UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dpto. de Ingeniería Electrónica

DISEÑO DE UN SINGLE-BOARD COMPUTER BASADO EN UNA CPU ARM CORTEX-A7 CON MEMORIA DDR3L Y CONECTIVIDAD USB, ETHERNET Y CAN

Trabajo Fin de Máster

Máster Universitario en Ingeniería de Sistemas Electrónicos

AUTOR/A: Luna Alcolea, Alfonso José

Tutor/a: Toledo Alarcón, José Francisco

CURSO ACADÉMICO: 2021/2022

Resumen

En la actualidad, muchos de los sistemas embebidos se utilizan en aplicaciones que cada vez requieren de mayores prestaciones, microprocesadores más potentes, memorias más rápidas y el uso de sistemas operativos como Linux.

Este tipo de sistemas a menudo se basan en un potente microprocesador o FPGA junto a uno o varios chips de memoria RAM conectados mediante interfaces de alta velocidad tales como DDR3 o DDR4. Además, estos sistemas suelen comunicarse mediante interfaces de mayor velocidad que las que se suelen encontrar en microcontroladores, tales como HDMI, MIPI CSI/DSI o PCI Express.

Esto implica que en el PCB haya una gran cantidad de calor generado, además de un gran número de pistas por las que circulan señales de alta velocidad, complicando bastante el diseño de este tipo de sistemas. Se hace imprescindible tener en cuenta distintos aspectos relacionados con el diseño térmico, compatibilidad electromagnética e integridad de señal, hasta el punto de que no es posible conseguir un producto que funcione de manera fiable si no se consideran todos estos aspectos desde el principio.

En este Trabajo Fin de Máster se abordarán las distintas etapas implicadas en el desarrollo de sistemas basados en este tipo de dispositivos mediante el diseño, desarrollo, fabricación y posterior testeo de un *Single-Board Computer* (SBC), documentando a lo largo del camino todas las etapas involucradas en el proceso.

Índice general

Parte I

INTRODUCCIÓN

Capítulo 1

Introducción y objetivos

1.1. Conceptos previos

En la actualidad, los sistemas embebidos se utilizan en una amplia variedad de aplicaciones entre las cuales se encuentran grandes diferencias en cuanto a sus necesidades de prestaciones, consumo energético y costes, entre otras.

Para hacer frente a estas necesidades tan dispares entre aplicaciones, en los últimos años los fabricantes de microprocesadores para este tipo de sistemas han tendido a hacer una distinción entre dos grandes grupos de dispositivos: los microcontroladores y los *System-On-Chip* (SoCs).

1.1.1. Sistemas basados en microcontroladores

Los microcontroladores suelen hacer referencia a dispositivos que integran en un mismo chip tanto el microprocesador como la memoria RAM e incluso la memoria Flash en la que es almacenado el programa, además de otros periféricos.

Se suelen utilizar en aplicaciones que no necesitan de grandes prestaciones y en las que lo que más se valora es conseguir el menor consumo, el menor tamaño y el menor coste posible.

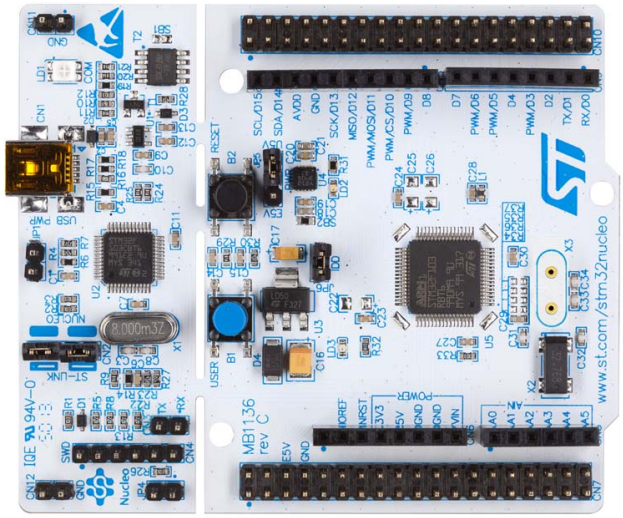


Figura 1.1: Placa ST NUCLEO basada en un microcontrolador.

Suelen funcionar a frecuencias relativamente bajas, menores de unos 200MHz, incluir memorias del orden de los KB o unos pocos MB, y tener un consumo bastante reducido, del orden de decenas de mA. En modos de bajo consumo pueden llegar a tener un consumo realmente bajo, del orden de los uA o nA, pudiendo alcanzar autonomías de meses o años alimentados mediante pilas o baterías. Dado lo poco que consumen, en la mayoría de ocasiones el calor que generan puede ser despreciable.

No suelen incorporar unidades de gestión de memoria, lo que por lo general los hace incompatibles con sistemas operativos como Linux. Normalmente se utilizan sin ningún tipo de sistema operativo o con microkernels de tiempo real.

Aunque también existen en el mercado microcontroladores que funcionan a frecuencias más elevadas y que admiten memorias externas de mayor capacidad, habitualmente las señales de mayor velocidad suelen quedar confinadas en el interior del propio chip, y las interfaces hacia otros componentes externos a los que se conectan suelen funcionar a frecuencias relativamente lentas.

Todo esto contribuye a facilitar en gran medida el diseño del hardware y a abaratar los costes de los sistemas basados en este tipo de dispositivos, ya que se relajan en gran medida las consideraciones a tener en cuenta sobre diseño de alta velocidad y sobre el calor generado.

1.1.2. Sistemas basados en System-On-Chips

Al igual que los microcontroladores, los SoCs integran en un mismo chip un microprocesador junto a una serie de periféricos que suelen ser de utilidad en gran parte de las aplicaciones. Sin embargo, se utilizan en aplicaciones en las que no prima tanto el consumo y que cada vez requieren de mayores prestaciones, microprocesadores más potentes y mayores cantidades de memoria.

La principal distinción respecto a un microcontrolador suele ser que los SoCs incorporan unidades de gestión de memoria, capaces de diferenciar la memoria física de la virtual, lo que los hace más adecuados para su uso con sistemas operativos como Linux. Estos sistemas requieren de mucha más cantidad de memoria, del orden de los cientos de MB o unos pocos GB, tanto de memoria RAM como de memoria Flash. Suelen funcionar a frecuencias más elevadas, del orden de cientos de MHz o unos pocos GHz, y tienen un consumo bastante mayor, del orden de los cientos de mA o varios A.

Dado que los microprocesadores y las memorias se fabrican utilizando tecnologías de circuitos integrados distintas, a menudo no es factible ni económico incorporar en un mismo chip tanto el microprocesador como las grandes cantidades de memoria necesarias. Por estos motivos, se utilizan uno o varios chips de memoria externa al SoC y conectada mediante interfaces de alta velocidad como DDR3 o DDR4 en el caso de la RAM, o como SDIO, UFS o incluso PCIe en el caso de la Flash.

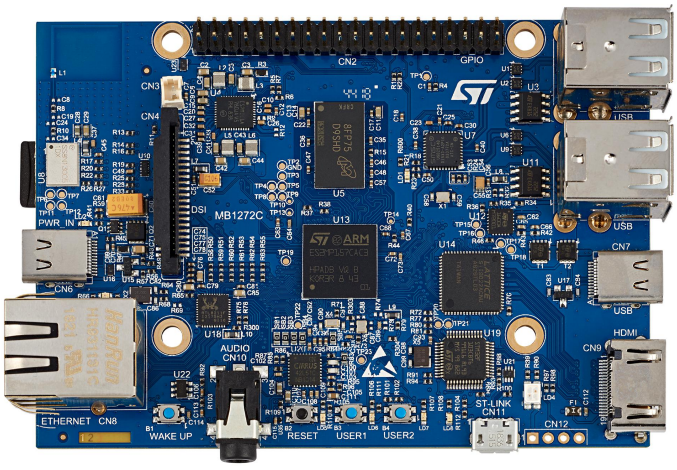


Figura 1.2: Placa STM32MP157A-DK1 basada en un SoC.

Además, estos sistemas suelen conectarse con el mundo exterior por medio de interfaces de mucha mayor velocidad que las que se suelen encontrar en microcontroladores, tales como Ethernet Gigabit, PCIe, HDMI o MIPI CSI/DSI.

Todo esto implica que en el PCB haya una gran cantidad de pistas por las que circulan señales de alta velocidad, además de una gran cantidad de calor generado, complicando bastante el diseño de este tipo de sistemas.

1.2. Motivación

Si bien ambos tipos de sistemas se benefician del uso de buenas prácticas en cuanto al diseño térmico, compatibilidad electromagnética e integridad de señal, en los sistemas del segundo tipo la aplicación de estas técnicas se hace indispensable hasta el punto de que no es posible conseguir un producto que funcione de manera fiable si no se consideran todos estos aspectos desde las primeras etapas del diseño.

La motivación para realizar este trabajo surge tras cursar las asignaturas *Diseño Térmico y Compatibilidad Electromagnética* [1] [2] y *Diseño de Circuitos Impresos* [3] del Máster, en las cuales se han estudiado todos estos conceptos.

Esto se une a que, recientemente, comienza a ser viable fabricar prototipos de sistemas con estas características aun con un presupuesto muy reducido, siempre que se impongan ciertas limitaciones que serán comentadas más adelante [4].

Por todo ello, el objetivo principal de este trabajo es poner en práctica el uso de estas técnicas mediante el diseño, desarrollo, fabricación y testeo de un *Single-Board Computer* (SBC), documentando todas las etapas involucradas en el proceso.

1.3. Estructura del documento

El trabajo realizado se ha dividido en 10 capítulos, cada uno se corresponde con uno de los principales objetivos o partes en las que se ha dividido el trabajo. Los capítulos se han organizado de forma cronológica, siguiendo en medida de lo posible el orden temporal en el que se han ido desarrollando.

Al comienzo de cada capítulo se ha incluido, en general, una breve introducción teórica de los conceptos necesarios, seguida de una explicación sobre cómo se han aplicado dichos conceptos al caso particular del diseño realizado.

1.4. Objetivos por capítulos

- **1. Introducción y objetivos.** Introducir el trabajo, la motivación para realizarlo y sus principales objetivos.
- **2. Estudio previo.** Estudiar y comparar los distintos SoCs y SBCs disponibles en el mercado, seleccionar los principales componentes y el factor de forma.
- **3. Emplazamiento y estudio térmico.** Decidir la ubicación de los principales componentes y estudiar la viabilidad del diseño a nivel mecánico y térmico.
- **4. Red de condensadores de desacoplo.** Diseñar y simular la red de distribución de energía o *Power Delivery Network* (PDN).
- **5. Diseño del esquemático y PCB.** Diseñar los esquemáticos, diseñar el stack-up y el PCB layout. Consideraciones para la reducción de costes y diseño para la fabricación o *Design for Manufacturing* (DFM).
- **6. Estudio de integridad de señal.** Realizar un análisis de integridad de señal y crosstalk de la interfaz de la memoria RAM DDR3L y otras interfaces.
- **7. Estudio de compatibilidad electromagnética.** Estudiar las normativas aplicables al producto, valorar aspectos a considerar para mejorar la compatibilidad electromagnética, realizar la simulación de las protecciones ESD.
- **8. Ensamblado y bring-up del PCB.** Fabricación del PCB, ensamblado de los componentes, *bring-up* o primeras pruebas de funcionamiento.
- **9. Instalación de Linux embebido.** Instalar una distribución de Linux embebido, comprobar el funcionamiento de la memoria y el resto de periféricos.
- **10. Conclusiones.** Comentar las conclusiones y los resultados obtenidos en el trabajo, así como posibles líneas para trabajos futuros.

Capítulo 2

Estudio previo

2.1. El mercado actual

2.1.1. El mercado de los SoCs

En la actualidad existen numerosos fabricantes de SoCs en el mercado. Nos centraremos en los SoCs que son capaces de ejecutar el sistema operativo Linux. Este tipo de SoCs normalmente tienen en común que integran uno o varios núcleos ARM Cortex-A, pero se diferencian en los distintos periféricos que llevan en función de las aplicaciones a las que vayan dirigidos. Algunos ejemplos son:

- En aplicaciones de electrónica de consumo, tales como tablets, set-top boxes, TV boxes o televisiones es común encontrar a fabricantes como Allwinner, Rockchip, Amlogic o Broadcom. Destacan en que suelen incorporar CPUs bastante potentes e incorporan periféricos multimedia tales como GPUs, decodificadores de audio/vídeo, codecs de audio y transceivers HDMI.
- Los SoCs de Qualcomm y Mediatek destacan en la conectividad inalámbrica, se utilizan casi exclusivamente en smartphones y en módulos inalámbricos que posteriormente se integran en otras aplicaciones en las que se requiere de conectividad a la red móvil, además de Wi-Fi, Bluetooth y GNSS. Los destinados a smartphones normalmente incorporan también CPUs bastante potentes además de periféricos multimedia.

- Los fabricados por NVIDIA destacan por la GPU. Se suelen encontrar en aplicaciones en las que los gráficos son importantes, tales como videoconsolas como la Nintendo Switch, y en aplicaciones aceleradas por CUDA, tales como las relacionadas con inteligencia artificial o procesado de imagen.
- En routers y otros dispositivos de red es habitual ver SoCs de marcas como Broadcom, Atheros o Marvell, suelen destacar por incorporar Wi-Fi o Bluetooth y numerosos puertos Ethernet.
- Los fabricados por Altera/Intel y Xilinx incorporan una FPGA o lógica programable posibilitando el desarrollar periféricos y aceleradores completamente personalizados.
- Fabricantes como ST, NXP, Texas Instruments o Microchip disponen de SoCs de propósito más general, destinados a aplicaciones más personalizadas o que se fabrican en menores cantidades. Suelen ser más adecuados para usos industriales, ya que incorporan una gran cantidad y variedad de interfaces tales como puertos serie, CAN o conversores ADC, que no es común encontrar en los anteriores. También suelen soportar mayores rangos de temperatura y tener ciclos de vida más largos, llegando algunos a venderse durante más de 10 años.

En este trabajo utilizaremos un SoC de los de este último punto. Este tipo de SoCs no suelen ser los más potentes, ni los más económicos. Aunque algunos incorporan GPUs e interfaces para LCDs, no suelen incorporar codec de audio ni transceiver HDMI. Esto los hace menos adecuados para aplicaciones multimedia, ya que requieren de otros componentes externos para conectarlos a una televisión, encareciendo el producto.

Por otro lado, incorporan muchas líneas GPIO e interfaces de propósito general. Además, prácticamente son los únicos que se pueden adquirir en pequeñas cantidades directamente en distribuidores occidentales, sin necesidad de firmar grandes acuerdos con los fabricantes. También disponen de toda la documentación fácilmente accesible y publicada en abierto. Esto los hace ideales para este trabajo.

2.1.2. El mercado de los SBCs

Una de las aplicaciones de este tipo de SoCs son los denominados *Single-Board Computers* (SBCs). A diferencia de un PC tradicional que suele tener una arquitectura modular basada en tarjetas de expansión y componentes intercambiables, un SBC consiste en una única placa en la que se suelda el SoC, la memoria RAM, y el resto de periféricos y conectores necesarios para conseguir un equipo funcional.

A pesar de que estos SoCs suelen ser menos potentes que un microprocesador de PC, los SBCs tienen grandes ventajas por su bajo coste, bajo consumo y pequeño tamaño, lo que está provocando que en muchas aplicaciones estén reemplazando a los PCs convencionales. La reciente salida del potente SoC Apple M1 basado en ARM o el reciente anuncio de Windows 11 para ARM, son prueba de que cada vez más PCs puedan acabar siendo reemplazados por este tipo de dispositivos.

Los SBCs no tienen una utilidad muy definida sino que en función del sistema operativo o del software que instale el usuario final, pueden ser utilizados para tareas muy distintas, de forma similar a como se haría con un PC convencional. Sin embargo, como hemos visto anteriormente, los SoCs no suelen diseñarse pensando en su uso en SBCs sino en aplicaciones embebidas mucho más específicas.

Esto hace que el SoC utilizado determine en gran parte la utilidad que se le puede dar al SBC. A modo de ejemplo, compararemos dos SBCs que son bastante populares en la actualidad, pero muy distintos en su enfoque, debido principalmente a las diferencias en los SoCs que utilizan.

2.1.2.1. Raspberry Pi

Es imposible hablar de SBCs sin hablar de la Raspberry Pi. La Raspberry Pi es el SBC más vendido actualmente con diferencia. Originalmente destinado a la enseñanza de informática en las escuelas y países en vías de desarrollo, acabó siendo más popular de lo esperado, siendo utilizado en numerosas aplicaciones en la actualidad.

Debido al gran número de unidades vendidas, son capaces de alcanzar acuerdos con fabricantes como Broadcom, empresa que proporciona el SoC utilizado. Aplicando economías de escala e incluso cierta personalización a nivel de SoC, consiguen comercializar un producto muy potente en relación al bajo precio por el que se vende.

Los SoCs en los que se basan las Raspberry Pi están claramente orientados a aplicaciones de electrónica de consumo. Disponen de una CPU potente, aceleración gráfica y decodificador de vídeo, así como dos transceivers HDMI capaces de manejar dos pantallas con 4K de resolución en esta última versión de la placa. Esto hace que sea un producto muy adecuado sobre todo para aplicaciones multimedia o para un uso cercano al de un PC convencional, conectado a dos monitores o a una televisión.

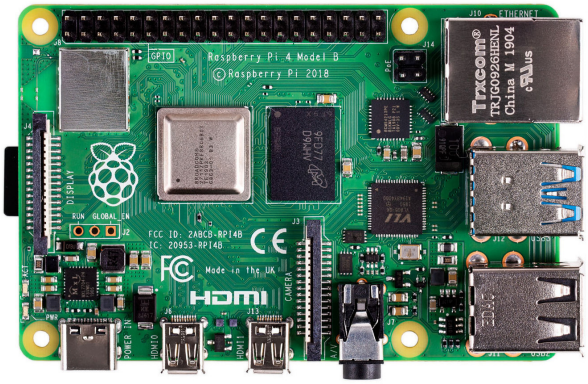


Figura 2.1: Raspberry Pi 4 Model B.

Otra de las utilidades muy populares que tiene es su uso como herramienta de desarrollo y prototipado. Los SBCs suelen disponer de líneas de E/S de propósito general (GPIO) e interfaces serie que permiten su conexión directa a otros componentes electrónicos de forma similar a como se haría con un microcontrolador, más fácil y accesible que con un PC tradicional.

Sin embargo, el SoC utilizado en la Raspberry Pi claramente no estaba enfocado a estas aplicaciones. Por un lado, dispone de avanzados periféricos tales como una GPU, decodificador de vídeo y dos transceivers HDMI. Pero por otro lado, solo dispone de dos canales PWM y unos pocos GPIOs y puertos serie accesibles en el conector de expansión, y no dispone de RTC ni ningún conversor ADC.

Esto hace que cuando se usa en aplicaciones un poco distintas a las que está orientada, a menudo sea necesario incorporarle muchos componentes externos para expandir su funcionalidad, tales como expansores de I/O, conversores ADC externos, codecs de audio o un RTC, periféricos que otros SoCs destinados principalmente a otros mercados traen incorporados en el mismo chip.

También cabe destacar que el SoC que utiliza no dispone de toda la documentación en abierto, ni es posible comprarlo en pequeñas cantidades. Tampoco están publicados los esquemáticos completos, el BOM ni el layout de la placa.

Esto no es muy importante a la hora de construir un prototipo o para realizar un proyecto del que solo se va a construir una unidad, pero hace que disminuya en gran medida su utilidad como placa de desarrollo o evaluación. En caso de querer comercializar un producto basado en un SoC de este tipo será necesario buscar otras alternativas, ya que no sería posible aprovechar el SoC ni otras partes de este diseño.

2.1.2.2. BeagleBone Black

La BeagleBone Black es otro SBC también bastante popular en la actualidad, aunque se ha vendido en mucha menor cantidad que la Raspberry Pi. Representa un enfoque similar en algunos aspectos pero muy distinto en otros.

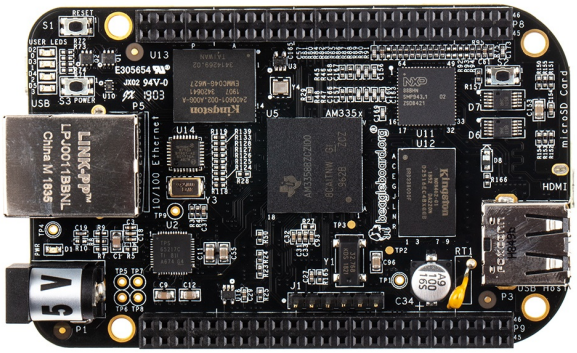


Figura 2.2: BeagleBone Black.

Está basado en un SoC Sitara de Texas Instruments, que podríamos englobar dentro del último punto de la lista de SoCs anterior. El uso de un SoC de este tipo, así como su menor número de ventas provocan que, aun siendo un dispositivo con una CPU considerablemente menos potente que la de la Raspberry Pi, se comercialice por un precio superior.

El SoC utilizado no destaca por sus capacidades multimedia. Si bien dispone de una GPU, esta es menos potente y no dispone de decodificadores de vídeo ni transceiver HDMI. Un chip adicional en la placa se encarga de la conversión entre la salida de vídeo RGB paralela y el conector HDMI, lo que contribuye a incrementar su coste, al contrario del SoC utilizado en la Raspbery Pi que integra los transceivers HDMI en el mismo SoC. También cabe destacar que incorpora un menor número de conectores USB y no dispone de conectividad inalámbrica.

Sin embargo, el SoC dispone de otros periféricos e interfaces que no se encuentran disponibles en la Raspberry Pi, tales como ADCs, RTC, distintos buses industriales como PROFIBUS y CAN, un mayor número de interfaces serie, por mencionar algunos. Además, la placa incorpora elementos como una memoria eMMC, mayor número de pines en los conectores de expansión, así como varios LEDs y botones que pueden ser controlados por el usuario. Esto hace que pueda ser más adecuada que la Raspberry Pi en otro tipo de aplicaciones, por ejemplo, las industriales.

