amazon.com, Inc. Se permite el uso, copia, modificación, publicación, distribución, volver a licenciar y comercializar, manteniendo siempre el aviso de derechos de autor.

**Licencias para el SE** Puesto que se pretende que el software realizado sea abierto, se publicará bajo la licencia Apache 2.0 [11]. Podemos ver sus características en la A.7.

Permisos	Condiciones	Limitaciones
Uso comercial	Aviso de licencia	Uso de marcas registradas
Modificación	y derechos de autor	Responsabilidad
Distribución	Declaración de los cambios	Garantía
Uso en patentes		
Uso privado		

Tabla A.7: Licencia Apache 2.0

**Utilización de la licencia Apache** Para indicar a terceros que el software realizado se encuentra bajo la licencia Apache, se expone en el directorio raíz un fichero que recibe el nombre de ‘License’ con el siguiente texto:

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Por supuesto se deberáN modificar los apartados que se encuentran entre corchetes e introducir el año y nombre del propietario del software. La inclusión de este texto en un trabajo, permite que quede constancia de que se le aplica la licencia de Apache 2.0 [27]

### Licencia para la documentación

En el caso de la licencia de la documentacion, escogemos una licencia diferente y mas adecuada como es el caso de Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional (CC BY-NC-SA 4.0) [5].

La licencia permite:

- Que la obra realizada pueda ser compartida libremente
- Que pueda ser copiada y redistribuida.
- Que se pueda editar y crear nuevos ficheros partiendo de la original.

Como obligación para todo aquel que utilice este proyecto se requiere exponer la autoría del autor del trabajo original y sea presentada bajo la misma licencia que posee la original.

Permisos	Condiciones	Limitaciones
Distribución	Crédito al autor	Responsabilidad
Modificación	Aviso de licencia	Uso de patentes
Uso privado	y derechos de autor Declaración de los cambios	Uso de marcas registradas Garantía
	No comercial	
	Mismo licenciamiento	

Tabla A.8: Licencia CC BY-NC-SA 4.0

El uso de la licencia queda expuesto mediante el logo de Creative Commons y un pequeño texto que lo acompaña, indicando a los lectores de la obra que está protegida bajo los derechos y obligaciones de esa licencia.



## *Apéndice B*

---

# Especificación de requisitos

---

## B.1. Introducción

En este apéndice se exponen los requisitos del software creado y la funcionalidad de este. Para ello, se da un descripción completa de las funciones que debe realizar el programa y de cómo un usuario podría utilizarlo. Se definen los pasos y opciones para utilizar correctamente el hardware del sistema empotrado y que cumpla así con su función. Por otro lado, se definen también los requisitos no funcionales. Para la exposición de estos requisitos se seguirán algunos estándares como el IEEE 8301998 [18] o el IEEE 291482011 [16].

### Audiencia destinataria

Este documento tiene como destinatario, cualquier persona que quiera iniciarse en el ámbito de los sistemas embebidos. Tanto alumnos, como profesores u otros usuarios interesados, podrán encontrar información sobre comunicaciones con periféricos y entre SE, y para qué y cómo pueden utilizarse.

Para la exposición de las siguientes secciones se utilizarán, por motivo de simplicidad, algunas abreviaciones:

**Sistema empotrado o embebido.** SE

**Software del sistema empotrado.** SW del SE

**Requisito de interfaces externas.** RE

**Requisito funcional.** RF

**Requisito no funcional.** RNF

**Casos de Uso.** CU

## Objetivos generales

En esta sección vamos a ver los objetivos para el funcionamiento del SE:

- Configurar un SE para que pueda conectarse en red mediante cable y comunicarse utilizando TCP/IP
- Gestionar el hardware de la planta piloto mediante envío de comandos software entre placas

A partir de estos dos objetivos principales vamos a continuar viendo en los demás apartados los requisitos que tiene la planta piloto.

## Tipos de usuario

Para la utilización del sistema empotrado no se necesita ningún tipo de conocimiento técnico por lo que cualquier persona interesada puede usarlo sin problemas. La mayor dificultad radica en conectar los componentes correctamente, conocer los casos de uso del SE y los pasos para utilizar la placa ‘maestro’. Teniendo en cuenta que junto con el software se proporcionan varios documentos:

- Manual del usuario.
- Manual del programador.
- Casos de uso de la planta piloto.

Cualquier usuario podría utilizarlo atendiendo a esos documentos.

## B.2. Catálogo de requisitos

### Requisitos externos

- **RE-1 Acceso a la red.** El SE tiene que ser capaz de usar Ethernet para conectarse en red. RE de alta prioridad.

- **RE-2 Transmisiones en red.** El SE tiene que ser capaz de usar el protocolo TCP/IP para comunicarse con otros sistemas embebidos. RE de alta prioridad.
- **RE-3 Uso de Botones.** Los seis botones disponibles en la placa más el shield deben poderse utilizar correctamente. RE de media prioridad.
- **RE-4 Uso de dispositivos ADC.** Tanto el sensor de temperatura como los potenciómetros deben realizar adecuadamente sus lecturas. RE de alta prioridad.
- **RE-5 Uso de pantalla LCD** La pantalla debe mostrar los mensajes requeridos correctamente. RE de media prioridad

## Requisitos Funcionales

- **RF-1 Recepción de comandos** El SE tiene que recibir comandos transmitidos mediante paquetes TCP con destino a su correspondiente dirección IP y puerto TCP abierto. RF de alta prioridad.
- **RF-2 Identificación de comandos** El SE tiene que ser capaz de identificar los comandos recibidos para poder ejecutar las acciones correctas. RF de alta prioridad
- **RF-3 Movimiento de los motores** Los motores deben realizar correctamente su giro dependiendo del comando recibido. RF de alta prioridad.
- **RF-4 Obtención de la velocidad de los motores** La obtención de los valores de velocidad de los motores debe ser recibida sin demora y correctamente. RF de media prioridad.
- **RF-5 Obtención de la temperatura** La obtención de los valores de temperatura debe ser recibida sin demora y correctamente. RF de media prioridad.
- **RF-6 Envío de comandos** Los comandos se deben enviar por red y no deben perderse en el proceso de comunicación. RF de alta prioridad.
- **RF-7 Parada de ambos motores** La parada de ambos motores al pulsar el ‘botón de emergencia’ debe ser rápida y completa. RF de alta prioridad.

- **RF-8 Comunicación I2C** La comunicación con la pantalla LCD debe ser mediante I2C. RF de alta prioridad.
- **RF-9 comunicación UART** La comunicación con los motores mediante UART debe ser rápida y confiable. RF de alta prioridad.

## Requisitos No Funcionales

- **RNF-1 Hardware del SE** El SE tiene que ser desarrollado con una placa de desarrollo FRDM-K64F. RNF de alta prioridad.
- **RNF-2 Rendimiento del SE** El SE tiene que ser capaz de realizar las acciones indicadas por el usuario sin demora. RNF de media prioridad.
- **RNF-3 Seguridad del SE** El SE tiene que asegurar que sus componentes no presentan riesgos eléctricos al usuario. RNF de alta prioridad.
- **RNF-4 Calidad del SW** El SW tiene que garantizar cierto nivel de calidad, tales como incluir comentarios que faciliten su comprensión para un mantenimiento o portabilidad posteriores. RNF de media prioridad.
- **RNF-5 Usabilidad del SE** La utilización del sistema debe ser sencilla para el usuario. RNF de alta prioridad.
- **RNF-6 Conectividad física de los componentes** Los elementos que componen el sistema deben estar conectados de manera adecuada. RNF de alta prioridad.

## B.3. Especificación de requisitos

### Diagrama de Casos de Uso

En la figura B.1 se muestran los casos de uso que un usuario puede llevar a cabo con la planta piloto.

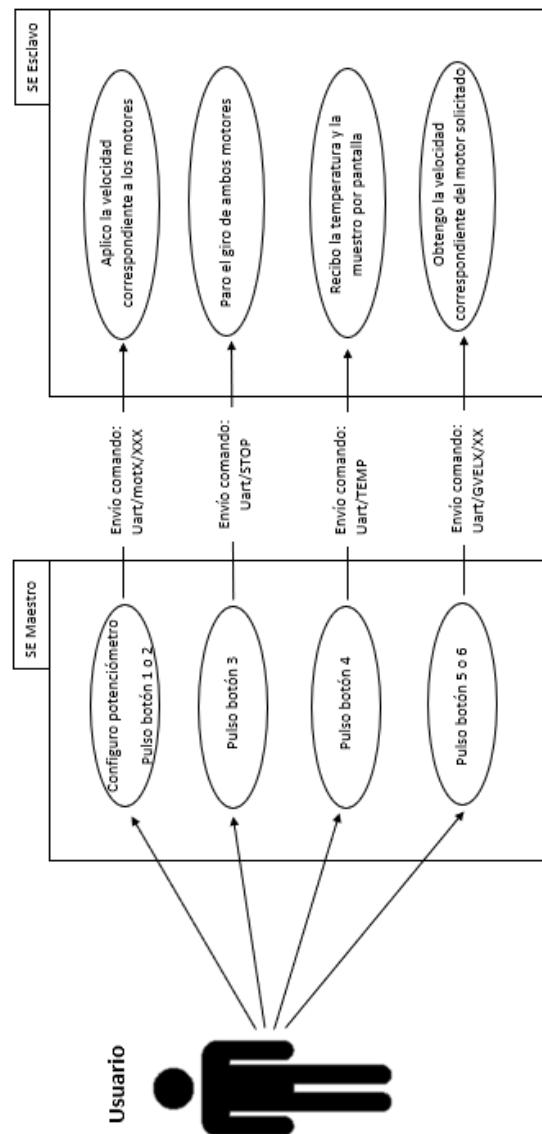


Figura B.1: Diagrama de casos de uso.

## Casos de Uso

Las siguientes tablas muestran los requisitos de los casos de uso:

<b>CU-1</b>	Set Speed A
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-3
<b>Descripción</b>	Fijar la velocidad del Motor A
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas.</li> <li>■ Todos los elementos del sistema deben estar conectados</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario ajusta el potenciómetro 1 según sus necesidades.</li> <li>2. El usuario pulsa el botón 1 para enviar el comando.</li> <li>3. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Postcondición</b>	El motor comienza a moverse
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará</li> <li>■ La lectura del potenciómetro puede no realizarse bien en algunas ocasiones.</li> </ul>
<b>Importancia</b>	Alta
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>■ Los cuatro leds de la placa shield se irán encendiendo a medida que avance el envío del comando</li> </ul>

Tabla B.1: CU-1 Set Speed A.

