

E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Diseño electrónico y verificación de un módulo OEM basado en el microcontrolador Renesas Synergy S-124



Grado en Ingeniería
en Tecnologías Industriales

Trabajo Fin de Grado

Jesús Ignacio de Miguel Valencia

Santiago J. Led Ramos

Pamplona, 27 de Junio de 2017



Agradecimientos

*A la empresa Embeblue por su
apoyo a lo largo de todo el proyecto*

ABSTRACT

The current thesis pretends to detail, step by step, the design, manufacturing and testing process of a Single Board Computer (SBC) based on OEM¹ hardware. The schematic of the electronic module which is going to be implemented focuses on the R7FS124773A01CFM MCU², of the Renesas S-124 series. Throughout the process, the company *Embeblue* supports it in different ways, such as manufacturing the module.

Firstly, the documentation and the inner functioning (power supply, ports, oscillators...) of the MCU are studied, since it will affect the hardware which will surround it. Besides, its basic programming concepts will be revised too. To perform these tasks, the Renesas DK-S124 board is used. This board is designed from the same MCU so it is a useful support for studying its schematics and for testing firmware.

The next step is the design of the electronic circuit schematics and their translation to a final Printed Circuit Board (PCB) using the *DesignSpark PCB* software. This hardware design will start from the MCU and the basic electronics it requires for its operation. Nevertheless, it is not just limited to this block, since for its integration in a functional card it needs additional electronics which assures an appropriate power supply. Besides, a block for USB data entry is added.

As for the electronics' PCB design, a special methodology is followed, designing separately (on different projects) the functional PCB blocks which have been mentioned. Later, they must be "assembled", step which is performed by *Embeblue*. One of the purposes of the thesis is to highlight the advantages of this PCB design philosophy.

The final target, once the board is manufactured, is checking its good operation by programming basic firmware.

KEYWORDS

- **PCB design**
- **Microcontroller Unit (MCU)**
- **Single Board Computer (SBC)**

¹ *Original Equipment Manufacturer*

² *Microcontroller Unit*

RESUMEN

El presente proyecto pretende detallar paso por paso el diseño, fabricación y verificación de una placa computadora basada en hardware OEM³. El esquema de la tarjeta electrónica a implementar se centra en el microcontrolador R7FS124773A01CFM, de la serie S-124 de Renesas. A lo largo del proceso se cuenta con el apoyo de la empresa *Embeblue*, que, entre otras cosas, se encarga de la fabricación del módulo.

Primeramente, se estudia la documentación y funcionamiento interno del microcontrolador (alimentación, puertos, osciladores...) que afectarán al hardware que le rodeará, así como las nociones básicas para su programación mediante el entorno *e2 studio*. Para esta tarea se dispone de la ayuda de la placa de desarrollo DK-S124, diseñada a partir del mismo microcontrolador, por lo que sirve de apoyo a la hora de basarse en algunas partes de sus esquemáticos y para realizar pruebas de *firmware*.

El siguiente paso es la preparación de los esquemáticos del circuito electrónico y su posterior traducción a un circuito impreso final mediante el software *DesignSpark PCB*. El hardware a diseñar partirá del microcontrolador y la electrónica básica que este requiere para su funcionamiento. Sin embargo, no se limita solamente a este bloque, ya que para su integración en una tarjeta funcional necesita de otro que asegure su correcta alimentación. Además, se decide añadir un bloque para la entrada de datos USB.

En el diseño PCB de esta electrónica se sigue una metodología especial, diseñando por separado los circuitos impresos de los bloques funcionales mencionados para su posterior “ensamblado” en el módulo final. Este ensamblado es llevado a cabo por *Embeblue*. Uno de los objetivos del proyecto es destacar las ventajas de esta filosofía de diseño PCB⁴.

El objetivo final, una vez fabricada la placa, es la verificación del correcto funcionamiento de esta mediante *firmware* básico.

PALABRAS CLAVE

- **Diseño PCB**
- **Microcontrolador**
- **Placa computadora**

³ *Original Equipment Manufacturer*

⁴ *Printed Circuit Board*

0. Índice

ABSTRACT	3
KEYWORDS	3
RESUMEN	4
PALABRAS CLAVE	4
0. Índice	6
Lista de Figuras	9
Lista de Tablas	13
Vídeos	13
1. Introducción	14
1.1. Objetivo del proyecto	14
1.2. Alcance y fases del proyecto	16
2. Herramientas utilizadas	17
2.1. Herramientas hardware	17
2.1.1. Microcontrolador Renesas R7FS124773A01CFM	17
2.1.2. Placa de desarrollo Renesas DK-S124	18
2.2. Herramientas software	21
2.2.1. Entorno de programación <i>e² studio</i>	21
2.2.2. Synergy Software Package	23
2.2.3. <i>DesignSpark PCB</i>	25
3. Reglas y consideraciones en el diseño PCB	27
3.1. Fundamentos del diseño PCB	28
3.1.1. Posicionamiento de componentes	28
3.1.2. Enrutado de pistas	29
3.1.3. <i>Pads</i> , máscara de soldadura y pasta de soldadura	30
3.1.4. <i>Vias</i>	32
3.1.5. Planos de masa	33

3.2.	Reglas de diseño impuestas	34
3.2.1.	Separaciones	34
3.2.2.	Pistas	35
3.2.3.	Vias	36
3.2.4.	Pasta de soldadura y máscara de soldadura	36
3.3.	Integridad de la señal	37
3.3.1.	Condensadores de desacoplo	38
3.3.2.	Condensadores de reserva (bulk).....	39
3.3.3.	Resistencias pull-up y pull-down.....	39
3.4.	Diseño con <i>DesignSparkPCB</i>	41
4.	Diseño PCB en bloques	44
5.	Diseño del módulo OEM	47
5.1.	Bloque del microcontrolador	49
5.1.1.	Diseño del esquemático	49
5.1.2.	Diseño PCB	55
5.2.	Bloque de alimentación	57
5.2.1.	Diseño del esquemático	58
5.2.2.	Diseño PCB	66
5.3.	Bloque USB	69
5.3.1.	Diseño del esquemático	70
5.3.2.	Diseño PCB	77
5.4.	Bloque externo unificador.....	79
5.4.1.	Diseño del esquemático	79
5.4.2.	Diseño PCB	85
5.5.	Lista de componentes	89
6.	Fabricación del módulo OEM	91
6.1.	Unión de los bloques PCB.....	91
6.2.	Envío de los archivos de diseño al fabricante.....	93
6.3.	Soldadura de componentes	95
6.4.	Aspecto final del módulo.....	98
7.	Verificación del módulo	100

7.1.	Revisión del enrutado y del sistema de alimentación	100
7.2.	Verificación mediante <i>firmware</i>	102
7.2.1.	Programa de prueba 1: parpadeo de un LED	104
7.2.2.	Programa de prueba 2: parpadeo de un LED + control de la frecuencia mediante una entrada digital	106
7.2.3.	Programa de prueba 3: parpadeo de un LED + control de la frecuencia mediante una entrada digital + uso de interrupciones externas	108
7.2.4.	Programa de prueba 4: “Hola mundo” en <i>LCD_Keypad_Shield</i>	110
7.2.5.	Programa de prueba 5: monitorización de variables analógicas en <i>LCD Keypad Shield</i>	114
8.	Conclusiones	118
9.	Bibliografía	119
ANEXOS	121
	<i>ANEXO I. ASIGNACIÓN DE PINES EN MICROCONTROLADOR R7FS124773A01CFM</i>	121

Lista de Figuras

Figura 2.1. Ordenación de pines R7FS124773A01CFM [1]	18
Figura 2.2. Vista TOP y partes de la placa de desarrollo DK-S124 [2]	19
Figura 2.3. Esquemático MCU y hardware requerido por este (condensadores de desacoplo, resistencias pull-up, cistales...) en placa DK-S124 [10]	20
Figura 2.4. Configuración de relojes en e2studio.....	21
Figura 2.5. Configuración de pines en e2studio.....	22
Figura 2.6. Configuración de librerías internas en e2studio y generación del código previo de configuración	22
Figura 2.7. Project Explorer: archivos de código de un proyecto en e2studio	23
Figura 2.8. Estructura de la plataforma Synergy: https://www.renesas.com/en-us/products/synergy/features.html	24
Figura 2.9. Esquema del Synergy Software Package (SSP): https://www.renesas.com/en-eu/doc/products/renesas-synergy/doc/R01PF0137ED0200.pdf 25	
Figura 2.10. Librerías de componentes para DesignSpark PCB.....	26
Figura 2.11. Esquemático y traducción a PCB en un proyecto de DesignSpark PCB.....	26
Figura 3.1. Partes de una PCB	27
Figura 3.2. Ratios S/C orientativos: área de PCB (S) – área de componentes (C)	28
Figura 3.3. Ortogonalidad de pistas en diferentes capas de las PCB	29
Figura 3.4. Diferentes anchuras de pistas.....	30
Figura 3.5. Componentes de agujero pasante o through hole (izquierda) y componentes de montaje superficial o SMD (derecha): https://tecnologiademontajesuperficial.es.tl/	31
Figura 3.6. Comparación de un condensador SMD (arriba) y una resistencia SMD (abajo) con sus respectivas huellas	31
Figura 3.7. Márgenes de pasta de soldadura (P) y máscara de soldadura (M) en un pad.....	32
Figura 3.8. Vías	32
Figura 3.9. “Vertido” del plano de masa tras el enrutado	33
Figura 3.10. Disposición de planos de masa entre diferentes partes del circuito	34
Figura 3.11. Márgenes de aceptación de niveles digitales. Integridad de señal [6]	37

Figura 3.12. Condensadores de desacoplo para el microcontrolador R7FS124773A01CFM en la placa DK-S124	38
Figura 3.13. Esquema de entrada de potencia en un circuito integrado.....	39
Figura 3.14. Conexión de una resistencia pull-up externa a un pin.....	40
Figura 3.15. Resistencia pull-up interna en un pin de un PIC.....	40
Figura 3.16. DesignSpark PCB: los componentes constan de su símbolo esquemático (arriba) y su huella PCB (abajo)	41
Figura 3.17. DesignSpark PCB: ejemplo de un esquemático.....	41
Figura 3.18. DesignSpark PCB: configuración inicial de las características de la PCB	42
Figura 3.19. DesignSpark PCB: disposición de los componentes fuera de la tarjeta con las conexiones impuestas en el esquemático (izquierda) y enrutado de pistas (derecha).....	42
Figura 4.1. Unión de bloques PCB	44
Figura 4.2. Proceso de creación de los bloques PCB: creación del marco (arriba), imposición de conexiones (medio) y traducción a PCB (abajo).....	46
Figura 5.1. Placa Arduino UNO.....	48
Figura 5.2. Bloque del microcontrolador: esquemático	50
Figura 5.3. Rebotes de señal generados en la pulsación manual	52
Figura 5.4. Oscilador Pierce: esquema de conexión del cristal de cuarzo (abajo).....	52
Figura 5.5. Conector de programación del módulo	53
Figura 5.6. LED de testeo del módulo	53
Figura 5.7. Bloque del microcontrolador: esquemático del marco	54
Figura 5.8. Huella del marco para el bloque del microcontrolador	55
Figura 5.9. Bloque del microcontrolador, vista PCB: cara TOP	56
Figura 5.10. Bloque del microcontrolador, vista PCB: cara BOTTOM	56
Figura 5.11. Bloque de alimentación: esquemático del marco.....	58
Figura 5.12. Bloque de alimentación: esquemático.....	59
Figura 5.13. Terminales del conector jack hembra para la entrada de corriente.....	60
Figura 5.14. Modelo simplificado de un regulador lineal ajustable.....	61
Figura 5.15. Reguladores de tensión: esquema de conexionado y relación de las resistencias para imponer una tensión de salida.....	62
Figura 5.16. Conexionado de AO y PMOS para la protección ante la conexión simultánea de dos entradas de potencia	64
Figura 5.17. Selector de tensión de alimentación (5V o 3.3V) en el módulo.....	65
Figura 5.18. Layout de los condensadores y resistencias externos del selector.....	66

Figura 5.19. Lengüeta disipadora de calor en un regulador lineal.....	67
Figura 5.20. Bloque de alimentación, vista PCB: cara TOP	68
Figura 5.21. Bloque de alimentación, vista PCB: cara BOTTOM	68
Figura 5.22. Cable USB cortado.....	69
Figura 5.23. Pulsador de RESET	69
Figura 5.24. Bloque USB: esquemático	71
Figura 5.25. Conector USB tipo B	72
Figura 5.26. Asignación de pines en el conector USB	72
Figura 5.27. Compensación del efecto del ruido en líneas de transmisión diferencial	73
Figura 5.28. Transmisión Non-Return to Zero, Invert on Ones (NRZI).....	73
Figura 5.29. Función del valor de la resistencia en función de la Temperatura para un termistor NTC y un termistor PTC.....	74
Figura 5.30. Asignación de pines para el pulsador RESET	75
Figura 5.31. Bloque USB: esquemático del marco	76
Figura 5.32. Trazado de las líneas USBDM y USBDP	77
Figura 5.33. Bloque USB, vista PCB: cara TOP	78
Figura 5.34. Bloque USB, vista PCB: cara BOTTOM.....	78
Figura 5.35. Asignación de pines a los conectores del Arduino UNO	79
Figura 5.36. Bloque externo unificador: esquemático de conectores externos.....	80
Figura 5.37. Asignación de pines en Atmega 168p, incluido en el Arduino UNO	81
Figura 5.38. Pines del microcontrolador a las hileras de conectores internas para la conexión de shields (los recuadrados).....	83
Figura 5.39. Bloque externo unificador: esquemático resistencias pull-up.....	84
Figura 5.40. Huella PCB de los conectores. Las dimensiones coinciden con las de los conectores en las placas Arduino UNO, LEONARDO Y MEGA.....	85
Figura 5.41. Ortogonalidad de pistas en las caras TOP y BOTTOM.....	86
Figura 5.42. Bloque externo unificador, vista PCB: cara TOP.....	87
Figura 5.43. Bloque externo unificador, vista PCB: cara BOTTOM	88
Figura 6.1. Módulo PCB fusionado: cara TOP	92
Figura 6.2. Módulo PCB fusionado: cara BOTTOM	92
Figura 6.3. Menú PCB Visualizaer.....	94
Figura 6.4. Vista previa del módulo PCB en PCB Visualizer: cara TOP	94
Figura 6.5. Vista previa del módulo PCB en PCB Visualizer: cara BOTTOM	95

Figura 6.6. Fijación del stencil sobre la placa PCB: https://www.youtube.com/watch?v=bSnb3D72pxY	96
Figura 6.7. Extensión de la pasta de soldadura sobre el Stencil con una espátula: https://www.youtube.com/watch?v=bSnb3D72pxY	96
Figura 6.8. Colocación de los componentes sobre sus respectivas huellas: https://www.youtube.com/watch?v=km1knuJin-o	97
Figura 6.9. Introducción de la PCB en el horno de soldadura SMD	97
Figura 7.1. Verificación de las pistas y planos mediante un multímetro	100
Figura 7.2. Alimentación de la tarjeta a través de puerto USB (izquierda), jack (centro) y de ambas entradas (derecha)	101
Figura 7.3. Tensión en la net '5V' en los tres modos de alimentación	101
Figura 7.4. Asignación de pines en programador J-Link EDU.....	103
Figura 7.5. Adaptador al conector JTAG/SWD y programador J-Link EDU	103
Figura 7.6. Diagrama de funcionamiento: programa 1	104
Figura 7.7. Parpadeo del LED de testeo	105
Figura 7.8. Configuración de los relojes internos y de las fuentes de los osciladores	105
Figura 7.9. Cristales de cuarzo externos	106
Figura 7.10. Diagrama de funcionamiento: programa 2	106
Figura 7.11. Pulsador para el control de la frecuencia de parpadeo del LED	107
Figura 7.12. Diagrama de funcionamiento: programa 3	108
Figura 7.13. LCD Keypad Shield	110
Figura 7.14. Diagrama de funcionamiento: introducción de datos en LCD_Keypad_Shield	111
Figura 7.15. Diagrama de funcionamiento: funciones para el manejo de la pantalla LCD	112
Figura 7.16. Diagrama de funcionamiento: funciones lcd_print() para escritura de secuencias de caracteres por pantalla	113
Figura 7.17. Diagrama de funcionamiento: lectura del nivel analógico mediante interrupción 114	
Figura 7.18. Diagrama de funcionamiento: menú interactivo en LCD KeypadShield con la lectura de los pulsadores en la variable btn	116
Figura 7.19. Visualización de las tensiones de los potenciómetros en la pantalla LCD	117

Lista de Tablas

Tabla 3.1. Distancias mínimas entre elementos de la PCB	35
Tabla 3.2. Parámetros de las pistas.....	35
Tabla 3.3. Dimensiones de las vías.....	36
Tabla 3.4. Márgenes de la Máscara de Soldadura y la Pasta de Soldadura	36
Tabla 5.1. Comparación de pines del Arduino UNO con los del R7FS124773A01CFM y asignación de pines de este último a las hileras de conectores internas	83
Tabla 5.2. Lista de componentes y precios	90

Vídeos

Verificación del módulo OEM:

<https://www.youtube.com/watch?v=6wDBlapPtSQ>

1. Introducción

1.1. Objetivo del proyecto

La idea de partida de este proyecto surgió del contacto con la empresa navarra Embeblue, que fue la que planteó los objetivos expuestos en este apartado. Embeblue se dedica al desarrollo de dispositivos electrónicos para el Internet de las Cosas, así como soluciones adaptadas a clientes para la producción de prototipos funcionales y preseries.

El objetivo principal del proyecto es profundizar en el proceso de desarrollo de un módulo electrónico funcional partiendo de un microcontrolador específico. Para ello se diseñará y testeará una placa con unas características que se irán describiendo, partiendo del microcontrolador R7FS124773A01CFM, de la serie S-124 de Renesas. Este microcontrolador pertenece a la plataforma Synergy, la cual está siendo actualmente desarrollada e impulsada por Renesas Electronics.

En cuanto a la funcionalidad del módulo a diseñar, se pretende conseguir una placa con el hardware necesario para posibilitar el trabajo con el citado microcontrolador, incluyendo, además, una entrada de alimentación con el acondicionamiento adecuado, un conector para la programación de *firmware* y conectores externos para el acceso a los pines del microcontrolador por parte del usuario. Adicionalmente, se incluye un conector USB, del cual, normalmente, disponen todas las tarjetas de características similares en el mercado.

Por tanto, se desea conseguir un diseño bastante cercano a una placa computadora u ordenador de placa reducida (*SBC*⁵). Este tipo de placas engloban todas las características de una computadora funcional en la propia tarjeta a partir de un sistema microprocesador o un sistema microcontrolador, siendo este último el utilizado en el presente proyecto. Estos módulos están alcanzando una gran popularidad en la actualidad debido a las grandes ventajas que ofrecen, sobre todo en cuanto a su tamaño reducido, su bajo precio y su versatilidad. Dichas características los hacen ideales en aplicaciones de prototipado o en determinados entornos industriales y sistemas embebidos como controladores e interfaces. Además, están impulsando el concepto “Hágalo usted mismo”, acercando la tecnología a usuarios con niveles básicos de electrónica y programación. Probablemente, las placas más conocidas y usadas, sobre todo en aplicaciones de complejidad menor, son las desarrolladas por Arduino, cuyos diseños sirven de inspiración en este proyecto.

⁵ *Single Board Computer*

Como reza el título del proyecto, el dispositivo a fabricar es un módulo OEM. Las siglas OEM provienen del inglés *Original Equipment Manufacturer*, traducido al castellano Fabricante de Equipamiento Original. Hacen referencia a que los componentes proceden de diferentes fabricantes, siendo la empresa que implementa el producto a partir de dichos componentes la encargada de distribuir el producto final bajo su nombre.

Una de las fases principales del proyecto es el diseño PCB o del circuito impreso que definirá el módulo. El objetivo es conseguir unos archivos de diseño finales (*gerbers*) con los que el fabricante tenga la información necesaria para implementar la tarjeta. Es de resaltar la metodología seguida en este diseño PCB, la cual es utilizada en la empresa Embeblue. Se trata de diseñar el circuito impreso en bloques funcionales separados para la integración posterior de sus archivos *gerber* en el módulo final, en lugar de crear el circuito global en un solo proyecto. La principal ventaja que ofrece este método es la posibilidad de reutilizar dichos bloques en diseños posteriores. Se profundizará en esta metodología en el apartado 4.

Se ha de recalcar que la tarea de fusión de los bloques PCB no forma parte de los objetivos de este proyecto, ya que esta se lleva a cabo en Embeblue a partir de software propio hecho a medida, de tal modo que no se incidirá en este apartado.

Teniendo en mente las características mencionadas del módulo, se trabajará en su diseño en diferentes fases, durante las cuales se profundizará en los siguientes aspectos:

- Aprendizaje de las particularidades internas del microcontrolador Renesas R7FS124773A01CFM, como pueden ser sus puertos, sus osciladores o sus características eléctricas.
- Diseño de un circuito electrónico para el desarrollo de una placa computadora básica.
- Selección de los componentes electrónicos adecuados para cada función dentro de los esquemáticos de la placa.
- Diseño PCB mediante herramientas ECAD⁶.
- Fases seguidas en la fabricación de una placa de circuito impreso.
- Programación de firmware básico en el entorno de programación *e2 studio*, proporcionado por Renesas.

⁶ *Electronic Computer-Aided Design*

1.2. Alcance y fases del proyecto

Teniendo en cuenta los objetivos del proyecto citados en el anterior apartado, se seguirá un proceso dividido en las siguientes fases:

- Revisión de la bibliografía y documentación relacionada con el microcontrolador y la placa de desarrollo DK-S124 que se dispone. Se trabajará con dicha placa para el aprendizaje de la programación y el funcionamiento del microcontrolador (periféricos, modos de funcionamiento...).
- Diseño electrónico de todos los esquemáticos que definirán la placa separados en bloques funcionales.
- Selección de la electrónica básica necesaria para el funcionamiento del módulo, y búsqueda de la disponibilidad comercial de dicha electrónica.
- Diseño PCB del módulo OEM mediante herramientas ECAD (*DesignSpark PCB*).
- Una vez fabricado, test del módulo OEM mediante el desarrollo de *firmware* básico de verificación en el entorno de programación *e2 studio*.

2. Herramientas utilizadas

2.1. Herramientas hardware

2.1.1. **Microcontrolador Renesas R7FS124773A01CFM**

El presente microcontrolador pertenece a la serie S-124 de Renesas. A continuación, se enumeran las características generales de esta serie:

- Bajo consumo (*Ultra-Low-Power MCU*).
- Frecuencia de reloj máxima: 32 MHz.
- Tensión de trabajo: de 1.6 V a 5.5V.
- Hasta 128 Kb de memoria flash de programa.
- 16 Kb de memoria SRAM.
- Conversor Analógico/Digital de 14 bits de resolución.
- Conversor Digital/Analógico de 12 bits de resolución.

El R7FS124773A01CFM es de montaje superficial, con empaquetado de tipo LQFP. Cuenta con 64 pines, de los cuales 51 son entradas/salidas de propósito general y están distribuidas en 6 puertos. En la Figura 2.1. se puede ver cómo se distribuyen los pines en el microcontrolador. Además, en el *Anexo I* se incluye la tabla de asignación de pines, donde se enumeran las funcionalidades de cada uno de estos.

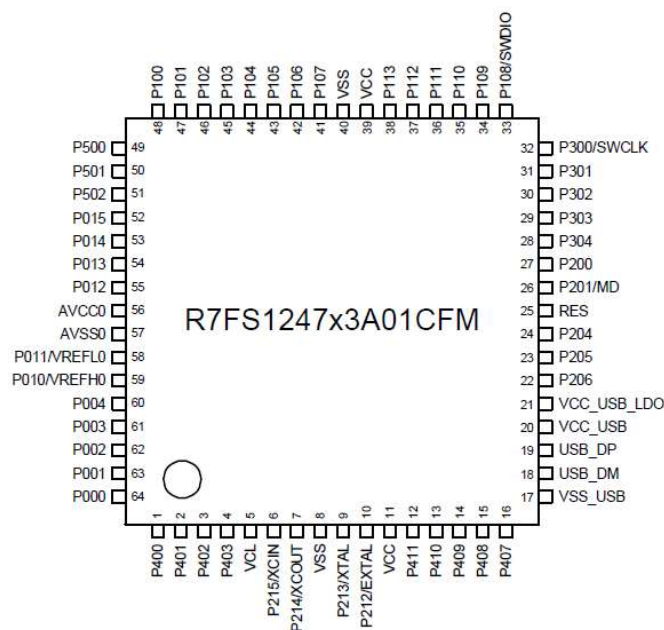


Figura 2.1. Ordenación de pines R7FS124773A01CFM [1]

Para información adicional sobre el microcontrolador, por ejemplo, acerca de sus osciladores, *timers* o interfaces de comunicación, se recomienda consultar su *datasheet* [1].

2.1.2. Placa de desarrollo Renesas DK-S124

La DK-S124 es una placa de desarrollo de Renesas que incorpora el microcontrolador R7FS124773A01CFM, en el que se basa el diseño del presente proyecto. Es por esto que resulta un importante apoyo, no sólo a la hora diseñar el hardware elemental para el funcionamiento del microcontrolador, pudiendo basarse en los esquemáticos de la placa, sino que además permite testear el software haciendo uso del entorno de programación *e²studio*.

Dispone de gran cantidad de periféricos, lo que permite poner a prueba muchas de las funcionalidades de las que dispone el microcontrolador. Entre otros periféricos, la placa cuenta con un panel táctil, un acelerómetro, leds de prueba, un potenciómetro, una toma para auriculares y diversos interfaces de comunicación (USB, I2C, SPI, CAN...).

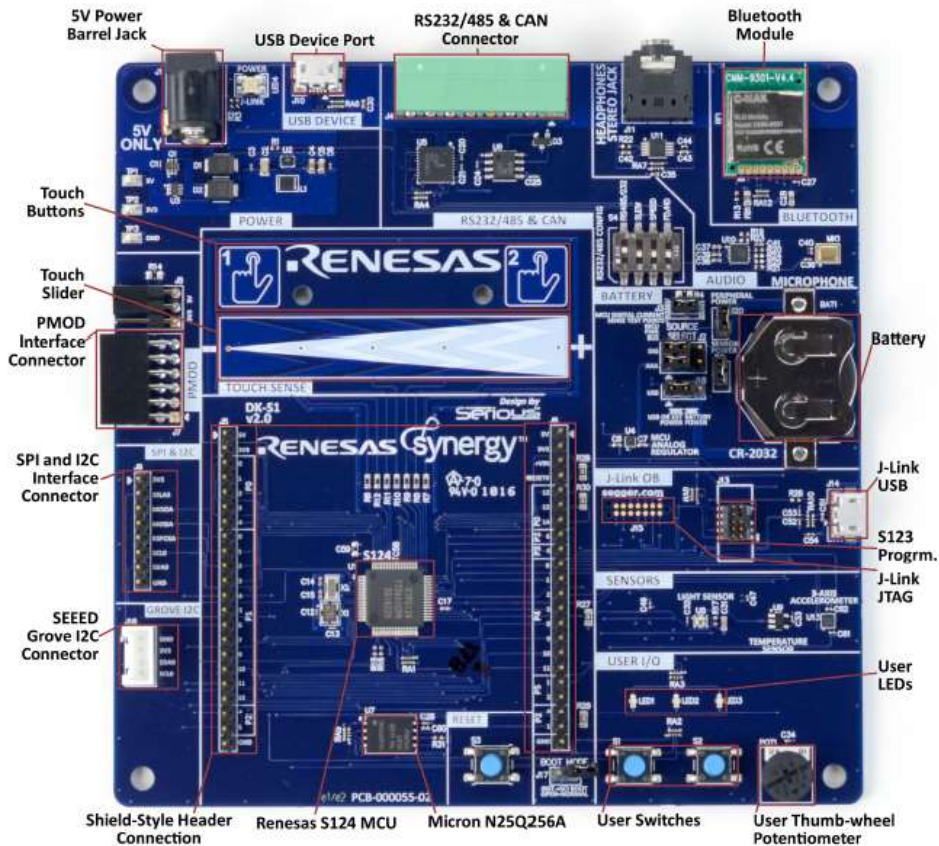


Figura 2.2. Vista TOP y partes de la placa de desarrollo DK-S124 [2]

Como se ha comentado, esta placa supone un apoyo importante en cuanto al diseño del hardware del módulo a implementar en este proyecto, ya que, como se verá, las conexiones de componentes que rodean al microcontrolador (condensadores, resistencias, osciladores...), se basan, en parte, en sus esquemáticos. En la Figura 2.3. se aprecia el circuito que rodea al R7FS124773A01CFM en la placa de desarrollo.

Si se desea mayor información sobre esta tarjeta, consúltese su manual de usuario [2].

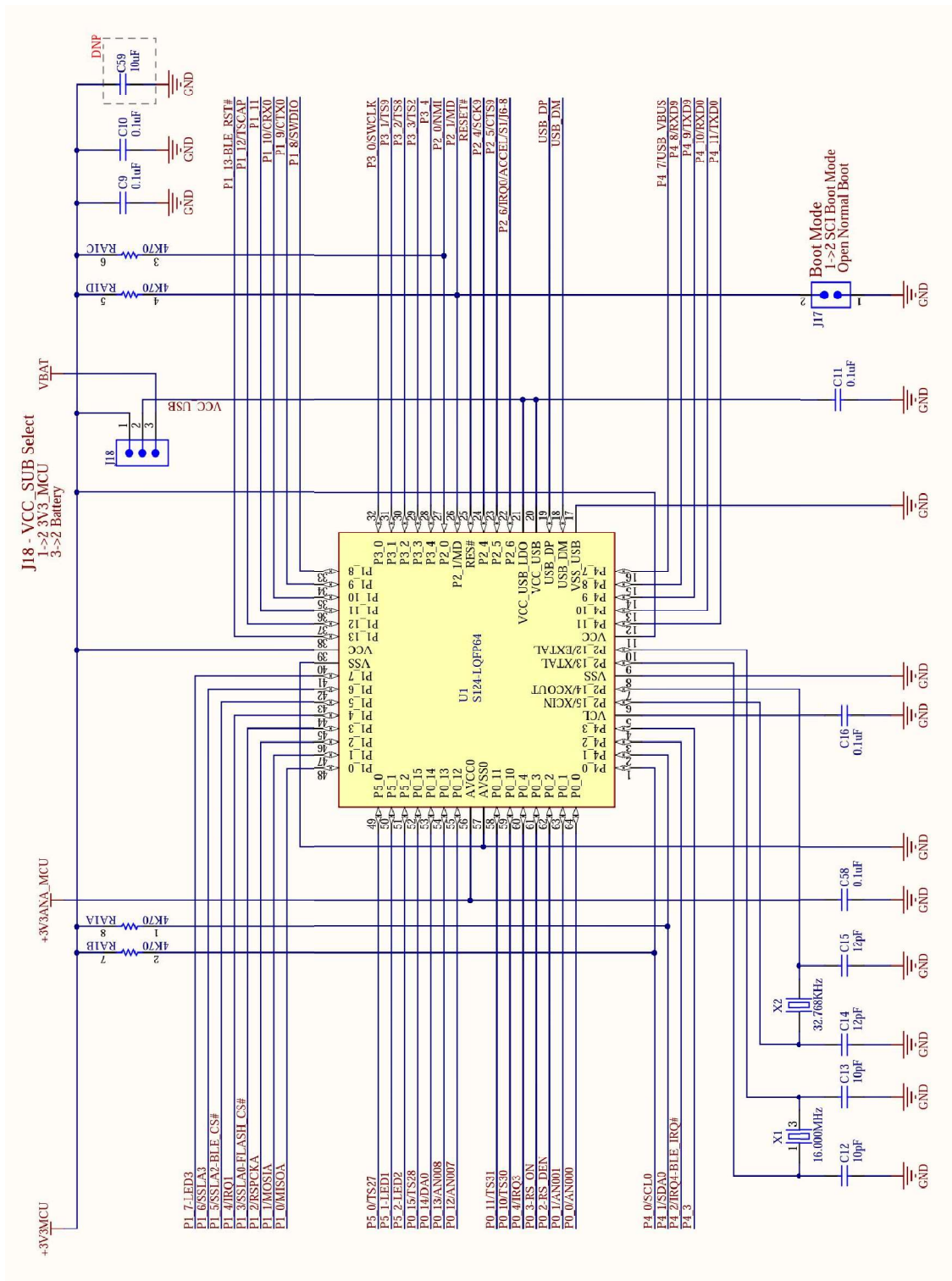


Figura 2.3. Esquemático MCU y hardware requerido por este (condensadores de desacoplo, resistencias pull-up, cistales...) en placa DK-S124 [10]

2.2. Herramientas software

2.2.1. Entorno de programación *e² studio*

El programa *e2 studio* es el entorno de programación que proporciona Renesas y que se utilizará a la hora de cargar el *firmware* necesario en el microcontrolador. Este entorno incluye el paquete *Synergy Software* (se detalla en el siguiente apartado), el cual está pensado para facilitar la labor de programación, proporcionando un amplio soporte que reduce labores de configuración previas al diseño de las aplicaciones (definir frecuencias de trabajo, pines cómo entradas y salidas, librerías internas...).

De hecho, este entorno permite modificar gráficamente las configuraciones iniciales del programa, como se observa en las imágenes incluidas a continuación. Así, una vez creado el proyecto, se establece la configuración de los relojes (pestaña *Clocks*), pines (pestaña *Pins*) y librerías internas que va a necesitar el programa (en la pestaña *Threads*), y lo único que quedará por hacer es pulsar en *Generate Project Content*, lo cual generará el código de base necesario en las carpetas del proyecto.