Otro aspecto a tener en cuenta es que dispone de un enfoque mucho más abierto que el de la Raspberry Pi. Todas las partes que utiliza, incluido el SoC, son componentes *off-the-shelf* los cuales pueden adquirirse en distintos distribuidores en pequeñas cantidades y disponen de toda la documentación muy accesible y publicada en abierto, sin necesidad de firmar acuerdos con los fabricantes.

El propio diseño del SBC también es *open-source*, estando disponibles todos los archivos fuente tales como el esquemático, el BOM y el layout del PCB, publicados bajo una licencia que permite su reutilización en otros proyectos. Esto facilita mucho su uso como placa de desarrollo o evaluación, permitiendo comenzar el desarrollo de un nuevo producto sobre ella, para finalmente acabar migrando a una placa a medida de la aplicación que podría estar basada en el mismo SoC, pudiendo reutilizar así gran parte del diseño tanto a nivel de hardware como de software.

Todo esto también la hace más adecuada para la enseñanza, especialmente a nivel de hardware o de Linux embebido, al ser un sistema más abierto y disponer de más documentación en este sentido, lo que facilita su estudio.

2.2. Objetivos del diseño

El enfoque de la placa a diseñar en este trabajo se asemeja más al de la BeagleBone Black que al de la Raspberry Pi, aunque toma aspectos de ambas, con algunas diferencias. Los principales objetivos del diseño a realizar son los siguientes:

- Debe tener un mínimo de complejidad que lo haga relevante de cara a la realización del trabajo, de forma que en el proceso de diseño sea posible aplicar y documentar los principios y técnicas que se querían estudiar y poner en práctica. Esto implica utilizar un SoC relativamente potente, con un encapsulado de tipo *Ball Grid Array* (BGA) y memoria DDR3 o DDR4.
- Utilizar solamente componentes *off-the-shelf*, que puedan adquirirse en pequeñas cantidades y cuya documentación esté publicada en abierto, sin necesidad de firmar acuerdos con el fabricante.
- Enfoque más industrial que multimedia. Incorporar conectividad como CAN, Ethernet y USB. De esta forma podría tener aplicaciones a modo de *gateway*, por ejemplo permitiendo comunicar una red CAN con una red Ethernet sin necesidad de componentes adicionales.
- Tratar de hacer accesibles todas las interfaces disponibles en el SoC pero que no se utilicen en la misma placa. De esta forma sería posible darle más utilidades conectándole una placa de expansión.
- Como resultado, disponer de una placa de desarrollo versátil sobre la que se puedan prototipar otros desarrollos dentro de una gran variedad de aplicaciones, pudiendo servir como base o punto de partida para futuros proyectos.

2.2.1. Reducción de costes

Otro de los objetivos del diseño ha consistido en mantener un coste de fabricación lo más bajo posible, especialmente hablando de la fabricación en tiradas muy pequeñas, del orden de unas 10 unidades. Este ha sido uno de los aspectos que más influencia ha tenido en el diseño. En este sentido, se han impuesto varias limitaciones:

- En cuanto al PCB, se tratará de usar componentes cuyo encapsulado permita el uso de un fabricante de PCBs económico sin necesidad de grandes prestaciones ni el uso de procesos muy avanzados. El diseño se realizará en solamente 4 o 6 capas. Se estudiarán ambas alternativas y la viabilidad de fabricarlo solamente con 4 capas. En capítulos posteriores se comentarán con más profundidad las consideraciones tomadas en este aspecto.
- En cuanto a los componentes, se tratará de abaratar y simplificar lo máximo posible el diseño, utilizando el menor número posible de componentes aunque sea a costa de perder algo de funcionalidad. Se intentará también reducir al máximo el número de líneas del BOM, evitando usar componentes distintos cuando sea posible utilizar varias unidades de la misma referencia.
- En cuanto al ensamblado, se facilitará lo máximo posible el montaje de los componentes. La mayoría de componentes se colocarán en la cara superior, especialmente los más pesados, dejando la cara inferior solamente para condensadores de desacoplo y otros pasivos que no quepan en la superior, evitando la necesidad de usar adhesivos para soldar ambas caras del PCB. Los pasivos serán 402 como mínimo y se mantendrá suficiente espacio entre los componentes para facilitar su colocación manual o con máquinas *Pick and Place* de baja precisión.

2.2.2. Selección del factor de forma

Se ha decidido utilizar un factor de forma similar al de la Raspberry Pi, respetando especialmente las dimensiones exteriores y el *pinout* del conector de expansión de 40 pines. De esta forma automáticamente el diseño se hace compatible con la inmensa cantidad de *shields* o placas de expansión ya existentes para esta plataforma.

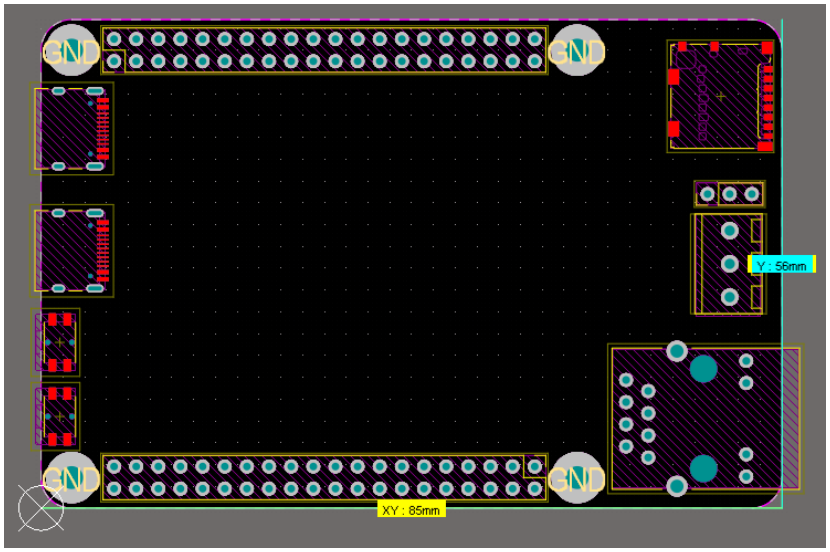


Figura 2.3: Factor de forma del SBC.

Además, se añade en la parte inferior un segundo conector de expansión de 40 pines, en el que quedarán accesibles el resto de señales del SoC que puedan resultar de interés, tales como la depuración JTAG, el audio I2S, la interfaz CSI para cámara o la interfaz LCD paralela. Estos pines también podrán usarse como líneas E/S de propósito general (GPIO) en caso de no necesitar dichas interfaces.

Esto permite simplificar y reducir los costes de la placa principal, a la vez que se deja abierta la posibilidad de ampliar la funcionalidad mediante placas de expansión. De este modo, en la placa de expansión sería posible incorporar elementos como una pantalla táctil, una cámara, codecs de audio y salida de vídeo VGA o HDMI, según necesite una aplicación en particular.

2.3. Selección de los componentes

2.3.1. Selección del SoC

En este tipo de producto la elección del microprocesador o SoC es la más importante, ya que condiciona en gran medida el resto del diseño. Se han valorado los distintos SoCs comercializados por ST, NXP, Texas Instruments y Atmel/Microchip. Finalmente se ha optado por utilizar un i.MX6ULL de NXP. A continuación, se detallan los criterios que se han tenido en cuenta y que han llevado a seleccionar este SoC.

2.3.1.1. Encapsulado y su influencia en el PCB

Uno de los requisitos del diseño es el uso de un SoC relativamente potente, con un chip de memoria RAM externa. A menudo este tipo de dispositivos solo se encuentran disponibles en encapsulados BGA, debido al gran número de pines de los que disponen, así como las mayores dimensiones del chip.

Los encapsulados BGA requieren de la colocación de vías entre los pads del chip, muy cercanas entre sí. También es necesario que las pistas sean capaces de pasar entre los distintos pads y entre las vías, así como el uso de múltiples capas para poder escapar las señales conforme nos adentramos en los pads más internos del chip.

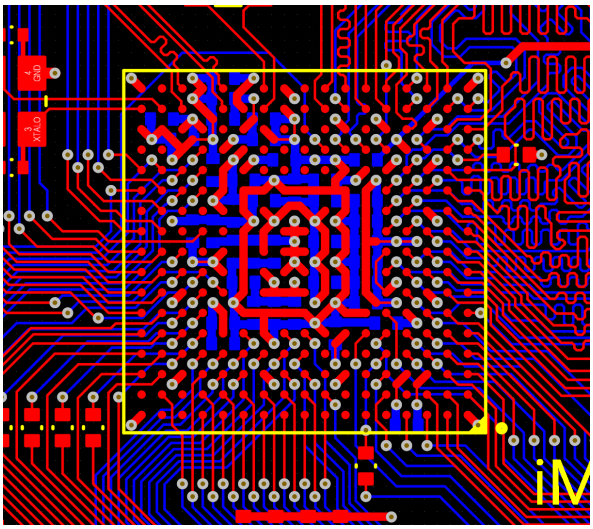


Figura 2.4: Fanout del SoC realizado en este trabajo.

Esto hace que el SoC sea el componente más exigente de todo el diseño, en cuanto a las especificaciones que debe cumplir el PCB para permitir rutar y escapar todas las señales adecuadamente. Dicho de otra forma, el coste de fabricación del PCB dependerá en gran medida de las características de este encapsulado.

A menor *pitch* o distancia entre centros de los pads, mejores especificaciones necesita el PCB. También se complica más el montaje, ya que se reduce el error admisible a la hora de colocar el componente. En el mercado se encuentran SoCs con un pitch de 0.8, 0.65 o 0.5mm. Nos limitaremos a los que utilizan un pitch de 0.8mm. Esto ya pone al límite las capacidades de los fabricantes de PCBs más económicos.

El *pinout* o la distribución de las señales en cada pin determina un mínimo de capas a utilizar. Como regla general, por cada dos filas que se quieran escapar alrededor de todo el chip hace falta una capa de señal adicional. Si un encapsulado está pensado para su uso en placas de pocas capas, las señales más imprescindibles estarán en los pines más exteriores, reservando los interiores para alimentación y masa.

Figura 2.5: Encapsulados BGA del i.MX6ULL.

El i.MX6ULL ha resultado ser bastante adecuado en estos aspectos. Está disponible en uno de los encapsulados más asequibles que se han encontrado, con 0.8mm de pitch y solo 289 pines. Casi todas las señales necesarias se encuentran en las 4 filas más exteriores, permitiendo escaparlas con tan solo 2 capas de señal. También es posible escapar algunas señales necesarias de la quinta fila sin añadir capas adicionales gracias a que algunos pads adyacentes no se utilizan o están destinados a masa.

Como contra, la distribución de pines del controlador de RAM no está muy bien organizada, lo que dificulta el rutado de la memoria respecto a otros SoCs en los que las conexiones desde los pads del SoC a los de la RAM son mucho más directas.

2.3.1.2. Prestaciones, periféricos y precio

En este caso se buscaba un procesador ARM Cortex-A relativamente potente. El i.MX6ULL incorpora un único núcleo ARM Cortex-A7 funcionando a una frecuencia de hasta 900MHz, lo cual es más que suficiente para las aplicaciones a las que estaría destinado este SBC. También incorpora un set de periféricos bastante genérico y versátil, permitiendo su uso en una gran variedad de aplicaciones.

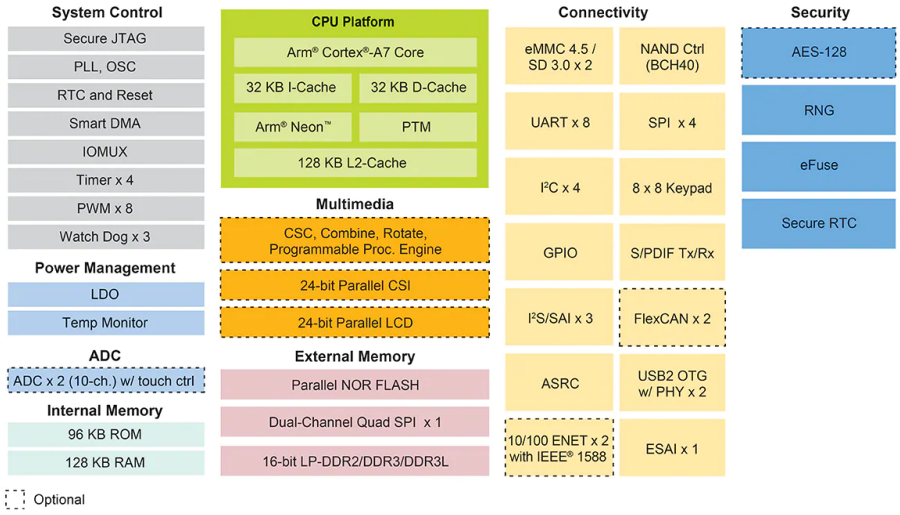


Figura 2.6: Diagrama de bloques del NXP i.MX6ULL.

Este microprocesador es de los más potentes que hay disponibles si tenemos en cuenta las limitaciones que nos hemos impuesto en cuanto a costes y al encapsulado. Se ha valorado el uso de otros SoC más potentes de este mismo fabricante, pero habría implicado el uso de un BGA de menor pitch y más numero de pads, además de más raíles de alimentación, complicando y encareciendo la fabricación del PCB.

Otra alternativa que se ha valorado es el STM32MP1 que también está disponible con un pitch de 0.8mm y en distintas configuraciones con uno o dos núcleos Cortex-A7 a 800MHz, así como una GPU y un coprocesador Cortex-M4 adicional. Si se fuesen a aprovechar ambos núcleos, la GPU o el Cortex-M4, el STM32MP1 sería una opción muy interesante a considerar.

Finalmente se ha escogido el i.MX6ULL debido a que en aplicaciones monohilo y que no usen la GPU cabe esperar que rinda de manera similar o superior, a la vez que consume menos y es más sencillo alimentarlo. También es más económico y dispone de mejor soporte al llevar más tiempo en el mercado.

Allwinner y Rockchip también disponen de SoCs muy potentes a bajo precio con encapsulados BGA de 0.8mm o incluso QFP, pero para este proyecto se han acabado descartando estos fabricantes debido a su menor disponibilidad y documentación.

2.3.1.3. Esquema de alimentación y consumo

A diferencia de los microcontroladores que suelen alimentarse con un único voltaje, estos SoCs requieren de varios raíles de alimentación a voltajes distintos, que además deben respetar cierto secuenciamiento durante el encendido. También tienen un consumo de corriente considerablemente mayor que el de los microcontroladores.

Por todo esto se suelen utilizar junto a un *Power Management IC* (PMIC). Los PMICs habitualmente se diseñan a medida pensando en su uso con un SoC o familia de SoCs concreta. En un mismo chip incorporan distintos reguladores que se encargan de proporcionar todas las tensiones necesarias y con su correspondiente secuenciamiento a la hora de encender, apagar o cambiar el estado del sistema.

Algunos modelos son capaces de comunicarse por I2C con el SoC y realizan un escalado del voltaje dinámico que permite variar el voltaje de alimentación del SoC en función de la frecuencia del microprocesador, así como desconectar algunos de los reguladores cuando el SoC entra en modo de bajo consumo. Estas funciones permiten ahorrar energía si baja la carga del microprocesador.

El inconveniente de estos PMICs es que son componentes especializados y relativamente caros en comparación con los reguladores convencionales. El i.MX6ULL ha resultado ser ventajoso en este aspecto ya que dispone de varios LDOs integrados requiriendo solo de dos tensiones distintas. Además, si se combina con una memoria DDR3L a 1.35V, es posible alimentar toda la placa con solo dos sencillos reguladores a 1.35V y 3.3V, sin necesidad de PMIC, simplificando enormemente el esquema de alimentación de la placa en comparación al de otros SoCs considerados [5].

2.3.1.4. Soporte, disponibilidad y future-proofing

Como se ha comentado anteriormente, para este trabajo solo se han valorado SoCs que puedan ser adquiridos en pequeñas cantidades y que dispongan de toda su documentación publicada de manera abierta sin necesidad firmar acuerdos con el fabricante. También es importante comprobar si el fabricante garantiza su disponibilidad durante los años necesarios para cubrir el ciclo de vida del producto.

Otro aspecto a considerar es su soporte en Linux. La familia del i.MX6ULL lleva tiempo en el mercado y dispone de buen soporte incluso en la rama principal de Linux, haciendo innecesario el uso de versiones del kernel parcheadas por el fabricante. Los drivers son open-source y no necesita de *binary blobs* para su funcionamiento.

Otro criterio importante a la hora de seleccionar un microprocesador es su compatibilidad con otros modelos dentro de la misma gama, tanto a nivel de encapsulado y eléctrico como a nivel de software y periféricos. De esta forma, en caso de problemas de stock de un modelo concreto, o si cambian las especificaciones del proyecto, sería posible cambiar a un modelo distinto pero compatible pin a pin.

En el caso del i.MX6, existen tres modelos distintos (i.MX6ULZ, i.MX6ULL, i.MX6UL) y diferentes submodelos, todos ellos compatibles entre sí. Se diferencian en el precio, frecuencia de funcionamiento y periféricos incluidos. Esto favorece el reutilizar gran parte de un mismo diseño para un rango de productos muy distintos.

2.3.2. Selección de la memoria RAM

2.3.2.1. Memoria SRAM

La memoria SRAM (Static RAM) está formada por celdas construidas con varios transistores (flip-flops) que almacenan cada bit almacenado. Esto hace que sea una memoria muy rápida, con bajo tiempo de acceso y muy sencilla de manejar a la hora de direccionarla y realizar lecturas y escrituras. Tiene un consumo muy reducido cuando está en reposo, pero mucho mayor durante los procesos de lectura o escritura cuando se producen las conmutaciones de los transistores.

Es común verla integrada en el mismo chip que el microprocesador aunque también se comercializa en chips individuales con capacidades que llegan hasta los 8MB. Su principal inconveniente es que es muy poco densa por lo que su coste/bit es muy alto, solo es factible utilizarla cuando se necesita en capacidades muy pequeñas.

En microcontroladores e incluso en FPGAs es común utilizarla como la única memoria RAM del sistema. Sin embargo, en los SoCs que estamos valorando en este trabajo se utiliza únicamente como caché integrada en el mismo SoC, ya que su capacidad no es suficientemente grande para usarla como la memoria principal.

2.3.2.2. Memoria DRAM

La memoria DRAM (Dynamic RAM) está formada por celdas construidas con un transistor y un condensador. El condensador puede estar cargado o descargado, lo que representa los dos estados del bit. Necesita circuitería adicional que refresque periódicamente los condensadores para evitar que se pierdan los datos, complicando la gestión de la memoria lo que conlleva peores tiempos de acceso que la memoria SRAM. Debido al continuo refresco el consumo es superior al de la SRAM cuando está en reposo, pero no se ve tan incrementado durante los ciclos de lectura o escritura.

La principal ventaja es su mayor densidad, permitiendo alcanzar capacidades mucho mayores a menor coste. Actualmente en el mercado se encuentran en capacidades de hasta 512MB por chip (2GB en chips multi-die) en DDR3 y 2GB por chip (8GB en chips multi-die) en DDR4.

Al contrario que la SRAM, no es común verla integrada en el mismo chip que el microprocesador, ya que se fabrican utilizando tecnologías de circuitos integrados distintas. Los *System-In-Package* (SiPs) integran en un mismo encapsulado tanto el die del SoC como el die de la RAM, pero por lo general las capacidades son menores y tienen un coste más alto que su equivalente en chips discretos.

Otro de los aspectos a tener en cuenta es el uso de chips multi-die. En ocasiones, a igualdad de capacidad es más económico utilizar varios chips de un solo die que un único chip multi-die. El inconveniente es que el uso de varios chips complica bastante la topología del rutado de la interfaz entre las memorias y el SoC, por lo que en caso de necesitar grandes capacidades, a veces puede compensar gastar más en un chip multi-die con tal de simplificar el PCB y reducir el área ocupada.

Actualmente la DDR4 tiene menor coste/MB que la DDR3, pero utilizaremos DDR3 ya que es la memoria soportada por el SoC. En concreto utilizaremos un único chip de memoria DDR3L capaz de funcionar a 1.35V en lugar de los 1.5V de la DDR3 convencional, lo que nos permitirá usar un único regulador a unos 1.35V para alimentar tanto el SoC como la DRAM. El diseño será compatible con chips DDR3L de 512MB y 1GB, lo que es más que suficiente para esta CPU. Por lo general, aplicaciones que pudiesen requerir de más RAM también requerirían un microprocesador más potente.

2.3.3. Selección de la memoria Flash

Actualmente en el mercado hay muchos tipos distintos de memoria no volátil, a continuación se comentan sus características así como sus ventajas e inconvenientes.

2.3.3.1. Memoria EEPROM

En la actualidad, sus principales ventajas son su bajo coste, alta fiabilidad y que sus celdas soportan una gran cantidad de ciclos de escritura, del orden de los 100K a 1M de ciclos. Son muy sencillas de manejar, ya que estas celdas pueden ser sobrescritas a nivel de byte, sin necesidad de sobrescribir páginas enteras como ocurre con las memorias Flash. Se suelen conectar mediante interfaces I2C y SPI.

Su inconveniente es su baja velocidad y que solo se encuentran en capacidades muy pequeñas, de unos 512KB como máximo, lo que las hace inviables para almacenar el sistema operativo o las aplicaciones. Se utilizan cuando es necesario almacenar de forma fiable y no volátil pequeños datos que son sobrescritos muy frecuentemente. Un ejemplo sería el cuenta kilómetros total y parcial de un coche.

2.3.3.2. Memoria NOR Flash

Estas memorias se encuentran en mayores capacidades, de hasta unos 128MB por chip. Se suelen conectar mediante interfaces serie SPI y QSPI, aunque también están disponibles con interfaz paralela. Son también memorias muy fiables aunque la durabilidad es menor, del orden de los 10K a 100K ciclos de escritura por celda.

Aunque se pueden direccionar a nivel de byte, están organizadas en páginas de varios KB, y a la hora sobrescribir un byte ya escrito es necesario borrar de antemano toda la página. Para no perder todos los datos almacenados en esa misma página es necesario copiarlos antes a otra memoria de forma temporal mientras se realiza el borrado, para posteriormente copiarlos de vuelta junto al nuevo dato. Esto hace que sean memorias muy lentas y complicadas de gestionar en cuanto a la escritura.