CU-2	Set Speed B
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-3
<b>Descripción</b>	Fijar la velocidad del Motor B
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas</li> <li>■ Todos los elementos del sistema deben estar conectados</li> </ul>
<b>Acciones</b>	<ol style="list-style-type: none"> <li>1. El usuario ajusta el potenciómetro 2 según sus necesidades.</li> <li>2. El usuario pulsa el botón 2 para enviar el comando.</li> <li>3. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Postcondición</b>	El motor comienza a moverse
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará</li> <li>■ La lectura del potenciómetro puede no realizarse bien en algunas ocasiones.</li> </ul>
<b>Importancia</b>	Alta
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>■ Los cuatro leds de la placa shield se irán encendiendo a medida que avance el envío del comando</li> </ul>

Tabla B.2: CU-2 Set Speed B.

<b>CU-3</b>	Parada de emergencia
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-7
<b>Descripción</b>	Parada de emergencia de ambos motores
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas</li> <li>■ Todos los elementos del sistema deben estar conectados</li> </ul> <ol style="list-style-type: none"> <li>1. El usuario pulsa el botón 3 para enviar el comando.</li> <li>2. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Acciones</b>	Ambos motores se paran
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará</li> <li>■ La lectura del potenciómetro puede no realizarse bien en algunas ocasiones.</li> </ul>
<b>Excepciones</b>	
<b>Importancia</b>	Alta
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>■ Los cuatro leds de la placa shield se encenderán a la vez.</li> </ul>

Tabla B.3: CU-3 Parada de Emergencia.

<b>CU-4</b>	Obtener temperatura
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-5
<b>Descripción</b>	El usuario solicita la temperatura ambiente.
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas.</li> <li>■ Todos los elementos del sistema deben estar conectados.</li> </ul> <ol style="list-style-type: none"> <li>1. El usuario pulsa el botón 4.</li> <li>2. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Acciones</b>	<ul style="list-style-type: none"> <li>■ La temperatura se muestra en la pantalla de la placa esclavo 1.</li> </ul>
<b>Postcondición</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará</li> </ul>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Las dos primeras lectura del sensor de temperatura son de calibración.</li> </ul>
<b>Importancia</b>	Alta
<b>Comentarios</b>	Los leds de la placa shield se encenderán a la vez

Tabla B.4: CU-4 Obtener Temperatura.

<b>CU-5</b>	Obtener Velocidad Motor A
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-4
<b>Descripción</b>	Obtener velocidad del Motor A
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas.</li> <li>■ Todos los elementos del sistema deben estar conectados</li> </ul> <ol style="list-style-type: none"> <li>1. El usuario pulsa el botón 5 para enviar el comando.</li> <li>2. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Acciones</b>	<ul style="list-style-type: none"> <li>■ Se muestra por la pantalla LCD la velocidad del motor A.</li> </ul>
<b>Postcondición</b>	
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará</li> </ul>
<b>Importancia</b>	Alta
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>■ Los cuatro leds de la placa shield se encenderán al mismo tiempo</li> </ul>

Tabla B.5: CU-5 Obtener Velocidad Motor A.

<b>CU-6</b>	Seleccionar SE
<b>Versión</b>	1.0
<b>Fecha</b>	2022-06
<b>Requisitos asociados</b>	RF-4
<b>Descripción</b>	Obtener velocidad del Motor B
<b>Precondición</b>	<ul style="list-style-type: none"> <li>■ Se debe tener conexión en red entre las placas.</li> <li>■ Todos los elementos del sistema deben estar conectados.</li> </ul> <ol style="list-style-type: none"> <li>1. El usuario pulsa el botón 6 para enviar el comando.</li> <li>2. Se corrobora mediante la pantalla LCD y los leds de la placa maestro que se ha enviado el comando.</li> </ol>
<b>Acciones</b>	<ul style="list-style-type: none"> <li>■ Se muestra por la pantalla LCD la velocidad del motor B.</li> </ul>
<b>Postcondición</b>	
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>■ Si alguno de los elementos del sistema no está bien conectado o no se dispone de corriente no funcionará.</li> </ul>
<b>Importancia</b>	Alta
<b>Comentarios</b>	<ul style="list-style-type: none"> <li>■ Los cuatro leds de la placa shield se encenderán al mismo tiempo.</li> </ul>

Tabla B.6: CU-6 Obtener Velocidad Motor B.



## *Apéndice C*

---

# **Especificación de diseño**

---

## **C.1. Introducción**

En el apéndice anterior se detallaron los requisitos del software. El siguiente paso es mostrar las especificaciones de diseño. En este apéndice se explicarán los motivos para las soluciones tomadas en relación al diseño de este programa. El diseño del software recoge las cuestiones relativas a la gestión de los datos y su presentación, a la división de las partes del código en otros módulos y cómo se estructura su funcionamiento. En definitiva, se expone cómo se estructura el software arquitectónicamente.

### **Ámbito del software**

El software desarrollado permite al usuario controlar motores y ser conocedor de la temperatura ambiente de la zona en la que está situado, mediante el manejo de un sistema empotrado que se comunica por red con otros SE que, a su vez, interactúan con los motores o medios de interacción con las personas como pantallas LCD o leds. El software es específico, como es habitual en el ámbito de los SE, para la planta piloto presentada. La interacción con el usuario es muy sencilla y se realiza mediante la pulsación de botones y el giro de potenciómetros.

## **C.2. Diseño de datos**

La base del proyecto es la comunicación entre sistemas empotrados. En esta comunicación se envían comandos que serán recibidos y gestionados por otros sistemas embebidos. Los comandos son cadenas de caracteres que

se procesarán por software hasta activar la acción que el SE debe realizar. Veamos el proceso en detalle.

### Transferencia de los comandos.

La comunicación entre placas para enviar y recibir comandos se implementa siguiendo el protocolo TCP Transmision Control Protocol. Este modelo se encarga de realizar la comunicación estableciendo la conexión. Para realizar esta labor enviara unos comandos de sincronización [7] tal y como veremos en la Figura C.1

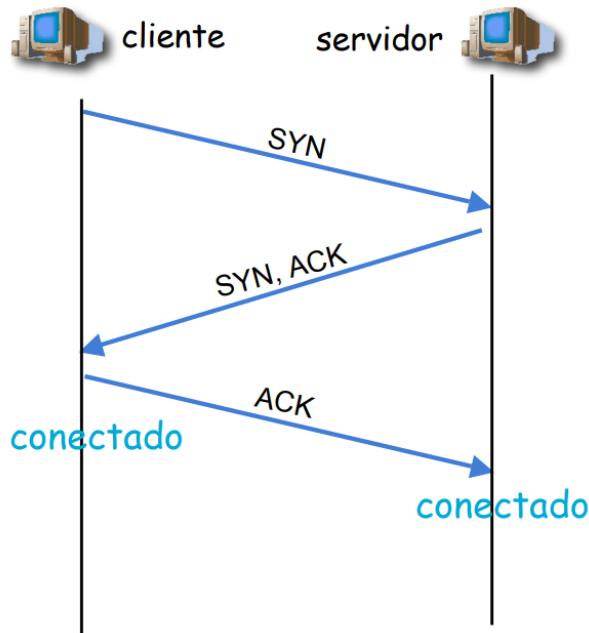


Figura C.1: Establecimiento de conexión TCP (*3 way Handshake*)

Los pasos a seguir para establecer la conexión son:

1. El cliente envía un segmento al servidor solicitando la conexión. Este segmento no contiene datos, tan solo la cabecera. A este segmento se le conoce como 'SYN'.
2. El servidor responde al segmento confirmando 'ACKnowledgement' la recepción del 'SYN'. Este segmento indica al cliente el deseo de establecer conexión (SYN). Al igual que el segmento anterior, no contiene

datos, solo cabecera. Tras ejecutar este proceso de inicialización, los comandos se envían encapsulados en los segmentos TCP.

3. Por último, el cliente envía confirmación al ‘SYN’ enviado por el servidor y se establece la conexión.

Una vez establecida la conexión, cliente y servidor pueden intercambiar segmentos con datos. Para interrumpir esa conexión se deben enviar los paquetes que se muestran en la C.2

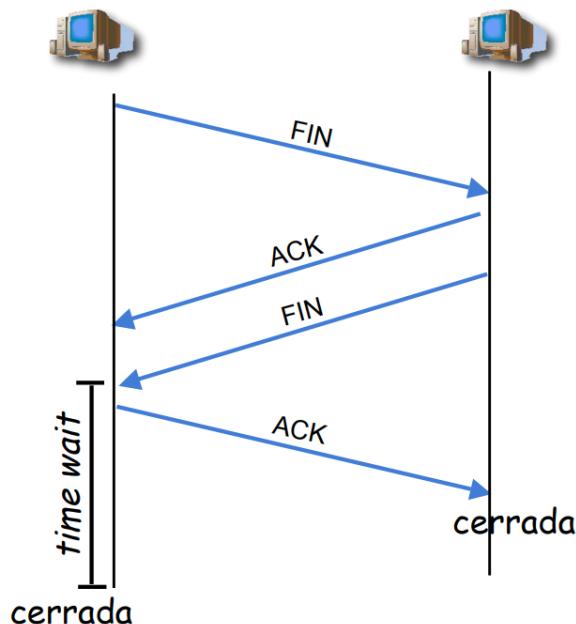


Figura C.2: Cierre de conexión TCP.

Los pasos siguientes describen como se interrumpe la conexión TCP:

1. Tanto el servidor como el cliente pueden tratar de terminar la conexión. Para ello se comienza enviando un comando a la otra parte llamado ‘FIN’ sin datos.
2. La parte que recibe el comando debe confirmar su recepción enviando un (ACK). La parte que envió el comando FIN ya no puede enviar más datos.
3. La misma parte que recibió el comando ‘FIN’, envía un segmento sin datos, solicitando el cierre.