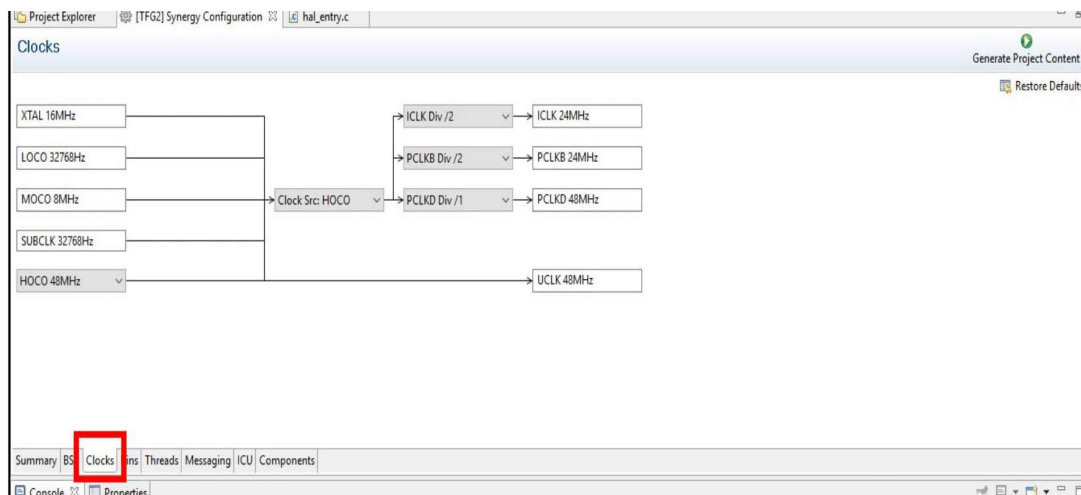


Figura 2.4. Configuración de relojes en *e2studio*

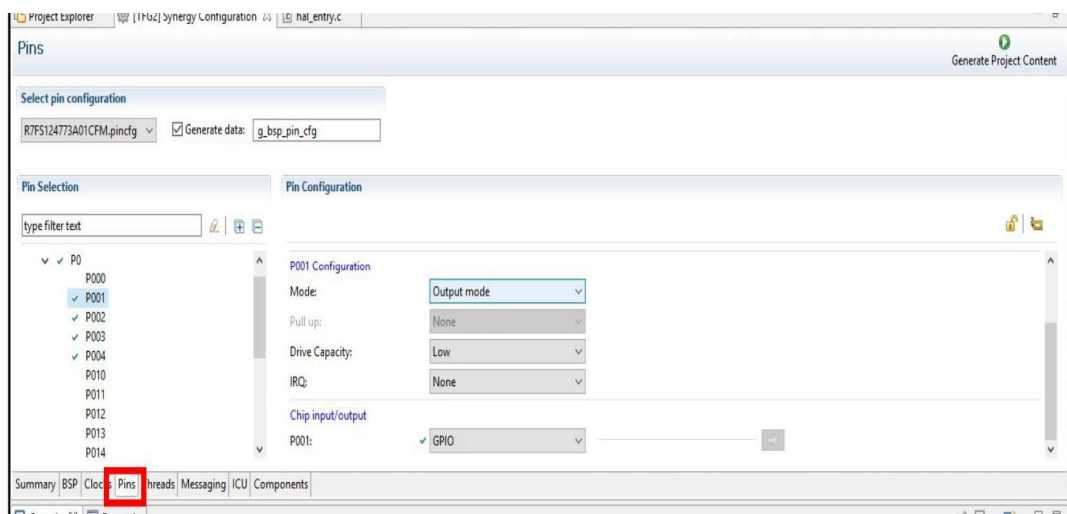


Figura 2.5. Configuración de pines en e2studio

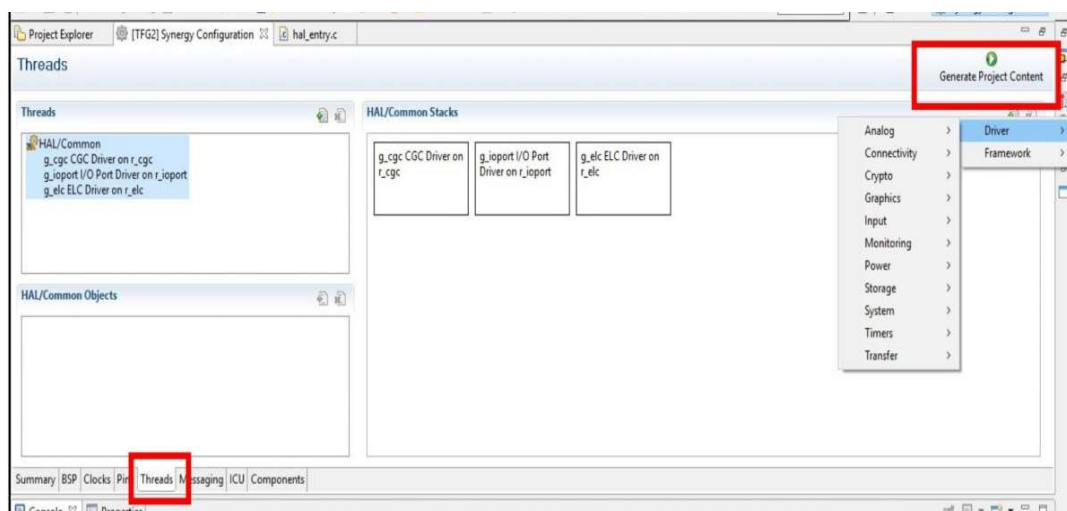


Figura 2.6. Configuración de librerías internas en e2studio
y generación del código previo de configuración

En el *Project Explorer* (Figura 2.7.) se muestran todos los proyectos guardados en el espacio de trabajo asignado y se puede comprobar cómo se generan los archivos de código pertinentes en las carpetas del proyecto en cuestión. Los archivos de código dentro de la carpeta *src* son los que modifica el usuario para implementar la aplicación en código C. El resto de la metodología de trabajo es similar a otros entornos basados en programación en C. La ventaja que ofrece este entorno es el ahorro de tiempo en tareas preliminares, a veces tediosas, lo que permite al programador centrarse en el aspecto más funcional de la aplicación.

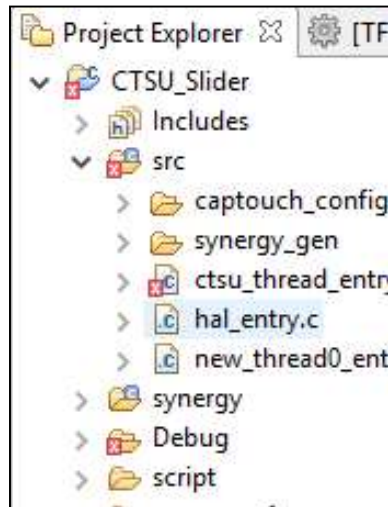


Figura 2.7. Project Explorer: archivos de código de un proyecto en e2studio

Para más información acerca de este entorno de programación consúltese el Manual de Usuario de *e2 studio* [3].

2.2.2. Synergy Software Package

Synergy Software Package forma parte de la plataforma *Renesas Synergy Platform*, un conjunto de dispositivos, software, herramientas y guías, con las que Renesas pretende dar un soporte fiable y estandarizado a la hora de desarrollar productos y aplicaciones, facilitando la labor del diseñador. Para ser más exactos, esta plataforma se divide en 5 elementos principales:

- **Synergy Software** (Software Synergy). Una plataforma completa y de calidad con APIs comunes.
- **Synergy Microcontrollers** (Microcontroladores Synergy). Familia de microcontroladores escalables basados en una arquitectura ARM Cortex.
- **Synergy Gallery** (Galería Synergy). Acceso Web a herramientas y software específico, y más.
- **Synergy Tools and Kits** (Herramientas y Kits Synergy). Herramientas y kits de desarrollo intuitivos.
- **Synergy Solutions** (Soluciones Synergy). Soluciones específicas para aplicaciones y productos.

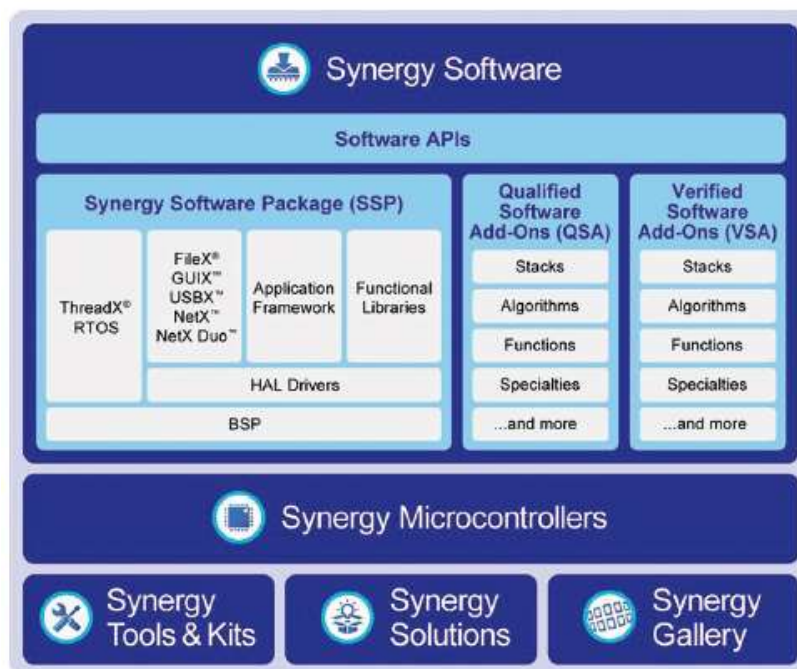


Figura 2.8. Estructura de la plataforma Synergy:
<https://www.renesas.com/en-us/products/synergy/features.html>

En concreto, el *Synergy Software Package* es un software desarrollado con el objetivo de crear programas fáciles de desarrollar y de mantener y está especialmente adaptado para explotar el potencial de los productos Synergy. Está formado por las siguientes capas:

- **Board Support Package, BSP** (Paquete de Soporte de la Placa). Se encuentra en la base de la arquitectura del software. Se trata de un paquete especialmente diseñado para cada kit de hardware que proporciona Renesas, en función del microcontrolador y los periféricos que emplee.
- **HAL⁷ Drivers** (Controladores HAL). Forma parte del sistema operativo y proporciona una capa consistente entre el hardware y el software sobre la que funcionan las aplicaciones. Es decir, las aplicaciones acceden a esta capa, simplificando la labor del programador, al poder disponer de una interfaz en la que la información del hardware se ha abstraído y sintetizado.
- **Application Frameworks** (Frameworks de aplicaciones). Es la capa que une la Capa de Abstracción de Hardware (HAL) con el Sistema Operativo en Tiempo Real (RTOS⁸), proporcionando mensajería de procesos internos, servicios de seguridad, comunicación serie, reproducción de audio...
- **Functional Libraries** (Librerías funcionales). Contienen software especialmente preparado para determinadas aplicaciones, como pueden ser de procesamiento de

⁷ HAL (*Hardware Abstraction Layer*): Capa de Abstracción de Hardware.

⁸ RTOS (Real Time Operating System): Sistema Operativo en Tiempo Real.

señal. No sólo reducen el tiempo de desarrollo sino que mejoran la estabilidad de la aplicación final.

- **ThreadX Real-Time Operating System** (Sistema Operativo en Tiempo Real ThreadX). Se trata del sistema operativo en tiempo real desarrollado para la familia de microcontroladores Synergy.
- **X- Ware stacks and middleware**⁹ (Pilas y middleware X-Ware). Comprenden archivos (FileX), interfaces gráficas de usuario (GUIX), comunicación USB y TCP/IP (USBX, NetX and NetX Duo), estando todo especialmente implementado para la plataforma Synergy.

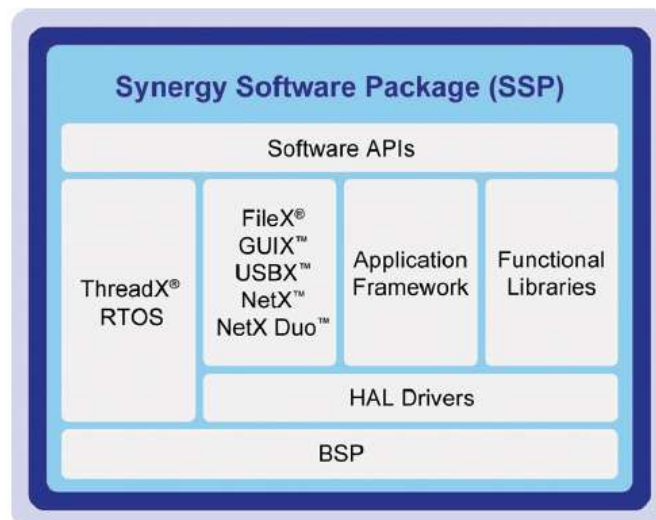


Figura 2.9. Esquema del Synergy Software Package (SSP):

<https://www.renesas.com/en-eu/doc/products/renesas-synergy/doc/R01PF0137ED0200.pdf>

2.2.3. DesignSpark PCB

En lo que respecta al diseño PCB se utiliza el programa *DesignSpark PCB*, desarrollado por RS. El proceso de diseño de circuitos impresos por medio de esta herramienta se detalla en el apartado 3.4. .

Este programa proporciona una amplia gama de librerías de componentes, la cual además se puede ampliar aún más mediante librerías online y creando librerías propias. En *DesignSpark PCB*, un componente incluye un símbolo esquemático y una huella PCB o *footprint*, la cual deberá ser acorde a las dimensiones del componente físico real que se quiere soldar encima. Cuando no se dispone de ninguna librería que contenga un componente, el usuario puede

⁹ *Middleware* o lógica de intercambio de información entre aplicaciones es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos.

crearlo a mano. Normalmente, los fabricantes proporcionan documentación sobre las dimensiones adecuadas de las huellas PCB.

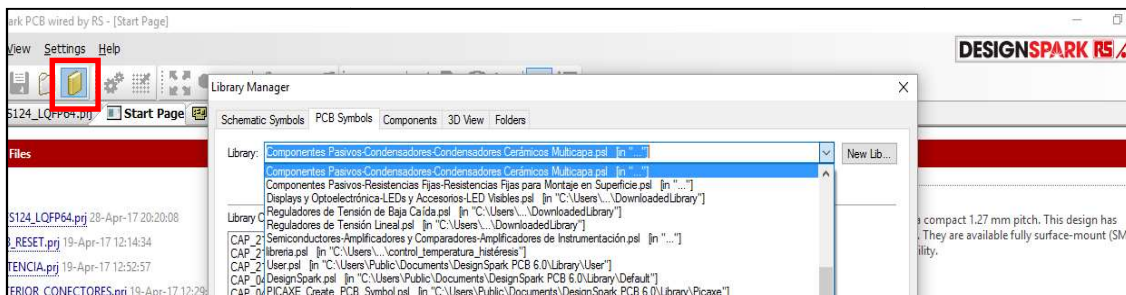


Figura 2.10. Librerías de componentes para DesignSpark PCB

La metodología de trabajo a seguir con este programa se basa en la creación del circuito o esquemático que impondrá las conexiones de los nodos y terminales según el diseño del usuario, y su posterior traducción al circuito impreso final. En este último, el usuario debe situar los componentes y enrutar las pistas a mano, o bien puede ayudarse de herramientas de *layout* y enrutado automático.

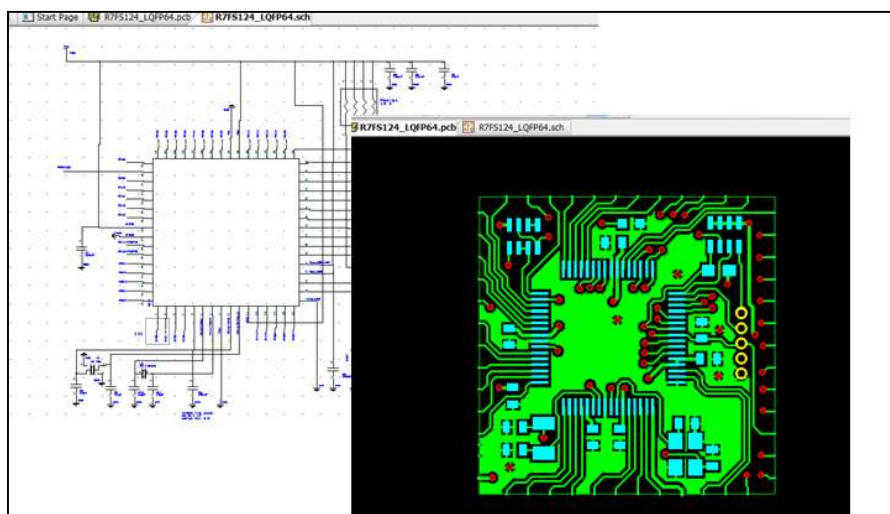


Figura 2.11. Esquemático y traducción a PCB en un proyecto de DesignSpark PCB

3. Reglas y consideraciones en el diseño PCB

Una PCB o Placa de Circuito Impreso es una tarjeta rígida o flexible en cuyas superficies se imprimen circuitos electrónicos mediante buses laminados de cobre, de tal modo que estos unen debidamente los terminales de los componentes que forman el circuito y que se sueldan encima. La placa sobre la que se imprimen los circuitos metalizados ha de ser de un material aislante, siendo el más común la fibra de vidrio. El cobre, que debe ser de alta pureza, por un lado, aporta las conexiones y, por otro, permite soldar los componentes. En la Figura 3.1. se observa un ejemplo de una PCB con todas las partes de las que se compone.

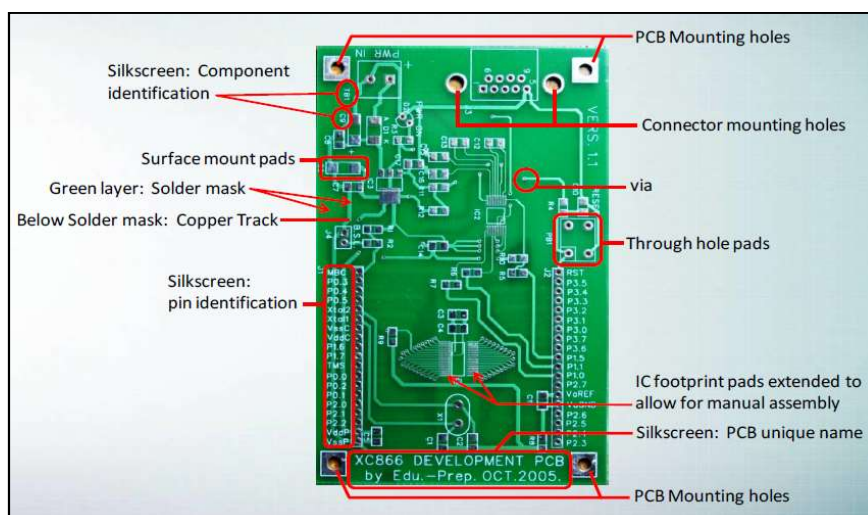


Figura 3.1. Partes de una PCB

A lo largo de los siguientes apartados se exponen una serie de consejos y premisas importantes que se deben tener en cuenta en un diseño PCB.

3.1. Fundamentos del diseño PCB

3.1.1. Posicionamiento de componentes

Una vez impuestas las conexiones de las *nets* en el esquemático del circuito, el posicionamiento o *layout* de los componentes que se pretenden soldar en la placa es el primer paso en el diseño PCB, previo al inicio del enrutado. Se debe prestar especial atención a este paso, ya que suele ser más difícil modificar la situación de los componentes conforme se avanza en el enrutado. A continuación, se exponen una serie de consejos y líneas a seguir que han de tenerse en consideración en esta labor:

- La primera decisión es determinar si se realizará un diseño con componentes a una cara o a dos caras. En función de ello y del número de componentes, así como de su tamaño y forma, se impondrá el tamaño y morfología de la PCB. Se debe prever suficiente espacio para distribuir los componentes sin interferencias o solapamientos entre ellos, teniendo en cuenta tanto los *pads*, como el espacio físico de los empaquetados. En la Figura 3.2. se exponen unos ratios S/C orientativos: área de PCB (S) – área de componentes (C).

Board Type	Single-sided	Double-sided PTH
Discrete components (ICs no more than 5% of the area)	2–3	1.5–2
Mixed (ICs from 35 to 50%)	2.5–4	2–3
IC board (discrete components no more than 20%)	4–6	2–3

Figura 3.2. Ratios S/C orientativos: área de PCB (S) – área de componentes (C)

- Una premisa en la disposición de los componentes es reducir la longitud de las pistas, lo cual no sólo reduce interferencias e impedancias parásitas, sino que también facilita el diseño, al estorbarse en menor medida unas pistas con otras en su camino. Para lograr pistas más cortas, se debe intentar colocar los componentes en orden según el flujo de la señal.
- En la medida de lo posible, los componentes deben quedar alineados, presentando un aspecto ordenado. Normalmente, los bordes de los componentes se orientan paralelos a los bordes de la PCB.
- A veces hay que contar con otras consideraciones particulares de los diferentes componentes, como por ejemplo, colocar los condensadores de desacoplo cerca de los pines del circuito integrado o procurar no introducir inductancias parásitas cuando resulten perjudiciales. En el caso de conectores a dispositivos externos (como puede

ser un puerto USB) se sitúan en una posición lo más práctica posible para su uso por parte del usuario, habitualmente cerca del borde de la placa.

3.1.2. Enrutado de pistas

El enrutado consiste en trazar las pistas de cobre que conectan unos terminales con otros en el circuito impreso. Conviene seguir una serie de pautas que, por un lado, eviten, en la medida de lo posible, el acoplamiento de interferencias o la introducción de impedancias parásitas que alejen al circuito del modelo ideal, y, por otro lado, favorezcan un diseño más eficiente, ordenado y estructurado. Se exponen algunos de estas recomendaciones a continuación:

- Se deben alejar las pistas de los bordes para evitar que se dañen al cortar la tarjeta (margen de al menos 1 mm).
- Del mismo modo, se debe respetar cierto margen con los taladros para evitar que las pistas se dañen durante el mecanizado.
- Es conveniente dejar cierta separación entre pistas y componentes cuando pueda darse lugar a interferencias entre ambos.
- Las consideraciones eléctricas, como la resistencia, capacitancia, inductancia y corriente máxima de las pistas, son especialmente importantes en pistas de alimentación y masa, en señales de alta frecuencia y en pistas muy largas.
- Normalmente, conviene minimizar la resistencia de las pistas. A mayor anchura y espesor de estas, mayor sección de la línea y, por tanto, menor resistencia.
- A mayor sección, es decir, a mayor anchura y espesor de la pista, mayor capacidad de conducción de corriente.
- Para evitar interferencias debidas a efectos capacitivos entre pistas pertenecientes a diferentes capas (como el *crosstalk*), se debe disponer un enrutado ortogonal en una capa respecto a la otra. De este modo, se reduce la capacitancia parásita entre pistas, ya que la superficie de cobre superpuesta con otras pistas de otra capa es menor.

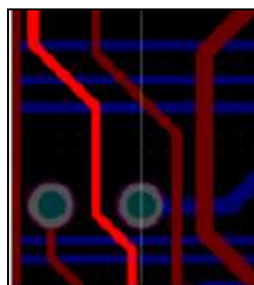


Figura 3.3. Ortogonalidad de pistas en diferentes capas de las PCB

- Hay que evitar siempre que sea posible la separación mínima impuesta entre conductores (separación siempre mayor a 0,1 mm).
- Para favorecer la integridad de la señal, evitando efectos como reflexiones de señal, los cambios de dirección en las pistas no deben realizarse mediante ángulos rectos o agudos (ángulos siempre mayores a 45°). Si queda alguna esquina en el diseño, se recomienda hacer chaflanes.
- Si una pista ha de pasar entre dos pads, procurar que esta quede equidistante entre ambos.
- La anchura de las pistas debe ser directamente proporcional a la corriente que se prevea que va a circular por el nodo en cuestión. Así, cabe esperar que las pistas correspondientes a alimentación y masa soporten intensidades mayores, por lo que deberán ser más anchas.

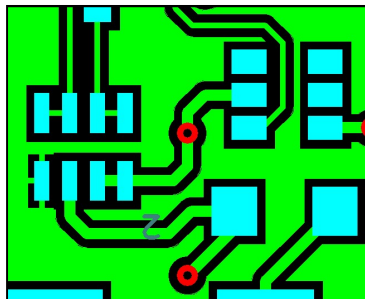


Figura 3.4. Diferentes anchuras de pistas

3.1.3. Pads, máscara de soldadura y pasta de soldadura

Los *pads* son áreas de cobre de la PCB que se sueldan a los terminales de los componentes. Hay que tener en cuenta que los componentes pueden ser de montaje superficial (*smd*), cuyos *pads* son simples superficies de cobre sobre los que reposan los pines, o de agujero pasante (*through hole*), para los cuales los *pads* deben incluir dos superficies de cobre superpuestas en ambas caras y conectadas por un orificio. En este último caso la soldadura se realizará por la cara opuesta al componente. La diferencia se ve claramente en la Figura 3.5. .

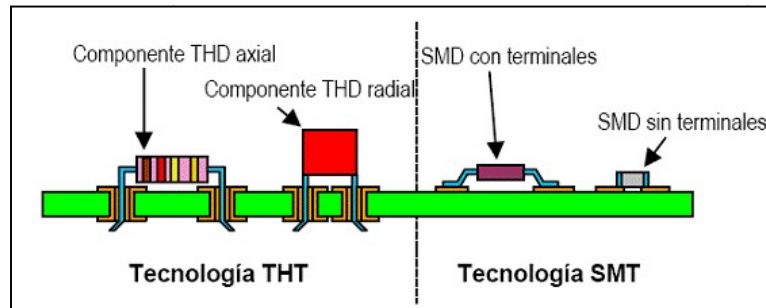


Figura 3.5. Componentes de agujero pasante o through hole (izquierda) y componentes de montaje superficial o SMD (derecha): <https://tecnologiademontajesuperficial.es.tl/>

La huella PCB o *footprint* es el dibujo que forman los *pads* sobre los que se suelda un componente. Los *pads* tienen un tamaño proporcional a los de los pines metalizados de los componentes y deben asegurar una buena soldadura posterior, procurando facilitar en la medida de lo posible esta tarea. Normalmente, los fabricantes aportan información sobre las dimensiones óptimas que deben tener las huellas PCB. A modo de ejemplo, la Figura 3.6. muestra una resistencia y un condensador *smd* montados sobre sus huellas, donde se aprecia como los *pads* suelen tener una superficie ligeramente superior a las de los pines que se sueldan sobre ellos.

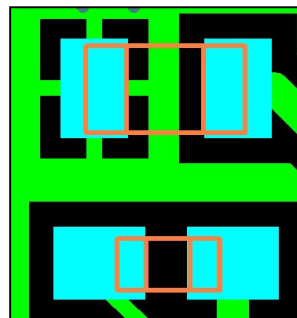


Figura 3.6. Comparación de un condensador SMD (arriba) y una resistencia SMD (abajo) con sus respectivas huellas

La máscara de soldadura es un recubrimiento de lacado que se les da a las PCBs para proteger al cobre de la corrosión y para prevenir posibles cortocircuitos. Obviamente, este no se debe aplicar sobre los *pads*, ya que el cobre en este caso debe quedar al descubierto para permitir la soldadura. Por esta razón, los archivos de diseño del circuito impreso incluyen una capa (*Solder Mask*) formada por el dibujo de todos los *pads*, indicando dónde no se debe aplicar el lacado. En realidad, el dibujo de los *pads* suele ser ligeramente superior a la superficie real de estos, ya que conviene dejar cierto margen en la superficie sin recubrir.

La pasta de soldadura es un producto compuesto principalmente de una aleación de estaño microgranulado mezclado, normalmente, con *flux*¹⁰. Se aplica sólo en los *pads*, existiendo en este caso otra capa (*Solder Paste*) entre los archivos de diseño similar a la de la máscara de soldadura, es decir, la cual indica dónde están los *pads* para aplicar sobre ellos el estaño. La diferencia es que los dibujos de los *pads* en esta capa suelen ser más pequeños que los *pads* reales, dejando cierto margen para no derramar estaño donde no se debe.

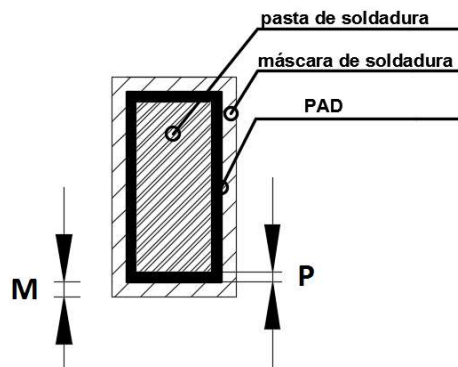


Figura 3.7. Márgenes de pasta de soldadura (P) y máscara de soldadura (M) en un pad

3.1.4. Vías

En diseños PCB con un mínimo de complejidad resulta inasumible diseñar el circuito impreso en una sola capa, ya que con frecuencia es imposible encontrar una solución que lo permita, debido a que unas pistas interfieren con otras. En estos casos se recurre a las *vías*. Las *vías* son orificios que conectan una pista en una capa con otra pista de otra capa diferente. Ambas pistas pertenecen a la misma *net*, por lo que el orificio debe ser conductor.

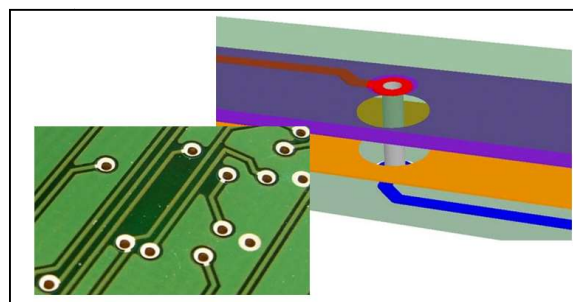


Figura 3.8. Vías

¹⁰ El *flux* o fundente es un producto químico utilizado en la soldadura de circuitos electrónicos. Sirve para prevenir de la corrosión que puede provocar la alta Temperatura alcanzada en el proceso y para favorecer la extensión del material de aportación (normalmente estaño) en las superficies a soldar.

Normalmente, las PCBs son confeccionadas por fabricantes especializados con vias plateadas (metalizadas), pero existen soluciones más rudimentarias cuando no se dispone de la maquinaria necesaria, como soldar en ambos extremos de la *via* un pequeño hilo de cobre.

3.1.5. Planos de masa

En un diseño PCB, cuando se finaliza el enrutado de las pistas, los espacios sobrantes que han quedado en la superficie suelen aprovecharse como planos de cobre conectados a alimentación o a masa.

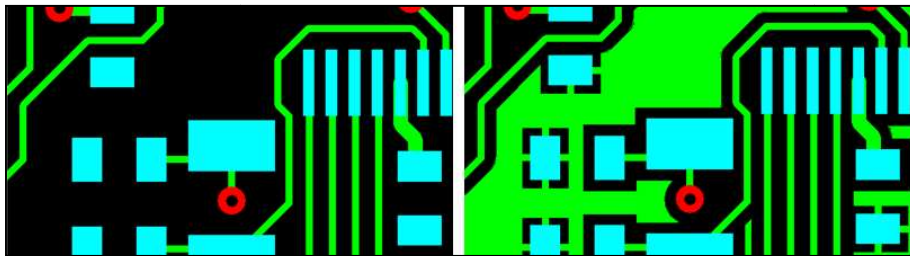


Figura 3.9. "Vertido" del plano de masa tras el enrutado

Este apartado se centra en los planos de masa y en porqué se utilizan, exponiéndose a continuación una serie de ventajas que aportan al diseño:

- Proporcionan una masa de baja impedancia, ofreciendo un camino de retorno de la alimentación que reduce el ruido en el circuito.
- Suponen un blindaje ante interferencias entre dos pistas cuando se sitúa entre ambas, reduciendo el efecto del *crossstalk* o diafonía.
- Mediante su uso se facilita el acceso de todos los componentes a masa, lo cual resulta bastante práctico, ya que muchos componentes deben tener acceso a esta net.

Se debe tener en consideración el efecto de circuitos más ruidosos, como pueden ser los bloques de potencia, en el plano de masa, el cual favorecerá que el ruido se propague. Para evitar que llegue a las partes del circuito más sensibles al ruido, se suele separar el plano de masa en diferentes planos para cada parte del circuito, unidos en el mismo punto, tal y como se ve en la Figura 3.10. .

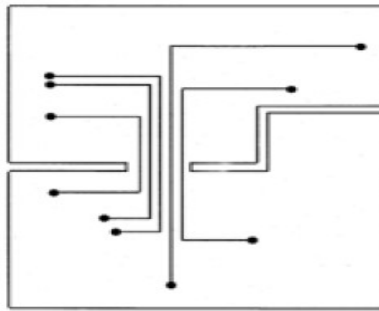


Figura 3.10. Disposición de planos de masa entre diferentes partes del circuito

3.2. Reglas de diseño impuestas

Para seguir unos estándares y tener cierta uniformidad en todos los diseños PCB de unas características similares, se suelen imponer una serie de reglas en cuanto a las dimensiones, márgenes y separaciones de los elementos que conforman el circuito impreso. Previamente al diseño de este proyecto, se asignan unos valores a seguir para estas restricciones, teniendo en cuenta las consideraciones apuntadas a lo largo del apartado 3.1. .

3.2.1. Separaciones

En la siguiente tabla se recogen las separaciones mínimas que se deben respetar entre los diferentes elementos de la PCB: pistas, pads, vias, silkscreen y bordes de la placa. Siempre que sea posible, se deben dejar separaciones mayores.