Sin embargo, son memorias bastante rápidas en lectura y con bajo tiempo de acceso. Por ejemplo una memoria QSPI a 133MHz puede llegar a alcanzar velocidades de unos 6MB/s en lectura aleatoria y 60MB/s en lectura secuencial. Además, se encuentran en capacidades suficientemente grandes como para almacenar en ellas el sistema operativo así como el resto de programas necesarios en algunas aplicaciones.

Es utilizada especialmente en dispositivos de bajo coste como routers, en los que se utiliza como la única memoria Flash del sistema. Durante el arranque, se aprovecha su alta velocidad de lectura secuencial para copiar la imagen del sistema al completo a la memoria RAM, y a partir de ahí el SO se ejecuta directamente desde la RAM sin realizar ninguna escritura en la Flash salvo para salvar configuraciones.

También es popular su uso en sistemas de alta fiabilidad. Estos sistemas pueden almacenar en la memoria NOR Flash el SO o algún sistema de recuperación de manera más fiable, dejando la memoria NAND solo para almacenar archivos de gran tamaño. De esta forma en caso de que se corrompa o se dañe la memoria NAND sigue siendo posible arrancar el sistema y tratar de recuperarlo o avisar del fallo.

2.3.3.3. Memoria NAND Flash

La principal ventaja respecto a la memoria NOR es su mayor densidad, estando disponible en capacidades mucho mayores y a menor precio. Está disponible en interfaces serie aunque es más común su uso con interfaces paralelas. En escritura es mucho más rápida que la NOR Flash, lo que la hace mucho más indicada que esta cuando se necesitan almacenar grandes cantidades de datos y a velocidades altas.

En lectura sin embargo es inferior a la NOR Flash, especialmente en cuanto a tiempos de acceso, lo que hace que sea inviable ejecutar código directamente desde ella sin antes copiarlo a otra memoria más rápida. Otro gran inconveniente es su menor fiabilidad. En las memorias NOR Flash todas las celdas se mantienen intactas a lo largo de toda su vida útil. En cambio, en la NAND Flash es común la presencia de bloques defectuosos que continúan apareciendo durante el uso normal de la memoria.

Esto complica bastante la gestión de la memoria, ya que es necesario cierta funcionalidad de corrección de errores y *wear leveling*, detectando bloques defectuosos y marcándolos como inutilizables. En el caso de la memoria NAND convencional, esta gestión debe realizarla por software el SoC, obligando a usar un sistema de archivos específico que tenga en cuenta todos estos aspectos. En la actualidad el uso de estas memorias se está reduciendo a favor de las memorias NAND gestionadas.

2.3.3.4. Memoria NAND gestionada

Son memorias NAND similares a las anteriores, con la diferencia de que en el mismo chip incluyen un controlador que se encarga de gestionar el wear leveling y la corrección de errores de manera transparente. Así, el software se despreocupa de estos aspectos y es posible utilizar estas memorias con sistemas de archivos convencionales. Este tipo de memorias se encuentran en distintos formatos e interfaces.

Uno de los formatos más populares actualmente son las memorias eMMC y SD. Las eMMC están disponibles en capacidades entre 2 y 128GB, mientras que las SD se encuentran en capacidades incluso más altas. Ambos formatos son similares y bastante compatibles entre sí, con la diferencia de que las memorias eMMC van en un encapsulado para soldar en el PCB y las memorias SD tienen forma de tarjeta extraíble.

Otro formato que normalmente queda reservado para los SoCs más potentes son las memorias UFS. Es común encontrarlas en los smartphones actuales. En UFS hay memorias disponibles entre 32 y 256GB que alcanzan velocidades de hasta 600MB/s.

En caso de necesitar mayores capacidades y velocidades y si el SoC dispone de interfaz PCI Express, es posible dar el salto a un SSD con formato NVMe. Estos discos incorporan en una tarjeta M.2 varios chips de memoria Flash junto a un controlador y a menudo también un chip de memoria DDR3 o DDR4 que se utiliza como caché.

2.3.3.5. Memorias selecionadas

El SoC utilizado en este trabajo es compatible con memorias NOR Flash, NAND Flash, eMMC y SD. Se ha evitado utilizar memorias NAND sin gestionar, así como memorias eMMC ya que solo están disponibles con un pitch de 0.5mm. En su lugar se utilizará un zócalo para tarjetas microSD, ya que es lo más económico.

También se incorporará una memoria NOR Flash de pequeña capacidad (16MB) conectada por QSPI. El sistema operativo podrá arrancar tanto de la memoria QSPI como de la microSD, según la posición de un pequeño interruptor en la placa.

2.3.4. Selección de los reguladores

Estos SoCs tienen un consumo considerablemente mayor que los microcontroladores, lo que hace casi imprescindible alimentarlos con reguladores conmutados en lugar de lineales. Como ya se ha comentado anteriormente, en lugar de un PMIC se ha decidido utilizar dos reguladores discretos. En un principio se ha valorado usar dos reguladores de salida fija a 1.35V y 3.3V. El inconveniente de este tipo de reguladores es que al ser más especializados tienen menor disponibilidad.

Por otro lado, en la línea de 1.35V realmente se necesita un voltaje ligeramente superior, de al menos 1.375V, para garantizar el funcionamiento del SoC a 900MHz según las especificaciones del fabricante. Por estos motivos finalmente se ha decidido utilizar dos TLV62569 de Texas Instruments capaces de entregar 2A cada uno.

Estos reguladores disponen de salida variable configurable con dos resistencias externas. Se han seleccionado los valores de forma que obtenemos un voltaje típico de 1.383V y 3.377V respectivamente. Dadas las tolerancias de la referencia interna del regulador y de las resistencias utilizadas, en el peor caso podríamos esperar una tensión en el rango 1.341 a 1.427V en el primer regulador y 3.255 a 3.502V en el segundo.

Estos reguladores tienen un encapsulado SOT-23-5 con un pinout bastante estándar. Esto permite que, en caso de falta de stock, puedan ser reemplazados por equivalentes del mismo u otros fabricantes, como el TLV62568 o el RT8059.

2.3.5. Otros componentes

Dados los problemas de aprovisionamiento que hay actualmente, el resto de componentes se han seleccionado siguiendo criterios sobre todo de precio y disponibilidad, así como compatibilidad con drivers existentes en Linux en el caso del transceiver Ethernet.

Se ha dado prioridad a componentes con stock abundante en distintos distribuidores, así como a componentes con encapsulados estándar, que sean fabricados por distintas marcas o de los que haya varias referencias compatibles entre sí o con la misma huella. De este modo si un componente deja de estar en stock se puede sustituir por otro sin cambios en el PCB.

2.4. Especificaciones finales del SBC

A continuación se resumen las especificaciones del diseño realizado.

CPU y memoria

- CPU: NXP i.MX6ULL (ARM Cortex-A7 @ 900MHz).
- RAM: 512MB/1GB DDR3L @ 400MHz (DDR3L-800).
- Flash: 16MB NOR Flash QSPI @ 104MHz.
- Conector para tarjeta microSD.
- Arranque configurable desde bootloader interno, QSPI Flash o microSD.

Alimentación

- Entrada de alimentación a 5V mediante conector USB-C.
- 2x Reguladores TLV62569 (1.35V, 3.3V) que proporcionan hasta 2A cada uno.
- Líneas de 5 y 3.3V accesibles en el conector de expansión.

Conectividad

- 2x Conector USB-C 2.0 High-Speed (480Mbps)
- 1x Conector Ethernet 10/100Mbps (transceiver LAN8720A).
- 1x Conector CAN 1Mbps (transceiver SN65HVD230).

Periféricos y expansión

- LED y pulsador controlables por el usuario.
- 2x Header de expansión incluyendo 60x GPIO, 4x ADC 12-bit, 5x UART, 4x I2C, 2x SPI, 2x I2S/SAI, 1x LCD 24-bit Interface, 1x Camera 8-bit Interface.
- Header superior compatible con placas de expansión de Raspberry Pi.

Parte II

DESARROLLO DEL PROYECTO

Capítulo 3

Emplazamiento y estudio térmico

3.1. Introducción

Una vez determinadas las dimensiones del PCB así como los principales componentes a utilizar, el objetivo de este capítulo es decidir la ubicación de estos componentes y comprobar en una etapa temprana que el diseño sea viable mecánica y térmicamente antes de invertir más tiempo en el esquemático y en el rutado del PCB.

3.1.1. Objetivo térmico del diseño

Debemos fijar de antemano el valor de temperatura ambiente y los valores máximos de temperatura que podemos tolerar en las distintas partes del producto, lo que dependerá en gran medida de la aplicación y usos a los que vaya destinado:

- **Temperatura ambiente:** Es necesario determinar el rango de temperatura ambiente en el que funcionará el dispositivo, especialmente la temperatura máxima. Fijaremos el valor máximo de temperatura ambiente en 35°C que es un valor típico para electrónica de consumo de interior. En otro tipo de aplicaciones este valor podría ser bastante más elevado.
- **Temperatura del PCB:** Es importante no superar la temperatura de transición vítrea (Tg) del PCB. A partir de esta temperatura el material base empieza a transformarse a un estado más blando y elástico pudiendo deformarse. En este caso el FR-4 que utiliza el fabricante tiene una Tg de 155°C.

- **Temperatura de los componentes:** Es necesario no superar la temperatura máxima que soporta cada componente. Por lo general esta será de unos 85°C, aunque algunos componentes como los osciladores son más delicados.
- **Temperatura en zonas de contacto:** Es importante fijar las temperaturas máximas en las zonas del PCB que puedan estar en contacto con otros elementos o que puedan ser tocadas o manipuladas por el usuario.

Llegado a este punto tenemos fijada la temperatura fría del sistema (el valor máximo de temperatura ambiente), así como el valor máximo de temperatura que podemos tolerar en cada punto. A partir de aquí será necesario diseñar el sistema de tal forma que pueda disipar el calor generado al ambiente a la vez que se mantienen dichas temperaturas por debajo de sus respectivos límites.

3.2. Emplazamiento de los componentes

Dadas las limitaciones que se han impuesto como tratar de utilizar solo dos capas de señal, así como situar los conectores y pines en lugares específicos para respetar el factor de forma de la Raspberry Pi, no hay mucho margen a la hora de situar los componentes. Si no se sitúan de la forma más óptima, se hace muy difícil rutar el diseño sin añadir más capas de señal.

Para facilitar la fabricación se ha dado prioridad a ubicar la mayor cantidad posible de componentes en la cara superior, sobre todo los más grandes y pesados. En la cara inferior solo se han situado pasivos como algunos condensadores de desacoplo.

Se ha planificado con antelación dónde colocar los principales componentes y por dónde pasar las pistas de las distintas interfaces, con el fin de ver si el diseño era viable en 4 capas. Los componentes se han distribuido por la placa tratando de que las distintas interfaces se puedan rutar en su totalidad o bien por la capa superior, o bien por la capa inferior, sin que haya cruces entre interfaces de la misma capa.

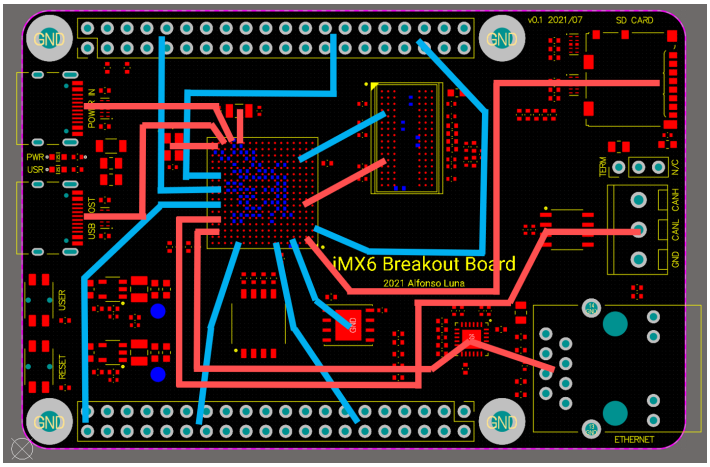


Figura 3.1: Emplazamiento y planificación del rutado.

Se ha valorado girar el SoC 90°a la izquierda, ya que así se acortarían todas las pistas de la capa superior. El problema es que entonces es necesario ubicar la memoria en la zona superior, y SoC y memoria ocupan casi toda la vertical de la placa, dificultando rutar la gran cantidad de pistas que van hacia los conectores de expansión por la cara inferior. Todas las señales que van hacia el conector superior saldrían de la parte inferior del micro y tendrían que dar toda la vuelta. Lo mismo ocurriría con todas las señales del LCD que ahora tienen conexión directa al conector inferior.

3.2.1. Equilibrio entre requisitos eléctricos y térmicos

Otro aspecto a tener en cuenta en el emplazamiento de los componentes es encontrar un equilibrio entre los requisitos eléctricos y los térmicos.

A nivel eléctrico, es preferible que los componentes estén lo más juntos posibles, mejorando así los aspectos relacionados con la compatibilidad electromagnética e integridad de señal. Sin embargo, a nivel térmico, conviene separar lo máximo posible los componentes que más consumen, consiguiendo así repartir mejor el calor en el PCB evitando puntos calientes.

En este caso, dadas las pequeñas dimensiones del PCB, no había mucho margen de elección en este aspecto. Se ha tratado de colocar el SoC y la RAM lo más cerca posible el uno del otro, manteniendo un mínimo de separación para facilitar el rutado de la interfaz y su *length-matching*. Los reguladores se han situado en una esquina del PCB, lo más lejos posible de estos componentes.

3.3. Introducción a la transferencia de calor

El calor se transfiere por medio de tres procesos distintos: conducción, convección y radiación. A continuación, se introducirán las nociones básicas de cada proceso y algunas particularidades para el caso concreto de un PCB.

3.3.1. Transferencia de calor por conducción

La conducción de calor está basada en el contacto directo entre los cuerpos, sin intercambio de materia. Dados dos cuerpos en contacto, el calor fluye desde el cuerpo de mayor temperatura al de menor temperatura. La conducción del calor viene determinada por la Ley de Fourier. Una vez estamos en régimen estacionario y si consideramos una única dirección, nos queda la siguiente expresión:

$$Q = k \cdot \frac{A}{L} \cdot \Delta T$$

Es decir, el flujo de calor Q (W) a través de un material es proporcional a la superficie perpendicular al flujo de calor A (m^2), a la conductividad térmica del material k $(\frac{W}{m \cdot K})$ y a la diferencia de temperatura en los extremos del mismo ΔT (°C), y es inversamente proporcional al espesor L (m).

La conductividad térmica k es la propiedad de los materiales que determina su capacidad para conducir el calor, a mejor conductor mayor será su conductividad térmica. En el caso del PCB, los materiales habituales que consideraremos serán:

- $k = 360$ $\frac{W}{m \cdot K}$ para el cobre, muy buen conductor.
- $k = 0.3$ $\frac{W}{m \cdot K}$ para el FR-4 y solder mask, muy malos conductores.
- $k = 0.02$ $\frac{W}{m \cdot K}$ para el aire, despreciable como conductor.

3.3.2. Transferencia de calor por convección

La convección es la transmisión del calor por el movimiento de la propia materia portadora del calor. Por ejemplo, al calentarse el aire en contacto con el PCB, este tiende a ascender, mientras que el aire frío desciende por los lados y ocupa el lugar que dejó el aire caliente. Requiere por tanto de un medio o fluido para la transferencia, que en nuestro caso habitualmente será el aire.

La transferencia de calor por convección está determinada por:

$$Q = h \cdot A \cdot \Delta T$$

Donde h ($\frac{W}{m^2 \cdot K}$) es el coeficiente de convección, A (m^2) es el área de contacto entre la superficie y el fluido, y ΔT (°C) es la diferencia de temperatura entre la superficie y el fluido.

Podemos diferenciar entre convección natural y convección forzada. En la convección natural el movimiento del fluido se debe a causas naturales, como la subida del fluido caliente y el descenso del fluido frío. En convección forzada se obliga al fluido a circular, por ejemplo con un ventilador. En este caso usaremos convección natural.

El coeficiente de convección h es muy complicado de estimar con precisión, ya que depende de la mecánica y movimientos del fluido así como de la geometría de cada problema. Para obtener un valor de h preciso sería necesario realizar complejas simulaciones CFD. Sin embargo, para un PCB de este tipo podemos tomar como referencia estos valores obtenidos a partir de correlaciones y experimentalmente:

- h = 4-8 $\frac{W}{m^2 \cdot K}$ para convección natural.
- h = 8-15 $\frac{W}{m^2 \cdot K}$ para convección natural, incluyendo también la radiación.
- h = 15-25 $\frac{W}{m^2 \cdot K}$ para convección forzada con un flujo de aire de 1m/s.
- h = 25-50 $\frac{W}{m^2 \cdot K}$ para convección forzada con un flujo de aire de 2m/s.

3.3.3. Transferencia de calor por radiación

La radiación es la transmisión de calor por medio de la emisión de ondas electromagnéticas o fotones. Todos los cuerpos que están a una temperatura mayor que el cero absoluto emiten calor a su entorno mediante la siguiente expresión:

$$Q = \varepsilon \cdot \sigma \cdot A \cdot (T_s^4 - T_{ent}^4)$$

Donde ε es la emisividad del material, $\sigma = 5{,}67 \cdot 10^{-8}$ $(\frac{W}{m^2 \cdot K^4})$ es la constante de Boltzmann, A (m^2) es el área de la superficie radiante, T_s (K) es la temperatura de la superficie radiante y T_{ent} (K) es la temperatura del entorno.

En el caso del PCB, los materiales habituales que consideraremos serán:

- $\varepsilon = 0.5$ para el cobre.
- $\varepsilon = 0.9$ para el FR-4 y solder mask.

Es interesante por tanto cubrir la mayoría del PCB con solder mask, evitando áreas de cobre desnudo, de esta forma el PCB será un mejor radiador.

3.3.4. Analogía eléctrica

Las tres expresiones anteriores tienen una forma similar, en la que el flujo de calor depende de una constante y de la diferencia de temperatura. Esto nos permite expresar estas ecuaciones como:

$$\Delta T = Q \cdot R$$

De manera análoga a la Ley de Ohm:

$$\Delta V = I \cdot R$$

Esto nos permite modelar y analizar un sistema térmico como si de un circuito eléctrico se tratase. El flujo de calor sería el equivalente a la corriente que circula por el circuito y las temperaturas en cada punto serían las tensiones en cada nodo.

La resistencia térmica por tanto sería el equivalente de la resistencia eléctrica, y expresaría la capacidad que tiene un cuerpo de oponerse al flujo de calor. Podemos determinar la capacidad para conducir el calor de los distintos elementos que forman el sistema y modelarlos en base a resistencias térmicas colocadas en serie o en paralelo.

La resistencia térmica se expresa en K/W o °C/W y podemos calcularla a partir de las expresiones de la conducción y la convección:

$$R_{cond} = \frac{L}{k \cdot A}$$

$$R_{conv} = \frac{1}{h \cdot A}$$

También es común que los fabricantes de componentes electrónicos y de disipadores especifiquen valores de resistencias térmicas en los datasheets. De esta forma podemos conocer, por ejemplo, cómo se transfiere el calor del encapsulado al PCB, o de la base del disipador al ambiente. En muchos casos estos valores de resistencias se obtienen experimentalmente.

3.4. Transferencia de calor en el PCB

3.4.1. Transferencia de calor dentro del PCB

Dentro del PCB la transferencia de calor se debe casi exclusivamente a la conducción. Consideraremos el PCB como un único sólido al que todos los componentes vierten el calor que generan, y dentro del cual el calor se distribuye por conducción.

3.4.1.1. Conducción transversal

Transversalmente el PCB está formado principalmente por capas de cobre y FR-4. Dado el pequeño espesor de las capas de cobre, en la práctica podemos despreciarlas, por lo que consideraremos que los 1.2mm de espesor del PCB son FR-4.

Si consideramos que el consumo máximo de la placa será de 2W, podemos estimar la diferencia de temperatura entre la cara superior y la cara inferior:

$$\Delta T = \frac{Q \cdot L}{k \cdot A} = \frac{2 \cdot 1{,}2 \cdot 10^{-3}}{0{,}3 \cdot 85 \cdot 10^{-3} \cdot 56 \cdot 10^{-3}} = 1{,}7°C$$

Podemos observar que la diferencia de temperatura es muy pequeña ya que, aunque el FR-4 tiene una conductividad térmica muy baja, el PCB es muy fino, por lo que la conducción transversal será bastante buena. En la práctica, podremos considerar que ambas caras se encuentran a la misma temperatura.

La conducción transversal se puede mejorar aún más añadiendo vías, que ayuden a conducir mejor el calor transversalmente entre los distintos planos de cobre.

3.4.1.2. Conducción lateral

Lateralmente la conducción debida al FR-4 será despreciable. Únicamente consideraremos las capas de cobre, especialmente las que contienen planos continuos de masa, y en menor medida los de alimentación ya que estos suelen tener discontinuidades.

Si queremos mejorar esta conducción lateral deberemos añadir más capas con planos continuos de cobre o aumentar el grosor de cada capa. En las simulaciones compararemos los resultados que podemos esperar según usemos 4, 6 o más capas.

Así conseguimos distribuir la temperatura de forma más homogénea a lo largo y ancho del PCB, reduciendo la formación de puntos calientes alrededor de los componentes que más calor generan. Es decir, podemos reducir mucho la temperatura máxima que alcanzan componentes como el SoC o la RAM, aunque a costa de aumentar ligeramente la temperatura del resto del PCB y otros componentes como los pasivos.

Cabe destacar que por mucho que mejoremos la conducción dentro del PCB, solo conseguiremos una superficie más homogénea en temperaturas, reduciendo los máximos, pero la temperatura media de todo el PCB se mantendrá constante. Para reducirla deberemos mejorar la transferencia de calor del PCB al ambiente.

3.4.2. Transferencia de calor del PCB al ambiente

Si bien la transferencia de calor dentro del PCB es por conducción, la conducción de calor del PCB al aire será completamente despreciable, ya que el aire tiene una conductividad térmica extremadamente baja. Por tanto, la evacuación del calor del PCB al ambiente se dará principalmente por convección y por radiación.

3.4.2.1. Radiación del PCB al ambiente

Para que el PCB irradie, se necesita que el entorno esté a temperaturas más bajas que la superficie del PCB. Cuando el PCB esté rodeado de otros elementos a igualdad de temperatura (por ejemplo, varios PCB dentro de un mismo chasis), no contaremos con la radiación.