4. Por último, el receptor del segundo segmento ‘FIN’, debe confirmar la recepción del segmento. Por si ese último ‘ACK’ se perdiera, el emisor del mensaje debe esperar un tiempo, incluso en algunas ocasiones podría tener que volver a enviarlo.

Con estos cuatro pasos se cerraría una conexión TCP.

## Envío y recepción de comandos

Todos los casos de uso para los que se puede utilizar la planta piloto se realizan mediante el envío de comandos. Desde la placa maestro, el usuario utiliza los botones y potenciómetros para configurar el envío de una petición a las otras placas que están conectadas mediante red. Una vez enviado el comando, una de las dos placas esclavo. El primer paso es descomponer el comando en distintas variables que procesarán hasta llegar a enviar una petición a los motores o reenviar información a la placa maestro o mostrarla por pantalla, en ambas se tramita el comando de la misma manera.

Los comandos tienen el siguiente formato “\*\*\*\*/\*\*\*\*/\*\*\*\*/”. Cada una de las barras hace de separador para guardar la información en cada variable, la longitud de los caracteres puede variar dependiendo del comando que se quiera enviar. Los comandos están determinados en el propio software y no pueden variar puesto que el usuario no los escribe tan solo maneja los botones de las placas para elegir cual mandar.

En la siguiente [C.1](#) vamos a ver cuáles son los comandos existentes en este proyecto

CMD	Arg1	Arg2	Arg3	Descripción
Set Speed 1	uart/	Vel1/	0/	El motor 1 gira hacia la izquierda
-	uart/	Vel1/	128/	El motor 1 se para
-	uart/	Vel1/	255/	El motor 1 gira hacia la derecha
Set Speed 2	uart/	Vel2/	0/	El motor 2 gira hacia la izquierda
-	uart/	Vel2/	128/	El motor 2 se para
-	uart/	Vel2/	255/	El motor 2 gira hacia la derecha
Parada de emergencia	uart/	stop/	-	Ambos motores se paran
Obtener temperatura	temp/	'X' /	-	El maestro envía la temperatura envía la temperatura
Obtener velocidad 1	uart/	GVel1/	-	El maestro pide la velocidad del motor A
Obtener velocidad 2	uart/	GVel2/	-	El maestro pide la velocidad del motor B
Obtener velocidad 1a	g_vel/	mot1/	'X' /	Esclavo devuelve la velocidad del motor A. 'X' podrá ser 0, 128, 255
Obtener velocidad 1b	g_vel/	mot2/	'X' /	Esclavo devuelve la velocidad del motor B. 'X' podrá ser 0, 128, 255

Tabla C.1: Lista de comandos

### Implementación de la conexión con lwIP

LwIP incluye la librería Netconn API [15] nos facilita varias funciones que podemos utilizar para el establecimiento de una conexión, envío de datos e interrupción de la conexión. Todos los SE de la planta piloto cuentan con recepción de comandos (*Listener*) y con envío de comandos (*Writer*). Los comandos utilizados para la recepción de datos en este proyecto son:

- `netconn_new`. Crea una conexión.
- `netconn_bind`. Vincula un netconn a una dirección IP y un puerto local específicos.
- `netconn_listen`. Establece una conexión TCP en modo de escucha.
- `netconn_accept`. Acepta una conexión entrante en una conexión TCP de escucha

- `netconn_recv`. Recibir datos (en forma de un `netbuf` que contiene un búfer de paquetes) de un `netconn`.
- `netconn_delete`. Cierra una ‘conexión’ `netconn` y libere sus recursos.

Por otro lado, para el envío de datos se han usado las siguientes funciones:

- `netconn_new`. Crea una conexión.
- `netconn_connect`. Conekte un `netconn` a una dirección IP y un puerto remotos específicos.
- `netconn_write`. Envía datos en un TCP `netconn` conectado
- `netconn_delete`. Cierra una ‘conexión’ `netconn` y libere sus recursos.

Estos son las funciones que se han utilizado en el software, sin embargo, existen más funciones en Netconn API para adaptarse a otras necesidades, [12].

### C.3. Diseño arquitectónico

Según el estándar IEEE 420102011 [17] la arquitectura del software se define como ‘conceptos fundamentales o propiedades de un sistema en su entorno encarnados por sus elementos, relaciones, y en los principios de su diseño y evolución’. Dicho esto, veamos de manera más específica el diseño arquitectónico de este sistema de SE.

#### Diseño arquitectónico del SE

Una de las grandes cualidades de los sistemas empotrados es que se pueden relacionar con el entorno mediante sensores. Esto hace que el software se pueda reajustar según los datos obtenidos por estos sensores.

En este caso como ya sabemos, la gestión de las tareas se realiza en tiempo real, lo cual requiere una configuración en las tareas algo más compleja que si dispusiéramos de un SE cíclico. Se deben asignar prioridades y es necesario utilizar un software que nos ayude en esta gestión, en este caso hemos utilizado FreeRtos.

Una buena práctica para este tipo de sistemas es la utilización de interrupciones, así evitamos que el programa tenga que realizar sondeos del

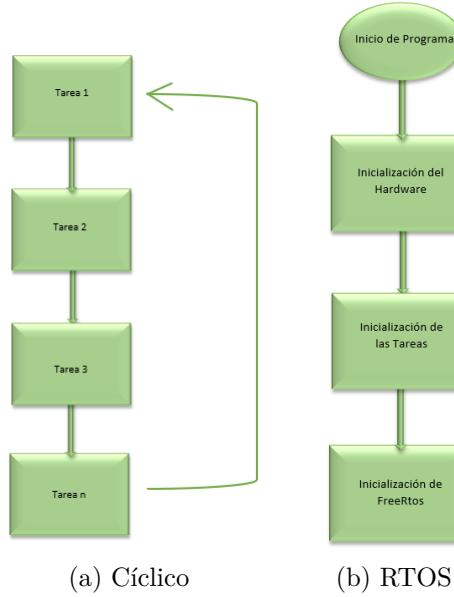


Figura C.3: Distintos algoritmos para la ejecución de un SE

entorno constantemente. Las interrupciones se pueden programar para activarse al pulsar un botón o al detectar algún cambio en el entorno mediante los sensores. En ese momento se activa la interrupción y se pone en marcha la tarea correspondiente.

Una alternativa a utilizar un RTOS es la utilización de una programación cíclica. El algoritmo cíclico recorre todas las tareas una a una y cuando termina con la última vuelve a empezar. Se le puede añadir un periodos, de ejecución máximos, conocidos como 'Quantum', consiguiendo así que una tarea no este en ejecución durante un periodo de tiempo excesivo. Sin embargo, como se comentó anteriormente, se decidió utilizar FreeRTOS. Esta decisión se basa en el bajo tiempo de respuesta que proporciona el SO. Mediante las prioridades elegimos además que tarea debe ejecutarse antes que otras, aportando flexibilidad a la ejecución del programa.

### Tareas programadas

Vamos a ver que tareas se inicializan en la tercera parte de la Figura C.3b para la correcta ejecución del software presentado.

**Tareas del maestro** Las tareas que incluye el software maestro son:

1. Init\_IP. Esta tarea se encarga de Inicializar la pila TCP/IP utilizando lwIP y establecer una IP fija según la dirección MAC de la placa. De esta manera podremos comunicarnos con otras placas en red.
2. Writer\_Thread. Se encarga de enviar los comandos que configura el usuario a las placas esclavo.
3. Listener\_Thread. Se encarga de establecer la conexión para poder recibir datos mediante un puerto y una IP determinados.
4. Temp\_thread. Se encarga de conseguir el voltaje del sensor de temperatura y enviárselo a la placa esclavo.

**Tareas del esclavo** Las tareas que incluye el software esclavo son:

1. init\_ip. Cumple las mismas funciones que en el maestro.
2. listener\_thread. Realiza la misma tarea que en el maestro.
3. uart\_task. Recibe el comando del SE maestro y se comunica con los motores enviando los bytes correspondientes al comando recibido.
4. temp\_task. Recibe el comando del SE maestro con el voltaje del sensor de temperatura. Se encarga de calcular la temperatura y mostrarla por pantalla.

## Diseño del uso de la placa

Los sistemas embebidos tienen como una de sus características principales la sencilla interacción con las personas, por ello se ha tratado de que el usuario que requiera utilizar el sistema, lo haga desde un solo SE y mediante botones y luces led pueda saber si envió la información. Su uso es simple y cuenta además tanto con leds en las tres placas como con una pantalla LCD que muestra un aviso al recibir un comando además de encender los leds de la placa en rojo y verde según este disponible para recibir un nuevo comando o este gestionando uno en ese momento.

## C.4. Diseño procedimental

En esta sección vamos a ver un diagrama de secuencia C.4 que muestra las interacciones entre los SE desde que el usuario configura y manda un comando hasta que se recibe y gestiona ese comando.