Distancias mínimas entre elementos de la PCB						
	Pistas	Pads	Vias	Silkscreen	Bordes	Unidades
Pistas	0.16					mm
Pads	0.16	0.16				mm
Vias	0.16	0.16	0.16			mm
Silkscreen	0	0.16	0.16	0.16		mm
Bordes	1	1	1	1	-	mm

Tabla 3.1. Distancias mínimas entre elementos de la PCB

3.2.2. Pistas

Como se explicó en el apartado 3.1.2., tiende a requerirse anchuras de pista mayores conforme mayor sea la corriente máxima prevista en la línea correspondiente. Es por ello que las pistas de alimentación (5V, 3.3V, VIN, AVCC0...) y masa (GND) tienen el doble de anchura que las de señal, ya que por ellas debe circular toda la corriente antes de bifurcarse en las diferentes pistas de señal (puertos del microcontrolador...).

Parámetros de las pistas				
Nets	Alimentación	Masa	Señal	Unidades
Anchura	0.32	0.32	0.16	mm
Espesor	18	18	18	μm

Tabla 3.2. Parámetros de las pistas

3.2.3. Vias

Las vias de este diseño son plateadas y presentan las dimensiones indicadas en la tabla inferior.

<i>Dimensiones de las vias</i>		
		Unidades
Diámetro del taladro	0.26	mm
Diámetro de la corona exterior	0.61	mm

Tabla 3.3. Dimensiones de las vias

3.2.4. Pasta de soldadura y máscara de soldadura

En la siguiente tabla se recogen los márgenes que se utilizan en todos los *pads* para las capas de la máscara de soldadura (*Solder Mask*) y la pasta de soldadura (*Solder Paste*), es decir, la separación entre los bordes de los dibujos correspondientes a los *pads* en ambas capas y los bordes de los *pads* en la capa de cobre (mirar Figura 3.7.).

<i>Márgenes de la Máscara de Soldadura y la Pasta de Soldadura</i>		
		Unidades
Margen Máscara de Soldadura (M)	+0.08	mm
Margen Pasta de Soldadura (P)	-0.05	mm

Tabla 3.4. Márgenes de la Máscara de Soldadura y la Pasta de Soldadura

3.3. Integridad de la señal

Idealmente, las señales digitales que manejan los microcontroladores son cuadradas. Normalmente, tienen un valor alto a 5 V o a 3.3 V y un valor bajo a 0 V, siendo el cambio entre ambos niveles instantáneo. Sin embargo, en sistemas reales se debe lidiar con diversos lastres que lo alejan del modelo teórico, como interferencias, retrasos o impedancias parásitas. Estas fuentes de error hacen peligrar la integridad de la señal.

En un sistema digital real existen unos márgenes de aceptación de nivel alto y bajo, es decir, existe un límite inferior para considerar un valor alto y un límite superior para considerar un valor bajo. Las fuentes de error antes mencionadas generan ruido que se acopla a la señal digital o retardos en esta. Esto puede llegar a provocar lecturas erróneas debido a desfases de los márgenes antes mencionados, tal y como se observa en la Figura 3.11. . El principio básico de la integridad de la señal es conseguir cierto factor de seguridad a la hora de atenerse a estos márgenes.

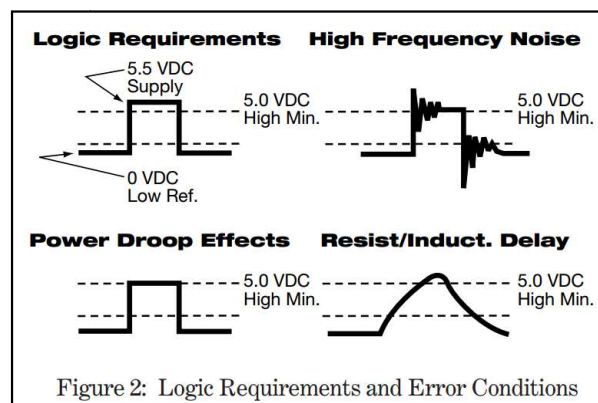


Figura 3.11. Márgenes de aceptación de niveles digitales. Integridad de señal [6]

Para proporcionar a los circuitos integrados señales de calidad libres de errores e incertidumbre se utilizan una serie de elementos y técnicas. Por ejemplo, cabe mencionar componentes como condensadores de desacoplo, condensadores de reserva y resistencias pull-up y pull-down.

3.3.1. Condensadores de desacoplo

El desacoplo nace de la necesidad de eliminar de las señales digitales del microcontrolador ruidos y sobretensiones generados en el sistema de alimentación.

Un circuito integrado como un microcontrolador, basado en conmutaciones de transistores CMOS, sólo demanda corriente cuando dichos transistores conmutan (sin contar con las corrientes de fuga). Estas conmutaciones se dan a gran velocidad y dan lugar a una entrada de corriente con cambios bruscos, lo cual provoca variaciones repentinas de tensión. Esto se entiende mediante la siguiente relación:

$$\frac{dV}{dt} = \frac{I}{C}$$

Según esta ecuación, para una demanda de corriente, I , dada, se darán picos de tensiones tanto mayores cuanto mayor sea I e inversamente proporcionales a la capacidad C a la entrada de esta corriente. Si no se colocara ningún condensador en esta entrada, esta capacidad C se correspondería con la capacidad parásita en ese punto del circuito. Esta, al tomar valores bajos, da lugar a picos de tensión altos, los cuales se trata de evitar, de ahí la necesidad del condensador de desacoplo. Por tanto, se colocan condensadores de desacoplo para imponer una capacidad C determinada en estas entradas que evite variaciones excesivas de tensión.

Otra forma de entender el razonamiento anterior es que el propósito del condensador de desacoplo es proveer de una fuente de corriente local de velocidad de respuesta muy rápida, durante un corto período de tiempo. Estos condensadores se deben colocar cerca de los pines de alimentación del circuito integrado en cuestión, ya que se trata de disponer de cargas almacenadas 'localmente' y de esta manera no se introducen otras impedancias parásitas indeseadas que les puedan afectar.

Para mayor información sobre las consideraciones a la hora de elegir el condensador de desacoplo adecuado se recomienda recurrir a documentación científica al respecto [6].

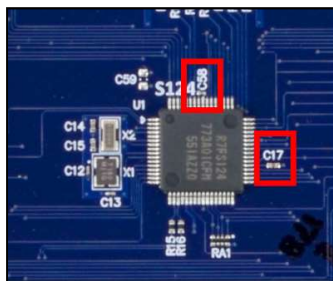


Figura 3.12. Condensadores de desacoplo para el microcontrolador R7FS124773A01CFM en la placa DK-S124

3.3.2. Condensadores de reserva (bulk)

Estos condensadores se colocan en paralelo a los condensadores de desacoplo y previamente a ellos, según el sentido del flujo de la señal desde la fuente de alimentación. En la Figura 3.13. se aprecia un esquema de la entrada de potencia a un circuito integrado que muestra la forma habitual de conectar ambos tipos de condensadores.

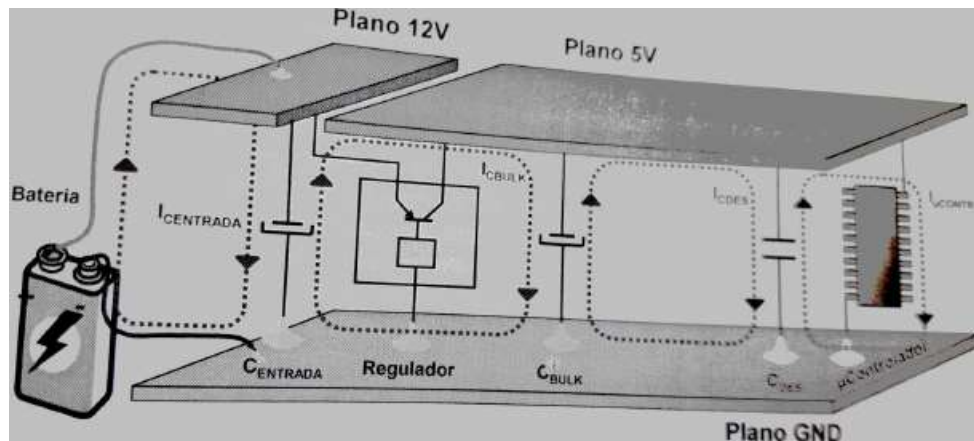


Figura 3.13. Esquema de entrada de potencia en un circuito integrado

Normalmente, se conecta un sólo condensador de reserva por bloque funcional. Su valor de capacidad debe ser de unas 20 veces la suma de todos los condensadores de desacoplo y debe tener una tensión de trabajo de al menos el doble de la utilizada en el circuito.

3.3.3. Resistencias pull-up y pull-down

Se trata de resistencias que se colocan en los pines de los microcontroladores, conectándolos con un nivel alto (5 V o 3.3 V) en el caso de las resistencias pull-up o con un nivel bajo (masa) en el caso de las resistencias pull-down, siendo más comunes las primeras.

Uno de sus objetivos es fijar la tensión de un pin de entrada cuando este no recibe señal al nivel alto o bajo, respectivamente, de tal modo que se eviten valores de tensión flotantes, ya que estos son perjudiciales. Esto se debe a que provocan consumos innecesarios de corriente al tratarse de circuitos integrados con lógica CMOS, dado que cuando los valores de tensión se encuentran indeterminados suelen oscilar en las regiones de conmutación de los transistores, provocando que se demande corriente cuando no se requiere. Por otro lado, se colocan en entradas/salidas en colector o drenador abierto, las cuales se dan en algunos tipos de buses de comunicación, como es el caso de terminales de señal en buses I2C.

Otra de las razones por las que se usan estas resistencias es para evitar la circulación de corriente excesiva dentro del circuito integrado ante la lectura de unos y ceros lógicos. El circuito de lectura puede trabajar con corrientes bastante bajas y, de este modo, se le libera de mayor intensidad de la necesaria, al desviarse gran parte de esta por la resistencia. En la Figura 3.14. se expone un ejemplo de conexión de una resistencia pull-up externa. El valor exacto de estas resistencias no es relevante pero suele rondar las decenas de kΩs, ya que, como se ha dicho, la corriente consumida por las entradas digitales es sumamente pequeña.

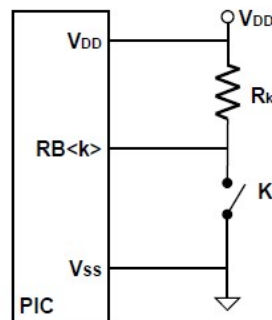


Figura 3.14. Conexión de una resistencia pull-up externa a un pin

Muchos terminales de los microcontroladores disponen de resistencias pull-up o pull-down internas, las cuales se pueden imponer o no mediante la programación del registro correspondiente y cuando el pin en cuestión funciona como entrada. En la Figura 3.15. se observa este caso dentro de un PIC. Estas resistencias internas son en realidad transistores N-MOS o P-MOS trabajando en su región óhmica. Normalmente, toman valores de alrededor de 20 kΩ y pueden ser menos efectivas que las externas.

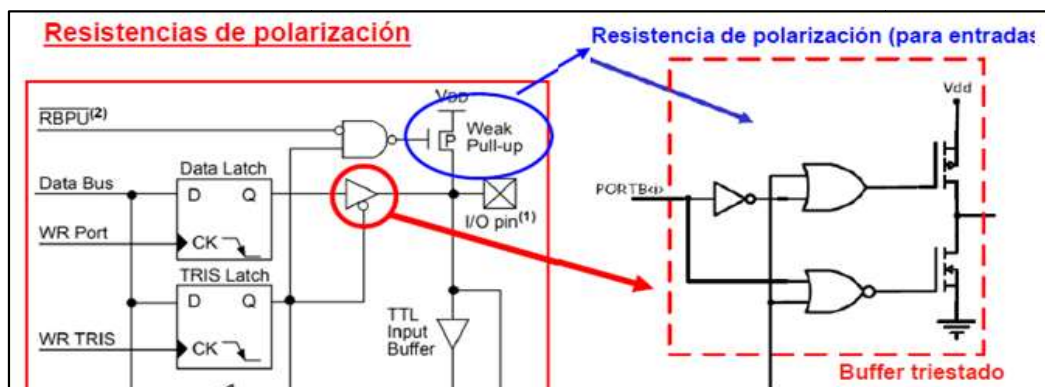


Figura 3.15. Resistencia pull-up interna en un pin de un PIC

3.4. Diseño con DesignSparkPCB

Como ya se dijo en el apartado 2.2.3., la herramienta software a emplear para diseñar el módulo OEM es *DesignSpark PCB*. La metodología que se sigue con este programa en el desarrollo de un proyecto es siempre la misma y se detalla paso por paso a continuación:

1. Se crean todos los componentes que incluye el circuito, cada uno de los cuales debe incluir el símbolo esquemático que lo representa y la huella sobre la que se soldará en la PCB. Para crear la huella con las dimensiones adecuadas se consulta la *datasheet* de cada componente.

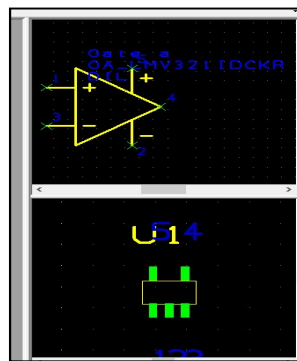


Figura 3.16. *DesignSpark PCB*: los componentes constan de su símbolo esquemático (arriba) y su huella PCB (abajo)

2. Se realiza el esquemático, donde se disponen todas las conexiones de los nodos del circuito de forma correcta.

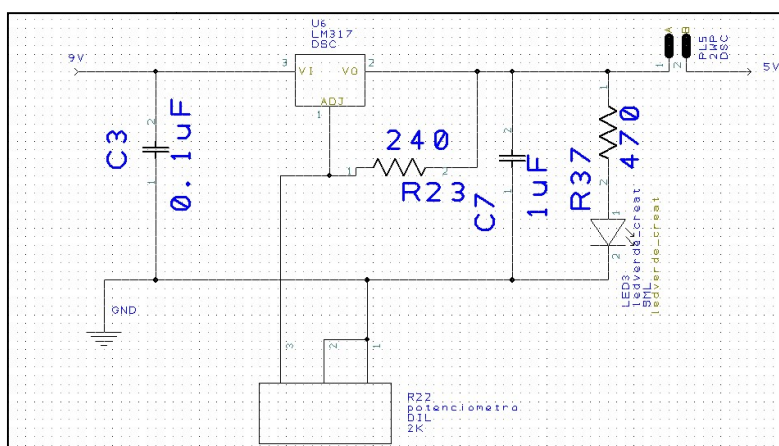


Figura 3.17. *DesignSpark PCB*: ejemplo de un esquemático

3. Se traduce el esquemático a la placa donde se imprimirá el diseño PCB, la cual tiene las dimensiones y características (número de capas, pasta y máscara de soldadura...) que se especifiquen.

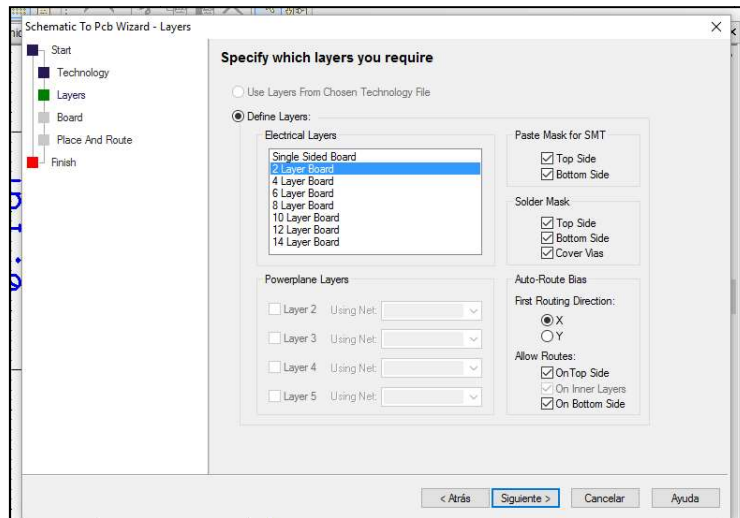


Figura 3.18. DesignSpark PCB: configuración inicial de las características de la PCB

4. Se crean estilos de diseño donde se especifican anchos de pista y espaciados entre diferentes elementos, entre otras cosas, para luego asignárselos a las diferentes *nets*.
5. Una vez en esta, se disponen de la forma más eficiente y lógica posible las huellas sobre la que se soldarán los componentes. En un principio se representan las conexiones entre las *nets* de forma simbólica, para luego trazar las pistas una a una.

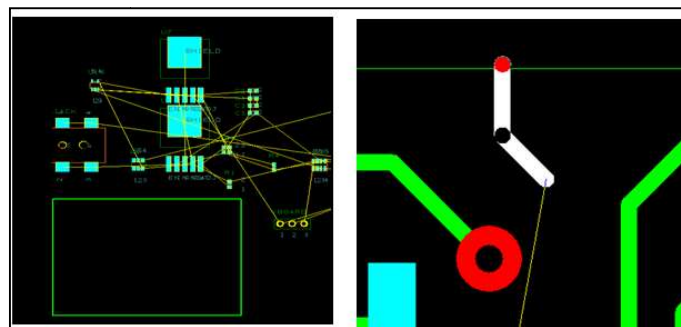


Figura 3.19. DesignSpark PCB: disposición de los componentes fuera de la tarjeta con las conexiones impuestas en el esquemático (izquierda) y enrutado de pistas (derecha)

6. Como se explicó anteriormente, suele ser recomendable distribuir planos de alimentación y masa entre las pistas.

7. Una vez terminado el diseño del circuito impreso, se comprueba que no presenta ninguna discrepancia respecto al esquemático del que proviene, para lo cual se selecciona la opción *Schematic PCB Check*. Así mismo, se revisan los fallos que puedan haberse cometido en cuanto a las reglas de diseño impuestas (espaciados, anchos de pista...) mediante la opción *Design Rule Check*. Si durante el diseño se decide cambiar algún componente, una vez realizados estos cambios, se debe seleccionar *Update Components* para aplicarlos a la PCB.

8. Una vez finalizado y revisado el diseño se deben generar los archivos de salida con toda la información necesaria para su confección, llamados archivos *gerber*. Estos archivos serán los que se envíen al fabricante para la confección de la placa.

4. Diseño PCB en bloques

Un elemento a destacar en este proyecto es el diseño PCB en bloques. En este proyecto se pretende demostrar las ventajas de usar esta metodología en el diseño de circuitos impresos, sobre todo en cuanto a ahorro de tiempo en el diseño.

Este método se basa en la creación de diferentes bloques PCB (implementados en diferentes proyectos) con funcionalidades específicas y que dan salida a determinadas *nets* de su interior a través de terminales distribuidos a lo largo de un marco. Este marco delimita la porción de PCB que comprende el bloque, siguiendo su contorno. A lo largo de este, como se ha dicho, se distribuirán los terminales a través de los que se interconectan todos los bloques entre sí. El objetivo es combinar estos bloques como si se tratara de un ensamblaje, de tal modo que se consiga un circuito global con unas características determinadas, las cuales vendrán determinadas por los bloques que incluya. Una buena analogía es pensar en este diseño final como si fuera un puzle en el que cada uno de los bloques es una pieza (delimitada por el marco) que habrá que encajar adecuadamente. La PCB completa deberá incluir al menos un bloque que cohesione los restantes entre sí, interconectando sus terminales debidamente.

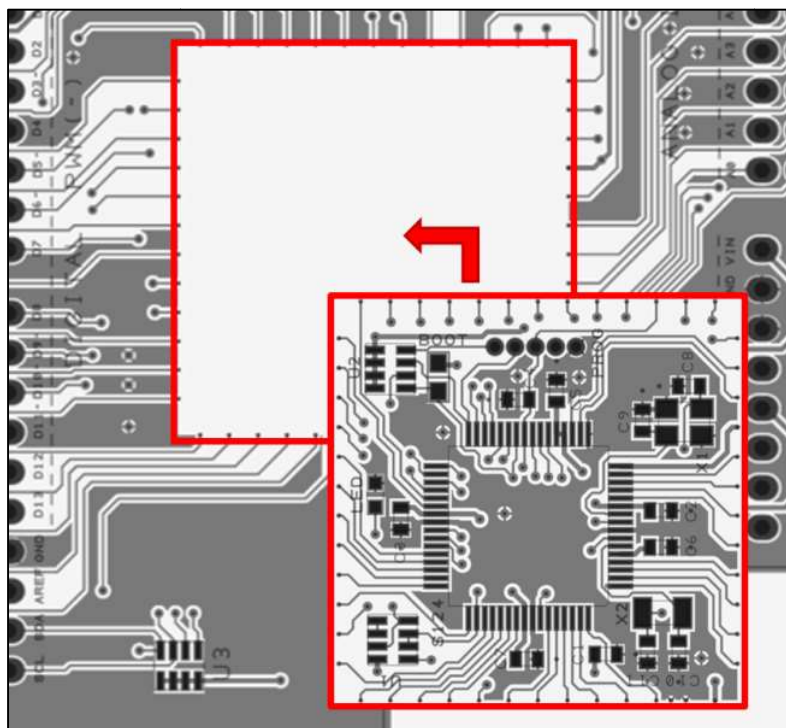


Figura 4.1. Unión de bloques PCB

De este modo, podemos pensar en el objetivo inicial de este proyecto, como es el diseño PCB que incluya el microcontrolador y el hardware básico para su funcionamiento, como la pieza central de un circuito PCB mayor, es decir, se trata de una unidad de control para hipotéticos periféricos dispuestos en diferentes bloques interconectados con este. El marco que delimita este bloque estará conformado por conexiones a los puertos del microcontrolador, las cuales podrán controlar periféricos externos, así como por las entradas de alimentación que este necesita. Para conseguir estas últimas, se necesitará un bloque PCB externo que contenga el hardware necesario para proporcionarlas.

Inciendo en los marcos mencionados, los cuales delimitan cada una de las “piezas” de la PCB, estos deben incluir tantos terminales en su contorno como *nets* se interconecten con otros bloques externos. En el entorno de desarrollo utilizado para el diseño PCB, el marco se crea como un componente más. El símbolo esquemático se conecta a los nodos correspondientes a cada uno de los terminales y la huella PCB consiste en *pads* distribuidos a lo largo de los límites que impone el marco. Estos *pads* deben tener, preferiblemente, la misma anchura que las pistas a las que se conectan, asegurándose así una buena continuidad de las pistas de un bloque a otro.

En la Figura 4.2. se muestra, a modo de ejemplo, los pasos básicos en el diseño de uno de estos bloques. Primero se debe crear el componente correspondiente al marco, con tantos terminales como *nets* del bloque se quieran conectar con el exterior. Tras ello, en el esquemático se indican las conexiones de estos terminales y en el diseño PCB, donde se encontrarán las conexiones ya determinadas, se realiza el enrutado.

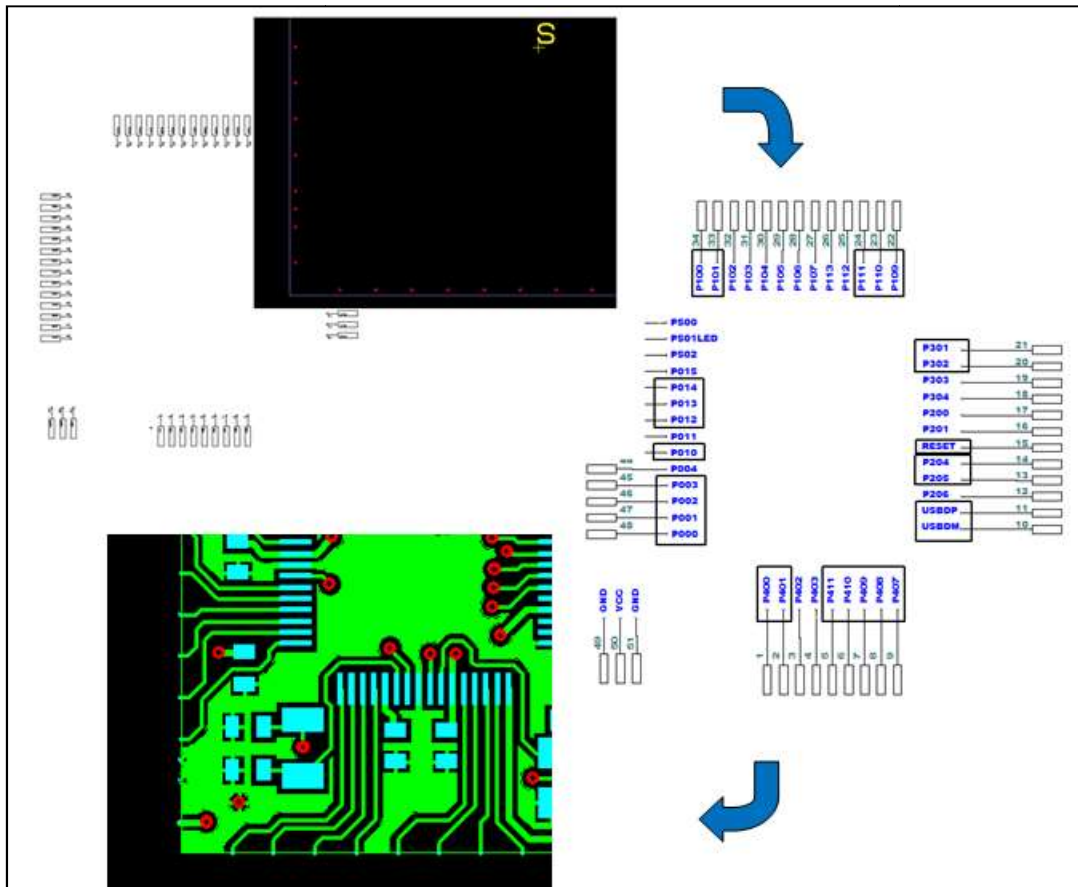


Figura 4.2. Proceso de creación de los bloques PCB: creación del marco (arriba), imposición de conexiones (medio) y traducción a PCB (abajo)

El objetivo de este proyecto solamente comprende la parte del diseño correspondiente al diseño de los bloques por separado. El siguiente paso sería fusionar los bloques en los archivos de salida finales para la fabricación del módulo, lo cual es realizado por Embeblue mediante un software hecho a medida en dicha empresa.

5. Diseño del módulo OEM

A la hora de plantear el diseño del módulo que se pretende fabricar conviene comenzar por recordar los objetivos fundamentales que este debe cumplir. Además, el proceso estará marcado en todo momento por la filosofía del diseño PCB en bloques, como se explicó en el apartado 4. Teniendo en cuenta esto, el diseño se dividirá en tres pasos fundamentales

- Realizar un bloque PCB que contenga el microcontrolador R7FS124773A01CFM, así como el hardware básico que nos permita utilizarlo sin restarle ninguna funcionalidad, permitiendo, además, su programación. Se trata de conseguir un bloque versátil que se pueda utilizar en más diseños, por lo que, por un lado, es preferible que tenga un tamaño reducido para hacerlo más manejable y, por otro lado, se pretende dar salida al mayor número posible de terminales del microcontrolador para su uso por parte de otros periféricos externos.
- Diseñar los bloques restantes que le aporten al microcontrolador otros requerimientos necesarios para su funcionamiento en un módulo PCB completo (entrada de potencia), así como características adicionales aportadas por otros periféricos que añadan funcionalidad a la placa (puerto USB).
- Conseguir un conjunto completo de bloques PCB que, una vez unidos, formen juntos el módulo buscado. Por tanto, una vez reunidas todas las piezas del circuito impreso necesarias, el siguiente paso es implementar los archivos de fabricación del módulo unido y completo.

A la hora de plantear las características del diseño y de los periféricos a añadir alrededor del microcontrolador, se decide apostar por la sencillez, ya que no se desea sumar más incertidumbre de la necesaria en el funcionamiento del circuito global, dado que el objetivo principal sigue siendo el cumplir los requerimientos básicos antes expuestos. Por otro lado, esta sencillez supone cierto ahorro económico en la fabricación del diseño final.

Pensando en esta sencillez se optó por inspirarse en, probablemente, las tarjetas de características similares más utilizadas y populares actualmente, como son las placas Arduino. Se pretende conseguir una distribución de los pines equivalente a la presente en tarjetas como el Arduino UNO, el Arduino Leonardo o el Arduino MEGA, buscando aportarle al módulo la ventaja de hacerlo compatible con las diferentes *shields* implementadas para las placas mencionadas.



Figura 5.1. Placa Arduino UNO

Por otra parte, cabe mencionar que se tratará de una placa con componentes en una sólo cara (al igual que las placas de Arduino mencionadas), lo cual simplifica el proceso de fabricación, sobre todo la labor de soldadura.

Siguiendo todas estas consideraciones, se decide que la PCB constará de cuatro bloques que se enumeran a continuación:

- Bloque del microcontrolador. Como se ha dicho, es, probablemente, el objetivo más importante del proyecto y contiene el R7FS124773A01 y el hardware adicional requerido.
- Bloque de alimentación. Contiene el conector de entrada de potencia y el acondicionamiento de la señal de alimentación a todas las tensiones requeridas.
- Bloque de conexión USB. Contiene un conector USB para posibilitar la comunicación por medio de este protocolo. Además, se incluye aquí un pulsador conectado al pin de RESET del microcontrolador.
- Bloque externo unificador. Es el bloque necesario para cohesionar los restantes, por lo que su dibujo deberá encajar a la perfección con el de los otros marcos. Todo diseño por bloques deberá incluir uno de este tipo diferente en cada caso. Además, se aprovechará este bloque para disponer una serie de conectores externos para el acceso a las diferentes *nets* por parte del usuario.

En los sucesivos apartados se detallan las características internas de cada uno de estos bloques.

5.1. Bloque del microcontrolador

Como se ha comentado en el apartado anterior, el objetivo fundamental de este circuito es disponer el hardware elemental necesario para conseguir una integridad de señal óptima a la entrada y salida de los puertos y una adecuada alimentación del microcontrolador. Además, se pretende incluir una conexión externa para la programación del mismo, un LED para testear su correcto funcionamiento, así como dos cristales externos de 16 MHz y 32,768 KHz, respectivamente.

Se intentará dar salida al mayor número posible de pines, por lo que todos aquellos que no se utilicen internamente, por ejemplo las conexiones a los cristales y al programador, y no sean terminales de alimentación del microcontrolador se conectarán al marco que delimita el bloque.

5.1.1. Diseño del esquemático

En la Figura 5.2. se muestra el esquemático completo de este bloque.

En lo que sigue de este apartado, se analizan una a una en detalle todas las partes del circuito.

Condensador de reserva¹¹ (C1)

Su valor es de 10 μ F. Se observa que es unas 20 veces superior a la suma de los valores de todos los condensadores de desacoplo.

Condensadores de desacoplo¹² (C2, C3, C5, C6, C7)

Se comprueban en la documentación los terminales de alimentación del microcontrolador que demandan este tipo de condensadores. Todos los condensadores de desacoplo tienen un valor de 100 nF, suficiente para contrarrestar las interferencias producidas por los flancos de tensión para las frecuencias de trabajo de este microcontrolador.

Resistencias pull-up¹³ (U1, U2)

Estas resistencias se colocan en terminales donde es importante evitar lecturas erróneas cuando se encuentran en reposo, como el pin de Reset, o los terminales de datos del conector de programación.

Filtro Reset (U1, C4)

Su objetivo es evitar rebotes de señal indeseados que provoquen la entrada de un '0' lógico cuando no se desea, ya que este resetearía el programa. Para ello, se conecta a una resistencia pull-up, que mantiene el nivel alto mientras no se acciona el pulsador, y a un condensador de 100 nF, que evita que se detecten múltiples pulsaciones debidas a rebotes durante el tiempo que se tarda en accionar el pulsador, consiguiéndose el envío de un sólo '0' lógico. Esta es una solución hardware, pero también se puede resolver el problema de los rebotes vía software.

¹¹ Mirar apartado 3.3.1.

¹² Mirar apartado 3.3.2.

¹³ Mirar apartado 3.3.3.

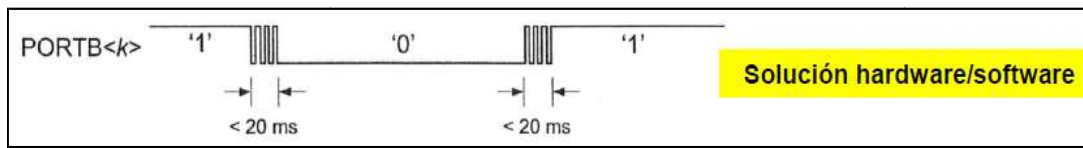


Figura 5.3. Rebotes de señal generados en la pulsación manual

Osciladores externos (X1, X2)

El microcontrolador R7FS124773A01CFM permite la introducción de dos señales de relojes externas a través de los pines XTAL y EXTAL (X1), por un lado, y los pines XCIN y XCOU (X2), por otro. La ventaja que ofrecen estas señales externas es que permiten no limitarse al uso de los osciladores internos del microcontrolador, que presentan limitaciones y suelen ser menos precisos, y posibilitan el generar pulsos periódicos a una frecuencia determinada externamente.

La práctica más común, como es el caso de este diseño, es colocar osciladores externos basados en cristales de cuarzo, cuya ventaja es la precisión de las señales de reloj que generan. Para estabilizar su frecuencia, deben conectarse a dos condensadores como se observa en la Figura 5.4., cuya capacitancia, llamada capacitancia de carga, debe consultarse en las *datasheets* de los cristales. Las frecuencias escogidas para los dos osciladores son 16 MHz y 32,768 KHz, siguiendo el diseño de la placa DK-S124. Además, a partir de divisores internos del microcontrolador se pueden obtener otras frecuencias a partir de las citadas.