En nuestro caso supondremos que la placa se utiliza sobre una superficie (por ejemplo, una mesa) que alcanza una temperatura cercana a la de la placa en la zona bajo la misma. Por este motivo, despreciaremos la radiación por la cara inferior y contaremos únicamente la superficie de la cara superior.

Para hacernos una idea del calor que podríamos disipar por radiación, supondremos que la temperatura ambiente es de 25°C y que la temperatura media del PCB es de 65°C:

$$Q = \varepsilon \cdot \sigma \cdot A \cdot (T_s^4 - T_{ent}^4) = 0{,}9 \cdot 5{,}67 \cdot 10^{-8} \cdot 85 \cdot 56 \cdot 10^{-6} \cdot (338^4 - 298^4) = 1{,}25W$$

3.4.2.2. Convección del PCB al ambiente

De forma similar a la radiación, supondremos que la placa se utiliza en posición horizontal sobre una mesa lo que dificulta o impide la convección bajo la misma. Por este motivo, despreciaremos también la convección por la cara inferior y contaremos únicamente con el área de la cara superior.

De igual forma que antes, podemos hacernos una idea del calor que podemos disipar por convección natural suponiendo que tendremos una temperatura ambiente de 25°C y una temperatura media en el PCB de 65°C:

$$Q = h \cdot A \cdot \Delta T = 8 \cdot 85 \cdot 56 \cdot 10^{-6} \cdot (65 - 25) = 1{,}52W$$

3.4.3. Conclusiones

De los cálculos realizados hasta ahora podemos extraer varias conclusiones.

Por un lado, ya podemos intuir que el diseño funcionará adecuadamente. Por medio de convección natural y radiación, sería viable evacuar los 2W que se estima que consumirá el PCB como máximo, manteniendo una ΔT razonable.

No obstante, estamos realizando muchas simplificaciones y considerando solo temperaturas medias, no hemos considerado las temperaturas máximas que llegarían a alcanzar los componentes. En siguientes apartados realizaremos dos simulaciones más elaboradas para comprobar las temperaturas máximas alcanzadas en cada punto.

Por otro lado, observamos que cuando usemos convección natural, ambos fenómenos (convección y radiación) serán relevantes. Podemos esperar que aproximadamente el 50 % del calor se disipe por convección y el otro 50 % por radiación. Debido a esto y para simplificar los cálculos posteriores en la simulación, en lugar de calcular la radiación, simplemente doblaremos el coeficiente de convección y consideraremos que el efecto de la radiación está incluido en dicho coeficiente.

También se observa que, si usásemos convección forzada, el calor disipado por convección sería muy superior al disipado por radiación, por lo que en ese caso la radiación directamente podríamos despreciarla.

Si quisiéramos disminuir la temperatura media del PCB sería necesario aumentar el flujo de calor que se disipa al ambiente por medio de convección o de radiación. Una forma de hacerlo sería aumentando la superficie de contacto del PCB con el ambiente, ya sea aumentando las dimensiones del propio PCB o añadiendo elementos adicionales como disipadores. Otra opción sería aumentar los coeficientes de convección añadiendo ventiladores, pasando así de convección natural a forzada.

En este tipo de sistemas, dadas las potencias tan bajas con las que se trabaja (menores a 5-10W), no suele ser necesario utilizar convección forzada salvo que necesitemos disipar toda la potencia en un volumen muy pequeño.

En este diseño trataremos de utilizar únicamente radiación y convección natural, añadiendo disipadores si fuese necesario aumentar la convección del PCB al ambiente, y mejorando la conducción en el PCB si fuese necesario reducir las temperaturas máximas que alcancen los componentes.

3.4.4. Sobre los disipadores

Con los disipadores podemos aumentar la superficie en la que es efectiva la convección, consiguiendo así evacuar más calor desde el encapsulado del chip o desde el PCB al ambiente sin necesidad de aumentar las dimensiones del PCB como tal.

Principalmente podemos encontrar dos tipos de disipadores, los *pin-fin* y los *straight-fin* o de extrusión. Los pin-fin pueden ser más efectivos si no conocemos de antemano la dirección del aire, ya que rinden similar en todas direcciones. Los de extrusión son más efectivos si el aire circula en una dirección determinada y los orientamos en dicha dirección, por ejemplo, si disponemos de un ventilador.

Preferiremos un disipador con una base lo más grande posible, tratando de cubrir el chip en su totalidad. También será preferible un disipador con las aletas más largas, aunque como conforme nos alejamos de la base va disminuyendo la temperatura de la aleta, a partir de cierta longitud no obtendremos una mejora relevante.

Determinar con exactitud el calor que evacuaría un disipador puede ser muy complicado, ya que depende de la geometría del disipador y de la mecánica del fluido. Por ello, los fabricantes de los disipadores proporcionan el valor de resistencia térmica desde la base del disipador al ambiente, normalmente medida experimentalmente.

Cuando trabajemos con convección natural, por lo general nos fiaremos de este valor de resistencia y lo utilizaremos en los cálculos, así como para comparar la efectividad entre distintos disipadores. Si tuviésemos convección forzada, este valor ya no sería muy relevante por lo que sería más conveniente realizar una simulación CFD.

3.4.5. Sobre las resistencias térmicas

Con las ecuaciones de la conducción, convección y radiación que hemos visto hasta ahora, podemos calcular fácilmente como se distribuye el calor por el PCB, así como la disipación de calor desde la superficie del PCB al ambiente. Sin embargo, nos faltaría por conocer la temperatura máxima que alcanzan los dies de cada chip.

En los datasheets de los componentes y de los disipadores podemos encontrar especificados ciertos valores de resistencias térmicas que han sido medidas por el fabricante. Estos valores permiten modelar toda la cadena de transferencia de calor a través de un componente como una serie de resistencias, desde la superficie del PCB bajo el componente hasta el ambiente, pasando por el encapsulado, el die y el disipador.

3.4.5.1. Relevancia de la resistencia Rja

Muchos fabricantes especifican en el datasheet la resistencia R_{ja} o *junction-ambient*, es decir, la resistencia térmica entre la unión de silicio y el ambiente. Antiguamente esta resistencia se utilizaba para calcular la temperatura del chip únicamente en función de la potencia consumida y de la temperatura ambiente usando la fórmula $T_j = T_a + P \cdot R_{ja}$, sin considerar el resto de componentes ni el PCB.

Estos componentes, normalmente en encapsulados grandes de agujero pasante, disipaban gran parte del calor que generaban al ambiente a través de su propio encapsulado. Era posible entonces despreciar el calor vertido por el componente al PCB y considerar que el componente se refrigeraba únicamente por sus propios medios.

Además, esta R_{ja} no contempla únicamente el flujo de calor del encapsulado al entorno, sino que tiene en cuenta la capacidad conjunta del encapsulado y de un PCB de tamaño normalizado, siguiendo un test normalizado, que poco tiene que ver con nuestro PCB en concreto. Esto hace que la R_{ja} no sea relevante en este caso.

En la actualidad, los encapsulados de los componentes son cada vez más pequeños y no tienen suficiente superficie expuesta al aire como para evacuar todo el calor que generan por convección y radiación. Estos componentes vierten la mayor parte del calor que generan al PCB, y es el PCB el que actúa como disipador.

3.4.5.2. Resistencias que consideraremos

Además del calor que disipa al ambiente el propio PCB, consideraremos también el flujo de calor desde la superficie del PCB al ambiente a través de los componentes modelados por las siguientes resistencias colocadas en serie:

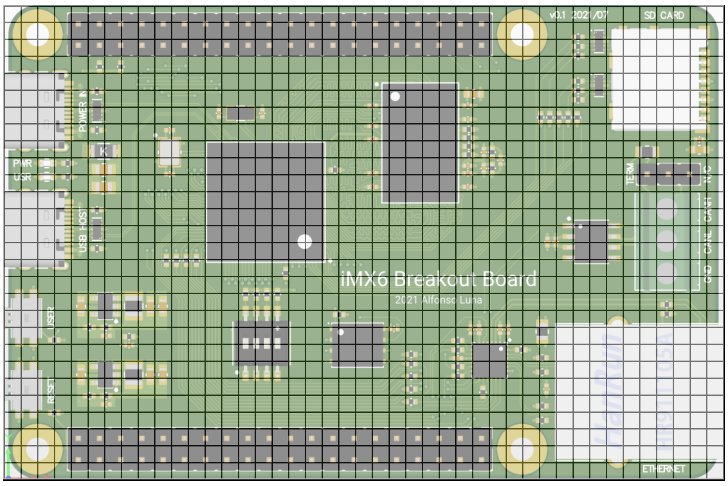
- **Rbj o *board-junction*:** De la superficie del PCB al die. Especificada por el fabricante del componente.
- **Rjc o *junction-case*:** Del die a la superficie del encapsulado. Especificada por el fabricante del componente.
- **Rcs o *case-sink*:** De la superficie del encapsulado a la base del disipador. Viene determinada por la almohadilla térmica.
- **Rsa o *sink-ambient*:** De la base del disipador al ambiente. Especificada por el fabricante del disipador.

3.5. Simulación con hoja de cálculo

3.5.1. Ecuaciones de las celdas

Se ha realizado una simulación utilizando una hoja de cálculo, en concreto con Microsoft Excel. De esta forma, a partir de la potencia consumida por cada componente, podremos estimar cuál será la distribución de temperaturas en toda la superficie del PCB, así como las temperaturas máximas en los dies de cada componente.

Se ha dividido el PCB en una cuadrícula con celdas de tamaño 2x2mm. Dado que las dimensiones del PCB son de 85x56mm, se ha utilizado una cuadrícula de 43x28:



En cada celda del PCB se ha aplicado un balance de potencia que relaciona la conducción a las celdas adyacentes con la convección y radiación al ambiente. Si además en dicha celda está presente uno de los componentes considerados, se añade a la ecuación un término de potencia vertida por el componente.

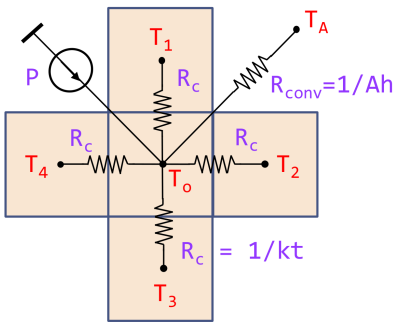


Figura 3.2: Modelo para una de las celdas del PCB [1].

3.5.1.1. Ecuación de una celda sin componente

Las celdas sin componente son las celdas en las que no se ha considerado que haya un componente que genere calor. En estas celdas solo tendremos en cuenta la conducción con las celdas adyacentes del PCB a través de las capas de cobre y la convección (incluyendo radiación) de la superficie del PCB al ambiente:

$$Q_{cond10} + Q_{cond20} + Q_{cond30} + Q_{cond40} + Q_{conv} = 0$$

$$\frac{(T_1 - T_0) + (T_2 - T_0) + (T_3 - T_0) + (T_4 - T_0)}{R_{cond}} + \frac{T_A - T_0}{R_{conv}} = 0$$

Para celdas en el interior del PCB, la ecuación es:

$$T_0 = \frac{R_{conv} \cdot R_{cond}}{4 \cdot R_{conv} + R_{cond}} \cdot \left(\frac{T_1 + T_2 + T_3 + T_4}{R_{cond}} + \frac{T_A}{R_{conv}} \right)$$

Para celdas en un borde del PCB, la ecuación es:

$$T_0 = \frac{R_{conv} \cdot R_{cond}}{3 \cdot R_{conv} + R_{cond}} \cdot \left(\frac{T_1 + T_2 + T_3}{R_{cond}} + \frac{T_A}{R_{conv}} \right)$$

Para celdas en una esquina del PCB, la ecuación es:

$$T_0 = \frac{R_{conv} \cdot R_{cond}}{2 \cdot R_{conv} + R_{cond}} \cdot \left(\frac{T_2 + T_3}{R_{cond}} + \frac{T_A}{R_{conv}} \right)$$

3.5.1.2. Ecuación de una celda con componente

A diferencia del caso anterior, en estas celdas la convección no se produce en la superficie del PCB sino en la superficie del encapsulado del componente. Por ello, añadiremos en el camino las resistencias Rbj y Rjc para modelar la transferencia de calor desde la superficie del PCB hasta la superficie del encapsulado.

Las resistencias Rbj y Rjc son distintas para cada componente y se han obtenido del datasheet de cada uno. Para determinar el valor de Rbj o Rjc asociado a una única celda, se multiplicará la resistencia total Rbj o Rjc del componente por el número de celdas que ocupa este en la cuadrícula.

Además, en estas celdas añadiremos al balance la potencia generada por el componente. La potencia P vertida por el componente en una celda será la potencia total consumida por el componente dividida entre el número de celdas que ocupa en la cuadrícula. A efectos de esta potencia ignoraremos Rbj y consideraremos que la potencia se genera en la superficie del PCB en lugar de en el die. De esta forma se consigue simplificar el modelo y poder trabajar utilizando una sola dimensión.

El balance de potencia en la celda es:

$$Q_{cond10} + Q_{cond20} + Q_{cond30} + Q_{cond40} + Q_{comp} + P = 0$$

$$\frac{(T_1 - T_0) + (T_2 - T_0) + (T_3 - T_0) + (T_4 - T_0)}{R_{cond}} + \frac{T_A - T_0}{R_{bj} + R_{jc} + R_{conv}} + P = 0$$

Si consideramos $R_{comp} = R_{bj} + R_{jc} + R_{conv}$, la ecuación de la celda es:

$$T_0 = \frac{R_{comp} \cdot R_{cond}}{4 \cdot R_{comp} + R_{cond}} \cdot \left(\frac{T_1 + T_2 + T_3 + T_4}{R_{cond}} + \frac{T_A}{R_{comp}} + P \right)$$

Esta temperatura será la de la superficie del PCB bajo el die, no la del die del componente. Posteriormente, podremos estimar la temperatura máxima en el die utilizando la expresión $T_{die} = T_0 + P_{total} \cdot R_{bj}$.

3.5.1.3. Ecuación de una celda con componente y disipador

Este caso es similar al anterior, solo que en lugar de considerar la convección del encapsulado al ambiente, tendremos en cuenta la cadena completa de resistencias, incluyendo la de la almohadilla térmica y la del disipador al ambiente.

$$\frac{(T_1 - T_0) + (T_2 - T_0) + (T_3 - T_0) + (T_4 - T_0)}{R_{cond}} + \frac{T_A - T_0}{R_{bj} + R_{jc} + R_{cs} + R_{sa}} + P = 0$$

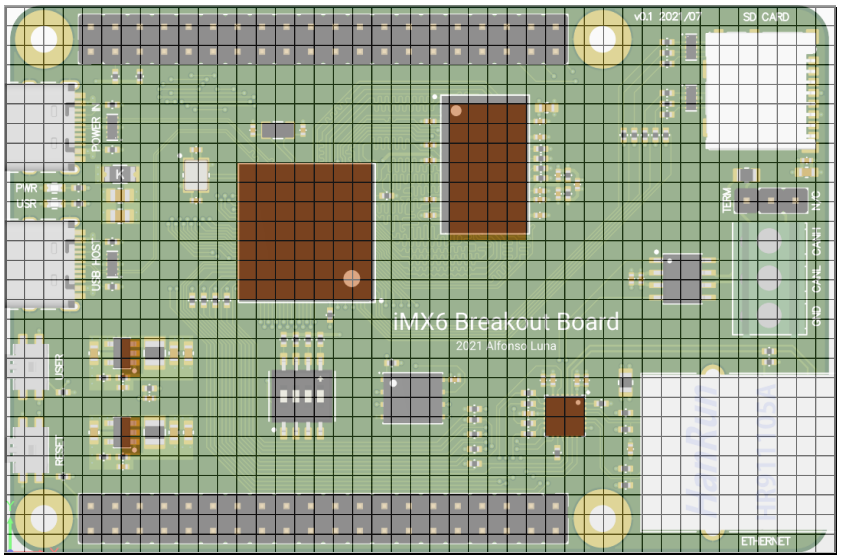
La ecuación de la celda será la misma que en el caso anterior, pero en este caso consideraremos $R_{comp2} = R_{bj} + R_{jc} + R_{cs} + R_{sa}$.

$$T_0 = \frac{R_{comp2} \cdot R_{cond}}{4 \cdot R_{comp2} + R_{cond}} \cdot \left(\frac{T_1 + T_2 + T_3 + T_4}{R_{cond}} + \frac{T_A}{R_{comp2}} + P \right)$$

3.5.2. Componentes considerados

A efectos de la generación de calor, solo se han tenido en cuenta los componentes que más consumen: El SoC, la RAM, los dos reguladores y el transceiver Ethernet, los cuales suman un consumo máximo algo menor a 2W. El resto de componentes se han ignorado ya que el calor que generan es despreciable.

En la siguiente imagen se muestra como se ha divido la cuadrícula según el tipo de celda. Las celdas coloreadas en naranja se corresponden con los componentes que generan calor, en estas celdas se ha aplicado la ecuación de una celda con componente. En el resto de celdas se ha aplicado la ecuación de una celda sin componente.



Para las celdas sin componente se han utilizado los siguientes datos:

PCB				
General	Tamaño de celda		2	mm
	Temperatura ambiente	Ta	35	ºC
Conducción	Conductividad térmica del cobre	k	360	W/m·K
	Espesor de una capa de cobre	t	17.5	um
	Número de capas con planos continuos	Nc	2	capas
	Resistencia de una celda = 1/K·Nc·t	Rcond	79.4	K/W
Convección	Coeficiente de convección (+ radiación)	h	15	W/m2·K
	Resistencia conv de una celda = 1/A·h	Rconv	16666.7	K/W
Coeficientes Precalculados	Ta/Rconv		0.0021	
	(Rconv·Rcond)/(4·Rconv+Rcond)		19.8177	
	(Rconv·Rcond)/(3·Rconv+Rcond)		26.4131	
	(Rconv·Rcond)/(2·Rconv+Rcond)		39.5883	

Para las celdas con componente se han utilizado los siguientes datos:

SoC i.MX6ULL sin Disipador				
Por Componente	Número de celdas ocupadas		49	celdas
	Potencia total	Ptotal	0.69	W
	Rbj total		21.8	K/W
	Rjc total		19.3	K/W
Por Celda	Potencia por celda	P	0.014	W
	Rbj por celda		1068.2	K/W
	Rjc por celda		945.7	K/W
	Rcomp = Rbj + Rjc + Rconv		18680.6	K/W
Coeficientes Precalculados	Ta/Rcomp		0.0019	
	(Rcomp·Rcond)/(4·Rcomp+Rcond)		19.8202	

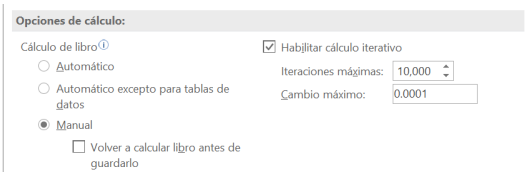
RAM DDR3L				
Por Componente	Número de celdas ocupadas		28	celdas
	Potencia total	Ptotal	0.345	W
	Rbj total		19.2	K/W
	Rjc total		3.9	K/W
Por Celda	Potencia por celda	P	0.012	W
	Rbj por celda		537.6	K/W
	Rjc por celda		109.2	K/W
	Rcomp = Rbj + Rjc + Rconv		17313.5	K/W
Coeficientes Precalculados	Ta/Rcomp		0.0020	
	(Rcomp·Rcond)/(4·Rcomp+Rcond)		19.8186	

VRM TLV62569				
Por Componente	Número de celdas ocupadas		2	celdas
	Consumo máximo en línea 1.35V		1.04	W
	Consumo máximo en línea 3.3V		1.65	W
	Eficiencia del convertidor		0.9	
	Potencia total convertidor 1.35V	Ptotal	0.12	W
	Potencia total convertidor 3.3V	Ptotal	0.18	W
	Rbj total		41.2	K/W
	Rjc total		137.5	K/W
Por Celda	Potencia por celda 1.35V	P	0.058	W
	Potencia por celda 3.3V	P	0.092	W
	Rbj por celda		82.4	K/W
	Rjc por celda		275	K/W
	Rcomp = Rbj + Rjc + Rconv		17024.1	K/W
Coeficientes Precalculados	Ta/Rcomp		0.0021	
	(Rcomp·Rcond)/(4·Rcomp+Rcond)		19.8182	

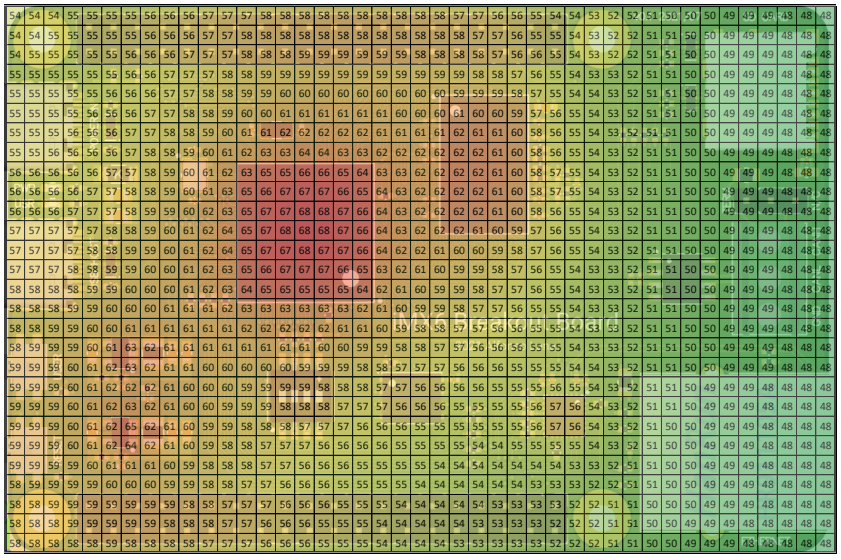
Ethernet PHY LAN8720A				
Por Componente	Número de celdas ocupadas		4	celdas
	Potencia total	Ptotal	0.164	W
	Rbj total		0	K/W
	Rjc total		12.6	K/W
Por Celda	Potencia por celda	P	0.041	W
	Rbj por celda		0	K/W
	Rjc por celda		50.4	K/W
	Rcomp = Rbj + Rjc + Rconv		16717.1	K/W
Coeficientes Precalculados	Ta/Rcomp		0.0021	
	(Rcomp·Rcond)/(4·Rcomp+Rcond)		26.4132	

3.5.3. Resultados de la simulación

Como hay referencias circulares entre las celdas, para poder ejecutar la simulación es necesario configurar Excel de forma que permita cálculo iterativo:



Tras aplicar a cada celda la ecuación correspondiente y pulsar F9 para realizar los cálculos, se obtienen los siguientes resultados:

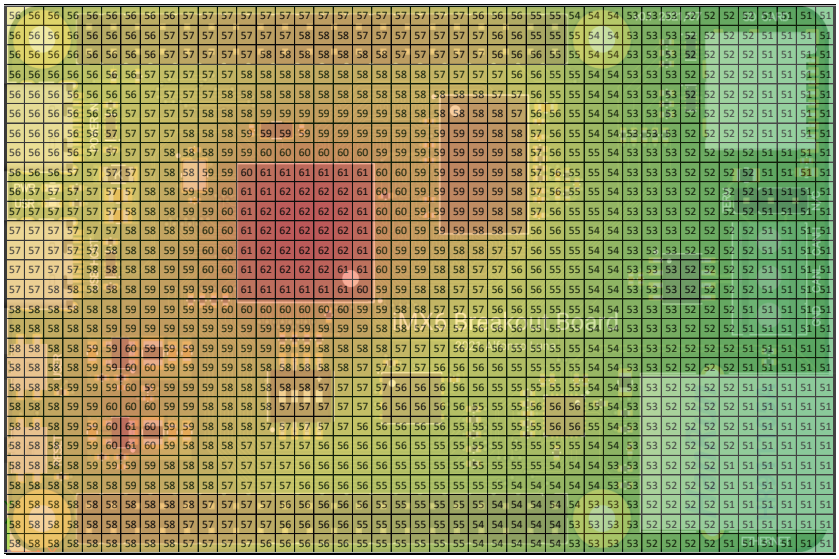


Resultados sin disipador, PCB de 4 capas			
PCB	Media PCB	56	ºC
	Máxima PCB	68	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	68	ºC
	RAM DDR3L	62	ºC
	TLV62569 1.35V	63	ºC
	TLV62569 3.3V	65	ºC
	Ethernet PHY	57	ºC
	CAN PHY	51	ºC
	QSPI Flash	57	ºC
	microSD	50	ºC
	Oscilador 32.768kHz	62	ºC
	Oscilador 24MHz	61	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	83	ºC
	RAM DDR3L	69	ºC
	TLV62569 1.35V	68	ºC
	TLV62569 3.3V	72	ºC
	Ethernet PHY	57	ºC

Los resultados obtenidos son suficientemente buenos. Aun con una temperatura ambiente de 35°C, la temperatura máxima en el PCB no supera los 68°C y las temperaturas en los dies se mantienen por debajo de 85°C. Además, si tenemos en cuenta que los datos utilizados son muy conservadores, considerando siempre el peor caso, en un uso normal de la placa cabe esperar que las temperaturas sean menores.