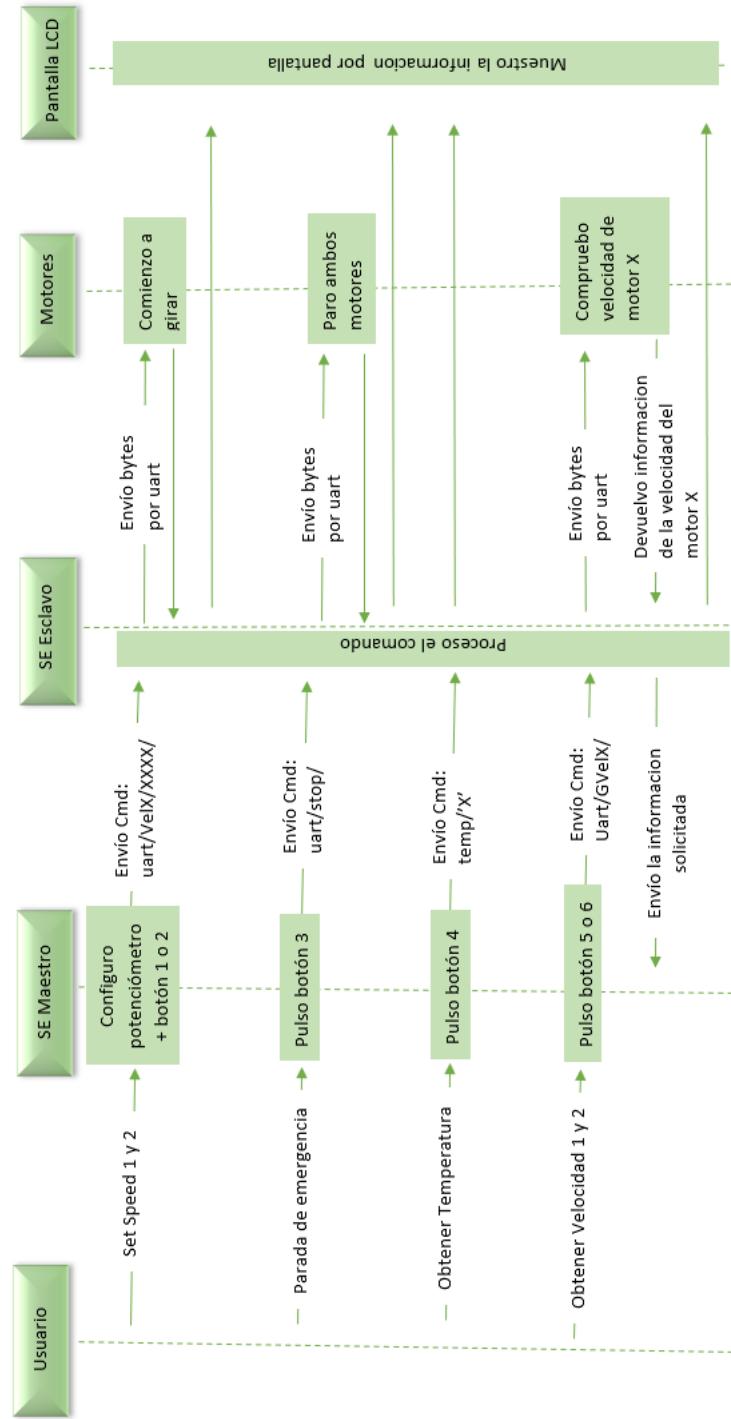


Figura C.4: Diagrama de secuencia de los casos de uso.



## *Apéndice D*

---

# **Documentación técnica de programación**

---

## **D.1. Introducción**

Este apéndice muestra las herramientas necesarias para entender y poder reutilizar el software presentado. Sin embargo, no es necesario utilizar las mismas herramientas, pero es muy recomendable. Para el desarrollo del trabajo se ha utilizado MCUXpresso. A continuación se muestra cómo instalar y configurar este IDE. Además, también veremos otras herramientas que nos ayudarán a comprobar el correcto funcionamiento del software.

## **D.2. Estructura de directorios**

Antes de comenzar la instalación de MCUXpresso veamos de qué ficheros disponemos en este proyecto. La estructura de directorios es:

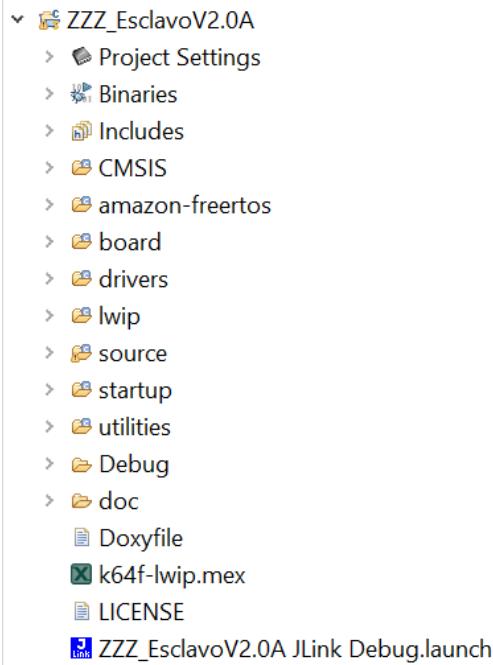


Figura D.1: Estructura de directorios.

/ Directorio raíz, contiene el resto de los directorios. Se incluyen ficheros como LICENSE que contiene los términos y condiciones del licenciamiento del software y el fichero .MEX que contiene los datos de las configuraciones de los pines, relojes y periféricos. Este último fichero lo genera el propio IDE. En la raíz también encontramos los archivos binarios para la ejecución del programa y el archivo para realizar el debug.

**/CMSIS/** Cortex Microcontroller Software Interface Standard (CMSIS). Reúne las fuentes que proporcionan interfaces al procesador y sus periféricos.

**/amazon-freertos/** Fuentes relacionadas con FreeRTOS, sistema operativo en tiempo real usado en el proyecto.

**/board/** Fuentes autogenerados por el IDE que permiten habilitar y configurar el hardware de la placa de desarrollo.

**/doc/** Documentación del proyecto.

**/drivers/** Controladores necesarios para trabajar con el hardware.

**/lwip/** Fuentes relativos a lwIP, la implementación de la pila de protocolos TCP/IP.

**/source/** Código fuente del proyecto. El fichero main.c es el que contiene el código del funcionamiento de las placas. Además se encuentran los ficheros que agrupan herramientas para completar la ejecución de las funciones utilizadas en el main.c.

**/startup/** Código de arranque generado por el IDE.

**/utilities/** Código generado por el IDE con utilidades auxiliares usadas para depuración o registro de eventos.

## D.3. Manual del programador

### Descarga e instalación de MCUXpresso

En primer lugar será necesario descargar el IDE desde la página oficial de NXP. Este software estará en la pestaña de desarrolladores. Para poder descargarlo será necesario tener una cuenta de NXP, la cual es gratuita y te puedes registrar fácilmente desde el sitio web. Dicho esto, iniciamos sesión en la página vamos a la pestaña en la que se encuentra el software y pinchamos en descargar [1].

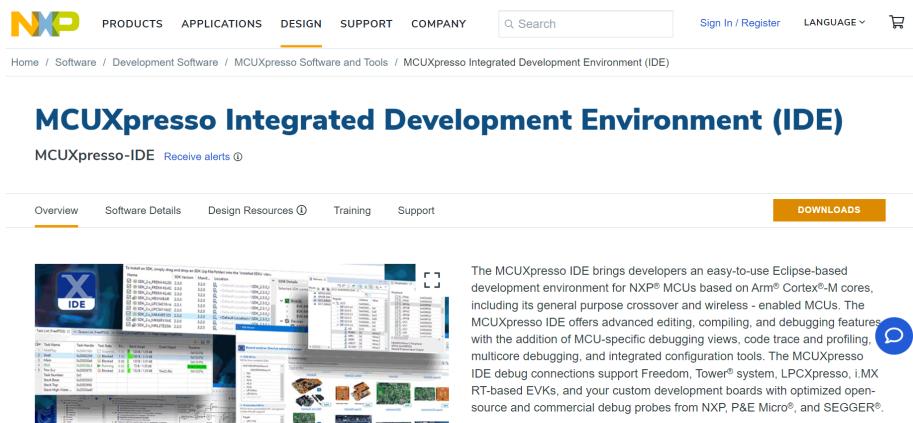


Figura D.2: Pagina de descarga del IDE.

Una vez hecho esto, elegimos el sistema operativo donde se ejecutará nuestro IDE. El instalador sigue los pasos habituales en este tipo de instalaciones, aceptar la licencia, elegir la ubicación donde se guardarán los archivos y controladores de programa, etc. Podemos dejar todas estas opciones por

defecto o variarlas a nuestro gusto. Al descargar el IDE viene con él la herramienta Config Tools que nos ayudará a realizar la configuración a bajo nivel de la placa.

Una vez tenemos instalado MCUXpresso, lo abrimos. Lo primero que nos pide es elegir una ruta donde guardar nuestros proyectos. Es recomendable que sea una ruta fácilmente accesible pues nos será de gran ayuda poder llegar rápidamente y poder importar y exportar de manera más sencilla los proyectos que necesitemos.

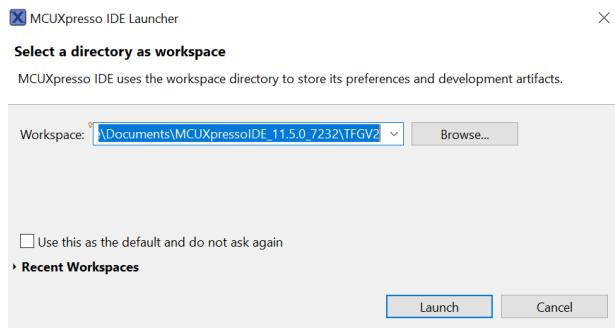


Figura D.3: Elección de la ruta para guardar los proyectos en local.

Tras elegir la ruta y pulsar ‘Launch’ encontraremos la siguiente interfaz:

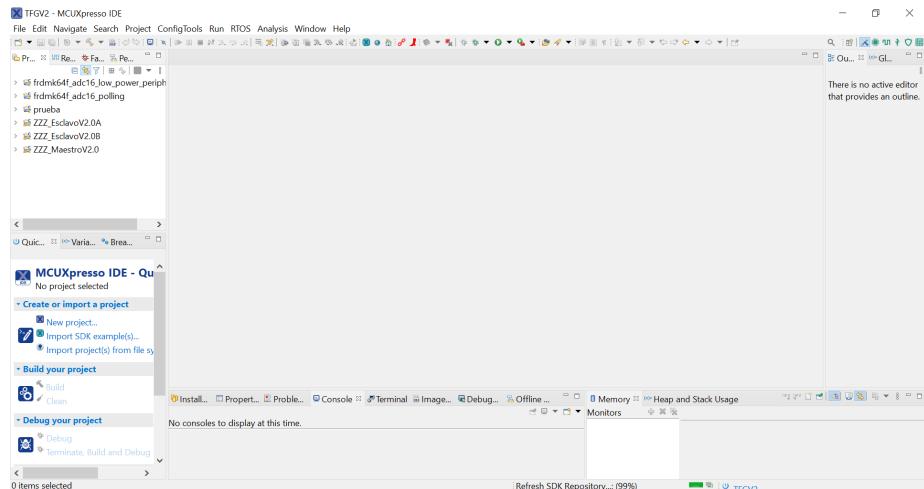


Figura D.4: Interfaz principal MCUXpresso.

Como se puede apreciar en la figura anterior, el IDE proporciona una interfaz semejante al IDE Eclipse.