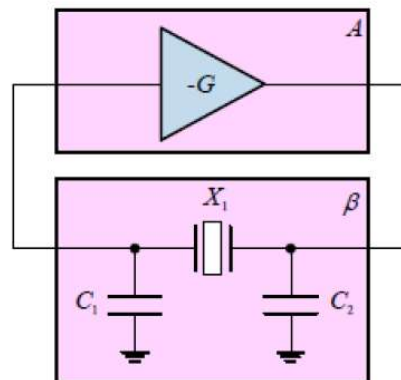


Figura 5.4. Oscilador Pierce: esquema de conexión del cristal de cuarzo (abajo)

Conector de programación (PROG)

Se trata de 5 pines macho para la conexión de un debugger o un programador externo que permitirá cargar el *firmware* necesario en el microcontrolador. Estos cinco pines son:

- **SWDIO** (*Serial Wire Data I/O*). Señal de entrada de datos al microcontrolador para su programación y depuración.
- **SWCLK** (*Serial Wire Data Clock*). Señal de reloj que coordina el flujo de datos durante la programación del microcontrolador y la depuración del software.
- **RESET**. Conectado al terminal de RESET del microcontrolador
- **VCC**. Conectado a la net correspondiente a la alimentación del microcontrolador.
- **GND**. Conectado a la masa de la PCB.

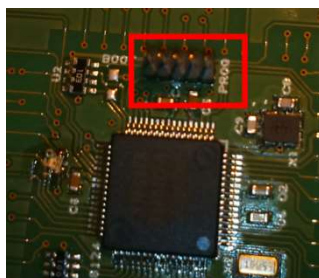


Figura 5.5. Conector de programación del módulo

LED de testeo

La función de este LED es aportar una señal física que permita testear el correcto funcionamiento del circuito y el software. La corriente que debe circular por el LED para su correcta iluminación debe ser de alrededor de 5 mA, por lo que, teniendo en cuenta que el '1' lógico por el cual se enciende es una tensión de 3.3 V o 5 V, la resistencia en serie debe ser de aproximadamente un valor entre 500 y 1000 Ω . Para este propósito se decide "amortizar" tres resistencias sobrantes de un array, tal y como se aprecia en el esquemático (Figura 5.2.), consiguiéndose 666.6 Ω .

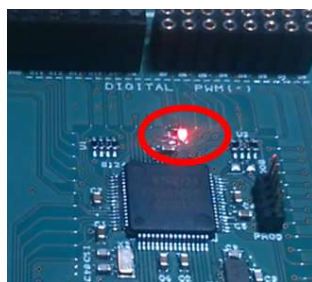


Figura 5.6. LED de testeo del módulo

Marco de conexiones

Como ya se ha dicho, se desea poder acceder desde otros bloques externos al mayor número posible de pines del microcontrolador, por lo que todos aquellos que no se utilicen en conexiones internas de este bloque se llevan hacia los terminales del marco, los cuales se distribuyen uniformemente en el contorno, tal y como se preparó previamente en la huella correspondiente. En la Figura 5.7. se observa el esquemático con los pines del microcontrolador que se conectan al marco.

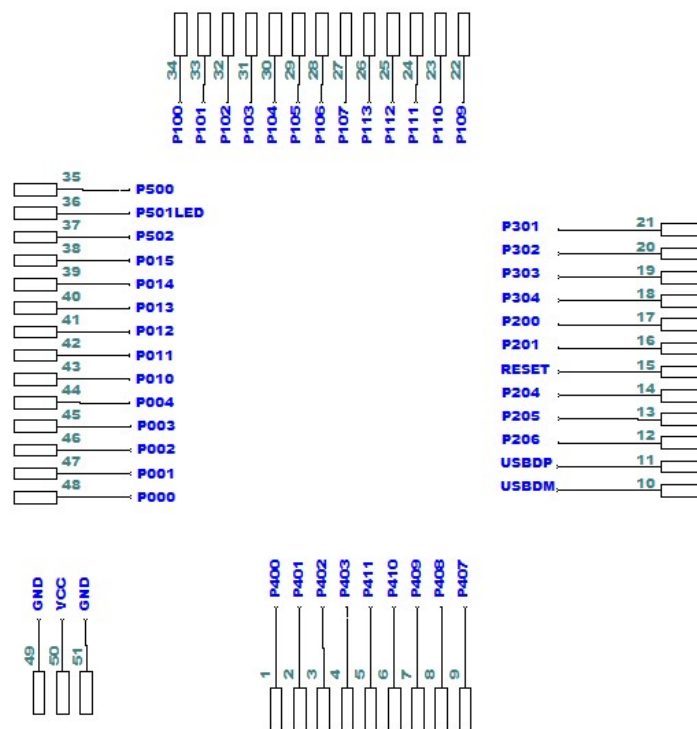


Figura 5.7. Bloque del microcontrolador: esquemático del marco

5.1.2. Diseño PCB

Además de seguir las recomendaciones expuestas en el apartado 3 sobre pautas a seguir en circuitos PCB, se enumeran una serie de apreciaciones que se tienen en cuenta en el diseño de esta parte de la placa:

- En la medida de lo posible, se busca reducir el tamaño del bloque, lo cual facilita la tarea de acoplarlo en otros circuitos impresos. Así, se dispone un marco cuadrado de 2.5x2.5 cm en el que se distribuyen las conexiones que irán al exterior del bloque. Sin contar las conexiones de entrada de masa (GND) y alimentación (VCC), se tienen 48 terminales en el marco, procedentes de entradas/salidas del microcontrolador. Se han excluido algunos pines que se utilizan internamente en el bloque, como por ejemplo los pines de programación.

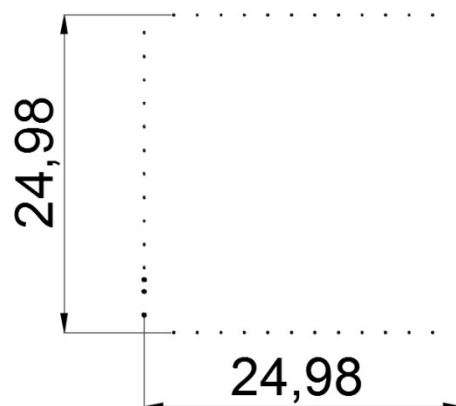


Figura 5.8. Huella del marco para el bloque del microcontrolador

- Las conexiones del marco con los pines del microcontrolador se disponen de una forma lógica, procurando en la medida de lo posible que cada terminal del marco quede cerca de su pin correspondiente, facilitándose así el diseño y reduciendo la longitud de las pistas.
- Los condensadores de desacoplo, se colocan lo más cerca posible de los pines que los demandan, reduciéndose así capacitancias parásitas e impedancias adicionales indeseadas.
- Las pistas USBDM y USBDP, al tratarse de un tipo de comunicación en modo diferencial, deben disponerse lo más paralelamente posible con una longitud de pista similar, de tal forma que tengan una impedancia igual o muy parecida.

Se muestra en la siguiente página el aspecto final del bloque del microcontrolador, superponiendo las diferentes capas en la vista TOP y en la vista BOTTOM. Además se incluyen dos ejes que muestran las dimensiones en mm.

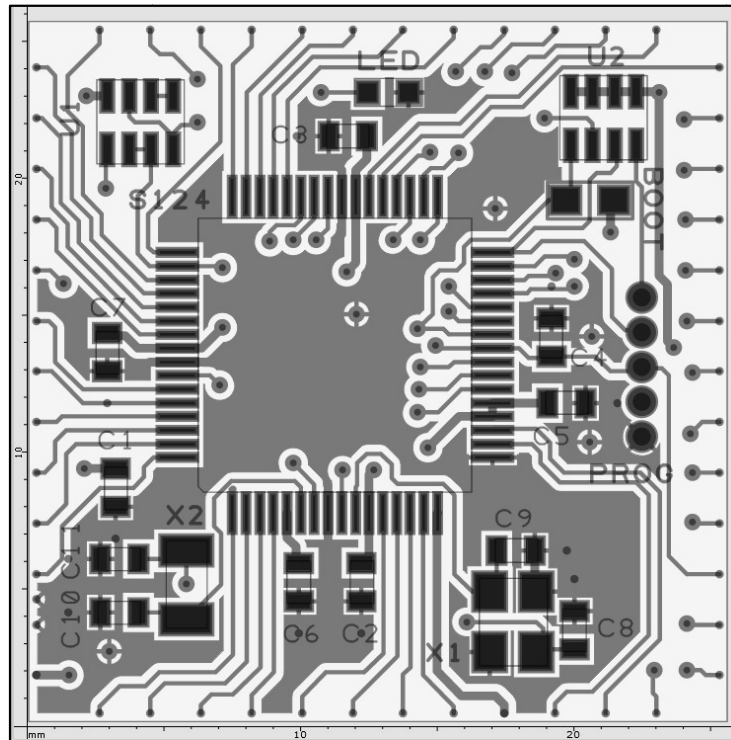


Figura 5.9. Bloque del microcontrolador, vista PCB: cara TOP

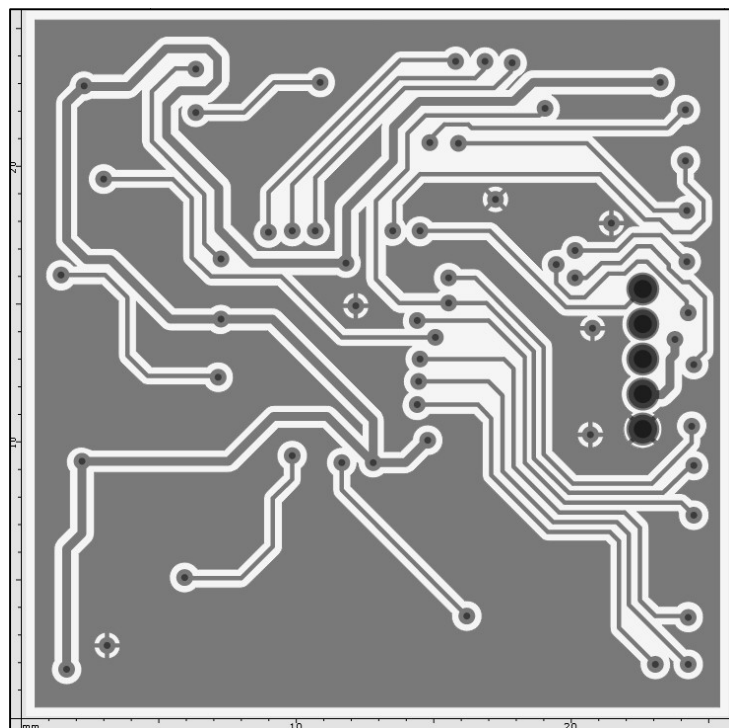


Figura 5.10. Bloque del microcontrolador, vista PCB: cara BOTTOM

5.2. Bloque de alimentación

El objetivo de este bloque es el de posibilitar la entrada de potencia a través de un conector tipo jack hembra, así como acondicionar esta señal de alimentación a las tensiones de trabajo compatibles con el circuito.

En la placa final, la entrada de potencia al circuito podrá provenir de las siguientes conexiones:

- Conector jack hembra a una fuente de tensión externa de un valor superior a 9 V. No es recomendable sobrepasar en más de 1 o 2 V dicho valor, ya que esto da lugar a un aumento en las pérdidas y en el recalentamiento de algunos componentes. Este conector se encuentra en el bloque de alimentación.
- Terminal de 5 V del puerto USB (ver apartado 5.3.).
- La misma tensión que introduciríamos a través del jack a través de los pines externos 'VIN' y 'GND'. El resultado sería el mismo en ambos casos, aunque no serviría de nada el diodo de protección a la entrada del jack ante inversiones de polaridad, por lo que se debe tener especial cuidado en este sentido.
- Tensión de 5 V a través de los pines externos '5V' y 'GND'. No se debe usar al mismo tiempo que la conexión USB. No se dispondría de tensión en el pin 'VIN' y tampoco funcionaría la protección del diodo a la entrada. No se recomienda.
- Tensión de 3.3 V a través de los pines externos '3.3V' y 'GND'. No se debe usar al mismo tiempo que la conexión USB. No se dispondría de tensión en el pin 'VIN' ni de la *net* de 5 V, y no funcionaría la protección ante inversiones de polaridad. No se recomienda.

De las opciones expuestas anteriormente, las más recomendables son las dos primeras, ya que la placa no pierde funcionalidad y es más segura, al funcionar todas las protecciones.

En el caso de conectar un potencial a través del jack o del pin 'VIN' se debe excluir la entrada de potencia a través del puerto USB. Esto es debido a que las diferencias en el valor en torno a 5 V que generan ambas fuentes externas pueden dañar el circuito, al desembocar ambas tensiones en la misma *net*, lo cual daría lugar a sobrecorrientes si no se desconectara una de ellas. Por tanto, se debe diseñar un sistema que desconecte la entrada de 5V del puerto USB cuando se detecte la entrada de potencia a través del jack o de 'VIN'. Esto se consigue mediante un amplificador operacional y un transistor P-MOS, como se explica más adelante.

5.2.1. Diseño del esquemático

En la Figura 5.11. se muestra el esquemático correspondiente al marco de conexiones de este bloque, en el que se aprecian los terminales que entran y salen de este, los cuales se enumeran a continuación:

- **VCC.** Tensión de salida para la alimentación del microcontrolador y del resto de componentes de la PCB, pudiendo ser de 3.3 V o 5 V.
- **5V.** Desemboca en el conector hembra correspondiente (en el bloque externo, ver apartado 5.4.).
- **3.3V.** Desemboca en el conector hembra correspondiente.
- **VIN.** Se trata de la tensión de entrada a través del jack. Desemboca en el conector hembra correspondiente.
- **GND.** Masa de la PCB, que debe ser la misma en todos los bloques. Así mismo, debe ir conectado a los terminales de masa de las entradas de tensión, es decir, del conector jack y del conector USB.
- **5V_USB.** Terminal procedente de los 5 V del puerto USB.

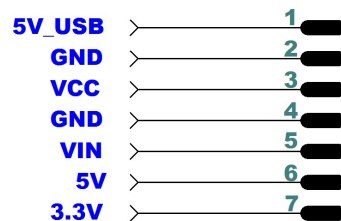


Figura 5.11. Bloque de alimentación: esquemático del marco

La PCB dispondrá de una *net* a 5 V y otra a 3.3 V (con salidas a conectores externos), del mismo modo que el Arduino UNO, lo cual obliga a incluir dos reguladores lineales con salidas de tensión iguales a dichos valores, respectivamente. Se aprovecha el disponer de dos *nets* con estos valores de tensión para tener la posibilidad de alimentar el microcontrolador a 5 V o a 3.3 V. Esto se decidirá externamente mediante un selector formado por tres pines macho, tal y como se observa en la parte derecha del esquemático (Figura 5.12.).

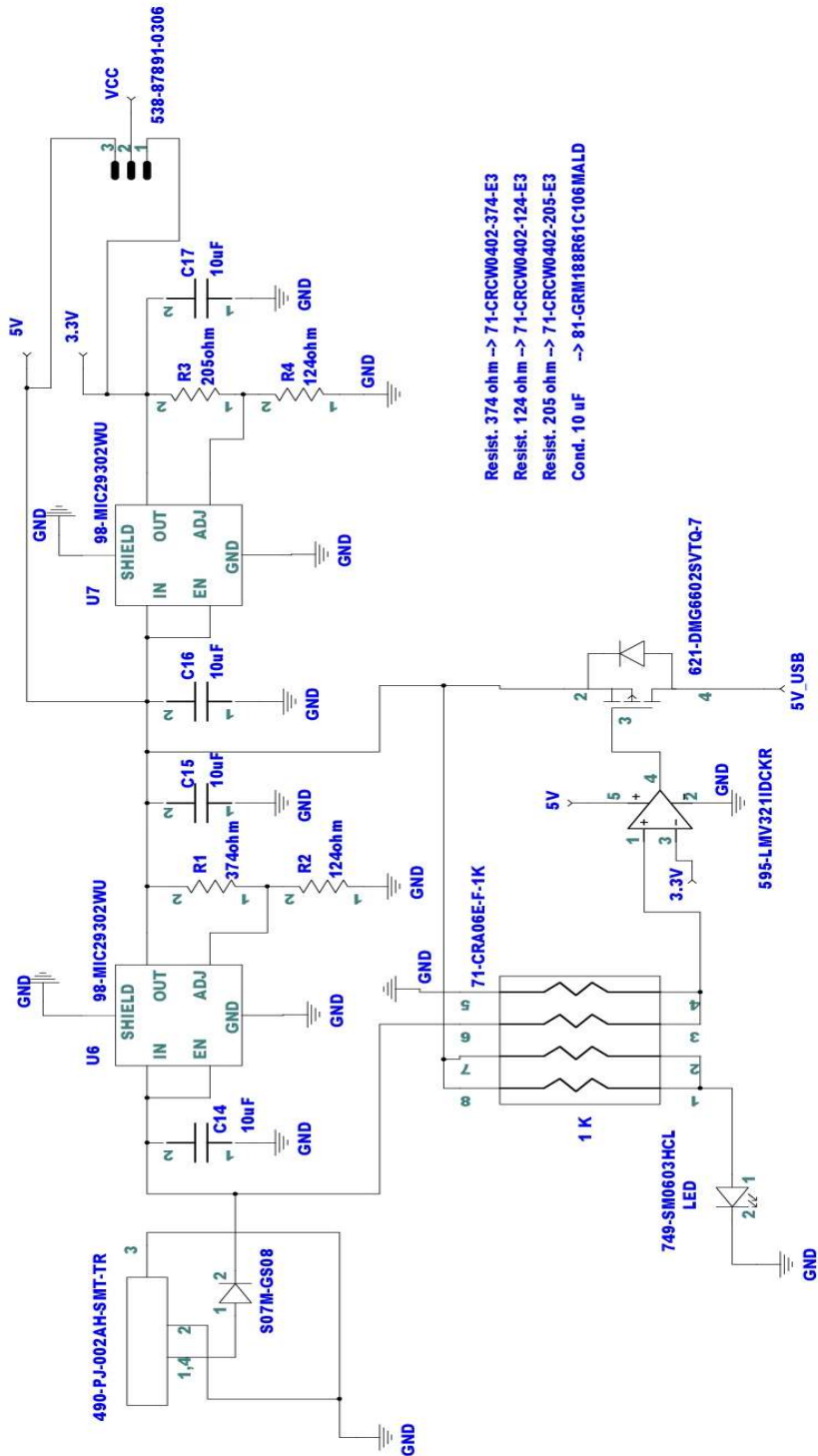


Figura 5.12. Bloque de alimentación: esquemático

A continuación, se exponen una a una las partes que conforman dicho circuito y sus características internas.

Entrada de potencia

Mediante un conector jack hembra (a la izquierda en la Figura 5.12.) se introduce potencia procedente de una fuente externa, la cual aporta una tensión continua. Esta fuente puede ser una batería o un convertidor AC/DC conectado a la red eléctrica.

Por otro lado, la tensión de entrada por este conector deberá encontrarse en un rango determinado, que no se debe rebasar ni superior ni inferiormente. Si es muy alta, se disipará mucha potencia a través del primer regulador lineal, ya que las pérdidas de este son directamente proporcionales a la diferencia de tensión que presenta entre la entrada y la salida. La salida de tensión en los reguladores ajustables está fijada, por lo que a mayor tensión a la entrada, mayores son las pérdidas. Si es muy baja, el sistema de protección ante la conexión al mismo tiempo de esta fuente y del puerto USB puede no funcionar correctamente, tal y como se verá más adelante. El valor de tensión más recomendable es de 9 V, evitando, sobre todo, un valor inferior.

En el diseño PCB se debe tener especial cuidado a la hora de asignar correctamente los pines de alimentación y masa en la huella correspondiente al conector jack, tal y como se muestra a continuación:

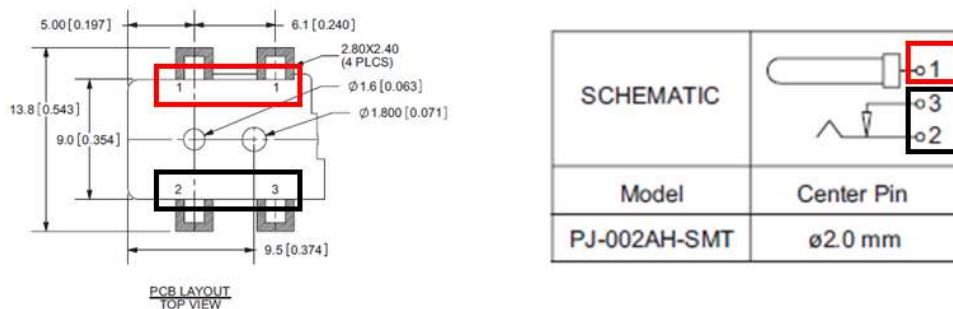


Figura 5.13. Terminales del conector jack hembra para la entrada de corriente

En cuanto al diodo en serie colocado a la entrada del terminal de alimentación, su objetivo es el de proteger al circuito ante posibles inversiones en la polaridad de la tensión de entrada o ante picos de corriente inversa debidos a cualquier defecto, bloqueando siempre esta corriente en el sentido hacia la fuente. No es la solución más eficiente, pero sí una de las más simples y baratas. Además, dado que en condiciones normales la placa no va a demandar una corriente excesiva, en este caso el criterio de la eficiencia pierde peso.

Se trata de un diodo de propósito general, con caída de tensión en directa de 1.1 V y corriente nominal en directa de 1.5 A. Para las corrientes de trabajo que va a demandar esta placa (normalmente, no llegará a 1 A en el peor de los casos) cumplirá su función y no disipará demasiada potencia

Reguladores lineales

Al requerir el diseño dos tensiones de funcionamiento (5 V y 3.3 V), se necesita recurrir a dos reguladores lineales de tensiones diferentes. Este tipo de reguladores suelen usarse en diseños cuyo consumo de potencia no supere los 10 W, como es el caso. Esto es porque cuando el consumo es bajo, se le da menor importancia a las pérdidas que tienen lugar en los convertidores. Para potencias mayores se suele optar por convertidores conmutados.

Se va a usar el mismo tipo de regulador para ambas tensiones objetivo. Se trata del regulador lineal ajustable MIC29302WU. Este tipo de reguladores incluye una patilla de ajuste (ADJ) que permite imponer una tensión determinada en la patilla de salida (OUT), valiéndose de dos resistencias de un valor concreto para lograr el valor buscado de tensión. A continuación se muestra un modelo muy simplificado de este tipo de reguladores, pero que permite demostrar en qué se basa su funcionamiento interno:

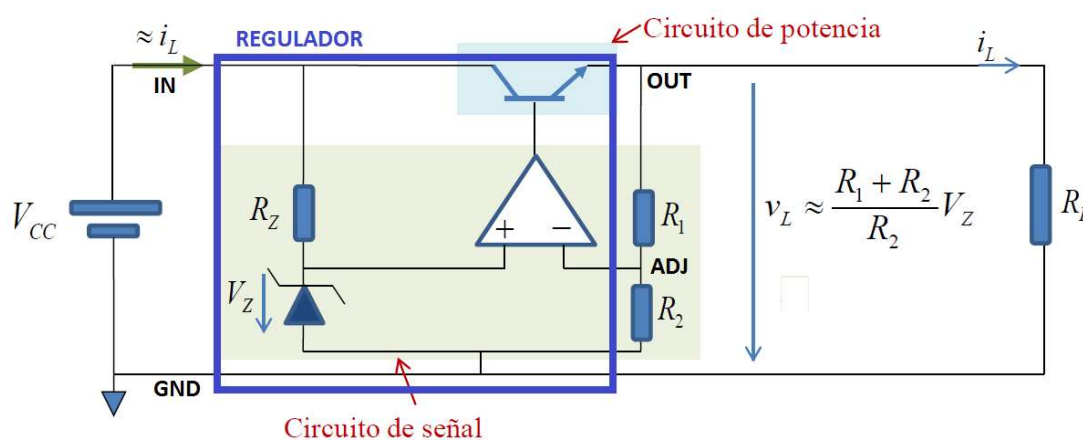


Figura 5.14. Modelo simplificado de un regulador lineal ajustable

Como se puede apreciar, la patilla ADJ está fijada a la tensión V_z . A partir de esta y de la conexión de las resistencias R_1 y R_2 , se consigue una tensión a la salida proporcional a V_z (mayor o igual que V_z). De este modo, imponiendo una tensión deseada a la salida V_{out} , la relación que deben guardar las resistencias es la siguiente:

$$R_1 = R_2 \cdot \left(\frac{V_{out}}{V_z} - 1 \right)$$

Volviendo al regulador en cuestión, a continuación se expone la relación de resistencias necesaria para imponer una tensión determinada a la salida, que, como se puede observar, es análoga a la fórmula teórica anterior:

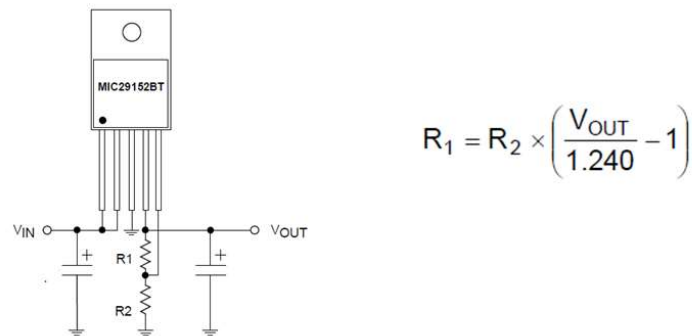


Figura 5.15. Reguladores de tensión: esquema de conexionado y relación de las resistencias para imponer una tensión de salida

No obstante, la relación anterior está idealizada, ya que considera nula la corriente de fuga por la patilla ADJ (I_{ADJ}). Si se incluye este término en los cálculos la relación es:

$$V_{out} = \left(\frac{R_1}{R_2} + 1 \right) \cdot 1,24 + R_1 \cdot I_{ADJ}$$

De esta fórmula se desprende que la tensión de salida está comprometida por el valor de las resistencias, cuyo valor no deberá ser demasiado alto para que el término dependiente de I_{ADJ} sea despreciable. Revisando la documentación, se recomienda que la corriente que circula por las resistencias R1 y R2 no sea inferior a 7 mA. En este diseño se impone un valor de 10 mA para esta corriente.

Se procede a calcular las resistencias R1, R2, R3 y R4 del diseño:

- **Regulador 9V – 5V:**

$$(1) \quad V_{out} = 5V \rightarrow R_1 = R_2 \cdot (5/1.24 - 1) = \rightarrow R_1 = 3.03 \cdot R_2$$

$$(2) \quad 5 / (R_1 + R_2) = 0,01 \text{ A} \rightarrow R_1 + R_2 = 500$$

$$(1), (2) \rightarrow \underline{R_1 = 376 \Omega, R_2 = 124 \Omega}$$

- **Regulador 5V – 3.3V:**

$$(3) \quad V_{out} = 3.3V \rightarrow R_3 = R_4 \cdot (3.3/1.24 - 1) = \rightarrow R_3 = 1.66 \cdot R_4$$

$$(4) \quad 3.3 / (R_3 + R_4) = 0,01 \text{ A} \rightarrow R_3 + R_4 = 330$$

$$(3), (4) \rightarrow \underline{R_3 = 206 \Omega, R_4 = 124 \Omega}$$

Para estos valores de resistencias, teniendo en cuenta que la potencia disipada es $P_d = \frac{V_{out}^2}{R}$, se puede ver fácilmente que las pérdidas a la salida de los reguladores son despreciables. Sin embargo, el circuito interno de los reguladores también da lugar a pérdidas, que, como ya se ha comentado, son directamente proporcionales a la diferencia de tensión entre la entrada y la salida, por lo que siempre que se pueda, se debe intentar reducir la tensión en la patilla de entrada. Un buen truco para ello, puede ser colocar una resistencia en serie previa a esta patilla. Para las potencias manejadas en este diseño, no se considera necesario.

LED indicador del encendido de la placa

Este LED indicador se debe encender cuando la placa se alimenta con una de las dos posibles fuentes externas, por lo que se conecta con la net de 5V, cuya tensión puede provenir de cualquiera de las dos.

Para encenderse, el LED utilizado necesita una corriente del orden de 10 mA y, en directa, provoca una caída de tensión de unos 2 V. Por tanto la resistencia a colocar en serie será aproximadamente:

$$(5-2) \text{ V} / 10 \text{ mA} = 300 \Omega$$

Se aprovechan para tal función dos resistencias en paralelo de un array de 1 k Ω , consiguiéndose una resistencia equivalente de 500 Ω .

Sistema de protección ante la conexión simultánea de dos entradas de potencia

Como se ha comentado anteriormente, la fuente de tensión de entrada puede provenir de una fuente externa conectada al jack o al puerto USB. En el primer caso, el circuito está preparado para una tensión de entrada de unos 9 V, por lo que se hace uso de un regulador lineal para su conversión a 5 V, como se explicó previamente. En el segundo caso, la entrada de tensión desemboca directamente (sin ningún acondicionamiento intermedio) a la salida de este regulador (ver Figura 5.12.) y se corresponde con el terminal de 5 V disponible en el bus USB. No obstante, este terminal no se conecta directamente a la net de 5 V, ya que, en el caso de alimentar la placa con ambas fuentes de entrada, las discrepancias existentes a la hora de aproximarse a 5 V (en ambos casos no se obtienen exactamente 5 V) darían lugar a sobrecorrientes que podrían dañar la placa.

Es por esto que se diseña un sistema para separar ambas nets en caso de su conexión simultánea, el cual se aprecia en la Figura 5.16.

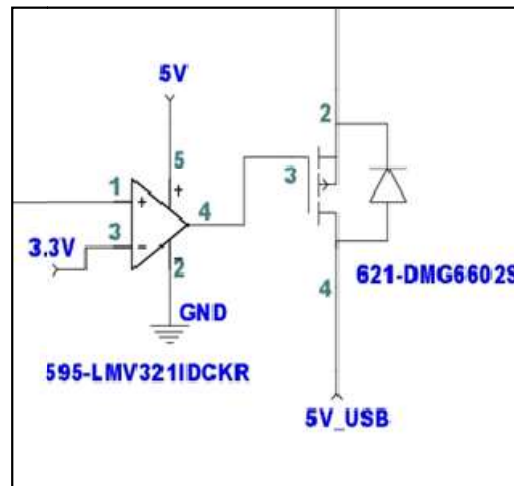


Figura 5.16. Conexión de AO y PMOS para la protección ante la conexión simultánea de dos entradas de potencia

El paso de corriente desde el bus USB se controla mediante un transistor PMOS que actúa como un interruptor.

Cuando entra tensión a través del jack, hay más tensión en la patilla no inversora que en la patilla inversora del Amplificador Operacional (AO), por lo que aparece un '1' lógico a la salida de este (tal y como están conectados los terminales de alimentación del AO, 5 V). Por tanto, al tener también 5 V en la fuente del PMOS, este no se polariza y se abre el circuito, cortándose el flujo procedente del puerto USB. Es importante resaltar que la entrada de tensión por el jack debe ser mayor o igual a unos 9 V para asegurar en la patilla no inversora una tensión mayor a 3.3 V (patilla inversora). En caso contrario el circuito no funcionaría correctamente tal y como se ha explicado.

Cuando no se introduce ninguna tensión por el jack, la patilla no inversora del AO se mantendrá a 0 V, teniendo un '0' (0 V) lógico a la salida. En este caso sí se polariza el PMOS y, por tanto, la placa sí puede alimentarse a través del puerto USB.

Para este propósito, se ha hecho uso de un Amplificador Operacional rail-to-rail con salida a 0 o a 5 V, tal y como se encuentra alimentado. El motivo es la no disponibilidad en la PCB de una *net* a -5 V, por lo que no se podría alimentar el AO con tensiones opuestas

Por tanto, se podrán dar los siguientes casos de conexión a una alimentación externa:

- Ninguna fuente externa conectada → Placa apagada.
- Conexión a puerto USB, jack desconectado → Placa alimentada a través del bus USB. PMOS polarizado.
- Conexión al jack, puerto USB desconectado → Placa alimentada a través del jack. PMOS en corte.

- Conexión al jack y al puerto USB → Placa alimentada a través del jack. Se corta el flujo de corriente procedente del puerto USB, PMOS en corte.

Cabe decir que no se han contemplado los casos en que la placa se alimenta a través de otros medios, como conectando 5 V o 3.3 V a los conectores correspondientes. Esto es porque en estos casos no funcionaría el sistema de protección expuesto, dado que no se dispondría de los 9 V que necesita el AO (sí se dispondrían en el caso de alimentación a través del terminal VIN, pero ya se ha dicho que no se recomienda por perder la protección ante inversiones de polaridad). Por tanto, sólo se recomienda alimentar la placa según los dos casos contemplados: alimentación a través del conector jack o a través del puerto USB.

Selector externo de la tensión de alimentación de la placa (VCC)

Este selector se basa en tres pines macho, de los cuales el central se conecta a la *net* de alimentación de la placa ('VCC') y los extremos se conectan a 5 V y a 3.3 V, respectivamente. Como se observa en la imagen, la forma de seleccionar externamente la tensión que alimentará el microcontrolador es cortocircuitar mediante un jumper el pin VCC con 5 V o con 3.3 V, según se desee.

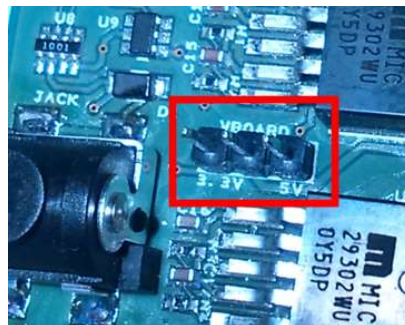


Figura 5.17. Selector de tensión de alimentación (5V o 3.3V) en el módulo

5.2.2. Diseño PCB

Se tienen en cuenta los siguientes puntos a la hora de plantear el diseño del presente bloque del circuito impreso:

- El conector jack se debe colocar en el borde de la placa para no interferir con otros componentes a la hora de acoplarle el conector macho.
- Los reguladores lineales, junto con el conector jack, son los componentes que más espacio ocupan en la PCB, por lo que son los primeros que se deben posicionar y los que determinan el *layout* del resto de componentes.
- Siguiendo las consignas del apartado 3.1.1., se procura, en la medida de lo posible, situar los componentes en función del flujo de la señal.
- Los condensadores a la entrada y salida de los reguladores se sitúan lo más cerca posible del patillaje correspondiente para no introducir impedancias parásitas.