3.5.4. Efecto de aumentar el espesor de cobre

En la simulación anterior hemos considerado un stackup de 4 capas con 2 planos de cobre continuos. Si aumentamos el stackup a 6 capas con 4 planos de cobre continuos, conseguimos los siguientes resultados:



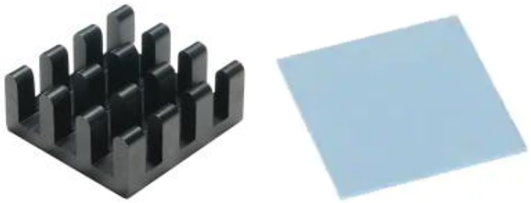
Resultados sin disipador, PCB de 6 capas			
PCB	Media PCB	56	ºC
	Máxima PCB	62	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	62	ºC
	RAM DDR3L	59	ºC
	TLV62569 1.35V	60	ºC
	TLV62569 3.3V	61	ºC
	Ethernet PHY	56	ºC
	CAN PHY	53	ºC
	QSPI Flash	57	ºC
	microSD	52	ºC
	Oscilador 32.768kHz	59	ºC
	Oscilador 24MHz	59	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	77	ºC
	RAM DDR3L	66	ºC
	TLV62569 1.35V	65	ºC
	TLV62569 3.3V	68	ºC
	Ethernet PHY	56	ºC

Se observa como la temperatura media en el PCB se mantiene constante aunque se aumente el espesor de cobre. Como se ha comentado anteriormente, esto es debido a que mejorando la conducción en el PCB se consigue distribuir las temperaturas de forma más homogénea, reduciendo los máximos y aumentando los mínimos, pero por lo general no se consigue evacuar más calor del PCB al ambiente.

Podemos ver una reducción de las temperaturas máximas de 6°C en el caso del SoC y 4°C en el regulador de 3.3V, a la vez que se han aumentado ligeramente las temperaturas en el lado derecho del PCB. No obstante, las temperaturas obtenidas para el caso de 4 capas ya eran lo suficientemente buenas.

3.5.5. Efecto de incluir un disipador

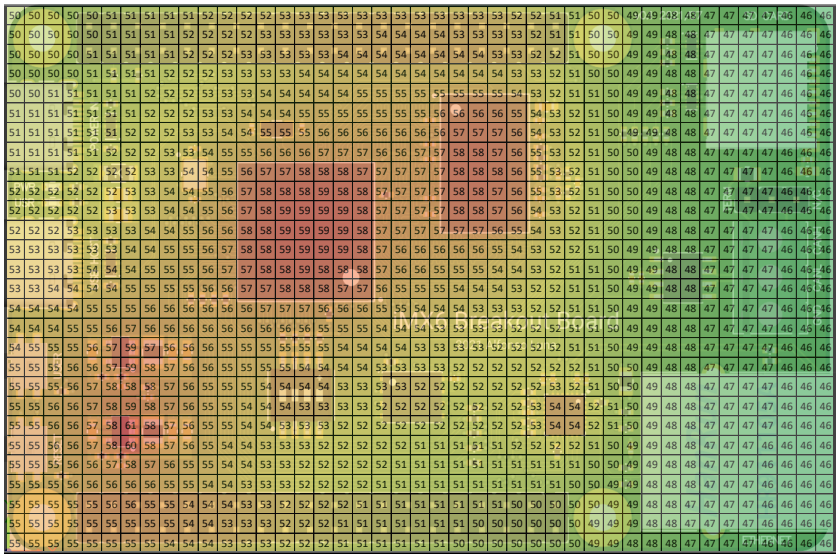
En este apartado se ha simulado el efecto en las temperaturas de añadir un disipador al SoC. A modo de ejemplo, se ha considerado un disipador Assmann V2020B unido mediante una almohadilla térmica TG-A6200-15-15-0.5 de t-Global Technology:



SoC i.MX6ULL con Disipador				
Almohadilla Térmica	Conductividad térmica almohadilla	k	6.2	W/m·K
	Espesor almohadilla	L	0.5	mm
	Longitud almohadilla		14	mm
	Rcs total = L/k*A		0.41	K/W
Disipador	Rsa total		27	K/W
Por Celda	Rcs por celda		20.2	K/W
	Rsa por celda		1323.0	K/W
	Rcomp2 = Rbj + Rjc + Rcs + Rsa		3357.1	K/W
Coeficientes Precalculados	Ta/Rcomp2		0.0104	
	(Rcomp2·Rcond)/(4·Rcomp2+Rcond)		19.7247	

Podemos ver que al añadir el disipador se produce una gran diferencia en la resistencia de convección asociada a las celdas del SoC. Sin disipador el valor de esta resistencia era de 18680 K/W, con disipador se reduce a 3357 K/W. Esto permitirá evacuar mucho más calor al ambiente en la zona del SoC, consiguiendo reducir la temperatura media del PCB, así como las temperaturas máximas de los componentes.

Para 4 capas obtenemos los siguientes resultados:

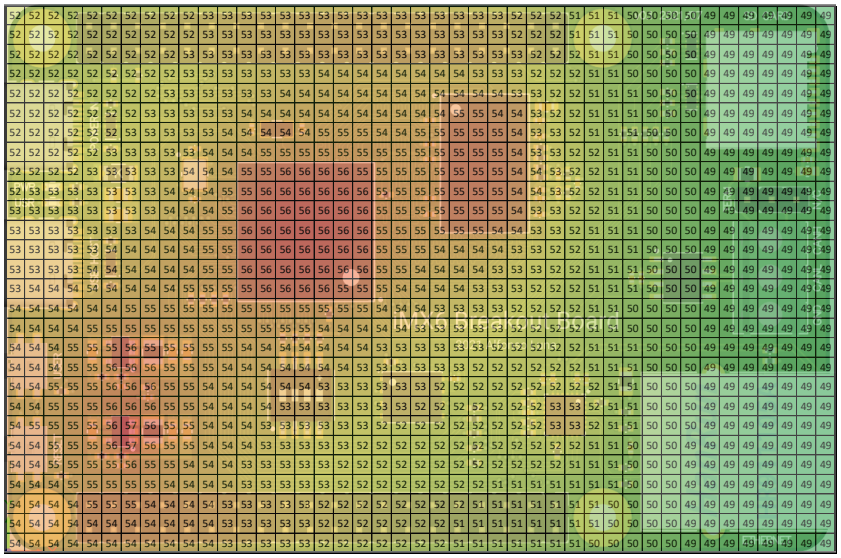


Resultados con disipador, PCB de 4 capas			
PCB	Media PCB	52	ºC
	Máxima PCB	61	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	59	ºC
	RAM DDR3L	58	ºC
	TLV62569 1.35V	59	ºC
	TLV62569 3.3V	61	ºC
	Ethernet PHY	54	ºC
	CAN PHY	48	ºC
	QSPI Flash	53	ºC
	microSD	47	ºC
	Oscilador 32.768kHz	55	ºC
	Oscilador 24MHz	55	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	74	ºC
	RAM DDR3L	65	ºC
	TLV62569 1.35V	64	ºC
	TLV62569 3.3V	68	ºC
	Ethernet PHY	54	ºC

De esta forma sí que se consigue reducir la temperatura media del PCB, en concreto de 56 a 52°C, ya que a través del disipador hemos aumentado el calor que se disipa al ambiente por convección. La temperatura máxima en el die del SoC se reduce considerablemente pasando de 83 a 74°C, casi 10°C de diferencia.

El resto de componentes, aunque en menor medida, también ven reducidas sus temperaturas con unos 4 o 6°C de diferencia. Esto se debe a que como gran parte del calor generado por el SoC ahora se evacúa por el disipador, el SoC vierte menos calor al PCB y las temperaturas de toda la superficie del PCB se ven reducidas.

Para 6 capas obtenemos los siguientes resultados:



Resultados con disipador, PCB de 6 capas			
PCB	Media PCB	52	ºC
	Máxima PCB	57	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	56	ºC
	RAM DDR3L	55	ºC
	TLV62569 1.35V	56	ºC
	TLV62569 3.3V	57	ºC
	Ethernet PHY	53	ºC
	CAN PHY	50	ºC
	QSPI Flash	53	ºC
	microSD	49	ºC
	Oscilador 32.768kHz	54	ºC
	Oscilador 24MHz	54	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	72	ºC
	RAM DDR3L	62	ºC
	TLV62569 1.35V	61	ºC
	TLV62569 3.3V	65	ºC
	Ethernet PHY	53	ºC

De forma similar al caso de 4 capas, al añadir el disipador conseguimos reducir la temperatura media del PCB de 56 a 52°C.

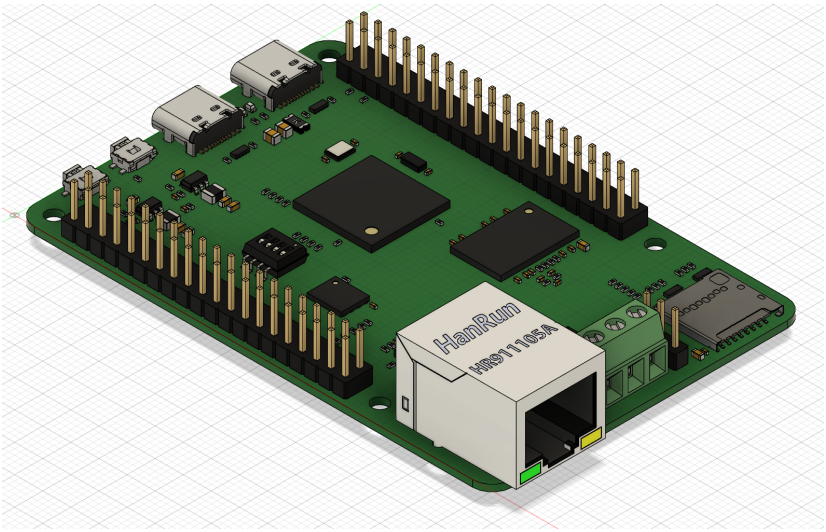
La temperatura máxima en el die del SoC se ve reducida de 77 a 72°C, solo 5°C de diferencia, en contraste con la bajada de 10°C que se conseguía en el PCB de 4 capas. Esto es debido a que el PCB de 6 capas es capaz de distribuir mejor el calor generado por el SoC por toda la superficie del PCB, por lo que la mejora conseguida al aumentar la convección sobre el SoC es menos significativa en este caso.

3.6. Simulación con Autodesk Fusion 360

Como alternativa a la hoja de cálculo, se ha realizado también una simulación térmica con Autodesk Fusion 360. La principal ventaja de esta simulación es que tiene en cuenta las 3 dimensiones y su efecto en el movimiento del fluido, por lo que esta simulación debería ser más precisa que la anterior en cuanto a la convección.

La configuración de la simulación es sencilla, no hay muchas opciones que se puedan ajustar. Se han asignado las cargas térmicas a los mismos componentes que en la simulación con hoja de cálculo, y del mismo valor de potencia que antes. La temperatura ambiente también se ha configurado en 35°C. La precisión de la simulación se ha configurado casi lo más baja posible, para que finalizase en menor tiempo.

Se ha partido del modelo 3D del PCB exportado de Altium Designer:



Para que la simulación se completase en un tiempo razonable, se ha tenido que simplificar en gran medida el modelo. Aun con estas simplificaciones, la simulación tarda alrededor de una hora en completarse, a pesar de que se ejecuta en la nube.

3.6.1. Simplificación del PCB

Se ha considerado que el PCB está sin rutar. No se han considerado pistas, vías ni orificios pequeños. Se ha considerado que usamos el stackup de 4 capas.

En un principio, se ha dividido el bloque del PCB en tres capas. A las dos capas más exteriores se les ha asignado material cobre con 17.5um de espesor, simulando los dos planos continuos de masa y alimentación. A la capa intermedia se le ha asignado material FR-4 hasta rellenar los 1.2mm de espesor total del PCB.

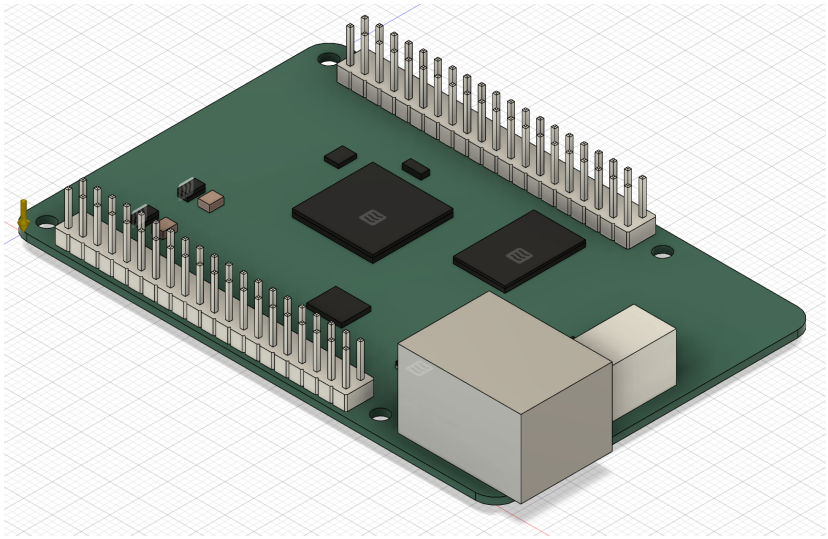
El problema de este enfoque ha sido que la simulación ignoraba completamente los finísimos planos de cobre, por lo que tenía en cuenta únicamente el FR-4 para la conducción en el PCB y por tanto los resultados obtenidos eran peores de lo esperado.

Como no es posible especificar el mallado que utiliza la simulación, finalmente se ha considerado el PCB como un único bloque sólido de material FR-4 y 1.2mm de espesor. Para que la conducción en este bloque se comporte de forma lo más parecida posible al PCB real, se ha modificado la conductividad térmica del material a por una media ponderada teniendo en cuenta el espesor de las capas de cobre y FR-4:

$$k_{pcb} = \frac{360 \cdot 0{,}0175 + 0{,}3 \cdot 1{,}165 + 360 \cdot 0{,}0175}{1{,}2} = 10{,}8 \frac{W}{m \cdot K}$$

3.6.2. Simplificación de los componentes

En cuanto a los componentes, se han eliminado todos los pasivos pequeños y el resto de componentes que no eran muy relevantes de cara a la simulación.



Los componentes considerados se han simplificado todo lo posible. Todos los circuitos integrados se han sustituido por cubos sólidos de ancho y largo equivalentes y pegados a la superficie del PCB, eliminando las bolas o patillas que tenían por debajo.

Se ha comprobado que las cargas térmicas no se generan en el centro del sólido en el que se configuran sino en el centro de su cara superior. Por este motivo, los componentes que generan calor se han dividido en dos partes, una encima de otra, tratando de que el espacio entre el die y el PCB fuese más cercano a la realidad.

En cuanto a los materiales, a los chips se les ha aplicado el material Discrete Component, el cual simula el material del encapsulado que rodea al die del componente. Esto permitirá medir las temperaturas máximas en el die directamente, sin tener que calcularlas a posteriori utilizando las resistencias Rbj. Por defecto, este material tenía una conductividad térmica de 6 $\frac{W}{m \cdot K}$. Este valor se ha modificado por otro más realista calculado a partir de la resistencia Rbj y dimensiones del SoC:

$$k_{comp} = \frac{L}{R_{bj} \cdot A} = \frac{0{,}8 \cdot 10^{-3}}{21{,}8 \cdot 14 \cdot 10^{-3} \cdot 14 \cdot 10^{-3}} = 0{,}2 \frac{W}{m \cdot K}$$

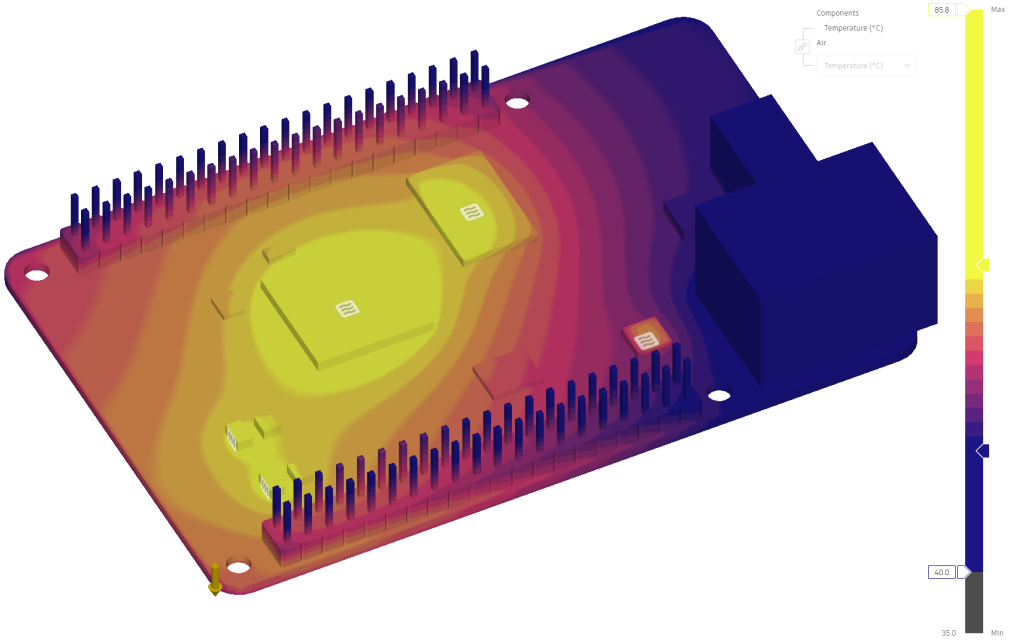
En el caso del transceiver Ethernet, a la mitad superior se le ha asignado Discrete Component pero a la mitad inferior se le ha asignado aluminio. De esta forma la transferencia de calor entre el die y el PCB será más realista, ya que este componente lleva un pad expuesto en su zona inferior por lo que tiene una Rbj de prácticamente 0.