## Descarga e instalación del SDK

Tras esta instalación del IDE, deberemos realizar el segundo paso en el que descargaremos el SDK. Podemos descargarlo también desde la web de NXP [2], en esta pestaña deberemos elegir el SDK correspondiente a la placa y comprobar que la versión sea superior a la 2.0.

A la hora de descargar el SDK es recomendable seleccionar todos sus paquetes por si los necesitáramos para futuros proyecto pero en caso de que no queramos descargarlos ahora podremos elegir solo algunos y descargarlos a ‘posteriori’ según los vayamos necesitando. Mínimo deberemos descargar FreeRTOS y CMSIS.

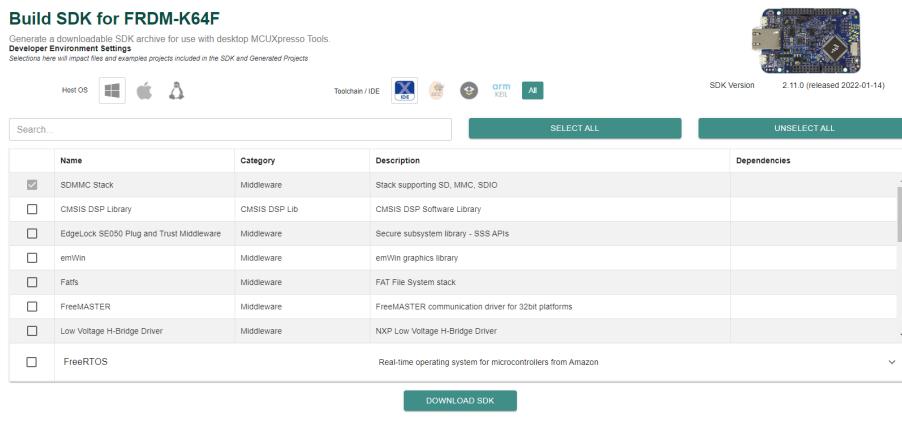


Figura D.5: Pagina de descarga del SDK para la placa K64F.

La descarga del SDK nos permite construir proyectos específicos para nuestra placa, permite además añadir los componentes necesarios según las funcionalidades del SE. Nos ofrece la posibilidad de descargar e instalar drivers, el tipo de sistema operativo o configurar el middleware. En nuestro caso deberemos seleccionar como mínimo el sistema operativo FreeRTOS y como drivers lwIP y ADC para la medición del sensor de temperatura y la lectura de los potenciómetros. Para poder elegir estos add-ons deberemos clicar donde se muestra en la Figura D.6.

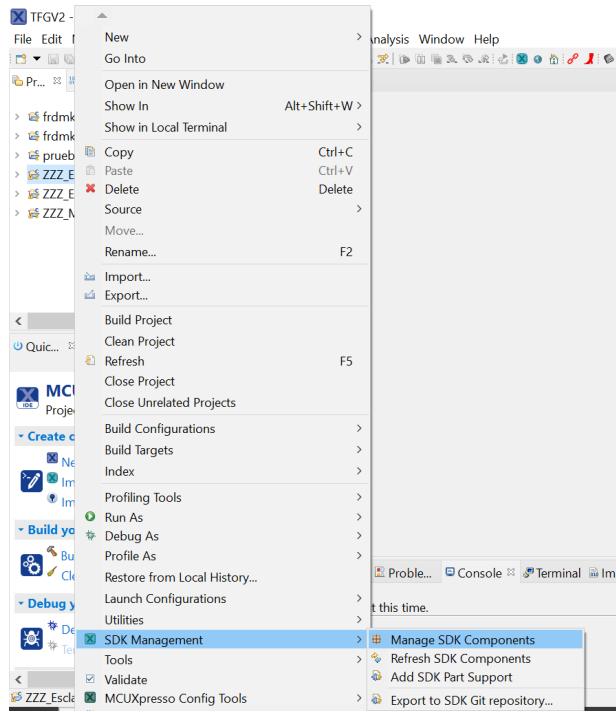


Figura D.6: Abrir el SDK.

Posteriormente se abrirá un interfaz como la de la Figura ??.

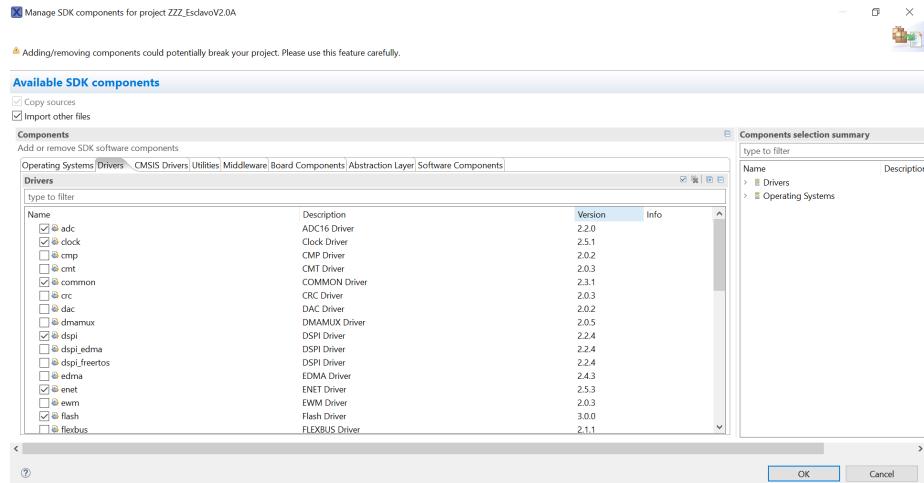


Figura D.7: Elección de los Drivers necesarios.

Donde podremos elegir los componentes que queramos. Como se ve en la figura disponemos de un menú con distintas opciones: *drivers*, *utilities*,

*middleware*. Además la interfaz cuenta con una barra de buscador que nos facilita el trabajo.

## Importación del proyecto

Importar un proyecto es bastante sencillo. Para ello iremos al repositorio de GitHub de este proyecto y descargamos los tres proyectos que componen el sistema en un ‘zip’. Una vez descargados tan solo tendremos que pulsar en la opción ‘import project from file system’ que aparece en la Figura ??:

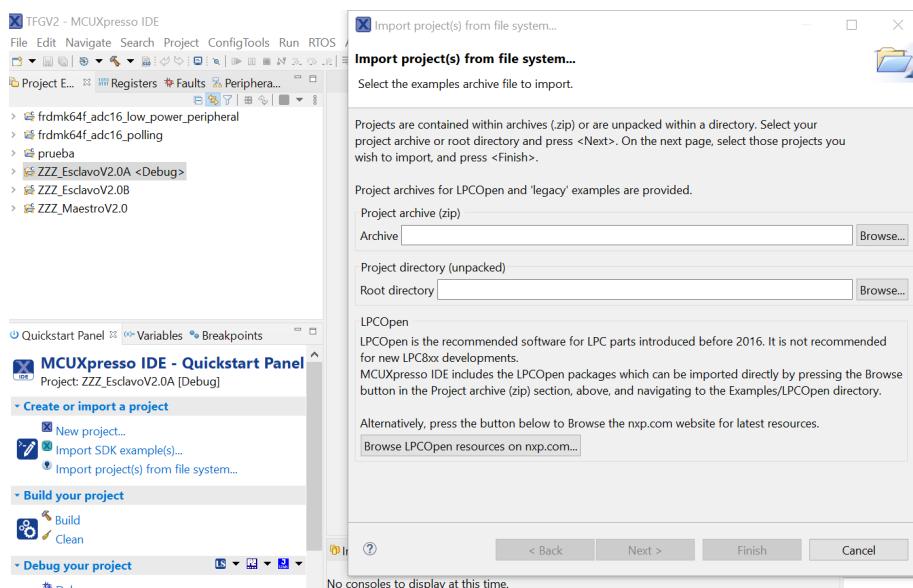


Figura D.8: Importación de un Proyecto.

Seleccionamos la ruta del proyecto o el archivo ‘.zip’ y pulsamos en ‘finish’. De esta manera ya tendríamos el proyecto importado en nuestro IDE. En caso de hacerlo con el ‘.zip’ nos saldrá otra pantalla más, en la que mi recomendación es que marquemos la opción de copiar el proyecto en la ruta de proyecto que elegimos previamente.

## ConfigTools: Configuración de pines, relojes y periféricos

La configuración a bajo nivel de la placa es una de las partes más importantes y a la vez más complejas de entender al principio, por ello voy a explicar brevemente cómo funcionan estas interfaces del IDE.

**Pines** Estas placas disponen de varios pines que son configurables permitiendo así la integración de varios periféricos y aumentando su funcionalidad. En la siguiente figura podemos observar la interfaz para configurarlo:

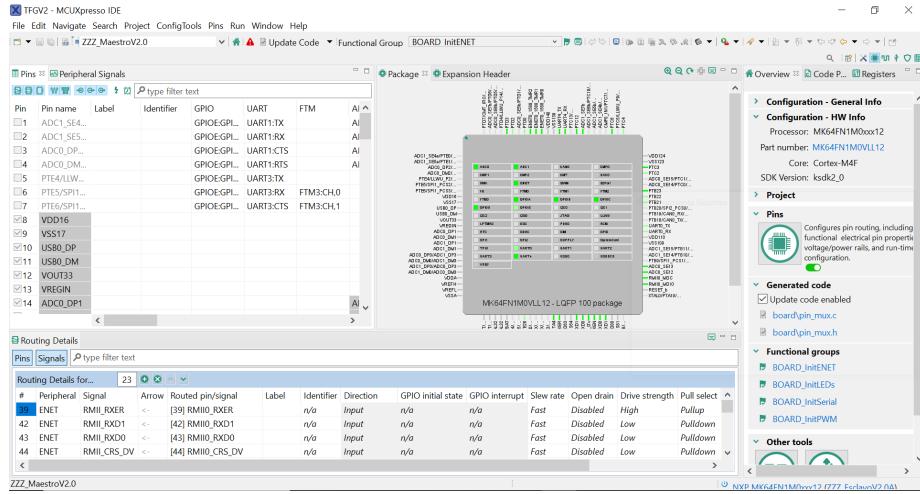


Figura D.9: Interfaz de los Pines en MCUXpresso.