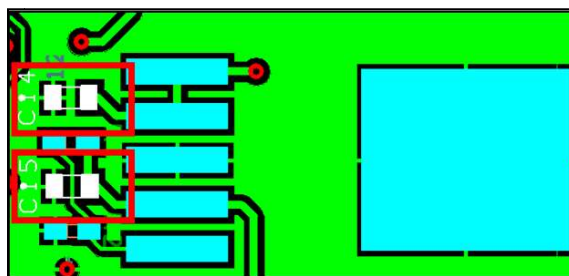


Figura 5.18. Layout de los condensadores y resistencias externos del selector

- Los reguladores lineales suelen tener, como es el caso, un terminal consistente en una lengüeta metalizada, cuya misión es evacuar el exceso de calor generado internamente debido a la potencia disipada. Este terminal debe conectarse a un pequeño plano de cobre en la propia PCB para disipar mejor el calor, el cual se incluye como otro pin más del componente a la hora de crear su *footprint*.

Estos planos se deben conectar a masa, siguiendo las recomendaciones del fabricante (el terminal de disipación está conectado a la masa del circuito interno del regulador). De este modo, se favorece una mejor evacuación del calor producido en los reguladores, evitando que estos alcancen Temperaturas demasiado elevadas, sobre todo ante corrientes altas.

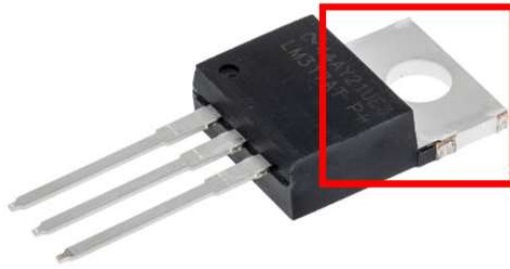


Figura 5.19. Lengüeta disipadora de calor en un regulador lineal

- Los terminales del marco se disponen en función de la dirección que deben tomar las pistas correspondientes. El bloque de alimentación se prevé que se encuentre bajo el bloque USB, según la vista en planta de la capa TOP (ver Figura 6.1.). Teniendo en cuenta esto, por ejemplo, la entrada de 5 V desde el puerto USB (5V_USB), se encuentra en el borde superior del marco.

A continuación, se exponen las vistas de la PCB que definen este bloque.

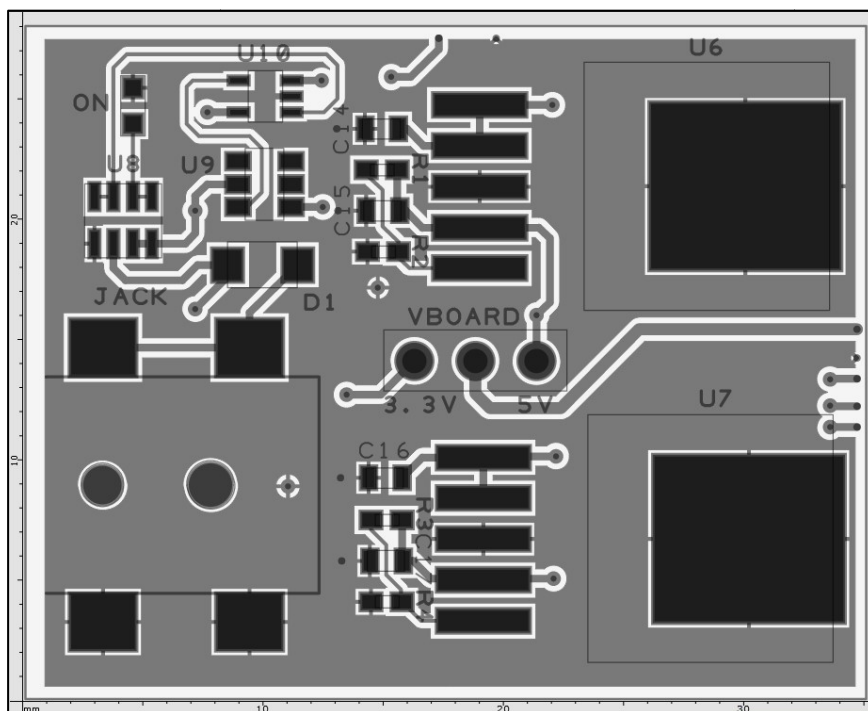


Figura 5.20. Bloque de alimentación, vista PCB: cara TOP

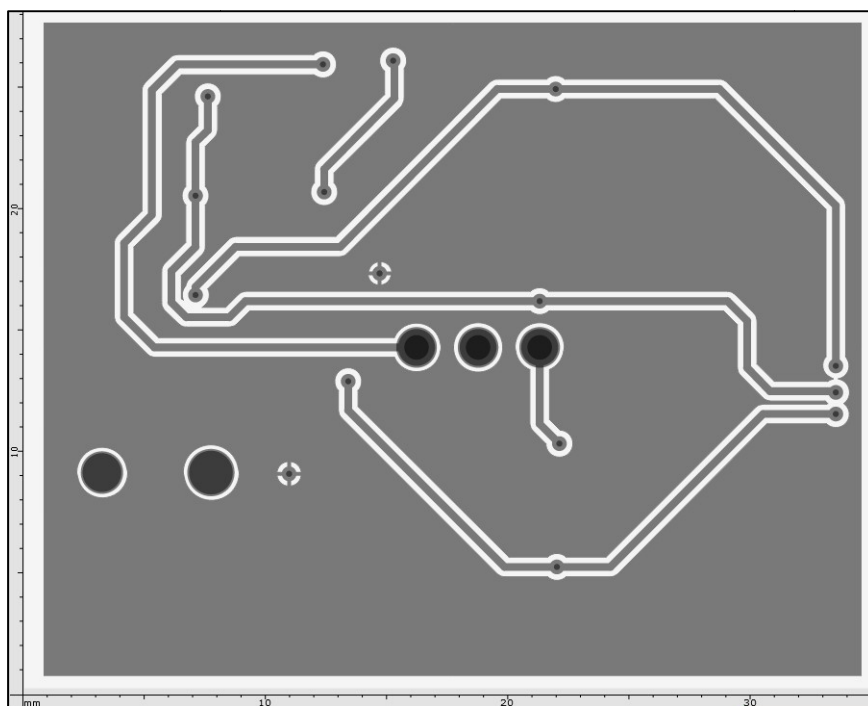


Figura 5.21. Bloque de alimentación, vista PCB: cara BOTTOM

5.3. Bloque USB

Con este bloque se pretende dar al módulo la posibilidad de comunicarse en tiempo real con otros dispositivos, como puede ser una computadora, a través de un puerto de comunicación en serie. Se opta por un protocolo de comunicación tan extendido como es el USB, el cual, por otro lado, proporciona una línea adicional de alimentación a 5 V. Por otro lado, la gran mayoría de placas computadora de características similares incorporan un conector USB y no se deseaba que esta fuera menos.

El USB (*Universal Serial Bus* o Bus Universal en Serie) es un tipo de bus industrial estandarizado con una serie de cables, conectores y protocolos definidos. Se utiliza tanto para comunicar, como para proporcionar alimentación a diferentes dispositivos electrónicos, normalmente computadoras con periféricos, como pueden ser teclados, ratones, impresoras, memorias USB, teléfonos móviles... El medio físico entre dispositivos es un cable de 4 hilos apantallados: uno de alimentación (5 V), otro de masa (GND) y un par trenzado de transmisión diferencial (D+ y D-).



Figura 5.22. Cable USB cortado

Además, se opta por incluir dentro de este bloque un pulsador de RESET, mediante el cual se puede reiniciar el programa cargado en el microcontrolador. Normalmente, todas las placas de estas características disponen de uno, o, al menos, deben incluir un medio externo de resetear el sistema.

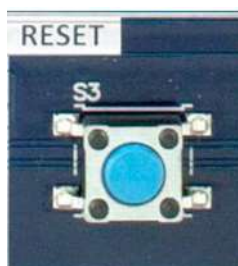


Figura 5.23. Pulsador de RESET

5.3.1. Diseño del esquemático

Se muestra en la Figura 5.24. el esquemático del presente bloque.

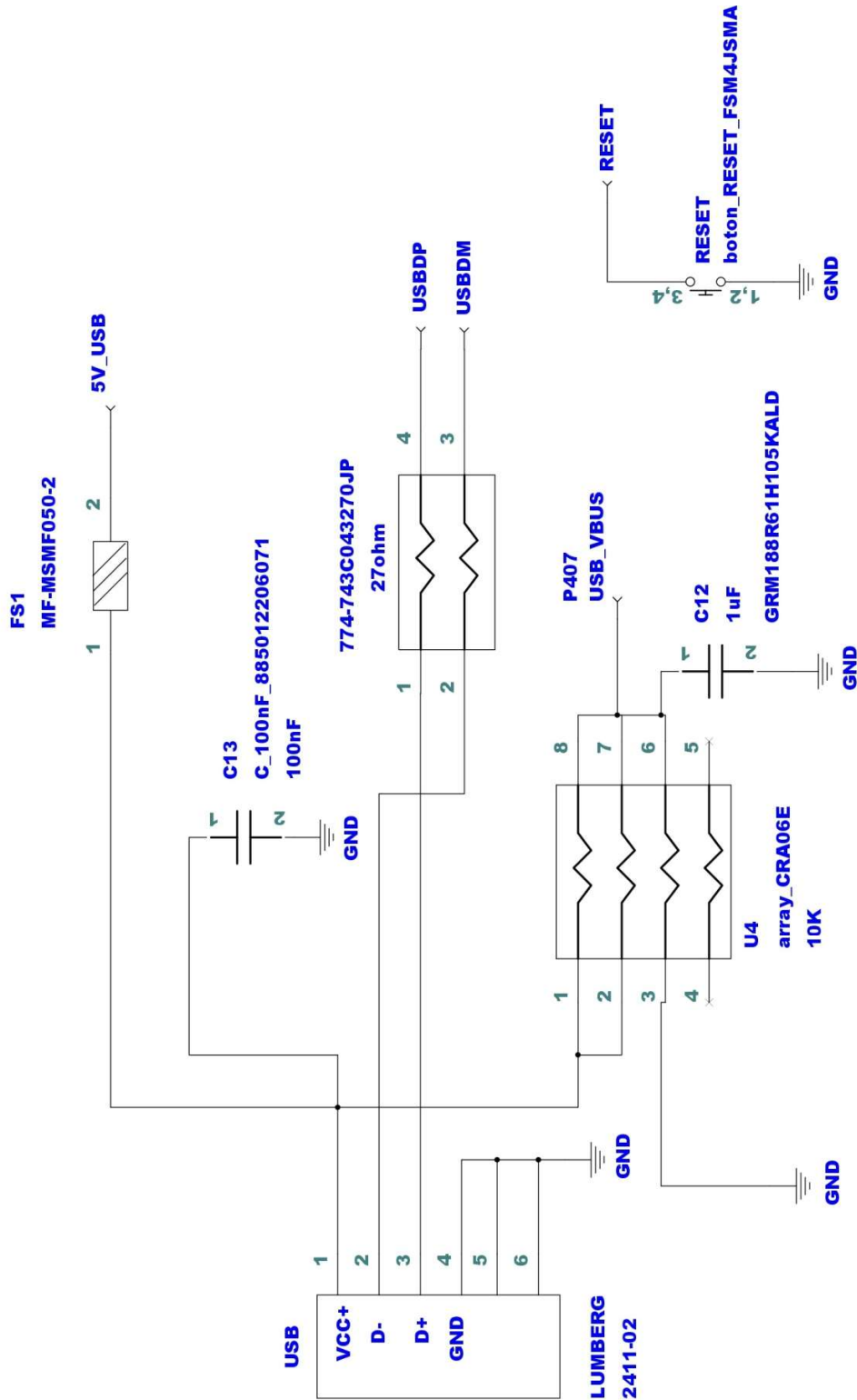


Figura 5.24. Bloque USB: esquemático

Conector USB

El puerto USB de entrada a la placa es un conector hembra de tipo B, el cual se puede ver en la Figura 5.25. .



Figura 5.25. Conector USB tipo B

En la Figura 5.26. se muestra la numeración de los pines de dicho conector y su correspondencia con cada uno de los cuatro hilos del bus USB.

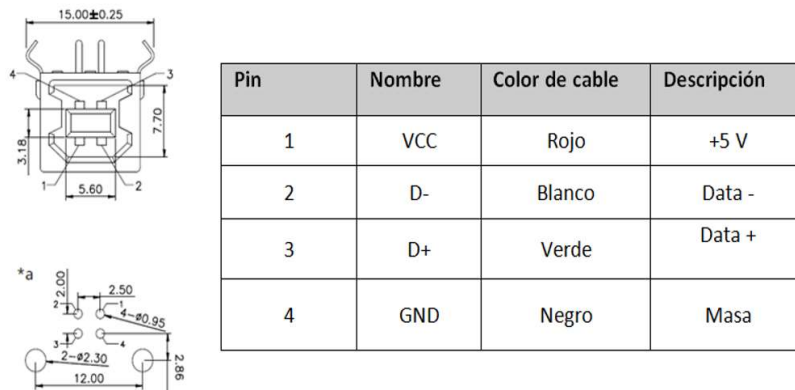


Figura 5.26. Asignación de pines en el conector USB

Por otro lado, las dos patillas laterales unidas a la carcasa del conector se conectan a masa.

Comunicación USB (Líneas de datos D+ y D-)

La comunicación USB es diferencial, lo que quiere decir que la señal digital que se codificará viene determinada por la diferencia de tensión entre los conductores D+ y D-, al contrario que en otros protocolos donde la señal es la de un sólo conductor referenciado a una tensión fija, normalmente masa. La principal ventaja de la transmisión diferencial es que es más robusta ante interferencias, ya que estas afectarán a ambos conductores en la misma medida,

manteniéndose constante la diferencia de tensión que determina el valor digital, tal y como se observa en la Figura 5.27. . Debido a esto, en circuitos impresos se deben disponer las pistas D+ y D- lo más paralelas y juntas que se pueda, de tal modo que el efecto de las interferencias sea similar en ambas.

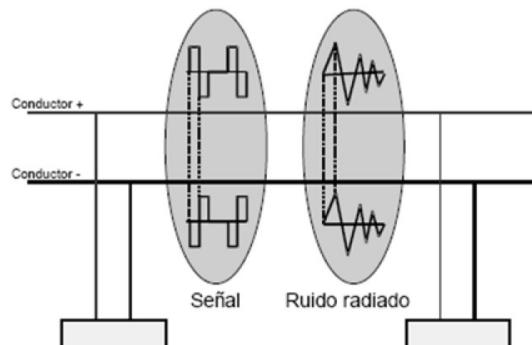


Figura 5.27. Compensación del efecto del ruido en líneas de transmisión diferencial

En concreto el bus USB utiliza transmisión NRZI (non-return to Zero, invert on ones), un tipo de transmisión polar basada en los flancos: ante cambios en la diferencia de tensión se detectan '1s' y cuando esta se mantiene con el ciclo de reloj se detectan '0s'. Así se aprecia en la Figura 5.28. .

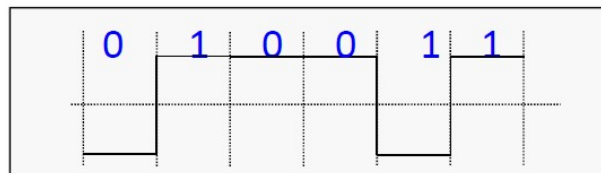


Figura 5.28. Transmisión Non-Return to Zero, Invert on Ones (NRZI)

Por tanto, de acuerdo a los párrafos anteriores, se utilizan dos líneas de transmisión de datos: D+ (USBDP) y D- (USBDM). Para que presenten la robustez ante interferencias deseada, estas líneas de tipo diferencial deben cumplir dos requerimientos principales:

- Han de seguir un trazado lo más similar posible, por lo que se trazan juntas y paralelas.
- Deben presentar una impedancia de línea parecida.

En el apartado 5.3.2. se muestra como se tienen en cuenta estos puntos en el diseño del circuito impreso. En cuanto a conseguir una impedancia de línea parecida en USBDP y USBDM, las resistencias de 27 ohmios en serie en estas líneas (ver Figura 5.24.) contribuyen a ello.

Alimentación a través del puerto USB

Con frecuencia, resulta de gran utilidad la disponibilidad de un puerto USB para la entrada de potencia, ya que, actualmente, es un tipo de conexionado altamente usado y puede no disponerse de otros tipos de adaptadores. Por esa razón, se decide disponer el hardware necesario para posibilitar la entrada de potencia a través de este puerto.

Esto se consigue gracias al terminal de 5 V del bus USB, tensión que puede ser perfectamente utilizada por los circuitos digitales del módulo. Cabe mencionar que la entrada de potencia a través de este terminal presenta una limitación en corriente. En concreto las versiones USB 3.0 aportan hasta 1000 mA, mientras que las versiones previas suelen aportar entre 500 mA y 900 mA. Las computadoras modernas suelen incorporar puertos USB 2.0 y 3.0.

En cualquier caso, a modo de protección ante sobrecorrientes se decide colocar un fusible de restablecimiento en serie a la entrada de potencia por el terminal de 5 V del puerto USB. Un fusible de restablecimiento es, simplemente, un termistor PTC¹⁴, es decir, una resistencia que aumenta su valor con la Temperatura. Así, ante corrientes altas, la subida de Temperatura provocará un aumento en el valor de la resistencia del componente. Por tanto, la función principal de este componente es disipar el exceso de potencia ante sobrecorrientes indeseadas. Las PTC no presentan, ni mucho menos, un crecimiento lineal con la Temperatura, propiedad que se explota en estos componentes de protección para conseguir un aumento abrupto de la resistencia ante un valor concreto de corriente.

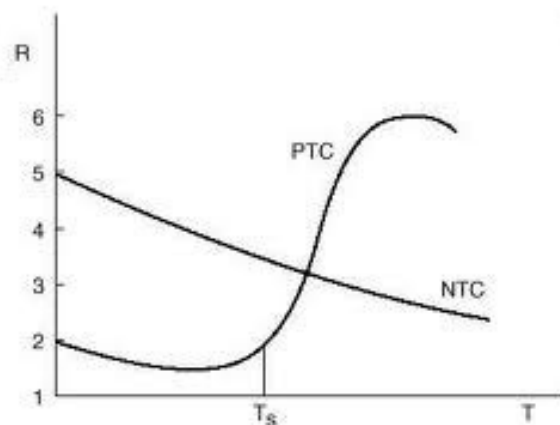


Figura 5.29. Función del valor de la resistencia en función de la Temperatura para un termistor NTC y un termistor PTC

¹⁴ PTC: Positive Temperature Coefficient

Pulsador RESET

Se decide colocar en este bloque del módulo el pulsador encargado de resetear el sistema al accionarlo. Se trata, simplemente, de un pulsador que cortocircuita la *net* RESET, proveniente del correspondiente pin del microcontrolador, con masa (GND). Al detectar un '0' lógico dicho terminal, se reinicia o resetea el firmware. Se recuerda que el bloque del microcontrolador ya incluye el filtro RESET, es decir, un condensador supresor de rebotes y una resistencia pull-up.

Se debe disponer el enrutado según la asignación de pines de la Figura 5.30., teniendo en cuenta que los pines 1 y 2 se encuentran cortocircuitados, al igual que los pines 3 y 4.

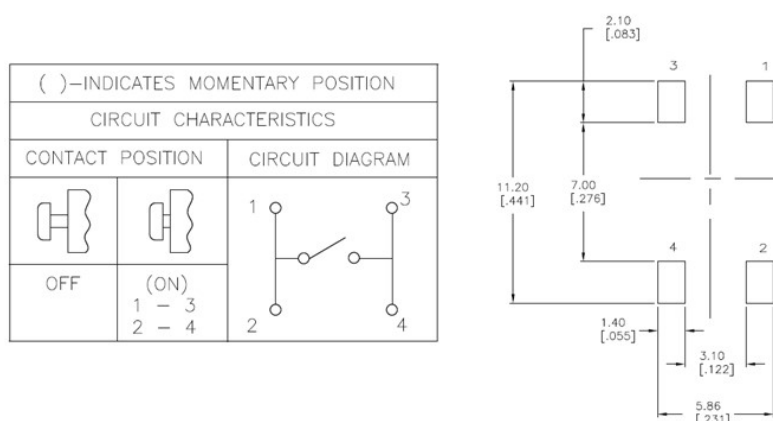


Figura 5.30. Asignación de pines para el pulsador RESET

Marco de conexiones

Se enumeran las *nets* a las que se debe dar entrada o salida en este bloque:

- **GND y 5V_USB.** La masa (GND) de la entrada de alimentación del puerto USB se debe referenciar a la misma *net* que la masa del resto de la placa. Se da salida también a la *net* 5V_USB, que se corresponde con la línea de 5 V del bus USB, para su acondicionamiento en el bloque de alimentación, como se explicó con anterioridad.
- **P407.** Esta *net* se corresponde con el pin de alimentación del módulo de comunicación USB del microcontrolador. Debe aportarse en este pin una tensión de 3.3 V, para lo cual se hace uso de un divisor de resistencias, logrando los 3.3 V a partir de los 5 del bus USB. Para este divisor de resistencias se utiliza un array de cuatro resistencias de 10 KΩ, tal y como se puede ver en la Figura 5.24. .
- **RESET.** Este terminal proviene del pin de RESET del microcontrolador. El pulsador aporta 0 V en este pin cada vez que se acciona, al cortocircuitarlo con masa.
- **USBDP y USBDM.** Se trata de las dos líneas de datos del bus USB, que deben dirigirse a los pines correspondientes del microcontrolador.

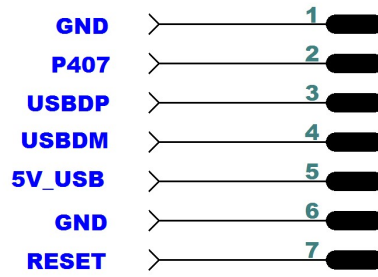


Figura 5.31. Bloque USB: esquemático del marco

5.3.2. Diseño PCB

Este bloque PCB tiene unas dimensiones aproximadas de 2 x 2.5 cm y se sitúa en la esquina superior izquierda en la vista en planta de la cara TOP. Se exponen a continuación las consideraciones que se tienen en el diseño PCB:

- Los componentes más grandes, como el conector USB o el pulsador de RESET, condicionan el *layout* del resto de componentes en este caso. El conector USB se posiciona en el borde izquierdo para facilitar su uso y el pulsador queda en la esquina superior izquierda de la placa. Las posiciones de estos componentes son casi las mismas que en el Arduino UNO.
- Los terminales del marco se sitúan en función de la dirección que deben tomar las respectivas pistas en su camino al resto de bloques, por ejemplo, al situarse el bloque de potencia bajo este bloque, la *net* correspondiente a los 5 V del bus USB (5V_USB) se sitúa en un terminal del marco de la parte inferior.
- Las pistas USBDP y USBDM, como ya se comentó, se trazan juntas y paralelas, igual que en el resto de la PCB.

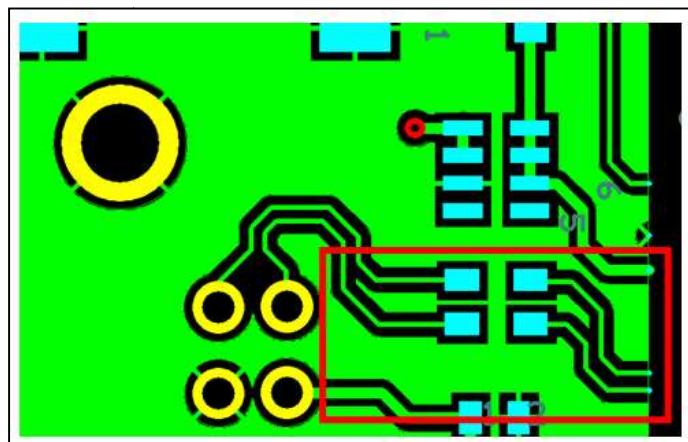


Figura 5.32. Trazado de las líneas USBDM y USBDP

El resultado final del diseño PCB, en lo que respecta a este apartado, se muestra en la siguiente página.

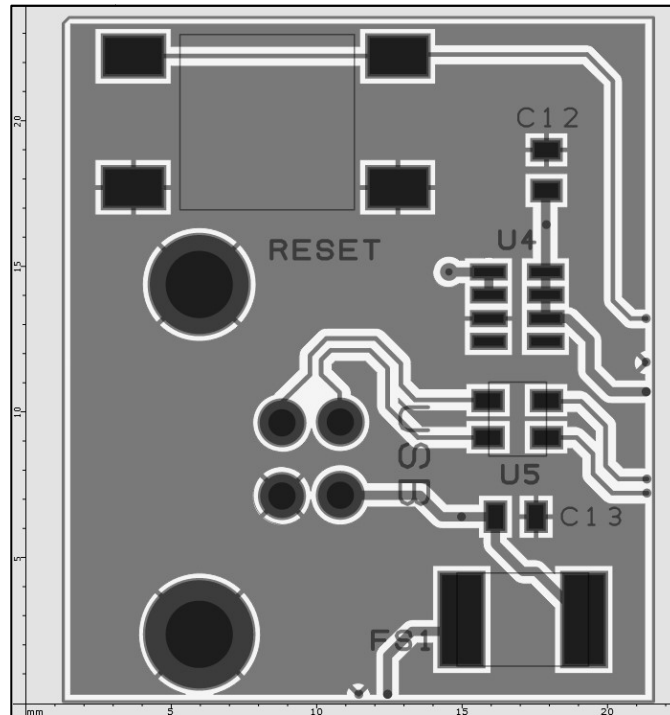


Figura 5.33. Bloque USB, vista PCB: cara TOP

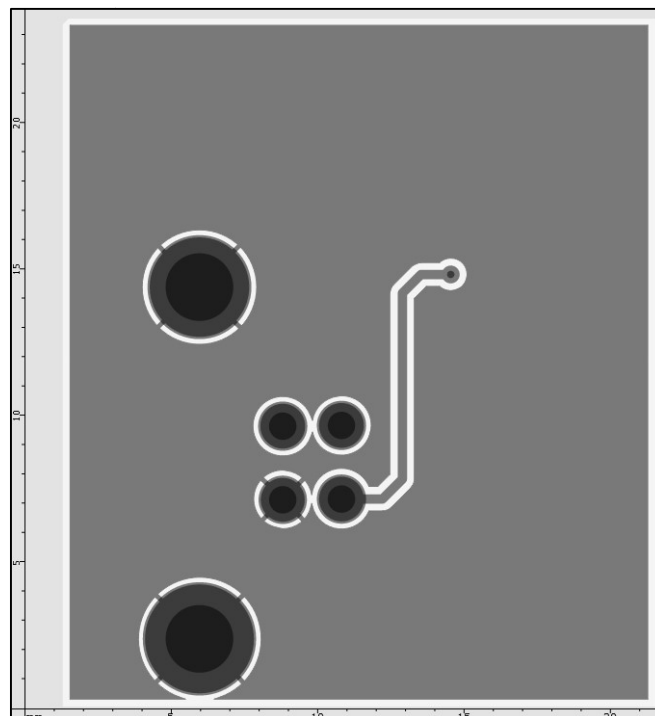


Figura 5.34. Bloque USB, vista PCB: cara BOTTOM

5.4. Bloque externo unificador

El objetivo de este bloque es el de unificar todos los bloques en la PCB global. De este modo, además de darle la forma final a la placa al incluir los bordes de esta, debe encajar a la perfección con los tres bloques restantes, conectando debidamente los terminales de todos los marcos. Es decir, es la pieza integradora de todos los bloques que conforman el módulo.

Por otro lado, siguiendo con el propósito de imitar los diseños de Arduino, debe contener las hileras de conectores hembra en la misma disposición que en las placas que se mencionó con anterioridad. Además de estos conectores, el único componente que incluirá será un array de resistencias de 10 K Ω , que se usará como resistencias pull-up para las líneas SDA y SCL de los buses I2C.

5.4.1. Diseño del esquemático

En la Figura 5.36. se puede ver el esquemático que muestra la conexión de los pines del microcontrolador a los conectores externos. El símbolo esquemático de los conectores es análogo a su huella PCB para que quede clara la disposición de las conexiones de las respectivas *nets*. Los 32 terminales desde el 1 al 32 presentan la misma localización que los conectores hembra del Arduino UNO, el Arduino Mega o el Arduino LEONARDO, de tal modo que permiten el acoplamiento de *shields* diseñadas para dichos módulos, tal y como se mencionó anteriormente.

Arduino Uno R3 Pinout

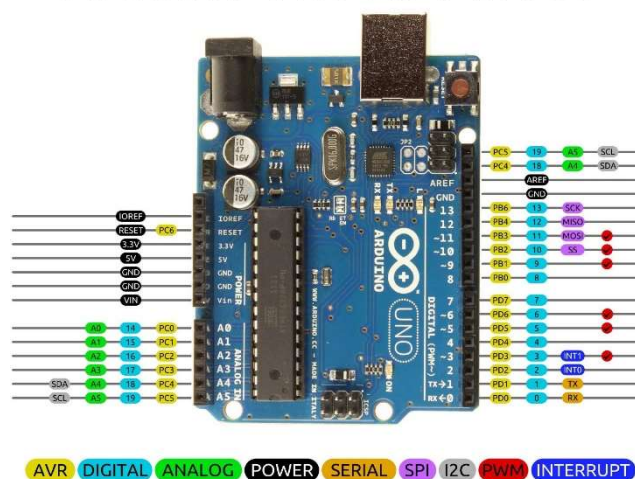


Figura 5.35. Asignación de pines a los conectores del Arduino UNO

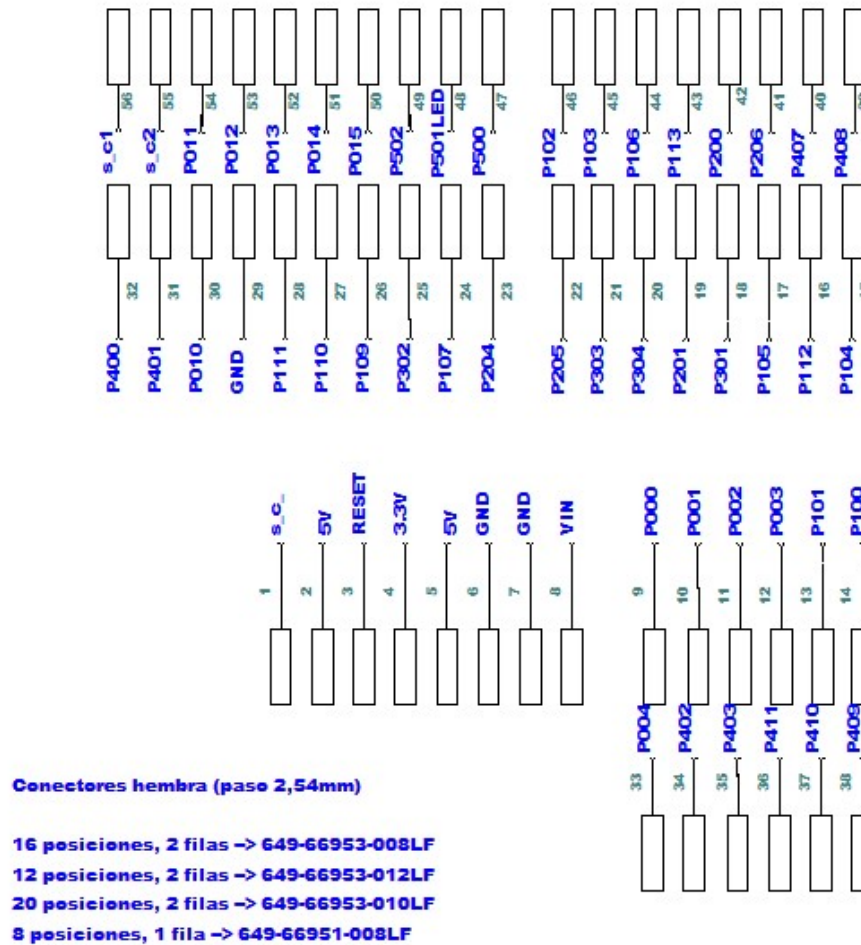


Figura 5.36. Bloque externo unificador: esquemático de conectores externos

Asignación de pines

La elección de la asignación de cada uno de los 32 conectores mencionados (del 1 al 32) a los pines del microcontrolador no es aleatoria, ya que, para conseguir la compatibilidad buscada con la electrónica de Arduino, se requiere que los pines análogos (en la misma posición análoga al Arduino UNO) de la placa a desarrollar aporten la misma funcionalidad. Por ejemplo, al ser los pines A0-A5 del Arduino UNO entradas analógicas, los conectores correspondientes de la placa a diseñar (del 9 al 14) se deben conectar a entradas analógicas del R7FS124773A01CFM.

El microcontrolador que se estudia para realizar la comparación es el Atmega168p, ya que es el que incorpora la placa Arduino UNO, placa que se tomará como referencia de aquí en adelante. En la Figura 5.37. se observa el esquema con la numeración de los pines, así como las funcionalidades de cada pin y su correspondencia con los conectores de la placa.