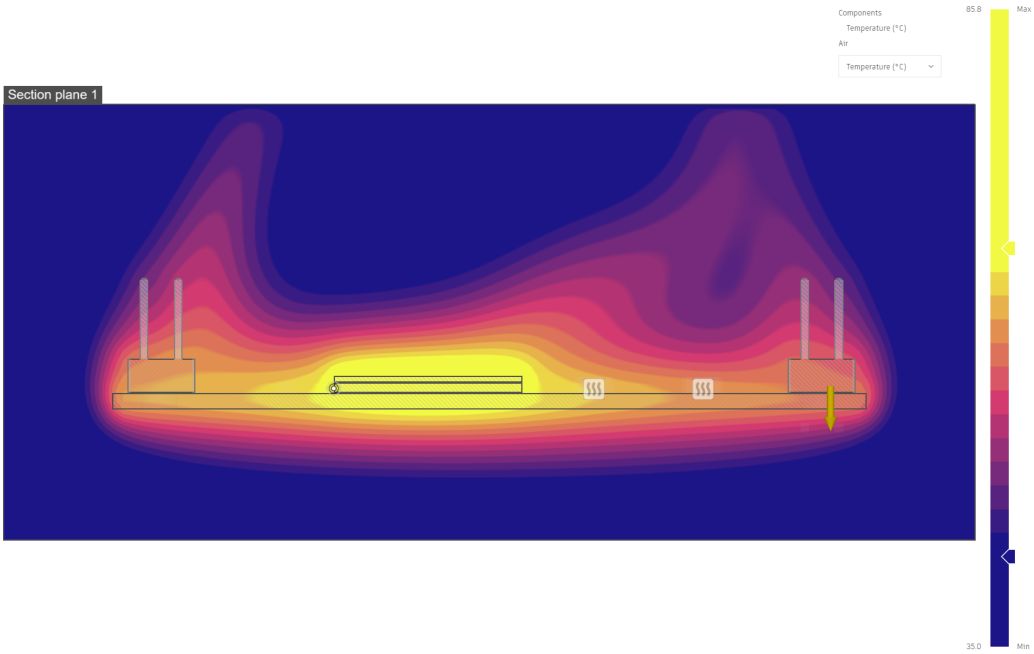
Los conectores de expansión y el conector Ethernet se han mantenido ya que por sus grandes dimensiones podrían ser relevantes de cara a mejorar la convección. En el caso del conector Ethernet, se ha simplificado el modelo real sustituyéndolo por un cubo hueco abierto por la cara inferior pegada al PCB. Los conectores de expansión se han simplificado eliminando los pines que sobresalían por debajo del PCB.

Al conector CAN y a los conectores de expansión se les ha aplicado plástico ABS. A los inductores se les ha aplicado cobre y al conector Ethernet, aluminio.

3.6.3. Resultados de la simulación

Tras ejecutar la simulación, los resultados obtenidos son los siguientes:





Resultados sin disipador, PCB de 4 capas				
		Excel	Fusion 360	
PCB	Máxima PCB	68	68	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	68	68	ºC
	RAM DDR3L	62	63	ºC
	TLV62569 1.35V	63	65	ºC
	TLV62569 3.3V	65	65	ºC
	Ethernet PHY	57	55	ºC
	CAN PHY	51	52	ºC
	QSPI Flash	57	59	ºC
	Oscilador 32.768kHz	62	64	ºC
	Oscilador 24MHz	61	63	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	83	70	ºC
	RAM DDR3L	69	64	ºC
	TLV62569 1.35V	68	72	ºC
	TLV62569 3.3V	72	75	ºC
	Ethernet PHY	57	58	ºC

En general, la distribución de temperaturas obtenida con Fusion 360 es muy similar a la de la hoja de cálculo. Las temperaturas obtenidas en la superficie del PCB son prácticamente idénticas, con apenas 1 o 2 °C de diferencia entre ambas simulaciones.

La mayor diferencia se encuentra en la temperatura del die del SoC. Esto se debe a que en esta zona la convección es mayor, mientras que en la simulación con Excel se ha utilizado el mismo coeficiente de convección para todo el PCB.

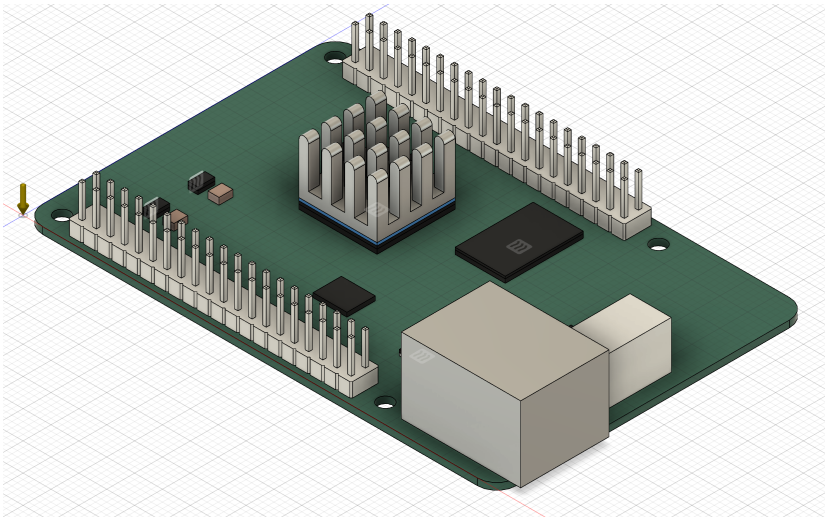
Además, el aire sobre la placa también contribuye a distribuir mejor las temperaturas, provocando un efecto similar al del aumento del espesor del cobre: el SoC está unos 13°C más fresco, pero el resto de componentes del PCB están 1 o 2 °C más calientes. Este efecto del aire no se tiene en cuenta en la simulación con Excel.

La diferencia también puede estar en el método de estimar las temperaturas máximas en los dies de los componentes. En la simulación con Excel, las temperaturas en el die se calcularon mediante la expresión $T_{die} = T_0 + P_{total} \cdot R_{bj}$, utilizando para cada componente el valor de Rbj especificado por su fabricante.

En cambio, en la simulación con Fusion, se considera que cada componente es un bloque de conductividad $k = 0{,}2\frac{W}{m \cdot K}$ y simplemente se ha medido la temperatura máxima en el sólido. Esto también provoca diferencias entre ambas simulaciones.

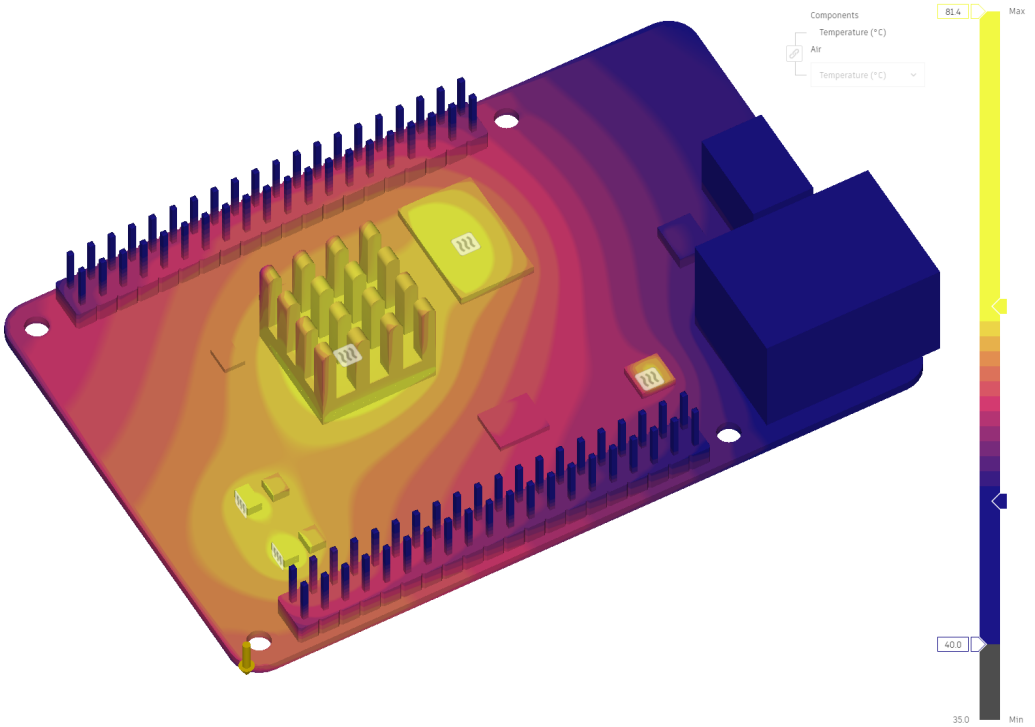
3.6.4. Efecto de incluir un disipador

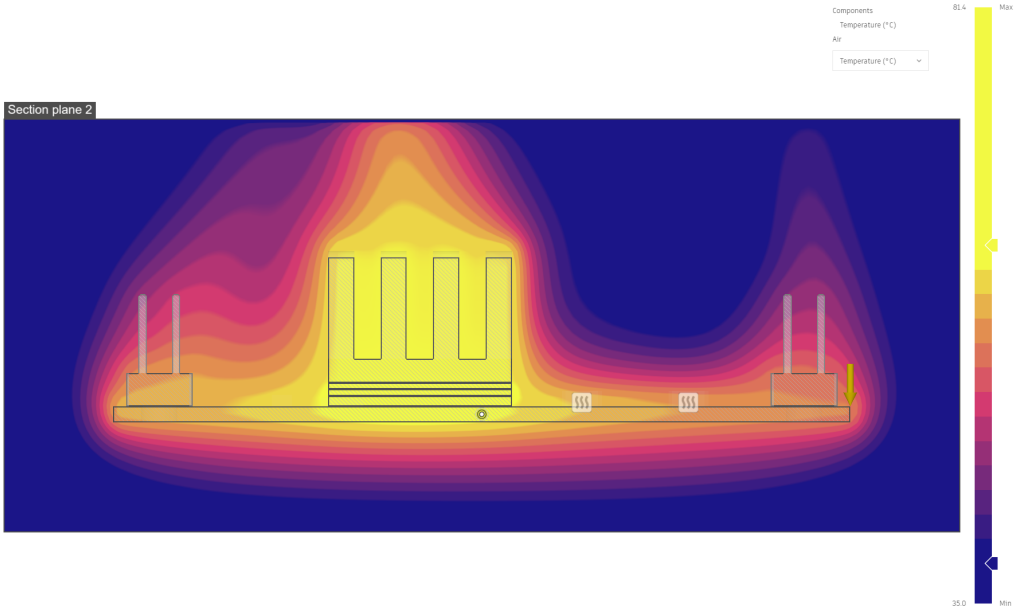
De igual forma que en la simulación con Excel, se ha comprobado el efecto de añadir un disipador. La almohadilla y disipador utilizados en esta simulación son los mismos que en la simulación con Excel. En este caso el disipador no se ha modelado con su resistencia térmica Rsa sino que se ha utilizado el modelo 3D que proporciona el fabricante del disipador y se le ha asignado como material aluminio. También se ha tenido en cuenta las dimensiones y conductividad térmica de la almohadilla.



Tras ejecutar la simulación, los resultados obtenidos son los siguientes:

Resultados con disipador, PCB de 4 capas				
		Excel	Fusion 360	
PCB	Máxima PCB	61	63	ºC
Temperaturas Máximas Superficie PCB	SoC i.MX6ULL	59	63	ºC
	RAM DDR3L	58	60	ºC
	TLV62569 1.35V	59	62	ºC
	TLV62569 3.3V	61	63	ºC
	Ethernet PHY	54	55	ºC
	CAN PHY	48	52	ºC
	QSPI Flash	53	56	ºC
	Oscilador 32.768kHz	55	60	ºC
	Oscilador 24MHz	55	59	ºC
Temperaturas Máximas Die Componente	SoC i.MX6ULL	74	63	ºC
	RAM DDR3L	65	61	ºC
	TLV62569 1.35V	64	67	ºC
	TLV62569 3.3V	68	71	ºC
	Ethernet PHY	54	57	ºC





Se observa que la distribución de temperaturas obtenida también es muy similar a la de la hoja de cálculo. La mayor diferencia también se encuentra en el SoC con 11 °C menos que en la simulación con Excel, mientras el resto de componentes están 2 o 4°C por encima. De aquí podemos concluir que la resistencia Rsa proporcionada por el fabricante del disipador y que hemos utilizado en Excel era bastante acertada.

Capítulo 4

Red de condensadores de desacoplo

4.1. Introducción

La red de distribución de alimentación o *Power Distribution Network* (PDN) consiste en todas las interconexiones desde el conector de alimentación de la placa hasta los pads de los distintos componentes, incluyendo todos los elementos intermedios tales como los reguladores, los condensadores de desacoplo, los planos de alimentación y masa, las vías y las pistas.

La correcta distribución de energía hacia los componentes que la consumen depende de todos estos elementos, por lo que es necesario diseñar toda la red correctamente para que los componentes funcionen de manera adecuada.

El principal criterio que se utiliza es la máxima caída o diferencia de tensión admisible en cada entrada de alimentación de cada componente. Si esta tensión varía significativamente y se sale de los límites especificados por el fabricante, el componente podría reiniciarse o dejar de funcionar con normalidad.

En función de esta caída de tensión admisible y de la corriente demandada por el dispositivo, utilizando la Ley de Ohm podemos obtener la impedancia máxima de la línea de alimentación. Por encima de esta impedancia máxima la diferencia de tensión sería excesiva y podría afectar al funcionamiento del componente.

Es importante tener en cuenta que los componentes digitales no tienen un consumo de corriente continuo sino pulsado, su consumo varía significativamente en cada conmutación de sus transistores internos así como en función de las distintas partes del chip que se estén usando en cada momento. Esto implica que es necesario garantizar esa máxima impedancia en todo el ancho de banda de esta corriente.

La propia inductancia de las pistas, vías y cables hasta la fuente de alimentación hacen que esta impedancia sea demasiado elevada, sobre todo a altas frecuencias. Estas inductancias deberán minimizarse todo lo posible, usando planos y pistas más cortas y anchas y varias vías en paralelo en los cambios de capa.

Aun así, se hace imprescindible añadir condensadores de desacoplo. Estos condensadores se colocan en cada línea de alimentación, lo más cerca posible de los pines de alimentación de cada componente. De esta forma los condensadores pueden hacer frente a los grandes picos de corriente a altas frecuencias sin que se produzca una gran caída en el voltaje, reduciendo así la impedancia de la línea a altas frecuencias.

El objetivo de este capítulo es determinar qué condensadores colocar y en qué cantidad para conseguir una impedancia de la red menor a la que se marque como objetivo para cada componente. Para ello nos ayudaremos de las recomendaciones de los fabricantes, así como de la hoja de cálculo PDN Tool de Intel que permite graficar la respuesta en frecuencia de estos condensadores.

4.1.1. Niveles de desacoplo

Para el análisis de la red se han considerado distintos elementos o niveles, que son efectivos en distintos rangos de frecuencia. De menor a mayor frecuencia en la que son efectivos, se han distinguido:

- **Nivel 1:** Capacidad de bulk a la entrada de alimentación de la placa
- **Nivel 2:** Reguladores de tensión discretos con sus condensadores
- **Nivel 3:** Condensadores bulk de 1-22uF, asociados a cada CI o grupo de pines
- **Nivel 4:** Condensadores de desacoplo de 100-220nF, asociados a cada pin
- **Nivel 5:** Capacidad de los planos de alimentación y capacidad on-die

4.1.2. Selección de los condensadores

Dado que no se necesitaban capacidades muy elevadas, se han utilizado solamente condensadores cerámicos del fabricante Samsung. Se han seleccionado varios valores para emplear en el diseño de forma que se cubra un rango amplio entre los 100nF y los 47uF con encapsulados que van desde el 402 hasta el 805.

Los valores exactos de capacidad no son muy importantes. Se han seleccionado varios valores intermedios dentro del rango de interés, basándose en cuales eran los valores recomendados por los fabricantes del SoC y de los reguladores, así como prefiriendo valores comunes que sean ampliamente utilizados y de los que haya mucha disponibilidad.

Los condensadores utilizados son los siguientes:

Capacidad	**Encap.**	**Tolerancia**	**Voltaje**	**Dieléc.**	**MPN**
100nF	402	10 %	16V	X7R	CL05B104KO5NNNC
220nF	402	10 %	16V	X7R	CL05B224KO5NNNC
1uF	402	10 %	16V	X5R	CL05A105KO5NNNC
4.7uF	402	10 %	10V	X5R	CL05A475KP5NRNC
10uF	402	20 %	6.3V	X5R	CL05A106MQ5NUNC
22uF	603	20 %	6.3V	X5R	CL10A226MQ8NRNC
47uF	805	20 %	6.3V	X5R	CL21A476MQYNNNE

Para minimizar las inductancias parásitas y reducir el área ocupada de PCB, para cada capacidad se ha tratado de elegir el encapsulado más pequeño posible, con el que era posible mantener un dieléctrico X7R o X5R.

4.2. Estudio por niveles

4.2.1. Nivel 1. Capacidad de bulk del PCB

Consideramos que tenemos la placa alimentada mediante un cable USB típico de 150cm de longitud, 1mm de separación entre las líneas de alimentación y masa, y 0.5mm de diámetro de cable. La inductancia del cable aproximadamente es:

$$L_{cable} = 4 \cdot l \cdot \ln\left(\frac{2H}{D}\right) = 4 \cdot 150cm \cdot \ln\left(\frac{2 \cdot 1mm}{0{,}5mm}\right) = 832nH$$

Aplicando un margen de seguridad, consideramos que la máxima variación de corriente que vamos a tener en toda la placa es de 2A, y que la máxima variación de tensión admisible en la línea de 5V es de 0.2V. Queremos una reactancia menor que:

$$X_{max} = \frac{\Delta V}{\Delta A} = \frac{0{,}2V}{2A} = 0{,}1\Omega = 100m\Omega$$

El cable alcanza esta reactancia a:

$$f_{ind-cable} = \frac{X_{max}}{2\pi \cdot L_{cable}} = \frac{0{,}1\Omega}{2\pi \cdot 832 \cdot 10^{-9}H} = 19000Hz = 19kHz$$

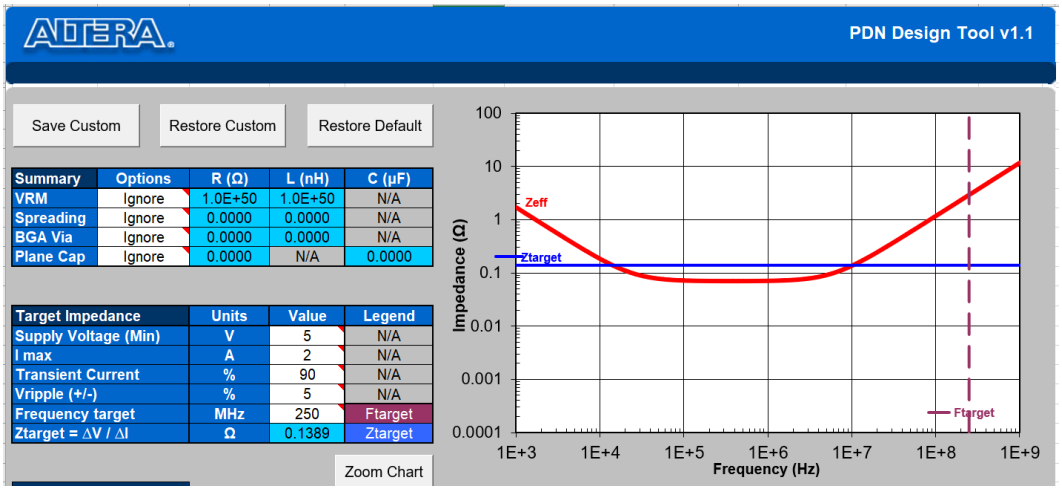
El condensador que presenta 100mΩ a 19kHz es:

$$C_{bulk} = \frac{1}{2\pi \cdot f_{ind-cable} \cdot X_{max}} = \frac{1}{2\pi \cdot 19000Hz \cdot 0{,}1\Omega} = 8{,}3 \cdot 10^{-5}F = 83uF$$

Aplicando más margen de seguridad, utilizaremos 2 condensadores de 47uF en la línea de 5V. El rango de acción de estos condensadores es toda la placa, por lo que su ubicación no es muy crítica, aunque preferiblemente deberían estar situados lo más cerca posible del conector de alimentación.

Situaremos un condensador al lado de cada puerto USB, ya que son los puntos donde más variación de corriente esperamos en la línea de 5V. Uno se utiliza como entrada de alimentación de la placa, y el otro para alimentar un periférico externo.

La capacidad bulk se ha simulado con PDN Tool de la siguiente forma, añadiendo los 2 condensadores de 47uF:



La simulación coincide con lo que hemos calculado anteriormente, el desacoplo de esta capacidad bulk empieza a ser efectivo por encima de los 20kHz y hasta los 5MHz aproximadamente. Por debajo de los 20kHz la inductancia del cable ya no sería tan relevante y podríamos depender de la fuente de alimentación externa a la placa.

4.2.2. Nivel 2. Reguladores de tensión

El diseño utiliza dos reguladores TLV62569 que convierten los 5V de entrada en dos tensiones de 1.35V y 3.3V que se utilizan para alimentar el resto del circuito.

El fabricante recomienda utilizar un condensador de entrada de 4.7uF o superior, y un condensador de salida de entre 10 y 47uF. Además, proporciona una tabla con los valores de condensador de salida que han probado expresamente y que recomiendan según el inductor y el voltaje de salida usados [6]:

8.2.2.3 Output Filter Design

The inductor and output capacitor together provide a low-pass filter. To simplify this process, Table 4 outlines possible inductor and capacitor value combinations. Checked cells represent combinations that are proven for stability by simulation and lab test. Further combinations should be checked for each individual application.

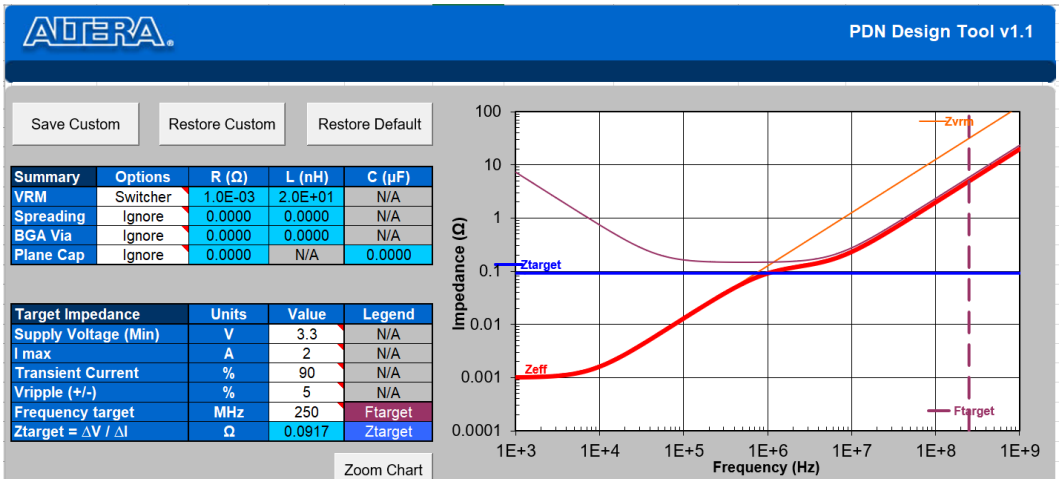
Table 4. Matrix of Output Capacitor and Inductor Combinations

V_{OUT} [V]	L [µH](1)	C_{OUT} [µF](2)				
		4.7	10	22	2 x 22	100
0.6 ≤ V_{OUT} < 1.2	1				+	
	2.2				++(3)	
1.2 ≤ V_{OUT} < 1.8	1			+	+	
	2.2			++(3)	+	
1.8 ≤ V_{OUT}	1		+	+	+	
	2.2		++(3)	+	+	

(1) Inductor tolerance and current de-rating is anticipated. The effective inductance can vary by +20% and -30%.
(2) Capacitance tolerance and bias voltage de-rating is anticipated. The effective capacitance can vary by +20% and -50%.
(3) This LC combination is the standard value and recommended for most applications.