En la parte izquierda de la figura vemos la lista de todos los pines que podemos configurar, que en el caso de esta placa son más de 100. Cada uno de estos pines tiene distintas configuraciones que podremos elegir según nuestras necesidades.

En la parte inferior nos muestra la lista de pines que ya hemos seleccionado y configurado.

Por último, en la parte de la derecha tenemos la imagen del controlador con todos sus pines. Los pines que ya se está utilizando aparecen remarcados en verde.

**Relojes** En el caso de este software se ha utilizado siempre el reloj configurado por defecto pero podemos configurar más relojes dependiendo del objetivo del periférico que vaya a usarlo. La Figura ?? nos muestra una imagen de la interfaz.

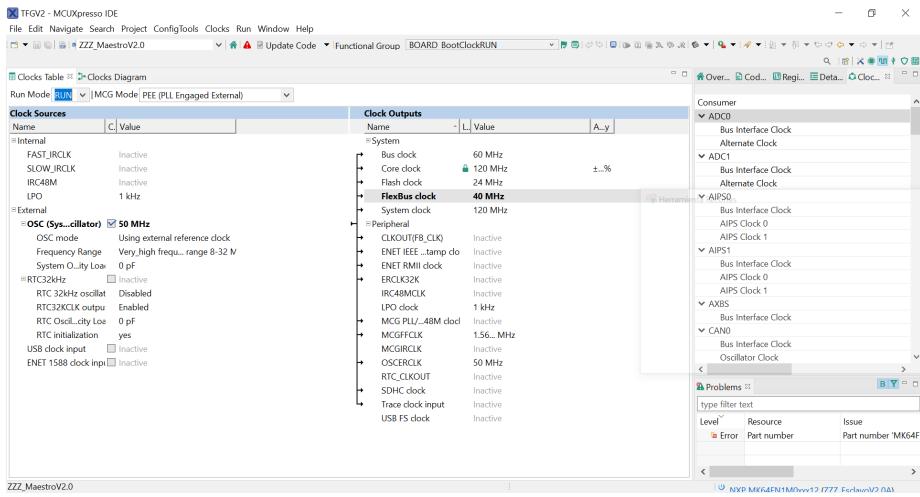


Figura D.10: Interfaz de la tabla de los Relojes en MCUXpresso.

En la figura anterior se ven los relojes asignados a distintos elementos de la placa y en la parte izquierda los asignados a los periféricos. Estos relojes se pueden cambiar en un rango de frecuencias que se muestran en el diagrama de relojes de la Figura ??.

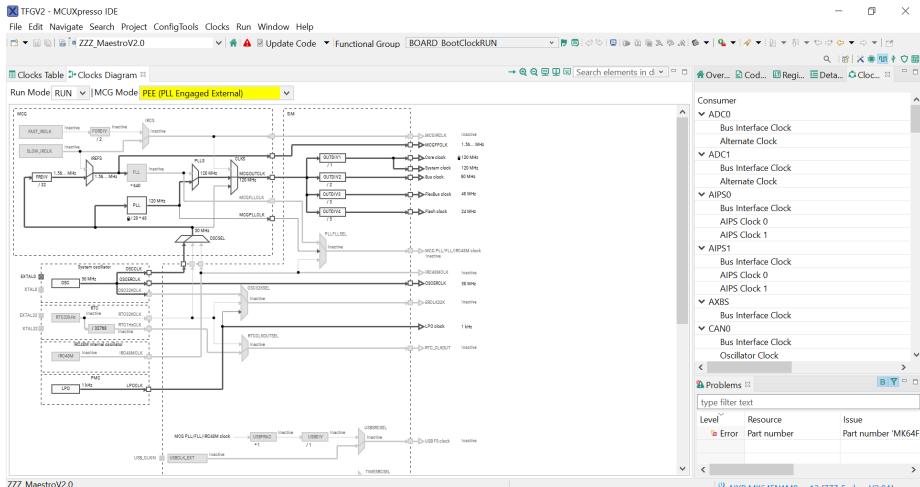


Figura D.11: Interfaz del diagrama de los relojes en MCUXpresso

**Periféricos** el microcontrolador permite conectar varios periféricos a la placa al mismo tiempo. En la Figura ?? se muestran cuales son:

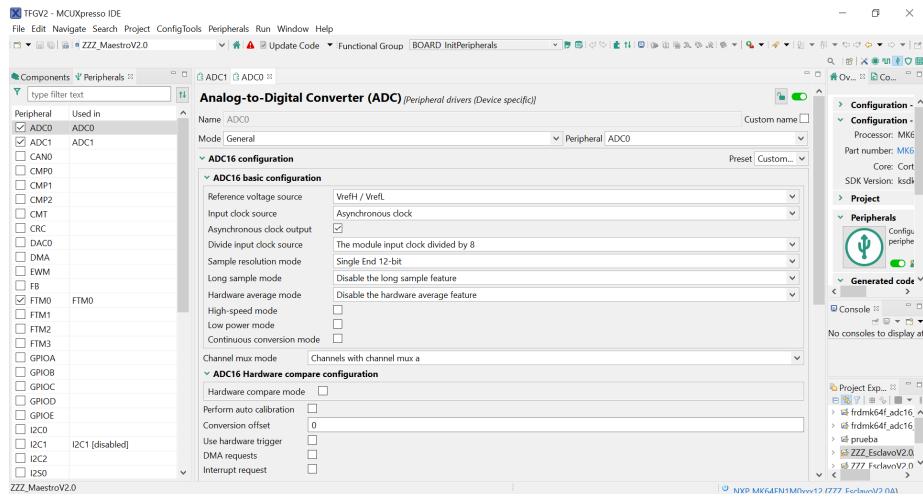


Figura D.12: Interfaz de los Periféricos en MCUXpresso.

Como se puede apreciar, hay algunos repetidos como por ejemplo UART puesto que esta placa permite tener hasta 4 comunicaciones UART configurables al mismo tiempo. En algunas ocasiones, podría darse el caso de que no se puedan añadir más periféricos de esta lista porque los pines que son configurables para ese fin están siendo utilizados, aunque esto solo pasara en casos de sistemas SE muy concretos. Cada una de las configuraciones de los periféricos tiene sus correspondientes opciones específicas. Estas configuraciones nos ayudan para que no tengamos que configurarlas programando y solo tengamos que hacer ‘click’ o elegir algunas de ellas.

## D.4. Compilación, instalación y ejecución del proyecto

Una vez que hemos importado el proyecto, vamos a la carpeta sources y vemos que el fichero tiene los datos fuente del proyecto. Para poder compilar clicaremos en la opción *build* para ver si el proyecto compila adecuadamente o tiene errores.

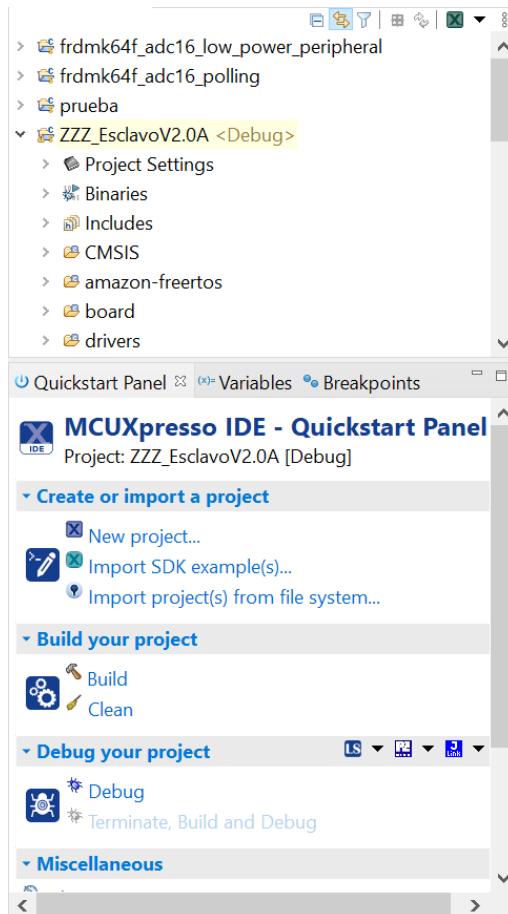


Figura D.13: Compilación del Proyecto.

Posteriormente deberemos conectar por USB la placa FRDM K64F al ordenador. Antes de seguir con el *debugueo*, es importante que la placa esté en modo OpenSDA, para ello, si es la primera vez que la usamos, deberemos pulsar el botón reset mientras conectamos la placa al ordenador. Tras esto se debería de abrir una carpeta llamada ‘bootloader’ donde deberemos copiar el fichero openSDA. Desconectamos y conectamos la placa de nuevo y la placa quedaría lista para realizar la escritura, o flash, de los archivos binarios del proyecto. Para este paso debemos pulsar la opción ‘debug’ para iniciar la depuración. Una vez comience la ejecución se abrirá una consola y podremos utilizar el SE sin problemas. La primera vez que usemos una placa el IDE nos solicita la identificación de la misma. Para ello deberemos seleccionar el uso de ‘Segger J-Link Probes’ que es compatible con OpenSDA y el adaptador serie y de depuración que viene integrado en la placa.

Dicho esto, es importante recalcar que estos métodos para compilar y depurar el programa no son los únicos, puesto que en el IDE se disponen de varios menús que en ocasiones repiten las mismas funcionalidades. Como dato a tener en cuenta, es interesante saber que la ejecución de la opción debug trae consigo la compilación del proyecto de manera automática si no se realizó manualmente antes de ejecutarse.

## D.5. Distribución de pines

En este apartado se muestran los pines usados para la comunicación de la pantalla LCD y de los motores. En la figura D.14 se muestran los pines de la placa FRDM K64F.