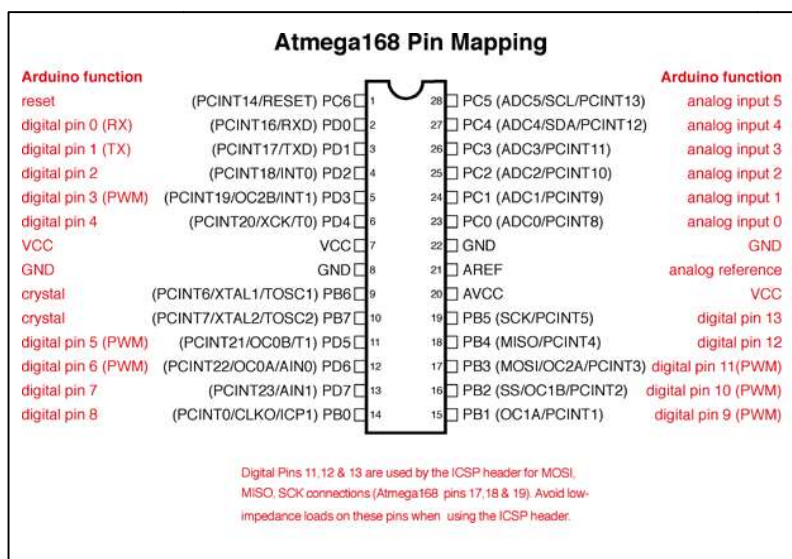


Figura 5.37. Asignación de pines en Atmega 168p, incluido en el Arduino UNO

Por tanto, para conseguir la asignación adecuada en cada conector habrá que fijarse, para cada posición, con qué pin del Atmega168p se corresponde, por un lado, y qué pines del microcontrolador R7FS124773A01CFM son compatibles con el anterior en cuanto a funcionalidad, por otro. Se exponen las funcionalidades principales de los pines del Arduino UNO que se tienen que tener en cuenta:

- **D0 (RX) y D1 (TX).** Comunicación serie.
- **D2 (INT0) y D3 (INT1).** Interrupciones externas.
- **D3, D5, D6, D9, D10 y D11.** Modulación por ancho de pulso (PWM).
- **D10 (SS), D11 (MOSI), D12 (MISO) y D13 (SCK).** Comunicación SPI.
- **A4 (SDA) y A5 (SCL).** Comunicación I2C.

Siguiendo este criterio, en la Tabla 5.1. se muestra la asignación de pines a cada uno de los 32 conectores, enfrentando las características de cada conector del Arduino UNO con las de los respectivos pines elegidos en el R7FS124773A01CFM, asegurando así, al menos, las mismas funcionalidades.

Pines Arduino	Pines S-124	Funcionalidad Arduino	Funcionalidad S124
1		-	
2		5V	
3	25	RES	RESET
4		3.3V	
5		5V	
6		GND	
7		GND	
8		VIN	
9	64	A0 Entrada analógica	P000 Entrada analógica : AN000
10	63	A1 Entrada analógica	P001 Entrada analógica : AN001
11	62	A2 Entrada analógica	Entrada analógica : AN002
12	61	A3 Entrada analógica	P003 Entrada analógica : AN003
13	47	A4 Entrada analógica TWI: SDA	P101 ¹⁵ Entrada analógica : AN021 I2C: SDA1_B
14	48	A5 Entrada analógica TWI: SCL	P100 ¹⁵ Entrada analógica : AN022 I2C: SCL1_B
15	44	D0 Serial: RX	P104 SCI: RXD0_C
16	37	D1 Serial: TX	P112 SCI: TXD0_C
17	43	D2 Interrupción externa	P105 Int.: IRQ0
18	31	D3 Interrupción externa PWM	P301 IRQ6 GTIOC4B_A
19	27	D4	P200
20	28	D5 PWM	P304 PWM: GTIOC1A_B
21	29	D6 PWM	P303 PWM: GTIOC1B_B
22	23	D7	P205
23	24	D8	P204
24	41	D9 PWM	P107 PWM: GTIOC0A_B
25	30	D10 PWM SPI: SS	P302 PWM: GTIOC4A_A SPI: SSLB3_B
26	34	D11 PWM SPI: MOSI	P109 PWM: GTIOC1A_A SPI: MOSIB_B
27	35	D12 SPI: MISO	P110 PWM: GTIOC1B_A

¹⁵ Requieren resistencias pull-up externas (bus I2C)

28	36	D13 SPI: SCK	SPI: MISOB_B P111 PWM: GTIOC3A_A SPI: RSPCKB_B
29		GND	
30	59	AREF	P010 VREFH
31	2	SDA	P401 ¹⁵ SPI: SDA0_A
32	1	SCL	P400 ¹⁵ SPI: SCLO_A

Tabla 5.1. Comparación de pines del Arduino UNO con los del R7FS124773A01CFM y asignación de pines de este último a las hileras de conectores internos

En la Figura 5.38. se resumen los pines del microcontrolador R7FS124773A01CFM que se indican en la tabla anterior.

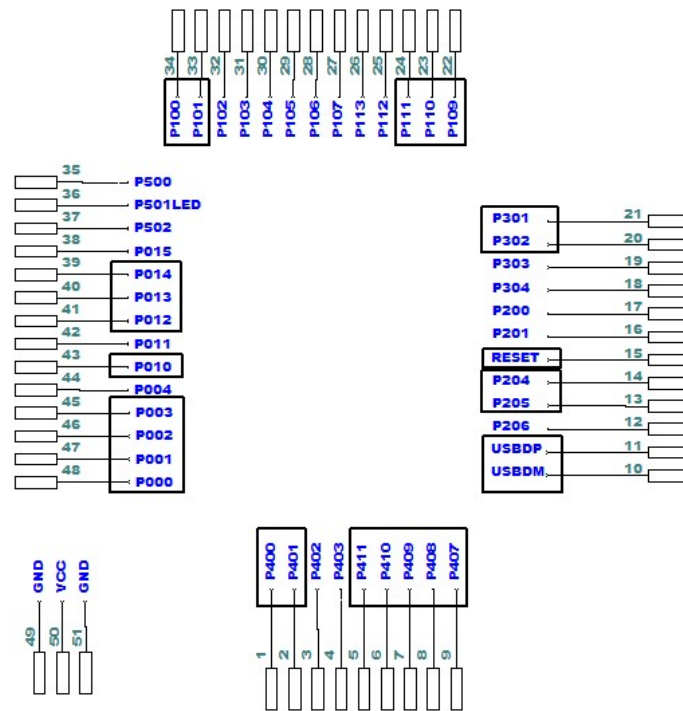


Figura 5.38. Pines del microcontrolador a las hileras de conectores internas para la conexión de shields (los recuadrados)

Resistencias pull-up

Por las características internas del bus, el puerto I2C requiere del uso de resistencias pull-up externas. Estas se incluyen externamente al microcontrolador dentro de este bloque. La Figura 5.39. muestra el esquemático de dichas resistencias, donde se ven los pines correspondientes al puerto I2C en el R7FS124773A01CFM. Al tratarse de cuatro terminales, se hace uso de un array de 4 resistencias de 10 K Ω .

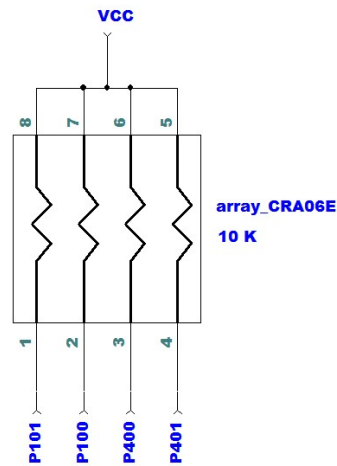


Figura 5.39. Bloque externo unificador: esquemático resistencias pull-up

5.4.2. Diseño PCB

En las imágenes del diseño PCB de este bloque, incluidas más adelante, se ve como este tiene la forma perfecta para encajar con los tres restantes, quedando enfrentados los terminales que corresponden en cada caso. Para conseguir esto, se incluyen los marcos de los otros bloques en el proyecto de *DesignSpark* de este, de tal modo que en el *layout* sólo queda situar estos donde se crea conveniente, como si se tratara de tres componentes más.

Durante el diseño de este circuito impreso se han tenido en consideración los siguientes puntos:

- Para elegir la ubicación del bloque del microcontrolador se procura que las pistas que unen sus terminales con los conectores externos sean lo más cortas posibles. Así, este bloque queda en una posición central de la placa. Por otro lado, también se elige la orientación intentando reducir la longitud de las pistas.
- Los bloques de alimentación y del conector USB quedan en el extremo izquierdo, como se ve en las imágenes posteriores. De este modo, tanto el conector de entrada de potencia, como el puerto USB, se sitúan casi en la misma posición, análogamente, que en la placa Arduino UNO.
- Una parte fundamental en este diseño, como ya se ha dicho, es disponer las filas de conectores hembra de tal modo que encajen perfectamente con las *shields* que proporciona Arduino. Para ello, se crea en una sola huella los *pads* donde se sueldan todos los conectores, imponiendo las mismas dimensiones que presentan en el Arduino UNO, las cuales se pueden ver en la Figura 5.40. .

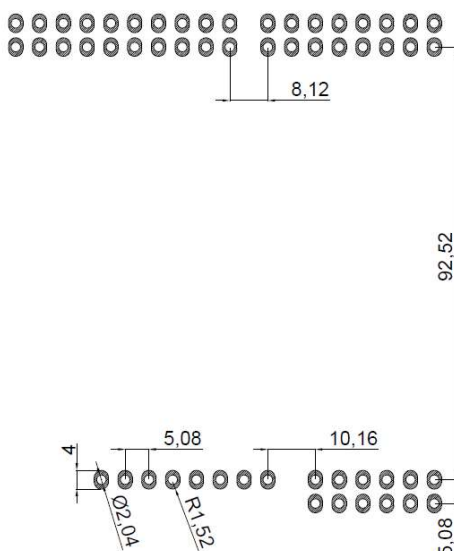


Figura 5.40. Huella PCB de los conectores. Las dimensiones coinciden con las de los conectores en las placas Arduino UNO, LEONARDO Y MEGA

- Al haber en este diseño un gran número de pistas en ambas caras, se tiene especial cuidado intentando trazar las pistas ortogonales en la capa TOP respecto a la BOTTOM.

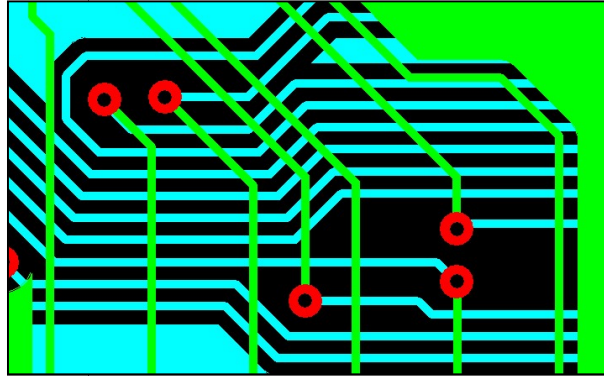


Figura 5.41. Ortogonalidad de pistas en las caras TOP y BOTTOM

- Las pistas USBDP y USBDM se mantienen juntas y paralelas en su recorrido, de igual modo que en el resto de bloques.

En las páginas que siguen se exponen todas las capas que conforman el diseño PCB de este bloque.

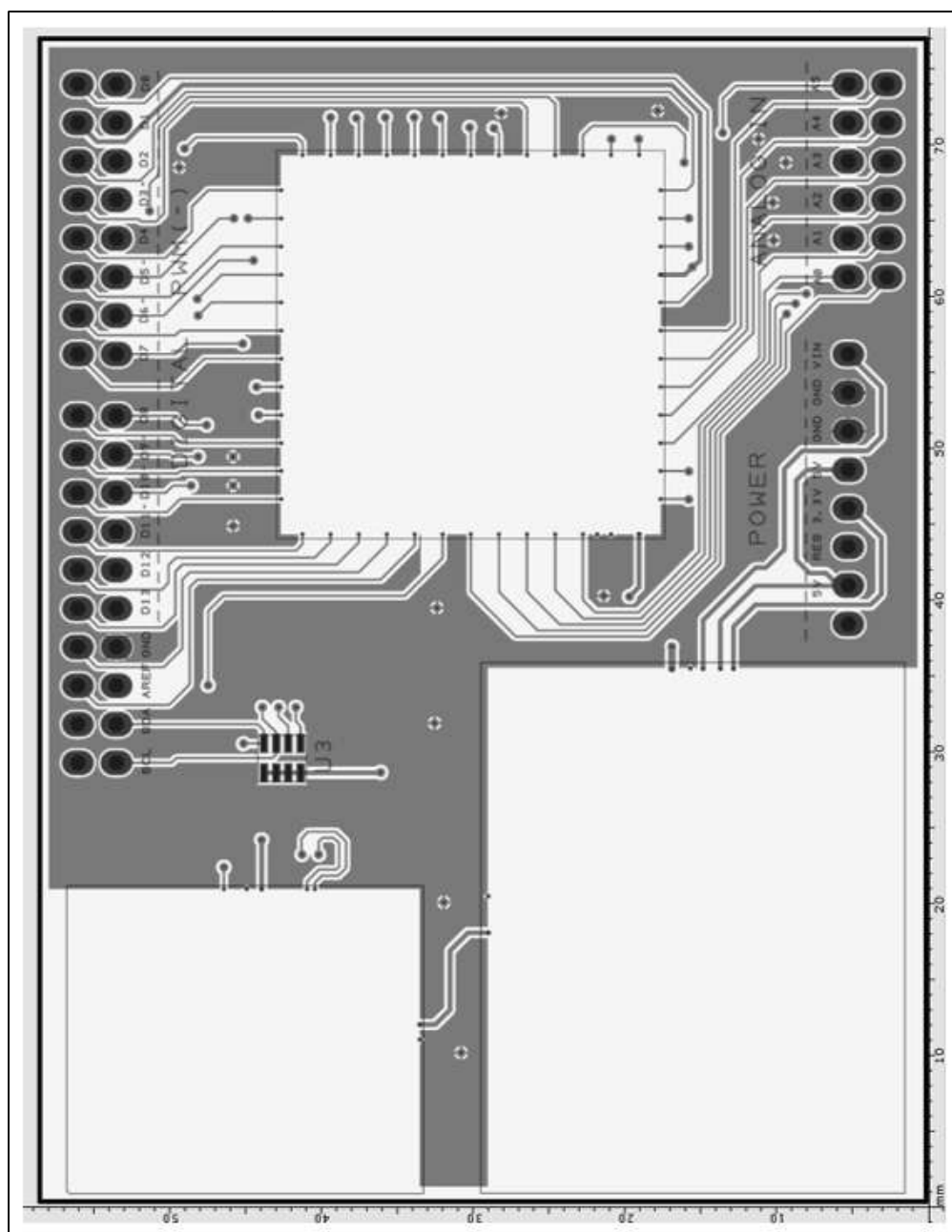


Figura 5.42. Bloque externo unificador, vista PCB: cara TOP

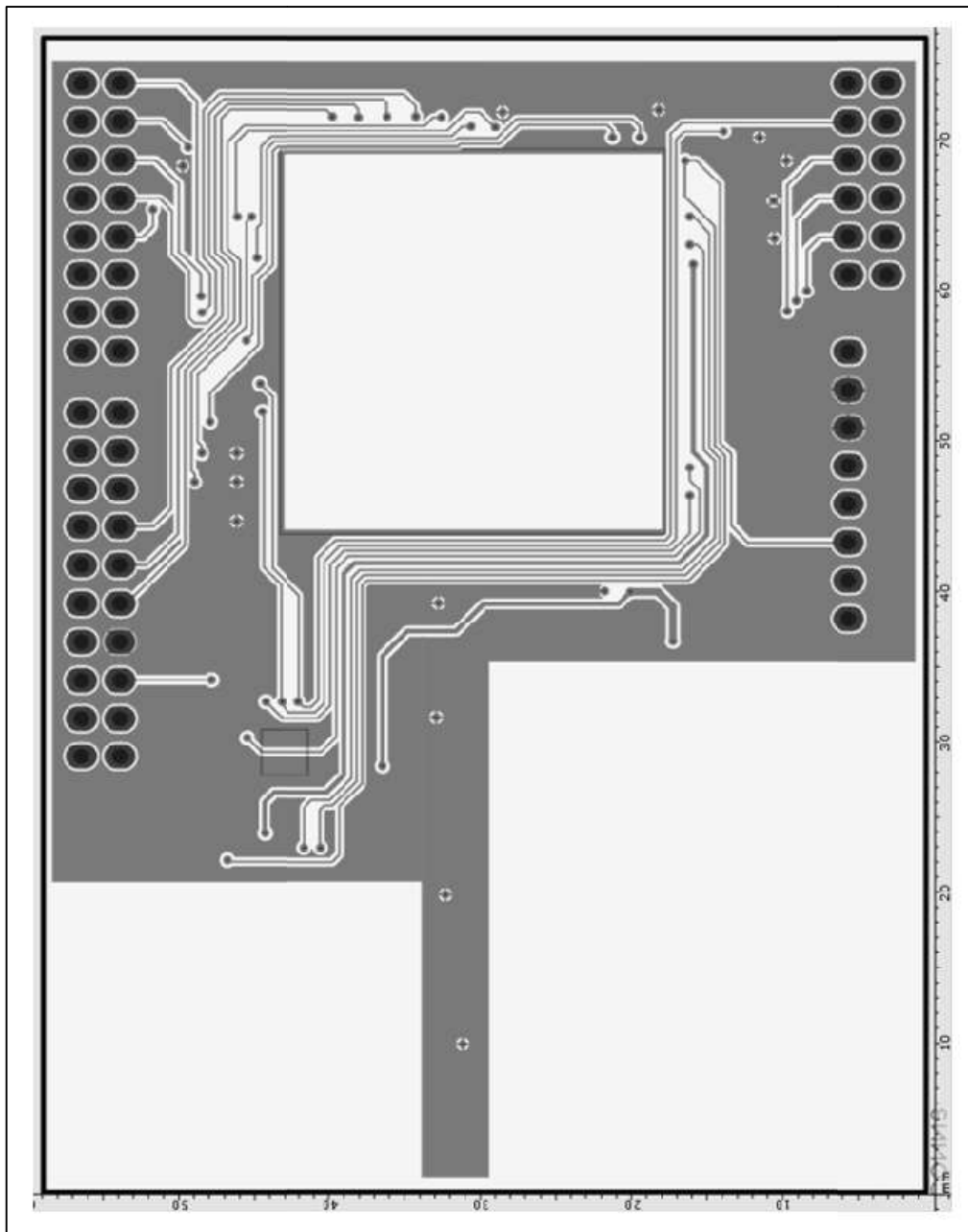


Figura 5.43. Bloque externo unificador, vista PCB: cara BOTTOM

5.5. Lista de componentes

	Descripción	Referencia	Cantidad	Precio (unid., €)	Precio (total, €)	
BLOQUE R7S124_LQFP64	<u>Microcontrolador /S1/WS1/LQFP/64 Pins</u>	Renesas R7FS124773A01CFM	1	6,34	6,34	
	<u>Condensador 100nF 0603 (1608 metric)</u>	Wurth Electronics 710-885012206071	6 (7 en total)	0,04	0,24	
	<u>Condensador 10µF 0603 (1608 metric)</u>	Murata Electronics 81-GRM188R61C106MALD	1 (5 en total)	0,321	0,321	
	<u>Condensador 10 pF 0603 (1608 metric)</u>	Murata Electronics 81-GRM185C1H100FA01J	2	0,113	0,226	
	<u>Condensador 12pF 0603 (1608 metric)</u>	Murata Electronics 81-GRM185C1H120FA01J	2	0,141	0,282	
	<u>Array x4 resistencias 1K</u>	Vishay/Dale 71-CRA06E-F-1K	1 (2 en total)	0,151	0,151	
	<u>Array x4 resistencias 10K</u>	Vishay/Dale 71-CRA06E-F-10K	1 (3 en total)	0,151	0,151	
	<u>LED 0603 (1608 metric)</u>	Bivar 749-SM0603HCL	1 (2 en total)	0,339	0,339	
	<u>Cristal 32,768 KHz 3,2x1,5mm</u>	IQD 449-LFXTAL009678REEL	1	0,321	0,321	
	<u>Conector de programación 5 posiciones, 1.27 mm paso</u>	Harwin 855-M52-040023V0545	1	0,66	0,66	
	<u>Cristal 16 MHz 3,2x2,5mm</u>	ABRACON 815-ABM8G-16-B4Y-T	1	0,849	0,849	
BLOQUE POTENCIA	<u>Conectpr jack de potencia DC</u>	CU1 490-PJ-002AH-SMT-TR	1	1,17	1,17	
	<u>Regulador de tensión ajustable LDO 3.0 A</u>	Microchip Technology / Micrel 98-MIC29302WU	2	2,42	4,84	
	<u>Trt 30V Vds 20V Vgs NMOS/PMOS</u>	Diodes Incorporated 621-DMG6602SVTQ-7	1	0,321	0,321	
	<u>Diodo propósito general 1,8x2,8mm</u>	Vishay Semiconductors S07M-GS08	1	0,509	0,509	
	<u>Resistencia 374 ohms 0402 (1005 metric)</u>	Vishay 71-CRCW0402-374-E3	1	0,094	0,094	
	<u>Resistencia 124 ohms 0402 (1005 metric)</u>	Vishay 71-CRCW0402-124-E3	2	0,094	0,188	
	<u>Resistencia 205 ohms 0402 (1005 metric)</u>	Vishay 71-CRCW0402-205-E3	1	0,094	0,094	
	<u>Condensador 10µF 0603 (1608 metric)</u>	Murata Electronics 81-GRM188R61C106MALD	4 (5 en total)	0,321	1,284	








	<u>Array resistencias 1K</u>	Vishay/Dale 71-CRA06E-F-1K	1 (2 en total)	0,151	0,151	
	<u>LED</u>	Bivar 749-SM0603HCL	1 (2 en total)	0,339	0,339	
	<u>Amplificador Operacional Low Voltage Rail-to-Rail</u>	Texas Instruments 595-LMV321IDCKR	1	0,594	0,594	
	<u>Conector x3 pines macho 2,54 mm paso (Selector 3,3V-5V)</u>	Molex 538-87891-0306	1	0,236	0,236	
BLOQUE USB + RESET	<u>Conector USB tipo B hembra</u>	LUMBERG 2411-02	1	0,56	0,56	
	<u>Array x4 resistencias 10K</u>	Vishay/Dale 71-CRA06E-F-10K	1	0,151	0,151	
	<u>Array x2 resistencias 27 ohms 2,54x2mm</u>	CTS Electronic Components 774-743C043270JP	1	0,50	0,50	
	<u>Pulsador (RESET)</u>	TE Connectivity/Alcoswitch 506-FSM4JSMA	1	0,207	0,207	
	<u>Condensador 1uF</u>	81-GRM188R61H105KALD	1	0,189	0,189	
	<u>Condensador 100nF 0603 (1608 metric)</u>	Würth Electronics 710-885012206071	1 (7 en total)	0,04	0,04	
	<u>Fusible de restablecimiento 1812 (4532 metric)</u>	Bourns 652-MF-MSMF050-2	1	0,208	0,208	
BLOQUE EXTERIOR-CONECTORES	<u>Array resistencias 10K</u>	Vishay/Dale 71-CRA06E-F-10K	1	0,151	0,151	
	<u>Conectores hembra, 16 posiciones, 2 filas, paso 2.54mm</u>	Amphenol FCI 649-66953-008LF	1	2,37	2,37	
	<u>Conectores hembra, 12 posiciones, 2 filas, paso 2.54mm</u>	Amphenol FCI 649-66953-012LF	1	3,31	3,31	
	<u>Conectores hembra, 20 posiciones, 2 filas, paso 2.54mm</u>	Amphenol FCI 649-66953-010LF	1	3,31	3,31	
	<u>Conectores hembra, 8 posiciones, 1 fila, paso 2.54mm</u>	Amphenol FCI 649-66951-008LF	1	2,13	2,13	
TOTAL:					32,918	

Tabla 5.2. Lista de componentes y precio

6. Fabricación del módulo OEM

En este apartado se describe el proceso que se sigue para fabricar el módulo OEM, una vez que se dispone de los *gerbers* definitivos de todos los bloques PCB que, fusionados, forman el conjunto completo. La realización de lo expuesto en este apartado no forma parte de los objetivos del proyecto, ya que es la empresa Embeblue la encargada de llevar a cabo la fabricación del módulo. En cualquier caso, se incluye una breve descripción de cada uno de los pasos seguidos:

- Unión de los cuatro bloques PCB en el conjunto que forma el circuito impreso definitivo a fabricar.
- Revisión y corrección, si esta fuera necesaria, de posibles errores en los *gerbers* definitivos y envío de estos al fabricante.
- Una vez que se dispone de la placa, proporcionada por el fabricante, soldadura de todos los componentes en ella.

6.1. Unión de los bloques PCB

Mediante un software hecho a medida por la empresa Embeblue, se sitúan los bloques en la posición adecuada, de tal modo que encajen perfectamente unos con otros.

Para ello, es necesario disponer del archivo de texto que indica las coordenadas de todos los componentes presentes en la PCB, el cual se puede generar desde el programa *DesignSpark PCB*. Por otro lado, los bloques internos no deben incluir ningún borde externo, ya que este viene determinado por el bloque unificador, el cual se describió en el apartado 5.4. Este bloque, además de integrar los restantes e incluir los conectores externos, determina el contorno de la placa. De no eliminarse estos bordes en el resto de bloques, se pueden encontrar problemas a la hora de fusionarlos.

A continuación, se muestran las caras TOP y BOTTOM de los *gerbers* fusionados formando el módulo final.

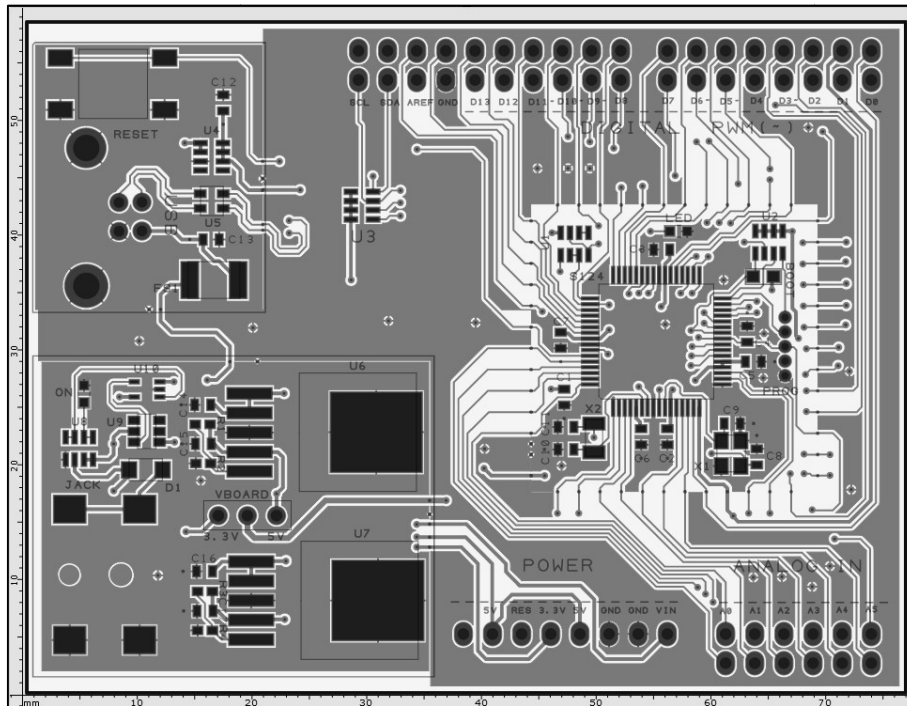


Figura 6.1. Módulo PCB fusionado: cara TOP

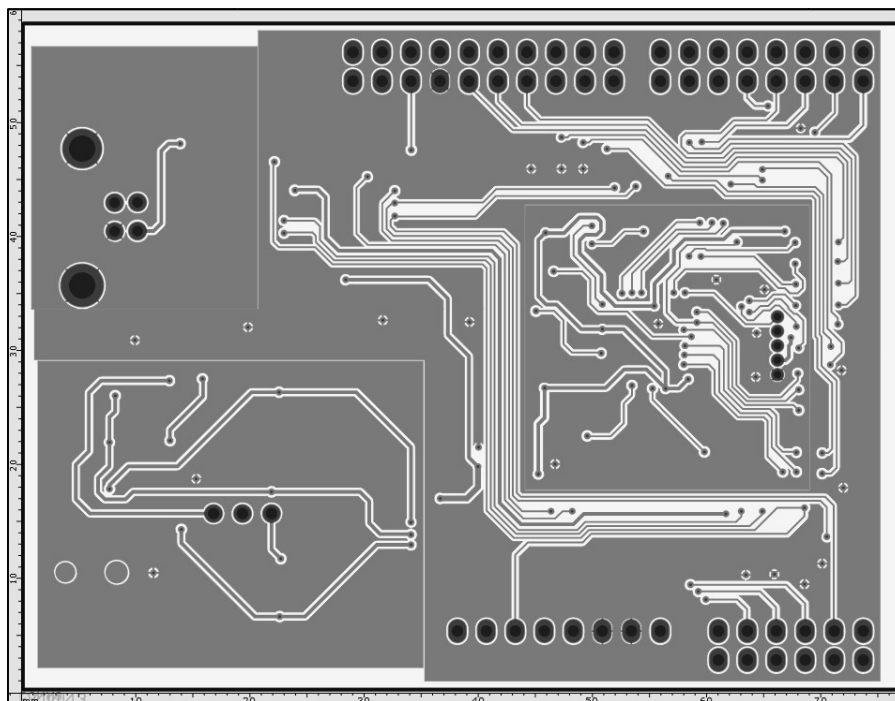


Figura 6.2. Módulo PCB fusionado: cara BOTTOM

Se pueden ver claramente las líneas que marcan los contornos de los bloques aunque, al encajarlos entre sí, se mantiene la continuidad de las pistas de unos a otros. La superposición correcta de los terminales de los marcos es el factor verdaderamente importante desde el punto de vista funcional.

Como línea de mejora en diseños futuros, se puede perfeccionar el conexionado de los planos de masa de unos bloques a otros, no sólo por estética, sino que, además, conviene mantener la mayor uniformidad posible en dichos planos. Por ejemplo, el plano de masa del bloque del microcontrolador está unido únicamente por dos terminales con el plano del bloque externo unificador. Una buena idea sería incluir terminales alargados que unan ambos planos a lo largo de todo el contorno.

6.2. Envío de los archivos de diseño al fabricante

La empresa que se encarga de fabricar la placa PCB es Eurocircuits, la cual se dedica a la fabricación de tarjetas de circuito impreso a medida y por encargo. Lo único que se debe hacer es seguir los pasos que se indican en su web para el envío de los archivos de diseño, adjuntándose estos durante el proceso.

Además, Eurocircuits proporciona una herramienta online llamada *PCB Visualizer*, que, además de mostrar una vista previa del aspecto y algunos parámetros de la tarjeta (dimensiones de las pistas, diámetro de los taladros...), permite al usuario detectar posibles errores en el diseño mediante la herramienta *PCB Checker*. Estos errores darían lugar a fallos en el hardware final si no se corrigieran, por tanto, en caso de detectarse, deben tomarse las medidas oportunas. Antes de enviar el pedido final, el usuario debe asegurarse de que no se detecta ningún error mediante esta herramienta. En las Figuras 6.3., 6.4. y 6.5. se muestran la interfaz de la herramienta *PCB Visualizer* y las vistas previas del aspecto final del módulo, respectivamente.