Siguiendo estas recomendaciones se ha optado por utilizar inductores de 2.2uH y situar junto a cada regulador un condensador de 22uF a la entrada y otro de 22uF a la salida. El rango de acción de este nivel también es toda la placa, por lo que la ubicación de los reguladores no es muy crítica, aunque preferiblemente deberán situarse lo más cerca posible del conector de alimentación. Sí es importante situar los condensadores del regulador lo más cerca posible de este.

El efecto de los reguladores en la impedancia de la red de desacoplo se ha simulado también con PDN Tool. En esta simulación solo se ha tenido en cuenta el efecto de un regulador y su condensador de 22uF a la salida, ignorando los condensadores a la entrada del regulador:



Con los componentes del nivel 1 y nivel 2 ya conseguimos un buen resultado en las líneas de 1.35 y 3.3V para frecuencias de 1MHz hacia abajo. Para frecuencias superiores a 1MHz, será necesario añadir condensadores de menor tamaño y más cercanos a los pines de cada componente.

4.2.3. Nivel 3. Condensadores bulk por IC

Por cada circuito integrado o por cada grupo importante de pines, se ha colocado un condensador bulk adicional de mayor capacidad, de entre 1 y 22uF. Dichos condensadores deberán situarse lo más cerca posible de las entradas de alimentación de cada componente, aunque siempre se dará prioridad a colocar más cerca los condensadores del nivel 4.

Estos condensadores actúan en un rango de frecuencias más alto que el de los niveles 1-2, aproximadamente entre 100kHz y 10MHz. La capacidad de estos condensadores se ha elegido según el consumo de cada componente o recomendaciones del fabricante. Por lo general, a mayor consumo máximo de corriente, mayor ha sido el condensador elegido.

4.2.4. Nivel 4. Condensadores de desacoplo por pin

El criterio que se ha seguido en este nivel es el de utilizar un condensador de pequeño valor (de 100 o 220nF) por cada pin de alimentación de cada componente. Cada uno de estos condensadores deberá colocarse lo más cerca posible de su pin correspondiente.

Estos condensadores son los que se encargan de desacoplar la red en el rango más alto de frecuencias, aproximadamente entre 1MHz y 100-250MHz. En los casos en los que era necesario reducir más la impedancia de la red a estas frecuencias, se han colocado varios de estos condensadores en paralelo, siempre del mismo valor para evitar las antirresonancias.

4.2.5. Nivel 5. Capacidad de los planos y on-die

Los niveles 1-4 son efectivos hasta los 100-250MHz, rango que cubre el ancho de banda de todos los componentes, con excepción del SoC y la RAM. Para el SoC y la RAM, por encima de los 100-250MHz los condensadores discretos no son muy efectivos, debido a la inductancia de sus propios encapsulados y de las vías. Sería necesario colocar demasiados condensadores en paralelo, más de los que caben en el espacio disponible. Por este motivo, para frecuencias tan altas dependeremos de la capacidad de los planos de alimentación y sobre todo de la capacidad on-die.

La capacidad de los planos de alimentación en este caso no será muy buena, ya que se usará un stackup de bajo coste de 4 o 6 capas con bastante separación entre los planos de alimentación y masa. En concreto, tendremos 0.865mm de separación en el caso de 4 capas y 0.365mm en el caso de 6 capas.

Podemos estimar la capacidad entre planos de la siguiente forma:

$$C_{4capas} = \frac{\varepsilon_0 \cdot \varepsilon_r \cdot A}{h_{4capas}} = \frac{8{,}85 \cdot 10^{-12} \cdot 4{,}5 \cdot 85 \cdot 10^{-3} \cdot 56 \cdot 10^{-3}}{0{,}865 \cdot 10^{-3}} = 224pF$$

$$C_{6capas} = \frac{\varepsilon_0 \cdot \varepsilon_r \cdot A}{h_{6capas}} = \frac{8{,}85 \cdot 10^{-12} \cdot 4{,}5 \cdot 85 \cdot 10^{-3} \cdot 56 \cdot 10^{-3}}{0{,}365 \cdot 10^{-3}} = 530pF$$

Si necesitamos una reactancia menor de 200mΩ para que el desacoplo sea efectivo, esta capacidad entre planos sería efectiva a partir de:

$$f_{4capas} = \frac{1}{2\pi \cdot X_{max} \cdot C_{4capas}} = \frac{1}{2\pi \cdot 0{,}2\Omega \cdot 224 \cdot 10^{-12}F} = 3{,}6GHz$$

$$f_{6capas} = \frac{1}{2\pi \cdot X_{max} \cdot C_{6capas}} = \frac{1}{2\pi \cdot 0{,}2\Omega \cdot 530 \cdot 10^{-12}F} = 1{,}5GHz$$

Si además tenemos en cuenta que a dichas frecuencias la impedancia realmente será mucho más elevada debido a las discontinuidades y antipads en los planos, así como a la inductancia de las vías y bolas del BGA, podemos concluir que la capacidad entre planos no será efectiva en este caso y podemos despreciarla.

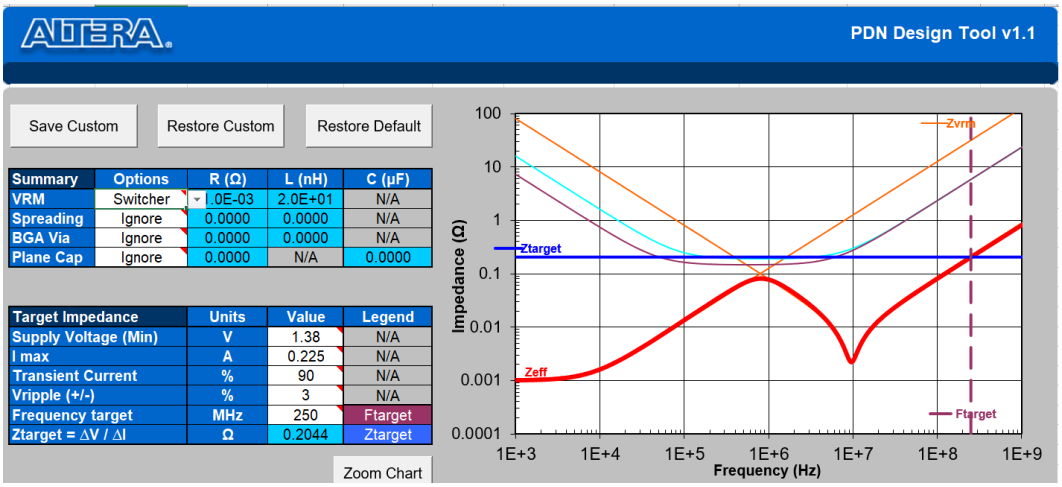
Para desacoplar las frecuencias >100-250MHz, dependeremos por tanto de que estos componentes incorporen cierta capacidad on-die en el propio chip.

4.3. Estudio por componentes

4.3.1. Niveles 3-5. Desacoplo de la RAM

En el caso de la RAM, el fabricante indica que la memoria incluye capacidad on-die y que no depende completamente de los condensadores externos para su funcionamiento [7]. Recomiendan colocar 4 condensadores a cada lado del chip, de 100nF o 1uF. El valor de 100nF se recomienda en diseños que tengan el regulador en la misma placa y suficiente capacidad de bulk adicional, y el de 1uF en el caso de que la fuente de alimentación de la placa esté situada en una placa externa.

En línea con las recomendaciones del fabricante y debido a que no hay espacio suficiente para colocar 18 condensadores de desacoplo, uno por cada pin, se han utilizado 8 condensadores de 220nF de forma que algunos pines comparten el condensador, además de un condensador bulk adicional de 22uF:



Componente	Pines	Voltaje	Condensadores
RAM DDR3L	VDD[1:9], VDDQ[1:9]	1V35	8x 220nF, 1x 22uF

La red de desacoplo obtenida es suficientemente buena hasta los 250MHz. A partir de dicha frecuencia, el desacoplo se realiza por la capacidad on-die del chip.

4.3.2. Niveles 3-5. Desacoplo del SoC

4.3.2.1. Reguladores de tensión integrados en el SoC

El SoC utilizado en el trabajo incorpora varios LDOs integrados en el mismo chip. Estos LDOs toman como entradas las líneas de 1.35V y 3.3V anteriores y las convierten al resto de tensiones que necesita el núcleo del SoC internamente [8] [9].

Las salidas de estos LDOs salen al exterior del SoC por distintos pines, que no se utilizan para alimentar otros componentes de la placa, sino que son utilizados solamente para colocar los condensadores de desacoplo correspondientes al SoC.

El fabricante no especifica expresamente que disponga de capacidad on-die, pero atendiendo a las recomendaciones que hace sobre los condensadores necesarios y a los que utilizan en la placa de referencia, podemos suponer que sí que tiene cierta capacidad on-die a la salida de estos LDOs, de lo contrario no sería posible realizar el desacoplo con los condensadores que indica el fabricante y en el espacio disponible.

4.3.1 Digital Regulators (LDO_ARM, LDO_SOC)

There are two digital LDO regulators ("Digital", because of the logic loads that they drive, not because of their construction). The advantages of the regulators are to reduce the input supply variation because of their input supply ripple rejection and their on-die trimming. This translates into more stable voltage for the on-chip logics.

Dado que las salidas de estos LDOs son utilizadas exclusivamente para alimentar el microprocesador del SoC, se ha decidido seguir en medida de lo posible las recomendaciones del fabricante que se indican en la guía de desarrollo hardware [10].

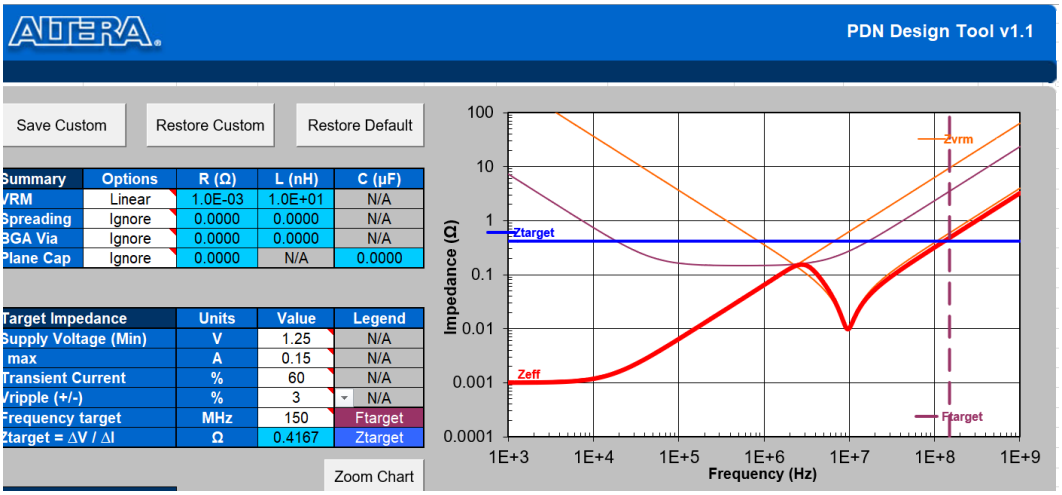
Debido a que no había suficiente espacio debajo del SoC, se han suprimido algunos condensadores bulk a la entrada de los LDOs, lo que no debería ser un problema ya que hay suficiente capacidad en dichas líneas. A la salida de los LDOs, cada condensador 603 de 22uF se ha sustituido por 2 condensadores 402 de 10uF para facilitar la organización de los componentes bajo el SoC. Respecto a los condensadores de 220nF se ha respetado estrictamente lo indicado por el fabricante.

Finalmente, los condensadores utilizados son los siguientes:

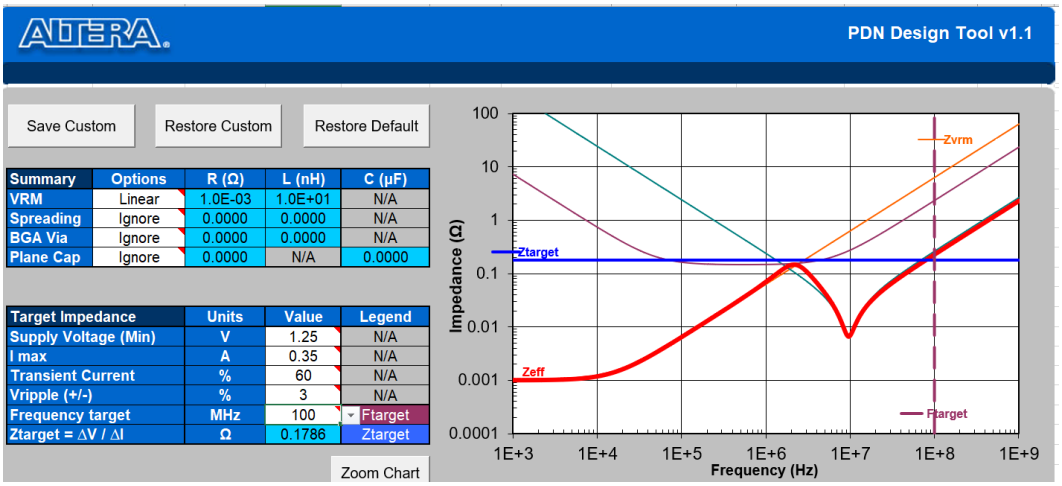
Pines	Voltaje	Condensadores
VDD_SOC_IN[1:6]	1V35	2x 220nF, 2x 10uF
VDD_HIGH_IN	3V3	1x 220nF, 1x 4.7uF
VDD_SNVS_IN	3V3	Compartido con VDD_HIGH_IN
VDD_ARM_CAP[1:4]	VDD_ARM	2x 220nF, 2x 10uF
VDD_SOC_CAP[1:10]	VDD_SOC	3x 220nF, 2x 10uF
VDD_HIGH_CAP[1:2]	2V5	1x 220nF, 1x 10uF
NVCC_PLL	1V1	1x 220nF, 1x 10uF
VDD_SNVS_CAP	VDD_SNVS	1x 220nF
USB_OTG[1:2]_VBUS	5V	1x 1uF
VDD_USB_CAP	VDD_USB	1x 100nF, 1x 10uF

Se ha simulado el desacoplo a la salida de los dos LDOs más importantes, según los datos obtenidos en el datasheet. Los pines VDD_SOC_IN[1:6] son la entrada para ambos reguladores, se indica que el consumo máximo en esta entrada son 500mA, pero no especifica el consumo en cada una de las dos salidas. Ponderando el número de pines que tiene cada salida, supondremos que en VDD_ARM_CAP tendremos un consumo máximo de 150mA y en VDD_SOC_CAP de 350mA.

Para el LDO_ARM (pines VDD_ARM_CAP[1:4]):



Para el LDO_SOC (pines VDD_SOC_CAP[1:10]):



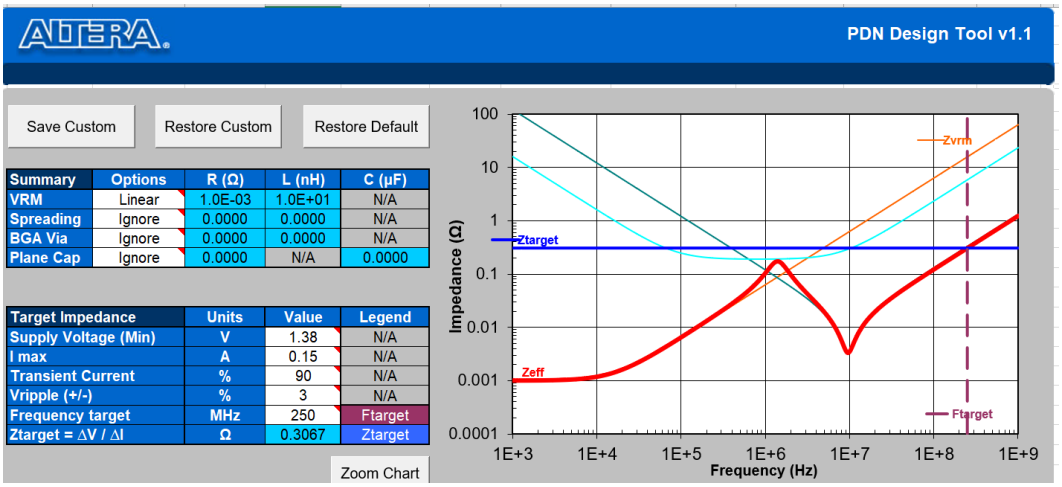
Con estos condensadores obtenemos un buen desacoplo hasta los 100-150MHz. A partir de dicha frecuencia, el desacoplo depende de la capacidad on-die del chip.

4.3.2.2. Desacoplo del resto del SoC

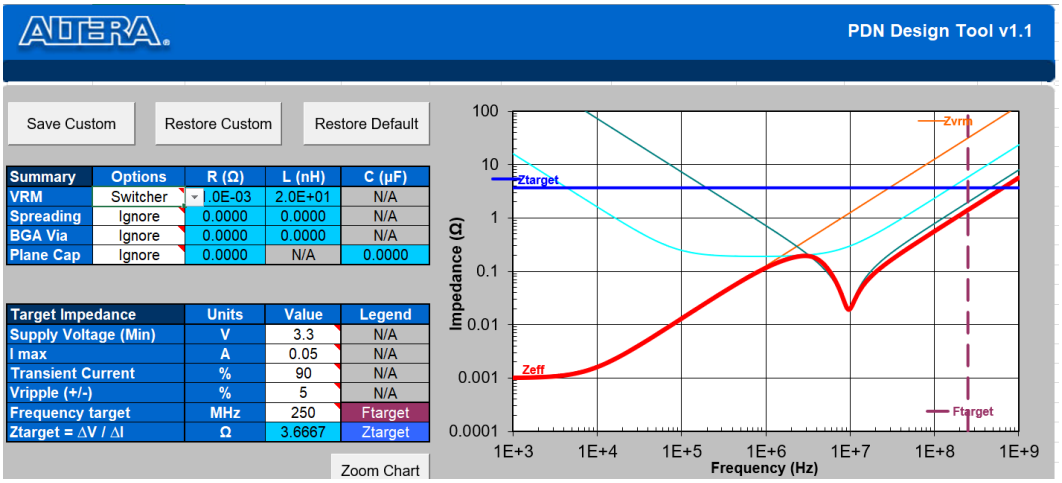
Si bien el núcleo del SoC se alimenta a través de los LDOs internos, el resto de periféricos del SoC (controlador de memoria, periféricos serie) se alimentan por separado y a tensiones diferentes, según queramos que funcione su I/O, en este caso a 1.35V el controlador de memoria y 3.3V el resto de periféricos.

Siguiendo las recomendaciones del fabricante, el criterio que se ha seguido en este caso es el de utilizar un condensador de 100-220nF por cada pin de alimentación del periférico, y un condensador bulk de 10uF en el caso de las entradas NVCC_DRAM que tienen una variación en el consumo de corriente mayor.

Para las entradas NVCC_DRAM[1:6]:



Para el resto de entradas NVCC_x a 3.3V:



Componente	Pines	Voltaje	Condensadores
i.MX6ULL	NVCC_DRAM[1:6]	1V35	5x 220nF, 1x 10uF
i.MX6ULL	NVCC_GPIO	3V3	1x 220nF
i.MX6ULL	NVCC_UART	3V3	Compartido con NVCC_GPIO
i.MX6ULL	NVCC_LCD	3V3	1x 220nF
i.MX6ULL	NVCC_CSI	3V3	1x 220nF
i.MX6ULL	NVCC_ENET	3V3	1x 220nF
i.MX6ULL	NVCC_SD1	3V3	1x 220nF
i.MX6ULL	NVCC_NAND	3V3	1x 220nF
i.MX6ULL	VDDA_ADC_3P3	3V3	1x 100nF

4.3.3. Niveles 3-4. Desacoplo del resto de componentes

El resto de componentes de la placa se alimentan de la línea de 3.3V, tienen un consumo máximo de unos 50mA y un ancho de banda menor a 250MHz.

Para estos componentes se ha seguido el mismo criterio que para las entradas NVCC_x del SoC, se ha colocado un único condensador de 100 o 220nF, y otro condensador de bulk en función de la corriente máxima de cada componente.

Los resultados obtenidos en la simulación anterior para las entradas NVCC_x a 3.3V son también aplicables a estos.

Componente	**Pines**	**Voltaje**	**Condensadores**
NOR Flash	VCC	3V3	1x 220nF
microSD	VDD	3V3	1x 220nF, 1x 22uF
CAN PHY	VCC	3V3	1x 100nF, 1x 1uF
ENET PHY	VDDIO	3V3	1x 100nF
ENET PHY	VDD1A, VDD2A	3V3_PHY	1x 100nF, 1x 1uF

Capítulo 5

Diseño del esquemático y PCB

5.1. Introducción al diseño de alta velocidad

El diseño de alta velocidad hace referencia a sistemas que utilizan señales de alta velocidad para transmitir los datos entre distintos componentes. Es el caso de la mayoría de SBCs y otros sistemas basados en SoCs o FPGAs en los que disponemos de memorias muy rápidas tales como DDR3 o interfaces tales como USB o HDMI.

5.1.1. Longitud crítica de una pista

En el PCB dispondremos de dos tipos de pistas: las que consideremos lentas o eléctricamente cortas, y las que consideremos de alta velocidad o eléctricamente largas.