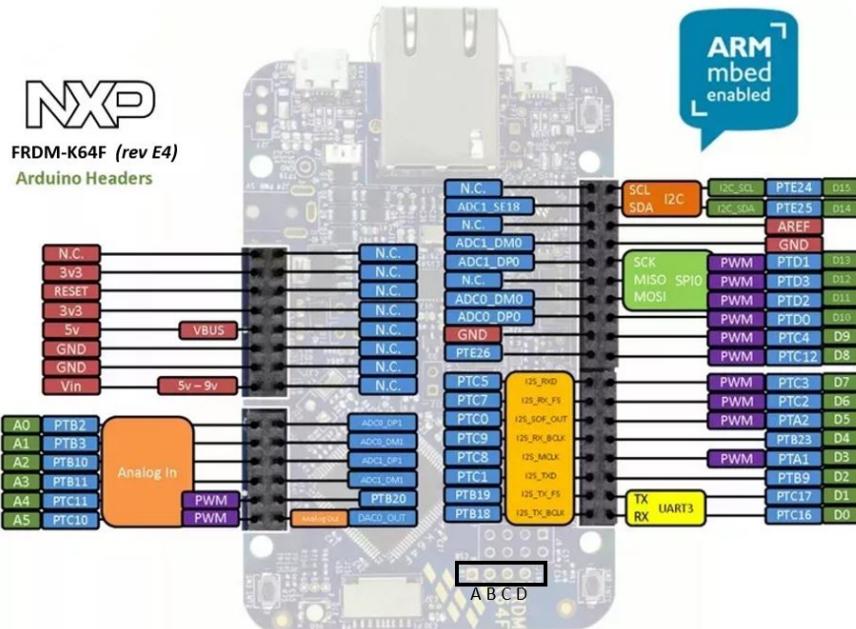


Figura D.14: Definición de los pines de la placa FRDM K64F.

La figura D.15 muestra las conexiones de la pantalla LCD y la placa controladora de los motores para poder entender mejor las siguientes tablas.

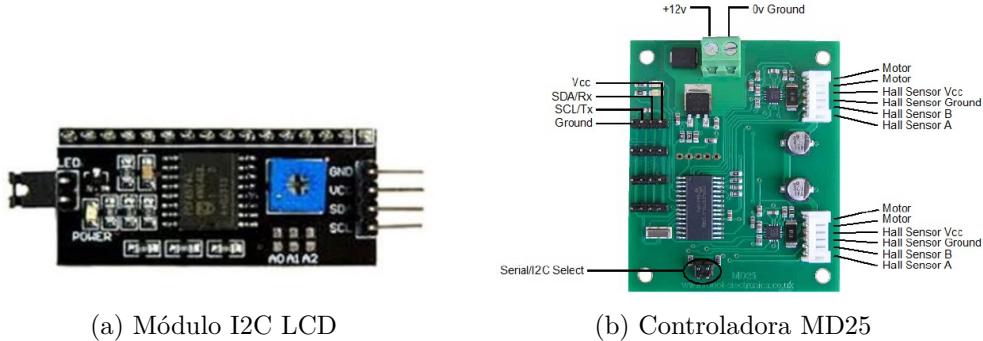


Figura D.15: Conexión de Pines

Los pines de la pantalla LCD se conectarán a los pines de la placa especificados en la tabla D.1

Pin K64F	Pin Pantalla LCD
PTE24	SCL
PTE25	SDA
5V	VCC
GND	TIERRA

Tabla D.1: Pines FRDM K64F y pantalla LCD.

En la parte inferior de la figura D.14 hay cuatro pines marcados con las letras: A,B,C,D, a los que se conectarán a la placa controladora de los motores. En la tabla D.2 se muestra su disposición.

Pin K64F	Pin Motor EMG30
A: 3v	VCC
B: Tierra	GND
C: PTC14, UART, RX	TX
D: PTC15, UART, TX	RX

Tabla D.2: Pines FRDM K64F y motores EMG30.

## D.6. Pruebas del sistema: Packet Sender

Otra herramienta que puede ayudar enormemente a los desarrolladores es packet sender. Para descargarla tendremos que ir a su página web oficial [3] y descargar la herramienta para el sistema operativo donde vayamos a utilizarla. Packet Sender permite enviar paquetes por el protocolo tcp, a una ip y un puerto determinados. Además se pueden guardar nuestros propios comandos de envío para reutilizarlos de forma más sencilla. De esta forma, conseguimos poder comprobar si las placas reciben los comandos adecuadamente y cómo se comportan al recibirlo. Es como tener otra placa en red pero los comandos se envían de forma más sencilla desde nuestro propio ordenador.

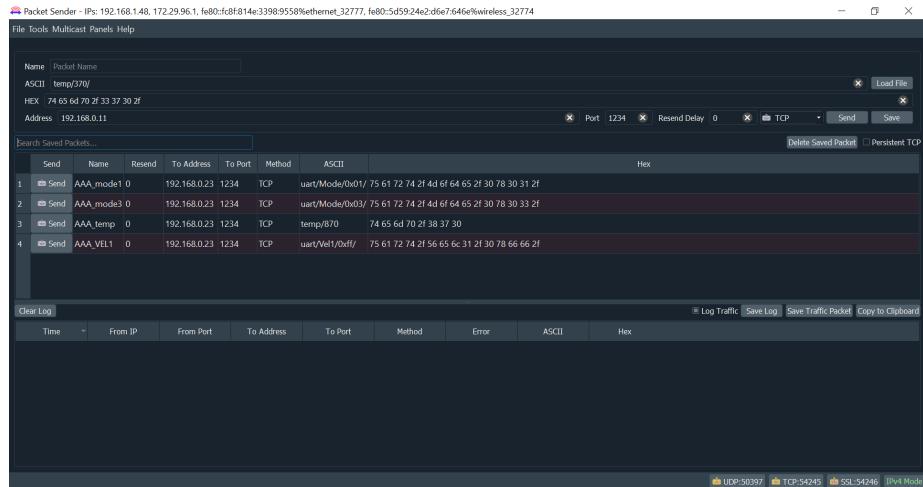


Figura D.16: Interfaz de la herramienta Packet Sender.

## *Apéndice E*

---

# **Documentación de usuario**

---

## **E.1. Introducción**

Esta sección recoge las instrucciones y conocimientos técnicos que un usuario debe conocer para poder utilizar el laboratorio presentado. Veremos cómo instalar el software en nuestra placa y cómo controlar la placa maestro. Se explicarán las funciones que realiza cada botón y cuáles son los usos que tiene el laboratorio.

## **E.2. Requisitos de usuarios**

Para poder utilizar el programa lo primero es disponer de todas las piezas que componen el laboratorio. Deberemos comprobar que todos los componentes están correctamente conectados entre sí. Para ello deberemos:

1. Disponer de tres cables ethernet con clavija RJ-45 y conectar cada uno de ellos a una placa y a un switch.
2. Comprobar que los motores están correctamente conectados a la placa controladora y que las placas FRDM k64F también están conectadas a la placa controladora de los motores. Es importante cerciorarnos de que todos los pines están debidamente conectados a su posición correspondiente.
3. Tendremos que asegurarnos que todos los componentes están conectados a una toma de corriente y reciben energía, por lo general si están recibiendo corriente tendrán algún led encendido.

4. También deberemos comprobar que el switch cumple con su función mirando si todos los puertos donde se conectan los cables de red están recibiendo y enviando datos.
5. Se deberá conectar la placa de expansión a la placa maestro, en este caso es muy sencillo porque solo existe una forma de conectarla para que todos los pines queden conectados.
6. Por último deberemos conectar los pines de la pantalla LCD a una de las placas maestro para poder ver de manera adecuada la información de los comandos recibidos.

Para la conexión de los pines de los motores y la pantalla se puede consultar el apéndice anterior, concretamente la sección D.5 Tras tener configurado el entorno, el siguiente paso será cargar el programa correspondiente a cada microcontrolador. Recordemos que disponemos de tres placas y tres *softwares* con algunas diferencias entre ellos. En el apéndice anterior D.4 ya vimos como podíamos realizar este procedimiento. Una vez se tienen los códigos cargados en las placas deberemos, esperar a que los tres microcontroladores adquieran su respectiva dirección IP y podremos comenzar a enviar comandos desde la placa maestro como veremos en el siguiente apartado.

### E.3. Manual del usuario

En el apéndice 3 B pudimos ver en los requisitos funcionales todos los usos de la placa y sus correspondientes pasos. En este apartado veremos de una forma más gráfica las utilidades de la misma.

En primer lugar vamos a visualizar y nombrar los botones de la placa maestro.

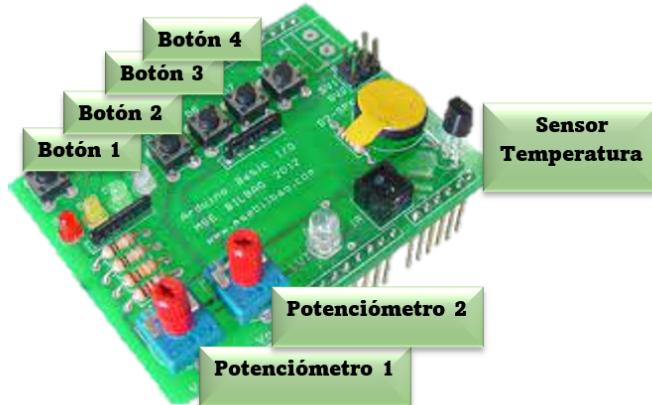


Figura E.1: Botones del shield de la placa maestro.



Figura E.2: Botones utilizados de la placa K64F.

Para configurar la velocidad del motor A deberemos mover el potenciómetro 1 de la placa según la velocidad (sentido del giro) que le queramos asignar. Si lo giramos hacia la derecha completamente el motor recibirá el valor 255 y girará hacia la derecha, si por el contrario lo giramos hacia la izquierda recibirá el valor de 0 y girará en el sentido opuesto. En caso de que dejemos el potenciómetro en un rango medio de su giro, el motor recibirá el dígito 128 y dejará el motor parado.

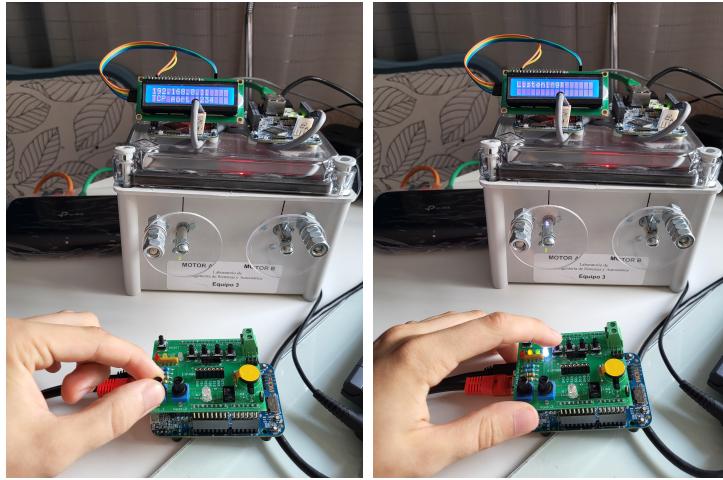


Figura E.3: Establecer velocidad motor A.