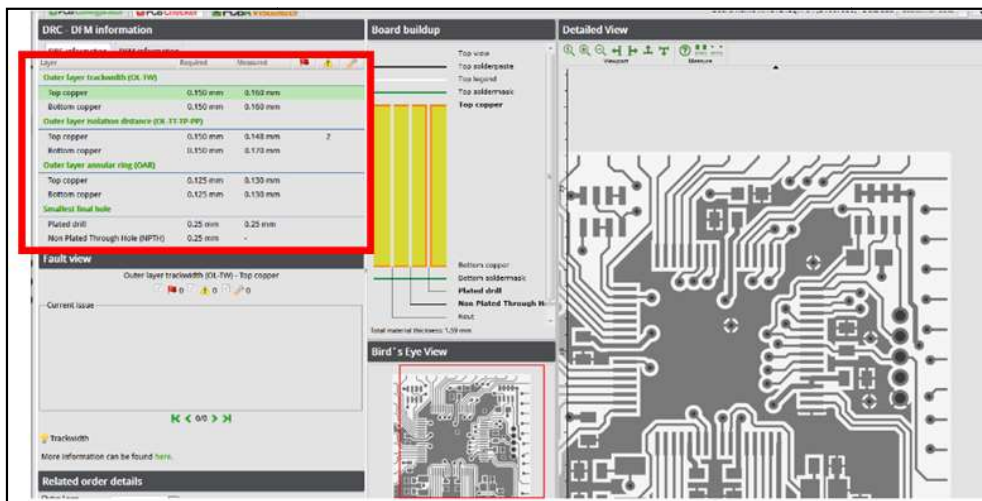


Figura 6.3. Menú PCB Visualizaer

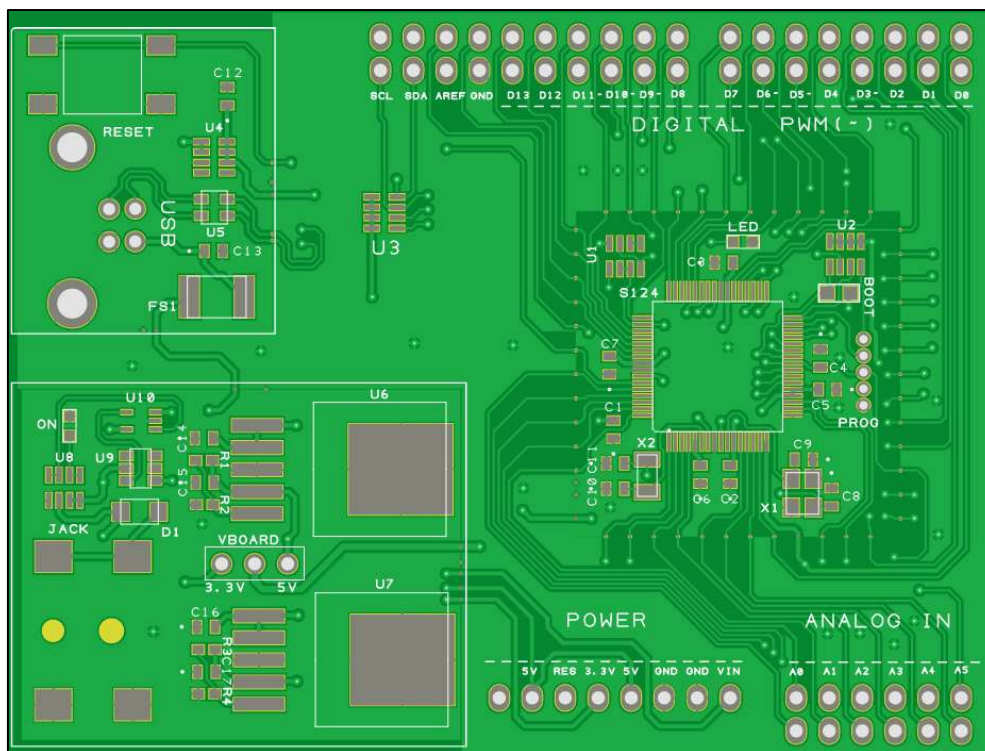


Figura 6.4. Vista previa del módulo PCB en PCB Visualizer: cara TOP

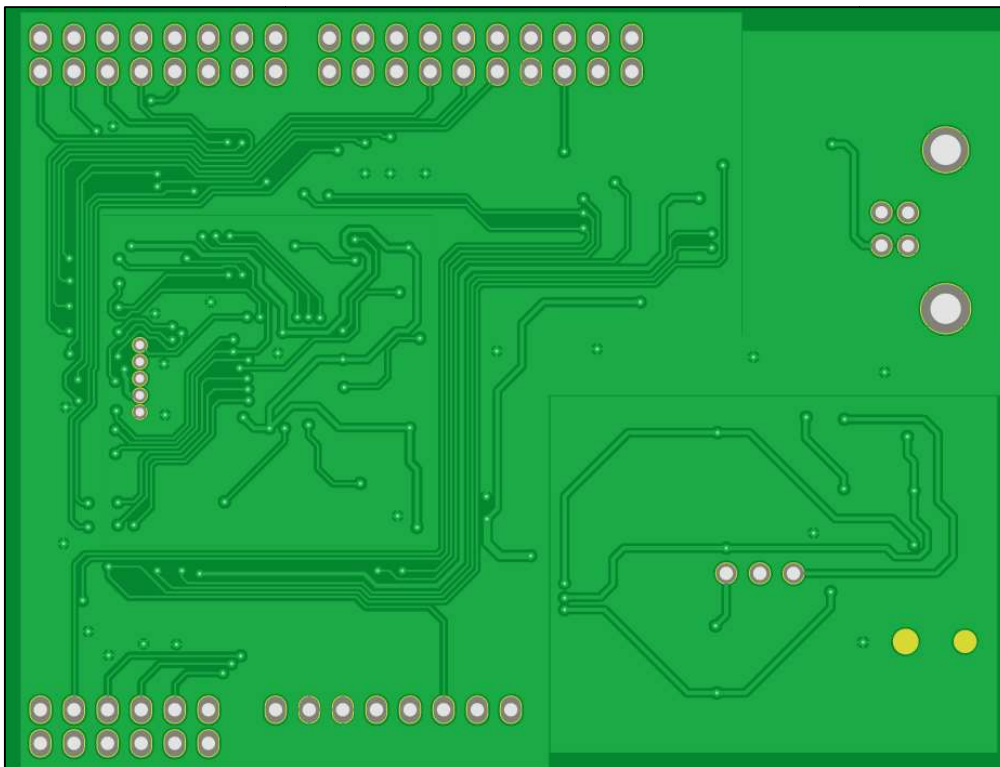


Figura 6.5. Vista previa del módulo PCB en PCB Visualizer: cara BOTTOM

6.3. Soldadura de componentes

Una vez se dispone de la tarjeta PCB fabricada por Eurocircuits y de todos los componentes (ver apartado 5.5.), el siguiente paso es la soldadura de los componentes sobre los *pads*. Este proceso, que se realiza íntegramente en la empresa Embeblue, se describe brevemente a continuación. Se recuerda que este diseño es a una cara (con todos los componentes en una cara), estando todos los *pads* smd en la cara TOP.

El soldado se realiza valiéndose de un horno de soldadura smd, similar al mostrado en la Figura 6.9. . Previamente a la introducción de la tarjeta en dicho horno se esparce la pasta de soldadura, valiéndose de una plantilla, y se colocan uno a uno los componentes en sus posiciones. Se detallan a continuación los pasos en orden:

- Se fija la placa sobre la superficie de trabajo, con el objetivo de que no se mueva durante el proceso.
- Se coloca la plantilla o *stencil* sobre la PCB, superpuesta de tal modo que los pads queden perfectamente situados en los huecos. El *stencil* es una plantilla metálica proporcionada por el fabricante y que se realiza a partir de la capa *Solder Paste* (descrita en el apartado 3.1.3). Contiene la posición y forma de todos los *pads* smd

(normalmente, con agujeros ligeramente más pequeños que los *pads*), de tal modo que sirve como referencia para el esparcimiento de la pasta de soldadura. Una vez, el *stencil* está perfectamente colocado se fija para que tampoco se mueva en el proceso.

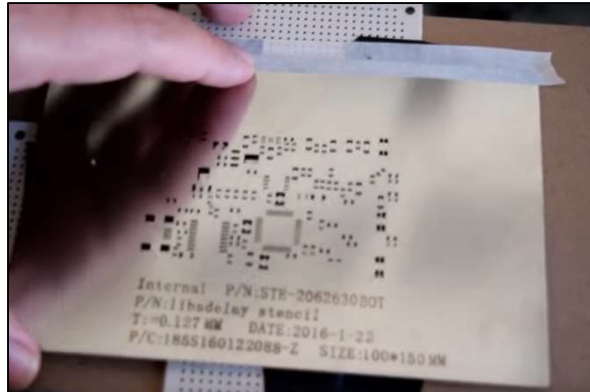


Figura 6.6. Fijación del stencil sobre la placa PCB:
<https://www.youtube.com/watch?v=bSnb3D72pxY>

- Con la plantilla colocada y fijada se extiende la pasta de soldadura (ver apartado 3.1.3.) ayudándose de una espátula. Los huecos en la plantilla hacen que la pasta se adhiera solamente en los *pads*. Hay que asegurarse de que se llenen todos los huecos, cuidando que sólo quede pasta encima de estos. Se debe mantener la plantilla plana sobre la PCB para que la pasta no se esparza por donde no debe. La pasta de soldadura es un producto espeso y que tiene una consistencia tal que los componentes se quedarán ligeramente pegados al situarlos encima.



Figura 6.7. Extensión de la pasta de soldadura sobre el Stencil con una espátula:
<https://www.youtube.com/watch?v=bSnb3D72pxY>

- Con cuidado, se colocan uno a uno los componentes sobre sus huellas, vigilando que tengan la orientación correcta para que cada terminal encaje con el *pad* que corresponda. Se suele hacer uso de herramientas como pinzas y una lupa, sobre todo

cuando se trata de componentes muy pequeños. Como se ha dicho, la pasta hará que los componentes se queden pegados, lo cual facilita el proceso.

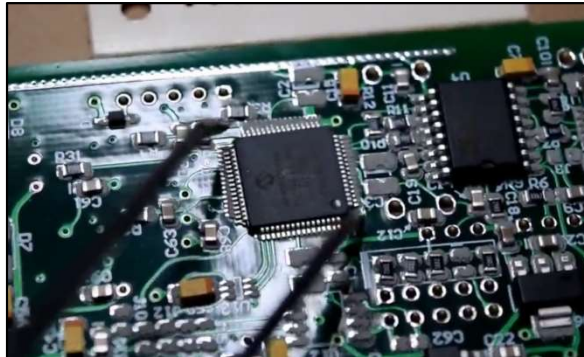


Figura 6.8. Colocación de los componentes sobre sus respectivas huellas:

<https://www.youtube.com/watch?v=km1knuJin-o>

- Con todos los componentes colocados en sus huellas y sobre la pasta de soldadura, se introduce la tarjeta en el horno, donde el estaño se funde y los componentes quedan soldados.



Figura 6.9. Introducción de la PCB en el horno de soldadura SMD

6.4. Aspecto final del módulo

En la Figura 6.10. se aprecia el aspecto final de la placa de circuito impreso tal y como se recibió del fabricante.



Figura 6.10. Placa PCB, previamente a la soldadura de componentes

Como se ha explicado en el apartado 6.3., el siguiente paso en la confección del módulo es la soldadura de todos los componentes siguiendo los pasos citados. Para ello se usa el *stencil* que se observa en la Figura 6.11.:

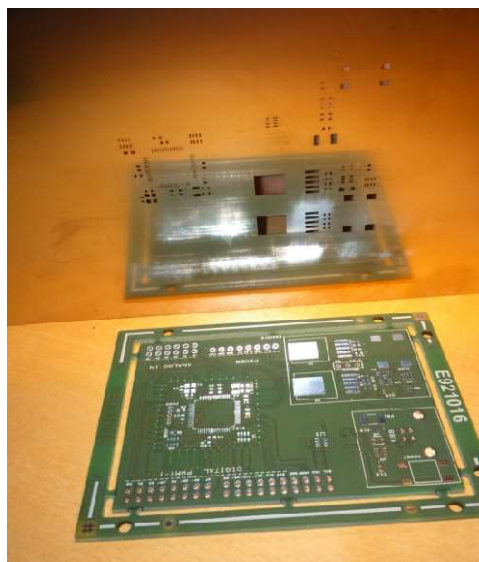


Figura 6.11. Stencil para módulo PCB

Cortesía de Embeblue, el módulo queda finalizado tras este último paso, presentando el aspecto que se puede ver en la Figura 6.12. y la 6.13. .

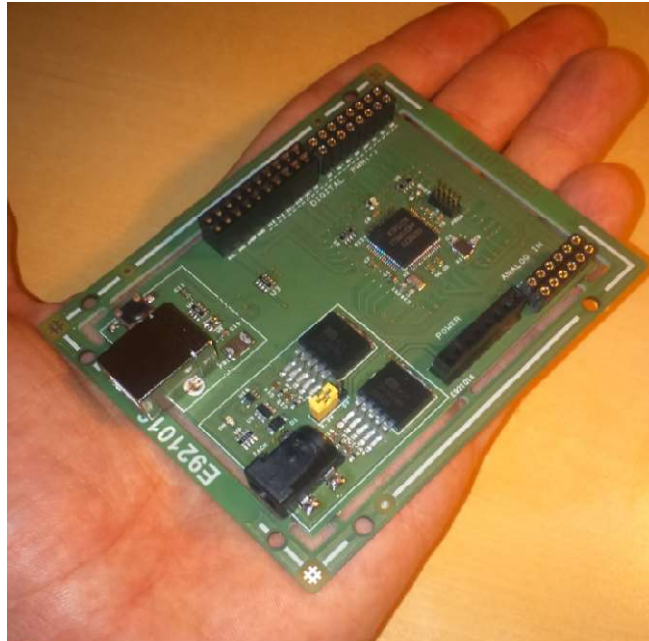


Figura 6.12. Módulo completo. Cara TOP

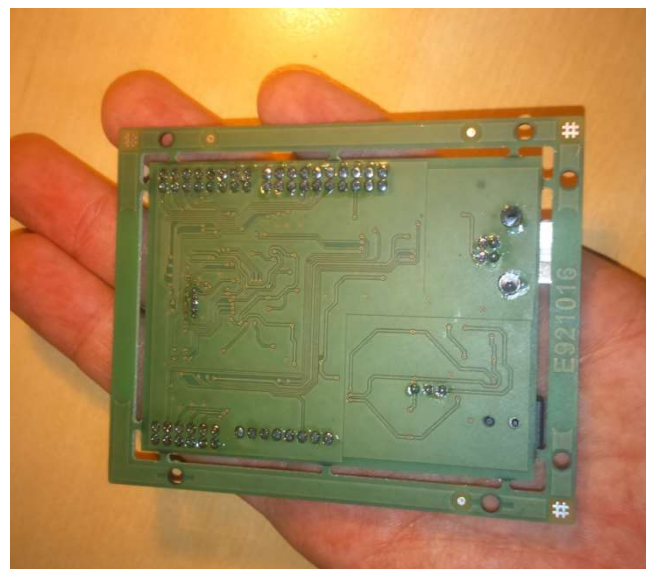


Figura 6.13. Módulo completo. Cara BOTTOM

7. Verificación del módulo

7.1. Revisión del enrutado y del sistema de alimentación

El primer paso para comprobar que todo se encuentra en orden en la tarjeta fabricada es la revisión de las *nets* para verificar que no hay cortes en las pistas, fallos en el diseño o contactos indeseados debidos al proceso de soldadura. Para ello se hace uso de un multímetro, el cual se puede configurar de tal modo que emita señales auditivas al detectar cortocircuitos entre sus bornes. De este modo, se puede comprobar que las pistas o planos desembocan donde deben, sin cortes en el conductor. El buen estado de algunas pistas se puede apreciar a simple vista pero es recomendable, por ejemplo, comprobar que todos los componentes activos reciben la alimentación correctamente en los terminales correspondientes.



Figura 7.1. Verificación de las pistas y planos mediante un multímetro

Tras dar el visto bueno al conexionado de la PCB, se procede a testear el sistema de alimentación. Para ello, se comprueban los tres modos de entrada de potencia para los que está pensada la placa:

- Alimentación a través del USB.
- Alimentación a través del *jack*.
- Alimentación simultánea a través del USB y del *jack*.

En la Figura 7.2. se muestra como el LED de entrada de potencia se enciende, como se esperaba, en los tres modos de alimentación.

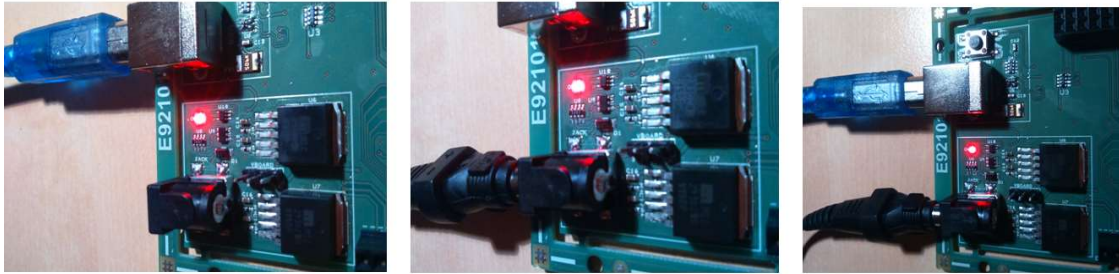


Figura 7.2. Alimentación de la tarjeta a través de puerto USB (izquierda), jack (centro) y de ambas entradas (derecha)

Además, en la Figura 7.3. se incluye la tensión en la *net* de '5V' en cada uno de los tres casos. Se observa que las tensiones son muy similares en todos los casos, tomando exactamente el mismo valor en el caso de alimentación a través del *jack* y en el caso de las dos fuentes conectadas. Esto es porque en este último caso se desconecta la entrada de potencia a través del puerto USB, tal y como debe hacer, por lo que el sistema de protección diseñado funciona correctamente.



Figura 7.3. Tensión en la *net* '5V' en los tres modos de alimentación

7.2. Verificación mediante *firmware*

En este apartado se pretende comprobar que el microcontrolador trabaja en óptimas condiciones mediante la puesta a prueba de algunos de sus módulos internos. Para ello, se cargan a través del conector de programación incorporado 5 programas de testeo, de menor a mayor complicación:

- **Programa 1.** Parpadeo de un LED conectado a una salida digital a una frecuencia constante. Se pretende verificar que las salidas digitales y el reloj interno funcionan correctamente. Además, se repite el programa utilizando los cristales de cuarzo externos como fuente de la señal de reloj para comprobar que estos realizan su función adecuadamente.
- **Programa 2.** Parpadeo de un LED y control de la frecuencia de parpadeo mediante un pulsador digital. Se añade al primer programa el uso de una entrada digital para comprobar que estas funcionan también correctamente.
- **Programa 3.** Parpadeo de un LED y control de su frecuencia mediante la interrupción externa de un pulsador digital. Al programa anterior se le añade el uso de una interrupción en el programa cada vez que se acciona el pulsador. Con ello se pretende testear el funcionamiento de las interrupciones externas.
- **Programa 4.** “Hola Mundo” en *LCD_Keypad_Shield*. Con este programa se quiere verificar la compatibilidad de la placa con una *shield* preparada para los módulos de Arduino. La *LCD_Keypad_Shield* contiene una pantalla LCD de 16x2 caracteres, además de una botonera conectada a un pin analógico. Se presenta con más detalle en el apartado correspondiente.
- **Programa 5.** Monitorización de variables analógicas en *LCD_Keypad_Shield*. Se pretende con este programa probar el módulo Analógico/Digital del microcontrolador.

En los siguientes subapartados se detalla el funcionamiento interno de los programas de prueba mencionados. Se incluye el link al video donde se pueden observar los mismos: <https://www.youtube.com/watch?v=6wDBlapPtSQ>. Como se puede comprobar en el vídeo, el resultado de los programas de verificación fue satisfactorio, ya que el módulo respondió perfectamente en todos los casos.

En cuanto a la forma de programar el microcontrolador, el módulo ofrece dos posibles interfaces para esta función:

- A través del puerto USB, lo cual exige la carga de un bootloader, posibilidad que ofrecen todos los microcontroladores de Synergy. Esta alternativa, por ejemplo, es la más habitual en las placas de Arduino.
- A través del conector JTAG/SWD incluido en el bloque del microcontrolador (ver apartado 5.1.1.).

Se descarta la primera opción, ya que no se cuenta con los conocimientos de programación necesarios para la programación del bootloader. Por tanto, se utiliza el conector JTAG¹⁶/SWD¹⁷, para lo cual se requiere de un programador o un debugger compatible con las características del microcontrolador. En este proyecto, se utiliza el programador J-Link EDU de Segger, que hace de acondicionamiento intermedio entre el puerto USB del ordenador y el conector JTAG/SWD del módulo. Se utiliza, en este caso, la interfaz SWD, para la cual se debe realizar el conexionado correcto de las salidas del J-Link correspondientes (5V, GND, SWDIO, SWCLK, RESET), según la asignación de pines mostrada en la Figura 7.4. .

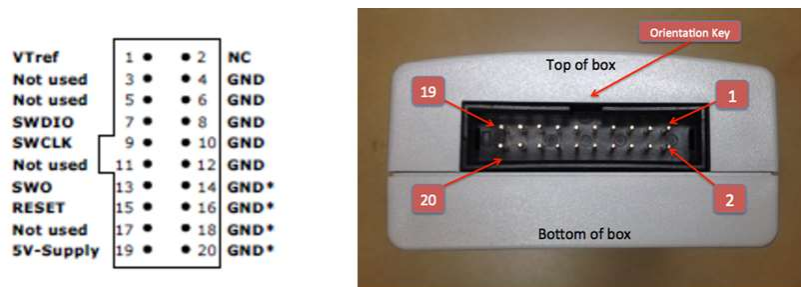


Figura 7.4. Asignación de pines en programador J-Link EDU

Además, se hace uso del adaptador que se puede ver en la Figura 7.5. para el acoplamiento en el conector JTAG/SWD, que tiene un paso demasiado pequeño en los pines para la conexión de cables normales.



Figura 7.5. Adaptador al conector JTAG/SWD y programador J-Link EDU

¹⁶ Joint Test Action Group

¹⁷ Serial Wire Debug

7.2.1. Programa de prueba 1: parpadeo de un LED

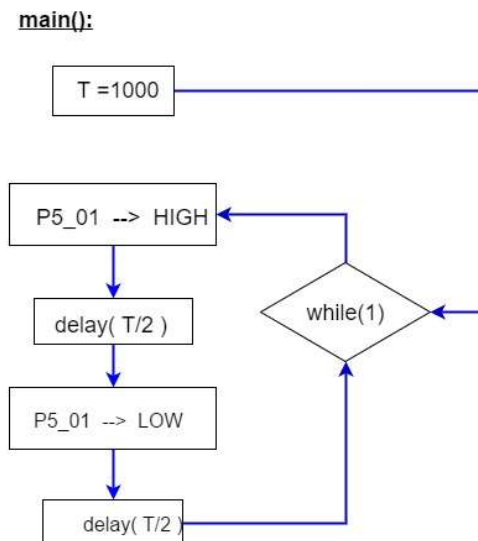


Figura 7.6. Diagrama de funcionamiento: programa 1

Como se puede ver en el diagrama de funcionamiento, se trata de conseguir un '1' y un '0' lógicos alternadamente en el pin P5_01, de tal modo que se consiga el parpadeo del LED conectado a dicho pin (ver esquemático de la Figura 5.2.) a una frecuencia constante $1 / T$. Se impone un período $T = 1000$ ms.

En este programa se hace uso de las siguientes funciones, definidas en el paquete software del entorno de programación (*Synergy Software Package*¹⁸):

- Función para escritura en **pinos digitales**. Se muestra el pin P5_01 a modo de ejemplo:

P5_01 = LOW ('0'):

```
g_ioport.p_api->pinWrite(IOPORT_PORT_05_PIN_01, IOPORT_LEVEL_LOW);
```

P5_01 = HIGH ('1'):

```
g_ioport.p_api->pinWrite(IOPORT_PORT_05_PIN_01, IOPORT_LEVEL_HIGH);
```

¹⁸ Ver apartado 2.2.2.

- Función para generar **retardos** (*delay*). En el siguiente ejemplo se impone un retraso de 1000 ms:

```
R_BSP_SoftwareDelay(BSP_DELAY_UNITS_MILLISECONDS, 1000);
```



Figura 7.7. Parpadeo del LED de testeo

Como con este programa también se quiere testear los cristales externos (X1 y X2), se vuelve a cargar el código, esta vez imponiendo la señal generada por estos componentes como fuente del reloj del sistema, tal y como se ve en la Figura 7.8. . En la Figura 7.9 se señalan los cristales en cuestión.



Figura 7.8. Configuración de los relojes internos y de las fuentes de los osciladores

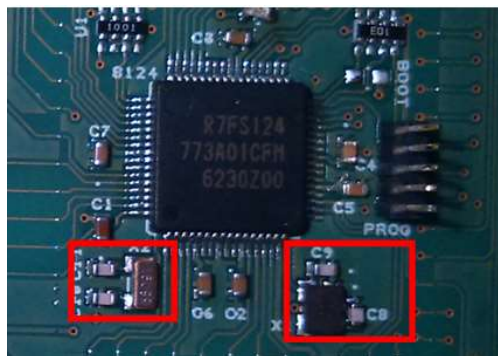


Figura 7.9. Cristales de cuarzo externos

7.2.2. Programa de prueba 2: parpadeo de un LED + control de la frecuencia mediante una entrada digital

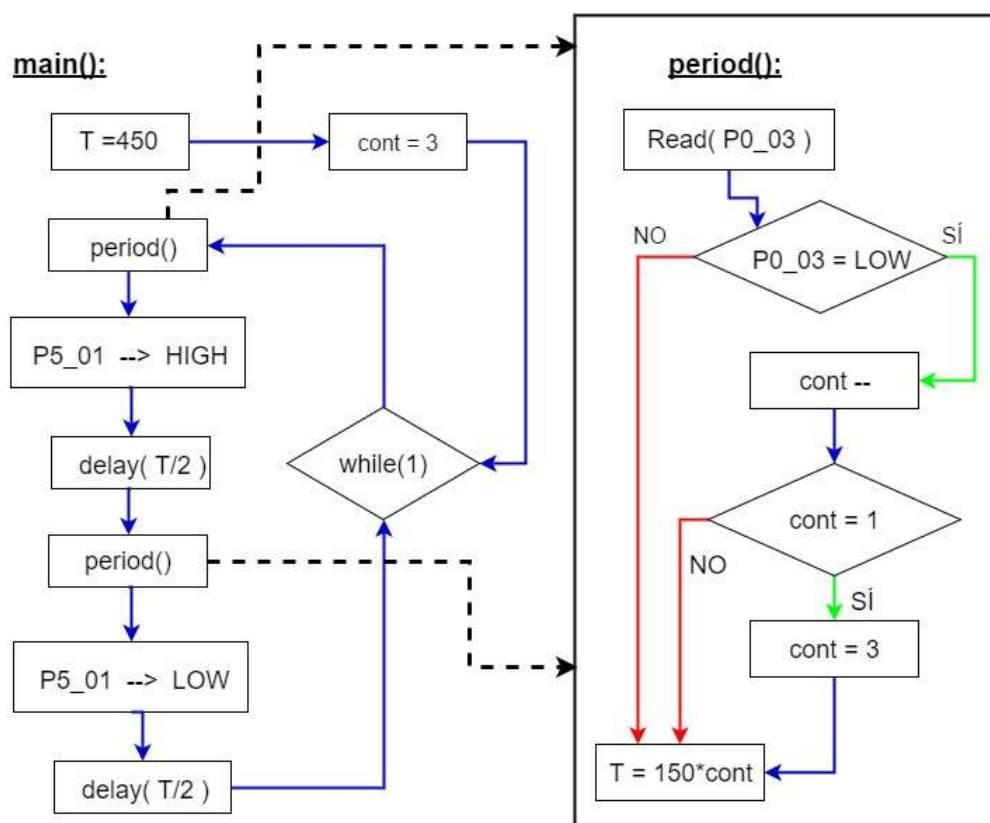


Figura 7.10. Diagrama de funcionamiento: programa 2

En este segundo ejemplo se mantiene la estructura del programa anterior pero se le añade la posibilidad de cambiar el período T en tiempo real. Ahora T es una variable que se puede ajustar mediante un pulsador conectado a la entrada digital PO_03. Al accionar el pulsador se cortocircuita la entrada correspondiente con masa ('0' lógico), mientras que en reposo se lee por esta un '1' lógico.

Siguiendo el diagrama de funcionamiento, se llama a la función *period()* cada medio período, dentro de la cual se comprueba si se ha accionado el pulsador. Si es así, se reduce el período en saltos de 150 ms, desde 450 s a 150 ms. Cuando se llega a este último valor se reinicia el ciclo al volver a pulsar el botón.

Como novedad, en este programa aparece la función de **lectura de entradas digitales**. A modo de ejemplo se muestra la expresión para la lectura del valor digital en el pin PO_03, valor que se guarda en la variable *level*:

```
g_ioport.p_api->pinRead(IOPORT_PORT_00_PIN_03, &level);
```

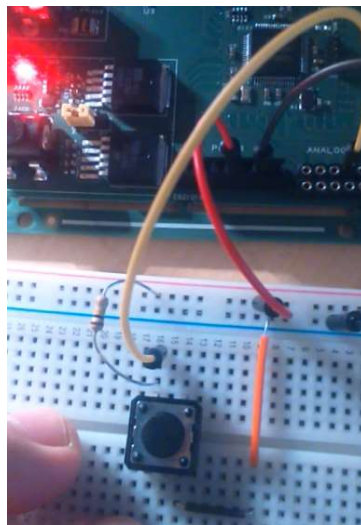


Figura 7.11. Pulsador para el control de la frecuencia de parpadeo del LED

7.2.3. Programa de prueba 3: parpadeo de un LED + control de la frecuencia mediante una entrada digital + uso de interrupciones externas

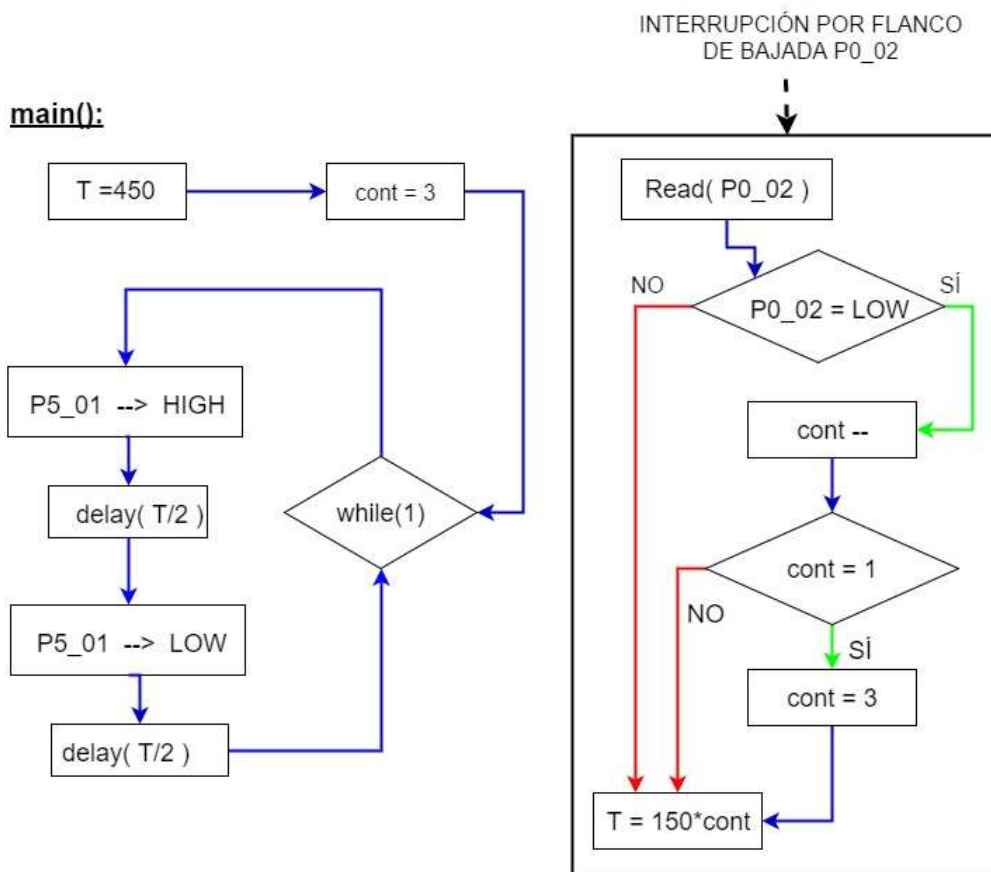


Figura 7.12. Diagrama de funcionamiento: programa 3

El objetivo de este programa es exactamente el mismo que el del anterior, es decir, controlar el parpadeo de un LED mediante un pulsador digital. Sin embargo, este es más eficiente, ya que no tiene que esperar a una función para comprobar el estado de la entrada digital, ni pierde tiempo dentro de esta en cada ciclo, ya que, directamente, se llama a la función cada vez que se pulsa el botón. Esto se consigue generando una interrupción por flanco de bajada en el pin de entrada correspondiente (P0_02).

Dentro de esta función interrupción (IRQ) se realizan las mismas instrucciones que en la función *period()*, es decir, se modifica un contador para variar con cada pulsación el período T, en saltos de 150 ms. Además, se añade dentro de esta función un retardo de 300 ms, para evitar que se llame repetidas veces a la función interrupción en cada pulsación. De este modo, se da tiempo a retirar el dedo y la función sólo se ejecuta una vez.

A continuación, se expone el código nuevo que introduce este programa respecto a los anteriores para el manejo de la interrupción:

- Declaración de la variable *status* y habilitación de la función interrupción:

```
ssp_err_t status;
```

```
status = g_sf_irq2.p_api->open(g_sf_irq2.p_ctrl, g_sf_irq2.p_cfg);
```

- Función interrupción:

```
void g_irq2_callback(external_irq_callback_args_t * p_args)
{
    SSP_PARAMETER_NOT_USED (p_args);

    // CÓDIGO DEL USUARIO
}
```

7.2.4. Programa de prueba 4: “Hola mundo” en *LCD_Keypad_Shield*

Con este programa se quiere probar la compatibilidad del módulo con una *shield* diseñada para Arduino, como es la *LCD Keypad Shield*. Esta *shield* consta de una pantalla LCD de 16 x 2 caracteres y 5 pulsadores que se nombran como sigue: *right*, *left*, *up*, *down* y *select*. Además, dispone de un pin de RESET (*rst*) conectado al pin análogo de Arduino.

Todos los pulsadores desembocan en el pin A0 de Arduino a través de un divisor de resistencias. De este modo, a todos los pulsadores les corresponde un valor de tensión determinado por este divisor de resistencias. Dicho valor se comprueba en el pin analógico A0, pudiendo así discriminar un pulsador de otro mediante *firmware*. Se aprovecha esta característica de la *LCD Keypad Shield* para testear el módulo analógico/digital del R7FS124773A01CFM.

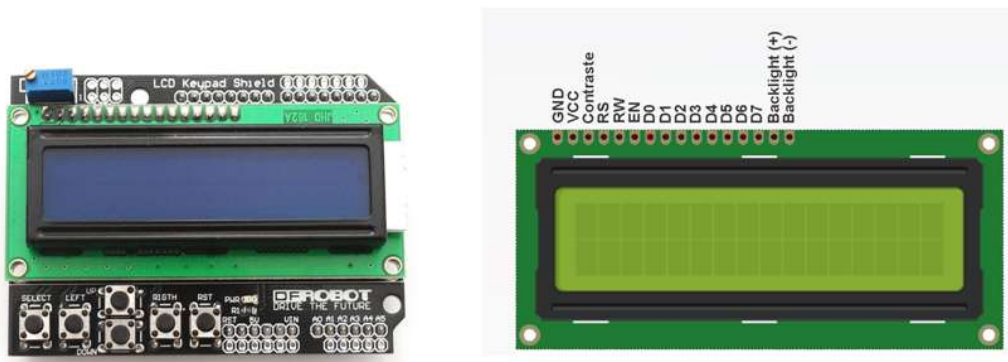


Figura 7.13. *LCD Keypad Shield*

Al no disponer de ninguna librería de esta *shield* compatible con el entorno de programación *e2studio*, se realiza todo el código que esta necesita a mano. La pantalla LCD lleva incorporado un microcontrolador propio que traduce los datos digitales que le llegan a caracteres en la pantalla. También existen secuencias de datos para realizar otras funciones, como imponer parámetros de configuración, desplazar el cursor de escritura en la pantalla... Las secuencias de datos se introducen a través de los pines D4, D5, D6, D7, R/W y E. Cada vez que se detecta un flanco de bajada en el pin E (*Enable*) se introducen los bits de los pines restantes. Se muestran dichos pines en la Figura 7.13. . Para mayor información acerca del código requerido por la pantalla LCD conviene revisar su *datasheet* [9].

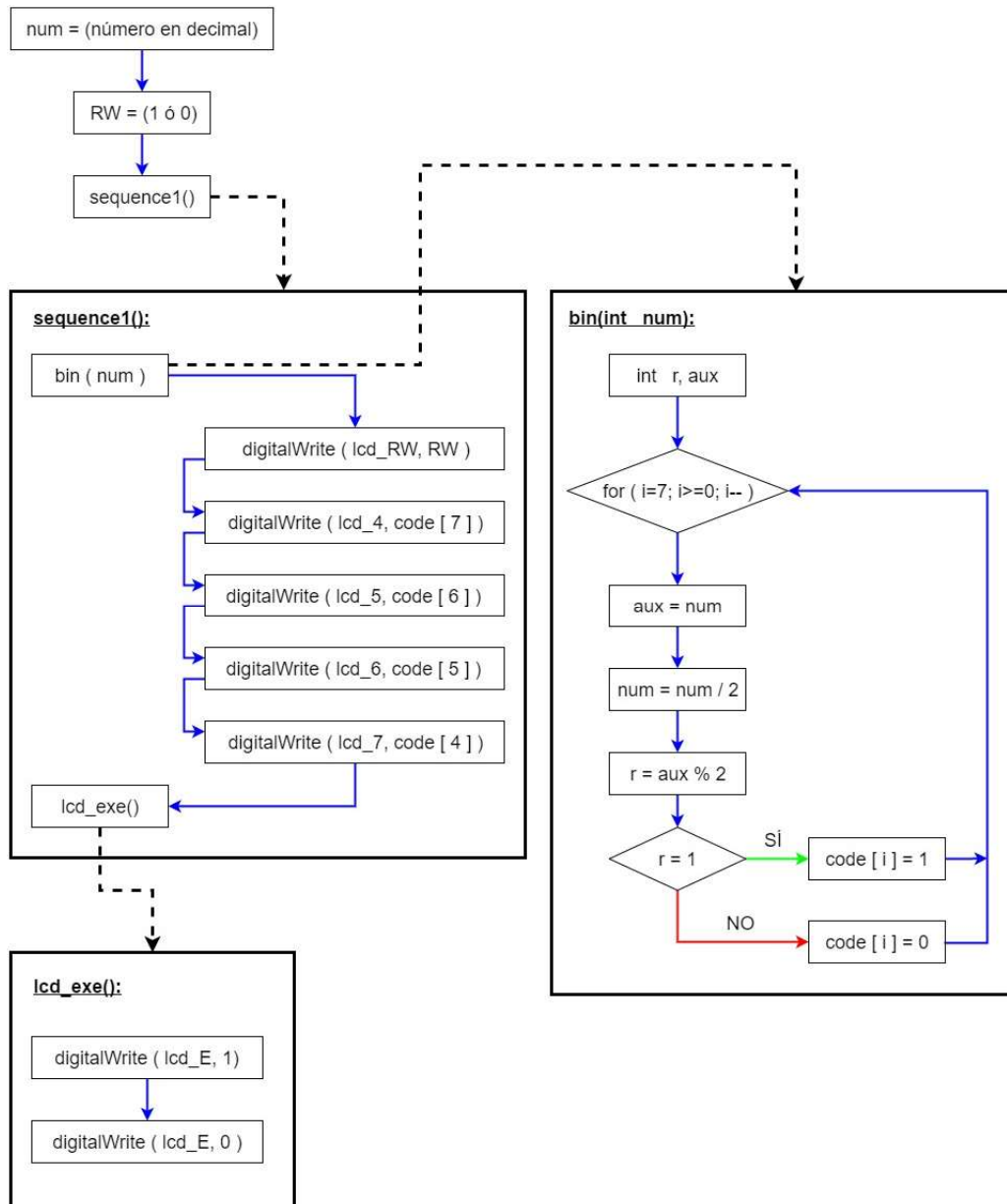


Figura 7.14. Diagrama de funcionamiento: introducción de datos en LCD_Keypad_Shield

En el diagrama de funcionamiento anterior se observa la secuencia para introducir los datos digitales por los pines R/W, D4, D5, D6, D7, definidos en el programa como las constantes *lcd_RW*, *lcd_4*, *lcd_5*, *lcd_6* y *lcd_7*, respectivamente. En la función *sequence1()* se da el valor digital que corresponde a estos pines en cada caso (definido en el código previamente) y se llama a la función *lcd_exe()*, donde se genera un pulso en el pin E que introduce dichos datos en la pantalla LCD. Para cada ejecución de la función *sequence1()* se debe definir el estado de *lcd_RW*: se pone a '1' para escritura y a '0' para configuración. Los datos en *lcd_4*, *lcd_5*,

lcd_6, lcd_7> se imponen dando un valor del 0 al 15 a la variable *num*. Esta variable se introduce en la función *bin(int num)*, que traduce este número de decimal a binario. El número binario resultante se guarda dentro de 4 posiciones de un vector de 8 posiciones de tipo *boolean (code[i])*, cada una de las cuales se corresponde con uno de los cuatro dígitos binarios.

Las librerías para manejar más cómodamente la LCD definen funciones que realizan diferentes acciones a partir de secuencias de datos en los pines mencionados. En este caso se definen las siguientes funciones:

- ***lcd_begin()***. Contiene la secuencia de números que configuran la pantalla LCD en el modo deseado e inicializan esta para poder monitorizar caracteres.
- ***lcd_home()***. Sitúa el cursor para escribir en la posición (0, 0), es decir, en la esquina superior izquierda.
- ***lcd_clear()***. Limpia la pantalla de todos los caracteres que estén escritos en ese momento.
- ***lcd_setCursor(int c, int r)***. Sitúa el cursor para escribir en la posición de entrada indicada. La variable *r (row)* indica la fila (0 ó 1) y *c (column)* indica la columna (de la 0 a la 15).
- ***lcd_print(char *c)***. Imprime la secuencia de caracteres tipo *char* en la pantalla. La frase que se escriba debe ir entre comillas dobles.
- ***lcd_scrollDisplayLeft()*** y ***lcd_scrollDisplayRight()***. Desplazan todo lo escrito en la LCD una posición a la izquierda y una posición a la derecha, respectivamente.

En el siguiente diagrama se muestra la forma de introducir la secuencia de números adecuada dentro de estas funciones.

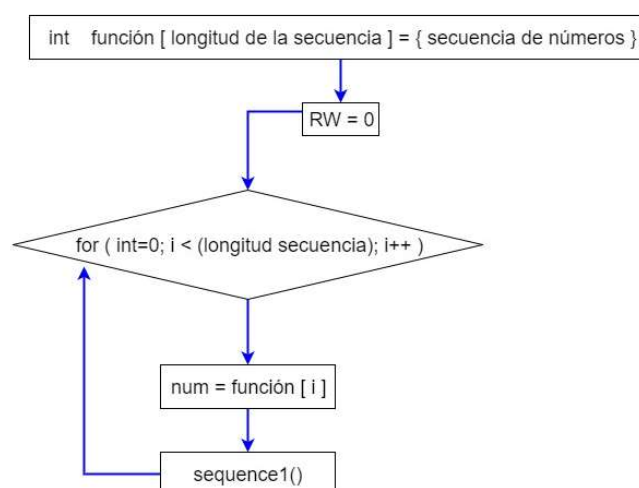


Figura 7.15. Diagrama de funcionamiento: funciones para el manejo de la pantalla LCD

Se puede apreciar cómo se introduce la secuencia de números en un vector, cuyos elementos, a su vez, se introducen uno a uno en la variable *num*, utilizada por la función *sequence1()*.

En el caso de la función *lcd_print()*, se sigue una metodología diferente, presentada en el siguiente diagrama de funcionamiento.

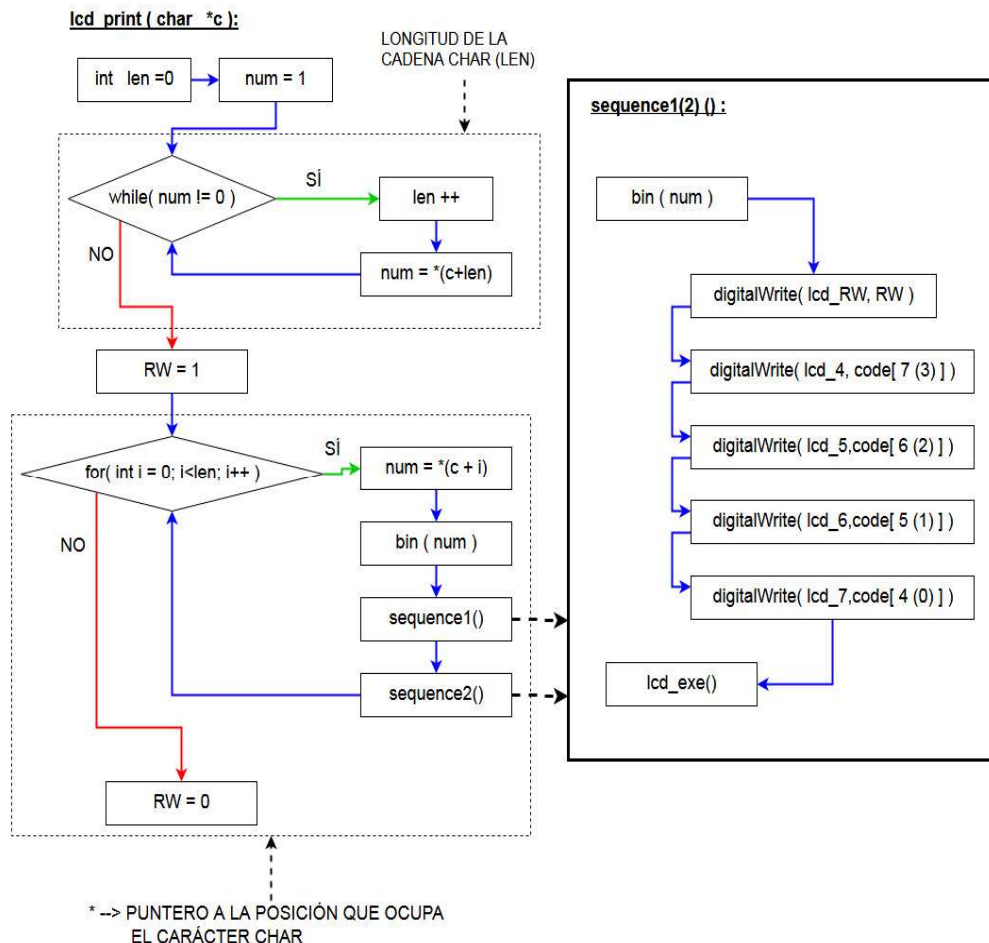


Figura 7.16. Diagrama de funcionamiento: funciones *lcd_print()* para escritura de secuencias de caracteres por pantalla

La variable de entrada (*c*) que se introduce a la función *lcd_print()* es una cadena de caracteres tipo *char*. Cada carácter tiene asignado un número según el código ASCII. Estos números siguen la misma ordenación que en el código que maneja la pantalla LCD, lo cual quiere decir que en la variable *num* bastará con introducir el número correspondiente a cada carácter en código ASCII. Para tratar con dicho número se utiliza un puntero, simbolizado en código C mediante un asterisco (*). Así, para acceder a la numeración ASCII del carácter *i* de la cadena *c* se escribe **(c + i)*. A partir de ahí se seguirá un procedimiento similar al del resto de funciones definidas para la LCD.

7.2.5. Programa de prueba 5: monitorización de variables analógicas en LCD Keypad Shield

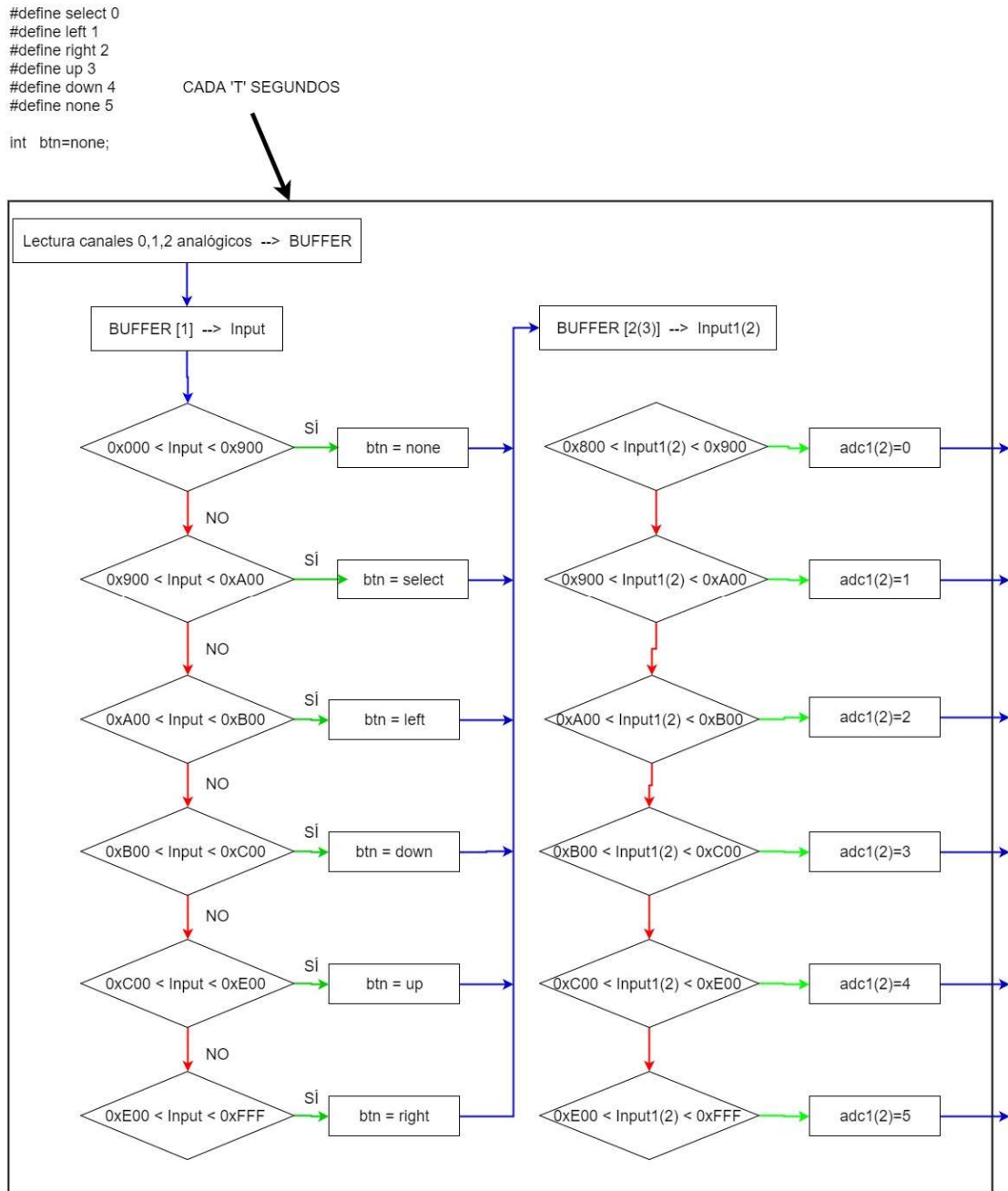


Figura 7.17. Diagrama de funcionamiento: lectura del nivel analógico mediante interrupción

El objetivo de este programa es verificar el funcionamiento adecuado del módulo Analógico/Digital mediante el uso de los pulsadores integrados en la *LCD Keypad Shield* y de dos potenciómetros conectados a los canales analógicos correspondientes a los pines P0_01 y P0_02. La entrada para la lectura de los pulsadores se encuentra en un único pin analógico. Mediante un divisor de resistencias se le asigna a cada uno de ellos un valor de tensión analógico diferente, lo que permite discriminarlos mediante software. En la Figura 7.17. se observa el proceso periódico de lectura de los valores analógicos (almacenados en 12 bits y representados en hexadecimal) que llegan secuencialmente al buffer, en el cual se almacenan según el orden de los canales. Se comprueba en qué rangos de tensión se encuentran las lecturas analógicas para almacenar en la variable *btn* el pulsador que se ha accionado (*up*, *down*, *right*, *left* y *select*) y en las variables *adc1* y *adc2* los respectivos valores de tensión de los potenciómetros.

```
int scroll=0,aux1,aux2;
bool K=1;
```

init_menu() :

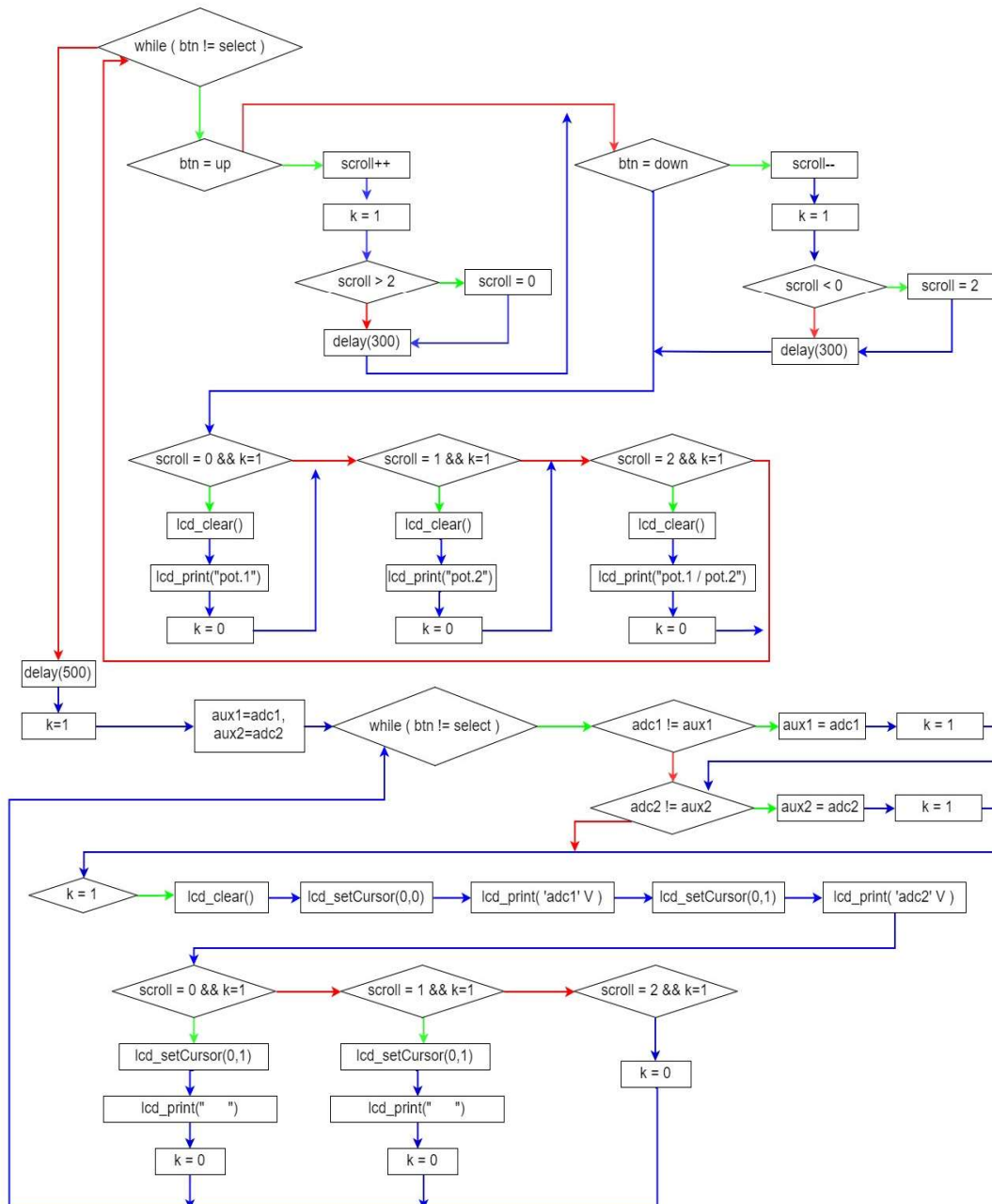


Figura 7.18. Diagrama de funcionamiento: menú interactivo en LCD KeypadShield con la lectura de los pulsadores en la variable btn

Se pretende monitorizar en tiempo real los valores de tensión impuestos por los potenciómetros en los pines correspondientes. Mediante el accionamiento de los pulsadores y el almacenamiento del valor de tensión correspondiente en la variable *btn* se interactúa con la pantalla LCD. Se crea un menú interactivo sencillo (ver Figura 7.18.) para cambiar entre tres modos de de visualización:

- Pot. 1. Visualización de la tensión del potenciómetro 1.
- Pot. 2. Visualización de la tensión del potenciómetro 2.
- Pot. 1 / Pot. 2. Visualización de la tensión de los potenciómetros 1 y 2, simultáneamente.

Mediante el uso de los pulsadores *up* y *down* se elige el modo y con el pulsador *select* se selecciona. Al rotar el potenciómetro se puede observar cómo cambia el valor de tensión en la pantalla simultáneamente.

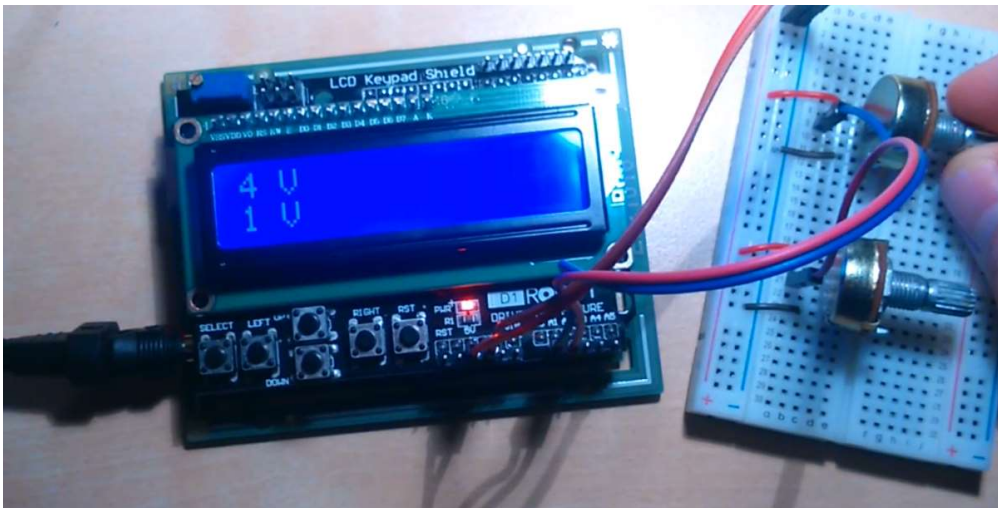


Figura 7.19. Visualización de las tensiones de los potenciómetros en la pantalla LCD

8. Conclusiones

En primer lugar se destaca que se han cumplido todos los objetivos planteados inicialmente. En resumen, se ha diseñado y fabricado el módulo OEM y, posteriormente, siguiendo los pasos que se han expuesto en el apartado 7, ha respondido positivamente a las pruebas a las que ha sido sometido. Por tanto, se puede afirmar que la verificación del módulo ha sido exitosa.

La conclusión más significativa del diseño logrado es que a partir de un microcontrolador y una electrónica adicional bastante básica se puede obtener un sistema funcional completo, apto para aplicaciones de prototipado, en sistemas embebidos o en determinados entornos industriales. Esta potencialidad creciente en cuanto a aplicaciones que ofrecen este tipo de módulos, cuyo diseño se ha demostrado que no requiere de una complejidad alta ni de un alto costo, es la razón de su popularidad cada vez mayor.

Por otro lado, cabe destacar las ventajas del método empleado en el diseño PCB, el cual se ha llevado a cabo en bloques funcionales separados. Si bien en este proyecto concreto puede haber supuesto un esfuerzo adicional, ya que se partía de cero, hay que resaltar que los bloques diseñados son perfectamente reutilizables en diseños posteriores, en los cuales se conseguirá un ahorro en cuanto a tiempo en el diseño. Por ejemplo, si se deseara diseñar un módulo que incluya unos periféricos determinados, se puede seguir la misma metodología utilizando el bloque del microcontrolador diseñado como unidad de control, lo cual reduciría significativamente el tiempo empleado.

9. Bibliografía

- [1] Renesas Electronics Corporation (2016). *S 124 Datasheet*. Recuperado de https://www.renesas.com/en-eu/doc/products/renesas-synergy/doc/r01ds0264eu0100_synergy_s124.pdf
- [2] Renesas Electronics Corporation (2016). *Development Kit DK-S124 v2.0. User's Manual: Hardware*. Recuperado de <https://www.renesas.com/en-eu/doc/products/renesas-synergy/doc/r12um0006eu0100-synergy-dk-s124.pdf>
- [3] Renesas Electronics Corporation (2016). *e2 studio Integrated Development Environment. User's Manual: Getting Started Guide*. Recuperado de https://www.renesas.com/ko-kr/doc/products/tool/doc/006/r20ut2771ej0400_e2_start_s.pdf
- [4] Renesas Electronics Corporation (2016). *Renesas Synergy™. Renesas Synergy Software Package 1.1.0*. Recuperado de <https://www.renesas.com/en-us/products/synergy/software.html#documents>
- [5] R.S. Khandpur (2005). *Printed Circuit Boards. Design, Fabrication, Assembly and Testing*.
- [6] AVX Corporation. *Introduction to Choosing MLC Capacitors For Bypass/Decoupling Applications*. Recuperado de http://bears.ece.ucsb.edu/class/ece224b/real_world_decoupling.pdf
- [7] Mouser Electronics. Catálogos de los Fabricantes (consultar referencias en el apartado 5.5.) <http://www.mouser.es/CatalogRequest/CatalogDownloads.aspx>
- [8] Arduino <https://www.arduino.cc/>

[9] HITACHI (1998). *HD44780U (LCD-II) Datasheet*. Recuperado de

<https://www.sparkfun.com/datasheets/LCD/HD44780.pdf>

[10] Archivos de diseño DK-S124

<https://www.renesas.com/en-eu/software/D6000220.html>

ANEXOS

ANEXO I. ASIGNACIÓN DE PINES EN EL MICROCONTROLADOR R7FS124773A01CFM (LQFP64)

Pin number					Power, System, Clock, Debug, CAC	I/O ports	Timers				Communication Interfaces				Analog		HMI	
LQFP64	LQFP48	QFN48	QFN40	LGA36			AGT	GPT_OPS, POEG	GPT	RTC	USBFS, CAN	SCI	IIC	SPI	ADC14	DAC12, ACMP, LP	CTS	Interrupt
1	1	1	1	C2	CACREF_C	P400	AGTIO1_D	GTIOC6A_A		SCK0_B/ SCK1_B	SCL0_A				TS20	IRQ0		
2	2	2	-	-		P401		GTETRGA_B	GTIOC6B_A	CTX0_B	SDA0_A				TS19	IRQ5		
3	-	-	-	-		P402				CRX0_B					TS18	IRQ4		
4	-	-	-	-		P403			GTIOC3A_B						TS17			
5	3	3	2	A1	VCL													
6	4	4	3	B1	XCIN	P215												
7	5	5	4	C1	XCOU	P214												
8	6	6	5	D1	VSS													
9	7	7	6	D3	XTAL	P213		GTETRGA_D								IRQ2		
10	8	8	7	D2	EXTAL	P212	AGTEE1	GTETRGA_D								IRQ3		
11	9	9	8	E1	VCC													
12	-	-	-	-		P411	AGTOA1	GTOVUP_B	GTIOC6A_B			MOSIA_B			TS07	IRQ4		
13	-	-	-	-		P410	AGTOB1	GTOVLO_B	GTIOC6B_B			MISOA_B			TS06	IRQ5		
14	10	10	-	-		P409		GTOWUP_B	GTIOC5A_B						TS05	IRQ6		
15	11	11	9	-		P408		GTOWLO_R	GTIOC5B_R						TS04	IRQ7		
16	12	12	10	E2		P407				RTCCOUT	USB_VBUS	CTS0_RT_S0_D/ SS0_D	SDA0_B	SSLB3_A	ADTRG0_B	TS03		
17	13	13	11	F1	VSS_USB													
18	14	14	12	F2						USB_DM								
19	15	15	13	F3						USB_DP								
20	16	16	14	F4	VCC_US_B													
21	17	17	15	F5	VCC_US_B_LDO													
22	18	18	-	-		P206		GTIU_A					SDA1_A	SSLB1_A	TS01	IRQ0		
23	-	-	-	-	CLKOUT_A	P205	AGTO1	GTIV_A	GTIOC4A_B				SCL1_A	SSLB0_A	TSCAP_A	IRQ1		
24	-	-	-	-	CACREF_A	P204	AGTIO1_A	GTIW_A	GTIOC4B_B				SCL0_B	RSPCKB_A	TS00			
25	19	19	16	E3	RES													
26	20	20	17	E4	MD	P201												
27	21	21	18	E5		P200										NMI		
28	-	-	-	-		P304			GTIOC1A_B									
29	-	-	-	-		P303			GTIOC1B_B						TS02			
30	22	22	-	-		P302		GTOUUP_A	GTIOC4A_A			SSLB3_B		TS08	IRQ5			
31	23	23	19	-		P301		GTOULO_A	GTIOC4B_A			SSLB2_B		TS09	IRQ6			
32	24	24	20	F6	SWCLK	P300		GTOUUP_C	GTIOC0A_A			SSLB1_B						
33	25	25	21	E6	SWDIO	P108		GTOULO_C	GTIOC0B_A			SSLB0_B						
34	26	26	22	D4	CLKOUT_B	P109		GTOUUP_A	GTIOC1A_A	CTX0_A			MOSIB_B		TS10			

LOFPA4	Pin number					Power System, Clock, Debug, CAC	I/O ports	Timers				Communication Interfaces				Analog		HMI	
	LOFPA8	QFN48	QFN40	LGA 36				AGT	GPT_OPS, POEG	GPT	RTC	USBFS, CAN	SD	IIC	SPI	ADC14	DACT12, ACMP/PLP	TSU	Interrupt
35	27	27	23	D5		P110		GTOVLO_A	GTIOC1B_A		CRX0_A	CTS0_RT SS0_C/ RXD0_B/ MISO0_B/ SCL9_B		MISOB_B		VCOUT	TS11	IRQ3	
36	28	28	24	D6		P111			GTIOC3A_A			SCK0_C/ SCK9_B		RSPCKB_B			TS12	IRQ4	
37	29	29	25	C6		P112			GTIOC3B_A			TXD0_C/ MOSI0_C/ SDA0_C					TSCAP_C		
38	-	-	-	-		P113													
39	30	30	-	-	VCC														
40	31	31	-	-	VSS														
41	-	-	-	-		P107			GTIOC0A_B									KR07	
42	-	-	-	-		P106			GTIOC0B_B					SSLA3_A				KR06	
43	-	-	-	-		P105		GTETRGA_C						SSLA2_A				KR05/ IRQ0	
44	32	32	26	-		P104		GTETRGA_B				RXD0_C/ MISO0_C/ SCL0_C		SSLA1_A			TS13	KR04/ IRQ1	
45	33	33	27	C3		P103		GTOVUP_A	GTIOC2A_A		CTX0_C	CTS0_RT SS0_A/ SS0_A		SSLA0_A	AN019	CMPREF1	TS14	KR03	
46	34	34	28	C4		P102	AGT00	GTOVLO_A	GTIOC2B_A		CRX0_C	SCK0_A		RSPCKA_A	AN020/ ADTRG0_A	CMPIN1	TS15	KR02	
47	35	35	29	C5		P101	AGTEE0	GTETRGA_A	GTIOC5A_A			TXD0_A/ MOSI0_A/ SDA0_A/ CTS1_RT S1_A/ SS1_A	SDA1_B	MOSIA_A	AN021	CMPREF0	TS16	KR01/ IRQ1	
48	36	36	30	B6		P100	AGTIO0_A	GTETRGA_A	GTIOC5B_A			RXD0_A/ MISO0_A/ SCL0_A/ SCK1_A	SCL1_B	MISOA_A	AN022	CMPIN0	TS26	KR00/ IRQ2	
49	37	37	-	-		P500	AGTOA0	GTIU_B	GTIOC2A_B						AN016		TS27		
50	-	-	-	-		P501	AGTOB0	GTIV_B	GTIOC2B_B						AN017				
51	-	-	-	-		P502		GTIW_B	GTIOC3B_B						AN018				
52	38	38	31	A6		P015									AN010		TS28	IRQ7	
53	39	39	32	A5		P014									AN009	DA0			
54	40	40	33	B5		P013									AN008				
55	41	41	34	B4		P012									AN007				
56	42	42	35	A4	AVCC0														
57	43	43	36	A3	AVSS0														
58	44	44	37	B3	VREFL0	P011									AN006		TS31		
59	45	45	38	A2	VREFH0	P010									AN005		TS30		
60	-	-	-	-		P004									AN004		TS25	IRQ3	
61	-	-	-	-		P003									AN003		TS24		
62	46	46	-	-		P002									AN002		TS23	IRQ2	
63	47	47	39	-		P001									AN001		TS22	IRQ7	
64	48	48	40	B2		P000									AN000		TS21	IRQ6	