Lo mismo pasa con el potenciómetro 2 y el botón 2. En ambos casos, al pulsar el botón 1 o dos, recibiremos un *feedback* en la placa esclavo mediante un mensaje en la pantalla que nos mostrará si se recibió el comando o no.

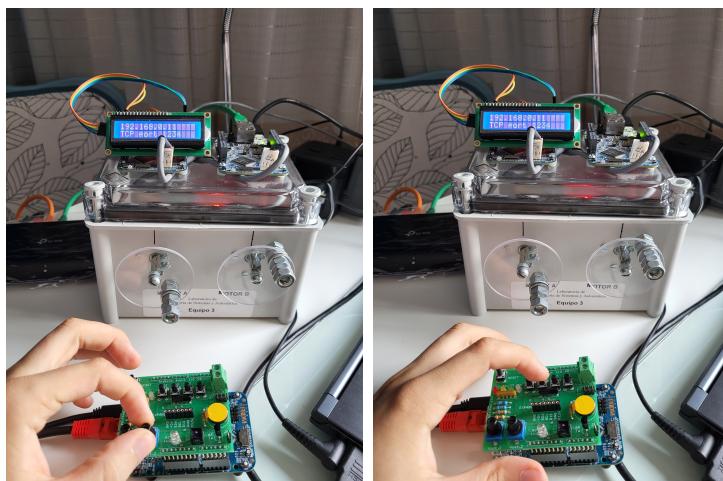


Figura E.4: Establecer velocidad motor B.

En caso de que queramos parar ambos motores utilizaremos el botón 3 a modo de parada de emergencia.

El botón 4 envía a la placa maestro la temperatura ambiente. Esta temperatura se muestra en grados centígrados por la pantalla del esclavo. Mensaje que sale por pantalla: “Temp actual: 23”.

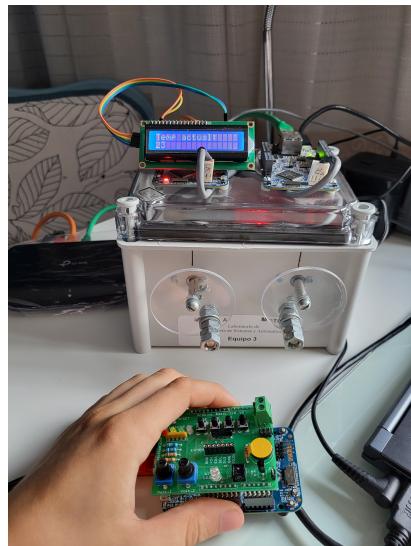
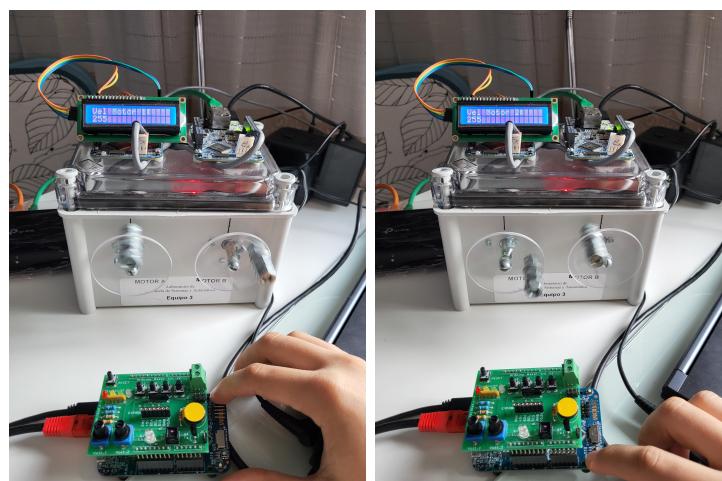


Figura E.5: Se muestra la temperatura por pantalla.

Por último, los botones 5 y 6 muestran la velocidad de los motores A y B respectivamente, esta información se mostrará por la pantalla de la placa esclavo. Mensaje que muestra la pantalla: “Vel motor 1: 255”.



(a) Obtengo velocidad 1

(b) Obtengo velocidad 1

Figura E.6: Obtener velocidades de los motores.



---

# Bibliografía

---

- [1] URL: <https://nxp.flexnetoperations.com/control/frse/product?entitlementId=627459067&lineNum=1&authContactId=171602057&authPartyId=181871057>. Link de descarga de MCUXpresso.
- [2] URL: <https://mcuxpresso.nxp.com/en/builder?hw=FRDM-K64F>. Link de descarga del sdk para la placa FRDM K64F.
- [3] URL: <https://packetsender.com/download>. Link de descarga de Packet Sender.
- [4] Boletín oficial del Estado. Ley 27/2014, de 27 de noviembre, del Impuesto sobre Sociedades. URL: <https://www.boe.es/buscar/doc.php?id=BOE-A-2014-12328>.
- [5] Creative Commons. CC BY-NC-SA 4.0. URL: <https://creativecommons.org/licenses/by-nc-sa/4.0/deed.es>.
- [6] Consejería de Hacienda, Junta de Castilla y León. DECRETO LEGISLATIVO 1/2013, de 12 de septiembre, por el que se aprueba el texto refundido de las disposiciones legales de la Comunidad de Castilla y León en materia de tributos propios y cedidos. URL: <http://bocyl.jcyl.es/boletines/2013/09/18/pdf/B0CYL-D-18092013-1.pdf>.
- [7] Universidad de Navarra. Establecer conexión TCP. URL: [https://www.tlm.unavarra.es/~daniel/docencia/ro\\_is/ro\\_is05\\_06/slides/Clase17-TCP.pdf](https://www.tlm.unavarra.es/~daniel/docencia/ro_is/ro_is05_06/slides/Clase17-TCP.pdf).

- [8] Jefatura del Estado. Ley 27/2014, de 27 de noviembre, del Impuesto sobre Sociedades. URL: <https://www.boe.es/buscar/pdf/2014/BOE-A-2014-12328-consolidado.pdf>.
- [9] MK Electrónica. BASIC I/O. URL: <https://mkelectronica.com/producto/basic-i-o/>.
- [10] farnell. Cable de Ethernet. URL: <https://es.farnell.com/videk/2996-1y/cable-de-conexion-cat6-amarillo/dp/1525699>.
- [11] The Apache Software Foundation. Apache License, Version 2.0. URL: <https://www.apache.org/licenses/LICENSE-2.0>.
- [12] lwIP wiki fandom. Netconn API. URL: [https://lwip.fandom.com/wiki/Netconn\\_API](https://lwip.fandom.com/wiki/Netconn_API).
- [13] Microsoft. Office 365. URL: <https://products.office.com/es-ES/business/office>.
- [14] Microsoft. Windows 10 Pro. URL: <https://www.microsoft.com/es-es/p/windows-10-pro/df77x4d43rkt/48DN>.
- [15] nongnu. Modulo Netconn API. URL: [https://www.nongnu.org/lwip/2\\_0\\_x/group\\_\\_netconn\\_\\_common.html](https://www.nongnu.org/lwip/2_0_x/group__netconn__common.html).
- [16] Institute of Electrical and Electronics Engineers. IEEE 29148-2011. URL: <https://standards.ieee.org/standard/29148-2011.html>.
- [17] Institute of Electrical and Electronics Engineers. IEEE 29148-2018. URL: <https://standards.ieee.org/standard/29148-2018.html>.
- [18] Institute of Electrical and Electronics Engineers. IEEE 830-1998. URL: <https://standards.ieee.org/standard/830-1998.html>.
- [19] PcComponentes. Switch TP-Link 5 puertos. URL: [https://www.pcccomponentes.com/tp-link-tl-sf1005d-switch-5-puertos-10-100-mbps-plugplay?gclid=EAIAIQobChMI6q29zoTL-AIVTY9oCR13Tgz-EAQYAiABEgJv3\\_D\\_BWE](https://www.pcccomponentes.com/tp-link-tl-sf1005d-switch-5-puertos-10-100-mbps-plugplay?gclid=EAIAIQobChMI6q29zoTL-AIVTY9oCR13Tgz-EAQYAiABEgJv3_D_BWE).
- [20] Sandorobotics. Controladora Motor MD25. URL: <https://sandorobotics.com/producto/md25/>.
- [21] Banco Santander. Calculadora sueldo neto. URL: [https://santandersmartbank.es/calculadora-sueldo-neto/#resultados\\_calculadora\\_nomina](https://santandersmartbank.es/calculadora-sueldo-neto/#resultados_calculadora_nomina).

- [22] Seguridad Social. Bases y tipos de cotización 2022. URL: <http://www.seg-social.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/36537>.
- [23] Farnell Components SL. Cable de Ethernet. URL: <https://es.farnell.com/videk/2996-1y/cable-de-conexion-cat6-amarillo/dp/1525699>.
- [24] Farnell Components SL. FRDM-K64F - Placa de Desarrollo. URL: <https://es.farnell.com/nxp/frdm-k64f/placa-desarrollo-ethernet-usb/dp/2406741>.
- [25] Farnell Components SL. Pantalla LCD Alfanumérica, 16 x 2. URL: <https://es.farnell.com/midas/mc21605c6w-bnmlwi-v2/pantalla-alfanum-rica-16x2-blanca/dp/2748649>.
- [26] Farnell Components SL. WIRE BUNDLE, BREADBOARD. URL: <https://es.farnell.com/adafruit-industries/153/wire-bundle-breadboard/dp/2409349>.
- [27] The Apache Software Foundation. TERMS AND CONDITIONS, Apache License, Version 2.0. URL: <https://www.apache.org/licenses/LICENSE-2.0.txt>.



Este obra está bajo una [licencia de Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional](#).