La línea entre lo que es considerado una pista eléctricamente corta o eléctricamente larga es difusa. Como criterio, para catalogar una pista como eléctricamente corta o larga nos basaremos en la longitud crítica de la pista [2]:

$$L_c = \frac{\lambda}{10}$$

Donde λ es la longitud de onda de la señal. Si la longitud de la pista es menor que L_c, los efectos de línea de transmisión serán poco relevantes por lo que podremos ignorarlos y considerar la pista como lenta. Si la longitud de la pista es mayor que L_c, la consideraremos como de alta velocidad y la trataremos como una línea de transmisión.

5.1.1.1. En señales analógicas

En señales analógicas, se puede calcular λ como:

$$\lambda = \frac{c}{f}$$

Donde c es la velocidad de la luz en el medio y f es la mayor frecuencia del ancho de banda de la señal. La velocidad de la luz en el medio se puede calcular como:

$$c = \frac{c_0}{\sqrt{\varepsilon_r}} = \frac{3 \cdot 10^8}{\sqrt{4}} = 1{,}5 \cdot 10^8 m/s$$

Donde $c_0 = 3 \cdot 10^8 m/s$ es la velocidad de la luz en el vacío y ε_r es la constante dieléctrica del PCB, en este caso tendremos $\varepsilon_r = 4$ para pistas en capas externas.

5.1.1.2. En señales digitales

En señales digitales, la frecuencia a considerar no es la frecuencia de reloj ni la tasa de bits. Debido a los abruptos flancos de subida y bajada de las señales digitales, el espectro de la señal se extiende hasta frecuencias mucho mayores.

El espectro de una señal digital se extiende de manera significativa hasta la denominada frecuencia de codo o *knee frequency*, a partir de la cual la amplitud de los armónicos comienza a decaer con mayor pendiente. Esta frecuencia se define como [11]:

$$f_{knee} = \frac{0{,}5}{t_t}$$

Donde t_t es el menor entre el tiempo de subida y de bajada de los flancos, definido como el tiempo de transición entre el 10 % y el 90 % de la amplitud del pulso.

De aquí resulta la siguiente expresión para calcular L_c en función de t_t:

$$L_c = \frac{\lambda_{knee}}{10} = \frac{c}{10 \cdot f_{knee}} = \frac{1{,}5 \cdot 10^8 \cdot t_t}{10 \cdot 0{,}5} = 3 \cdot 10^7 \cdot t_t$$

Dados los órdenes de magnitud con los que se trabaja habitualmente, es más conveniente utilizar la expresión en la siguiente forma:

$$L_c[cm] = 3 \cdot t_t[ns]$$

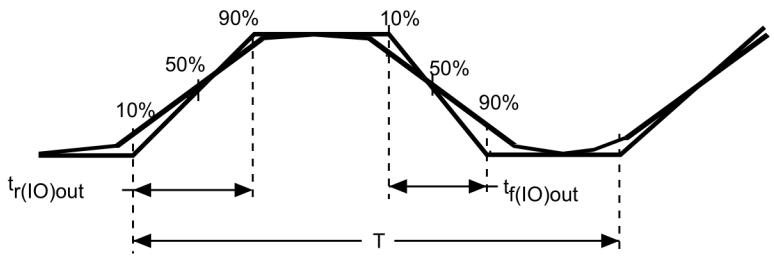
Los tiempos de subida y bajada de los flancos dependen del driver que maneja la señal en cuestión y podemos encontrarlos especificados en el datasheet de cada componente. Usaremos los valores del datasheet como referencia, aunque en función de la carga que haya conectada al driver, t_t podría ser más elevado.

5.1.1.3. El caso de un microcontrolador

A modo de ejemplo, estudiaremos el caso de un microcontrolador STM32F405RG. En este caso, es posible configurar la fuerza del driver de cada pin utilizando un registro de configuración. En función de la configuración del driver y de la carga conectada al pin, en el datasheet se especifican los tiempos de subida y bajada de los flancos y la frecuencia máxima de I/O que se puede conseguir en el pin [12].

A partir de esta información podemos calcular la frecuencia de codo y la longitud crítica de la pista para cada uno de los casos especificados en el datasheet:

t_t **(ns)**	f_{io-max} **(MHz)**	f_{knee} **(MHz)**	L_c **(cm)**
2.5	180	200	7.5
4	100	125	12
6	50	83	18
10	25	50	30
20	12.5	25	60
100	2	5	300



Maximum frequency is achieved if (t_r + t_f) ≤ (2/3)T and if the duty cycle is (45-55%) when loaded by CL specified in the table " *I/O AC characteristics*".

Habitualmente las frecuencias utilizadas para comunicar un microcontrolador con otros periféricos suelen ser menores que unos 50MHz, por lo que si no usamos drivers más rápidos de lo necesario, la longitud crítica será mayor que unos 18cm.

A menudo no dispondremos de pistas tan largas en el PCB, lo que nos permite ignorar los efectos de línea de transmisión cuando se trabaja con microcontroladores.

5.1.1.4. El caso del SoC iMX6ULL

El SoC utilizado en este trabajo dispone de distintos tipos de drivers según la interfaz a la que vayan destinados. En los pines de propósito general, la fuerza del driver también es configurable. Para $V_{io} = 3.3$V y $C_L = 22$pF obtenemos las siguientes longitudes críticas en función de la configuración del driver:

Drive	**Slew Rate**	t_t **(ns)**	f_{io-max} **(MHz)**	f_{knee} **(MHz)**	L_c **(cm)**
Max	Fast	1.7	197	296	5
High	Fast	2.1	156	234	6
Medium	Fast	2.5	132	198	8
Max	Slow	2.7	123	184	8
Low	Fast	3.4	97	145	10
High	Slow	4.0	84	125	12
Medium	Slow	4.5	74	111	14
Low	Slow	5.2	65	97	15

Los valores obtenidos son similares a los que podemos esperar en un microcontrolador. Dadas las frecuencias que usaremos y las pequeñas dimensiones del PCB realizado en el trabajo, las señales que utilicen este tipo de drivers las consideraremos lentas e ignoraremos los efectos de línea de transmisión:

- En el caso de la memoria QSPI, la frecuencia máxima de funcionamiento es de 104 MHz por lo que la L_c será mayor de 8 cm. Las pistas en el PCB son menores de 3 cm por lo que podemos considerarlas lentas.

- En el caso de la memoria microSD, la interfaz funciona a 25 o 50 MHz aunque la especificación establece un t_t máximo de 3 ns. L_c es de unos 8cm. Las pistas en el PCB son menores de 6 cm por lo que podemos considerarlas como lentas.
- En el caso del Ethernet, la interfaz RMII entre el SoC y el PHY LAN8720A funciona a 50 MHz. Todas las pistas son menores a 8 cm y es de esperar que L_c sea mayor que esta longitud, por lo que podemos considerarlas como lentas.
- En el caso del CAN y el resto de señales que van hacia los pin headers, todas las pistas del PCB son menores de 10 cm. Dadas las frecuencias a las que funcionan estas interfaces también podemos considerarlas lentas.

Aunque consideremos como lentas las interfaces entre el SoC y los PHYs del Ethernet y CAN, las interfaces entre los PHYs y los conectores las consideraremos pares diferenciales de alta velocidad, de acuerdo con la especificación de cada protocolo y dado que a estos conectores se espera tener conectado un cable de gran longitud.

Por otro lado, el SoC dispone de drivers más rápidos para interfaces concretas como DDR3L y USB. En estos casos, las especificaciones de la interfaz no vienen especificadas en detalle en el datasheet del SoC sino que tendremos que acudir a los documentos de la especificación de cada protocolo para conocerlas en profundidad.

En el caso de DDR3L-800, tendremos un *slew rate* de 1.75 a 5 V/ns para pistas *single-ended* y de 3.5 a 12 V/ns para pares diferenciales [13]. Podemos esperar un t_t de 100 a 800 ps, lo que nos da una L_c de 0.3 a 2.4 cm. Las pistas en el PCB tienen una longitud de unos 3 cm por lo que tendremos que considerarlas eléctricamente largas.

En el caso de USB 2.0, dada la longitud de los cables y de acuerdo con la especificación consideraremos todas las pistas como pares diferenciales de alta velocidad. El valor de t_t depende del modo de funcionamiento [14]:

Mode	**Bit Rate (Mbps)**	t_t **(ns)**	L_c **(cm)**
USB 2.0 Low Speed	1.5	75-300	225-900
USB 2.0 Full Speed	12	4-20	12-60
USB 2.0 High Speed	480	>0.5	>1.5

5.1.2. Diferencias entre pistas lentas y rápidas

5.1.2.1. Pistas lentas o eléctricamente cortas

Las pistas que consideremos lentas o eléctricamente cortas las consideraremos como conexiones casi ideales, que se encargan de unir eléctricamente dos puntos distintos del circuito. Por lo general, no necesitarán de ninguna atención en especial.

Tampoco será necesario prestar especial atención al camino de la corriente de retorno, que volverá por el camino más ancho y corto posible, buscando siempre el camino de menor resistencia.

Solo en caso de que circule una gran corriente por ellas, será necesario tratar de aumentar la anchura de la pista o reducir su longitud, con el fin de reducir su resistencia y evitar una caída de voltaje excesiva o sobrecalentamiento.

Figura 5.1: Camino de la corriente de retorno en pistas lentas y rápidas [2].

5.1.2.2. Pistas de alta velocidad o eléctricamente largas

Las pistas que consideremos de alta velocidad o eléctricamente largas, sin embargo, no podremos considerarlas como meros conductores ideales, sino que deberemos tratarlas como líneas de transmisión. Idealmente, la impedancia característica de la pista deberá coincidir con la impedancia de salida de la fuente y la impedancia de entrada de la carga.

Será necesario darle forma a la pista de manera que tenga una impedancia característica concreta y que esta impedancia se mantenga inalterada a lo largo de toda la longitud de la pista. De lo contrario, las variaciones en la impedancia provocarán reflexiones y otros efectos indeseados en la señal, que degradarán la forma de la señal hasta el punto de que el sistema deje de funcionar correctamente.

Las corrientes de retorno circularán por el camino de menor inductancia, esto es, por un plano de referencia justo bajo o sobre la pista de ida. Será muy importante por tanto que no haya discontinuidades en los planos de referencia ya que provocarían variaciones en la impedancia de la línea y por tanto la degradación de la señal.

En caso de que una pista de señal cambie de una capa a otra, se producirá también un cambio del plano de referencia por el que circula la corriente de retorno. Para evitar que el camino de retorno se interrumpa, será necesario proporcionar un camino alternativo a esta corriente de retorno.

Para ello será necesario colocar una vía que una ambos planos de referencia lo más cerca posible de la vía de señal, conocida como vía de *stitching*. En caso de que los planos de referencia estén a distinto potencial, es posible utilizar un condensador de stitching, aunque en este caso se producirá una mayor degradación de la señal.

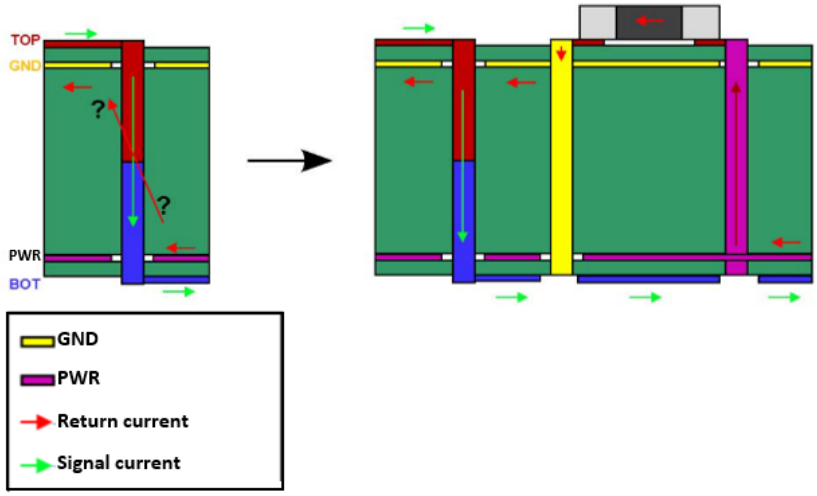


Figura 5.2: Uso de condensadores de stitching [15].

Otro aspecto a tener en cuenta es que una señal tarda un cierto tiempo en recorrer toda una pista desde la fuente hasta su destino. En el stackup utilizado en este trabajo se ha calculado en Altium un retardo de 6ps/mm, es decir, un mm adicional de pista conlleva retrasar en 6ps la llegada de la señal a su destino.

Dadas las altas frecuencias a las que se trabaja, las diferencias en la longitud de las pistas que forman un par diferencial o un bus paralelo podrían provocar errores en el muestreo de los datos al no llegar todos al mismo tiempo. Será necesario realizar un *length matching* de las pistas del bus para que todas tengan la misma longitud.

En la memoria DDR3L-800 utilizada el periodo de un bit es de 1250ps. Esto hace que se tenga cierto margen: una diferencia de longitud de unos 5-10mm entre la pista más corta y la más larga puede ser asumible [4]. Aun así, se realizará un *length matching* más estricto de acuerdo con las reglas indicadas por el fabricante.

5.1.3. Impedancia característica de una pista

Las pistas de alta velocidad las consideraremos líneas de transmisión. Esto implica que se debe diseñar la pista de forma que tenga una impedancia característica concreta y que esta se mantenga inalterada a lo largo de toda su longitud.

La impedancia característica de una línea viene dada por la relación entre la autoinductancia y la capacidad por unidad de longitud. La línea de transmisión se puede modelar como un circuito formado por diferentes etapas LC sucesivas. La relación entre L_0 y C_0 debe mantenerse inalterada a lo largo de toda la linea de transmisión [2].

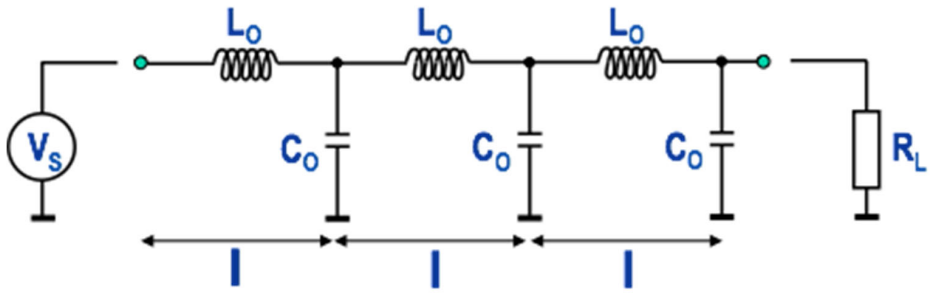


Figura 5.3: Modelo distribuido de una línea de transmisión [2].

A partir de la relación entre L_0 y C_0 es posible obtener la impedancia característica de la línea, así como el retardo de propagación de la señal:

$$Z_0 = \sqrt{\frac{L_0}{C_0}}$$

$$t_{p0} = \sqrt{L_0 \cdot C_0}$$

Los valores de L_0 y C_0 dependen de la construcción física de la línea. En la práctica, para una pista de un PCB estimaremos los valores de Z_0 y t_{p0} con un programa de ordenador a partir de la geometría de las pistas y las características del stackup del PCB. Los valores típicos serán de 40-60 Ω para Z_0 y 6-7 ps/mm para t_{p0}.

Los parámetros que más influyen en la impedancia de línea son la anchura de pista (w), el espesor de dieléctrico entre pista y plano de referencia (h) y la constante dieléctrica del material (ε_r). Otros parámetros como el espesor de cobre de la pista y el espesor y constante dieléctrica del solder mask también influyen en menor medida.

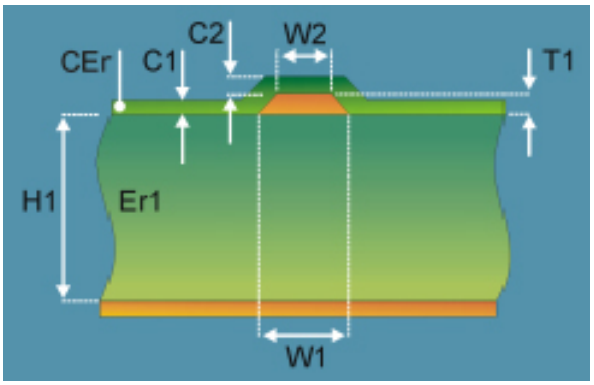


Figura 5.4: Parámetros de una pista en capas externas o *microstrip* [16].

Será por tanto imprescindible seleccionar correctamente el stackup del PCB de forma que sea posible conseguir las impedancias características que necesitamos para las distintas pistas del diseño, así como asegurar que se dispone de un plano de referencia sin discontinuidades junto a cada pista considerada de alta velocidad.

5.2. Diseño del stackup del PCB

5.2.1. Selección del fabricante del PCB

Para que el PCB pueda ser fabricado en tiradas muy pequeñas de la manera más económica posible, se ha evitado diseñar un stackup a medida. En su lugar se ha utilizado un stackup estándar del fabricante JLCPCB.

JLCPCB ofrece dos tipos de stackup, JLC7628 y JLC2313 [17] [16]. El JLC2313 se encuentra disponible en 4 y 6 capas mientras que el JLC7628 solo está disponible en 4 capas. La principal diferencia es el espesor de prepreg, siendo de unos 0.1mm en el caso del JLC2313 y de unos 0.2mm en el caso del JLC7628.

Se ha comprobado que el espesor del prepreg influye en gran medida a la hora de determinar la impedancia de la pista. Por ello se ha decidido utilizar el JLC2313 ya que, para conseguir una determinada impedancia de pista, la anchura de pista requerida es mucho menor. El JLC7628 necesitaría pistas demasiado anchas haciendo inviable rutar el diseño, especialmente bajo los componentes con encapsulado BGA.

5.2.2. Propuestas de stackup

5.2.2.1. Propuesta de 4 capas

En la propuesta de 4 capas se utilizaría la siguiente distribución:

JLC2313 Stackup (4-Layer, 1.2mm thickness)					
Copper Layer Nº	**Name**	**Material**	**Weight (oz)**	**Thickness (mm)**	**Dielectric Constant**
	Top Silkscreen	Silkscreen			
	Top Solder Mask	Solder Mask		0.0203	3.80
1	Signal	Copper	1.0	0.0350	
		Prepreg 2313		0.0889	4.05
2	Plane (GND)	Copper	0.5	0.0175	
		Core		0.8650	4.50
3	Plane (Power)	Copper	0.5	0.0175	
		Prepreg 2313		0.0889	4.05
4	Signal	Copper	1.0	0.0350	
	Bottom Solder Mask	Solder Mask		0.0203	3.80
	Bottom Silkscreen	Silkscreen			
			Total Thickness	**1.1884**	

En la capa 2 se dispone de un plano continuo de masa. La capa 3 estaría dividida en dos planos de alimentación de 1.35V y 3.3V. Tanto el SoC como la RAM se alimentan de la misma línea de 1.35V, lo que permite tener en la capa 3 un plano de 1.35V bastante grande y sin discontinuidades bajo las pistas que unen SoC y RAM.

Las pistas de señal de alta velocidad serían rutadas en las capas 1 o 4 y usarían como referencia el plano GND de capa 2 o el plano de 1.35V de capa 3 según corresponda. Se dará prioridad a rutar todo lo posible en la capa 1 al tener una mejor referencia, pero por falta de espacio muchas pistas serán rutadas en la capa 4.

En caso de que una pista de alta velocidad cambie de la capa 1 a la 4, es necesario tener un condensador de *stitching* lo más cerca posible de la vía [15]. Este condensador degradará el retorno de la señal ya que el ancho de banda en el que es efectivo es limitado. Se ha tratado de evitar en lo posible esta situación, rutando casi toda la pista o bien en la capa 1, o bien en la 4, sin cambios de capa excepto junto a los pads debajo del propio chip y muy cerca de los condensadores de desacoplo.

5.2.2.2. Propuesta de 6 capas

En la propuesta de 6 capas se utilizaría la siguiente distribución:

JLC2313 Stackup (6-Layer, 1.2mm thickness)					
Copper Layer Nº	**Name**	**Material**	**Weight (oz)**	**Thickness (mm)**	**Dielectric Constant**
	Top Silkscreen	Silkscreen			
	Top Solder Mask	Solder Mask		0.0203	3.80
1	Signal	Copper	1.0	0.0350	
		Prepreg 2313		0.0889	4.05
2	Plane (GND)	Copper	0.5	0.0175	
		Core		0.3650	4.50
3	Plane (Power)	Copper	0.5	0.0175	
		Prepreg 2116		0.1270	4.25
4	Signal (Slow)	Copper	0.5	0.0175	
		Core		0.3650	4.50
5	Plane (GND)	Copper	0.5	0.0175	
		Prepreg 2313		0.0889	4.05
6	Signal	Copper	1.0	0.0350	
	Bottom Solder Mask	Solder Mask		0.0203	3.80
	Bottom Silkscreen	Silkscreen			
			Total Thickness	**1.2154**	

Este stackup proporciona varias ventajas respecto al de 4 capas:

- Ambas capas externas tendrían ahora planos de masa como referencia. Esto permitiría rutar las pistas de alta velocidad de manera indistinta en las capas 1 y 6, sin preocuparse de las posibles discontinuidades en el plano de alimentación.
- Cuando una pista de alta velocidad necesite cambiar de la capa 1 a la 6, para dar continuidad al camino de la corriente de retorno es suficiente con añadir una vía de stitching conectando ambos planos de referencia lo más cerca posible de la vía de señal. De esta forma se mejora la integridad de señal, la señal se degradará menos que haciendo el cambio de capa a través de un condensador.
- Mejor conductividad térmica lateral en el PCB gracias a las capas extra de cobre con planos continuos. Tal y como se ha determinado en el capítulo 3, de esta forma se reduciría la temperatura máxima que alcanzan los componentes.
- El plano de alimentación estaría más cerca del de masa. Tal y como se ha determinado en el capítulo 4, de esta forma habría mayor capacidad entre estos planos y realizarían una mejor función de desacoplo en altas frecuencias.
- Se gana una tercera capa de señal, la capa 4, que podría ser utilizada para rutar señales de baja frecuencia o que no requieran tener la impedancia controlada.

5.2.3. Selección final del stackup

5.2.3.1. Diferencias en el precio

En una tirada relativamente grande, la diferencia de coste entre un PCB de 4 y 6 capas puede no ser muy significativa. Sin embargo, si solo se fabrican unos pocos prototipos, la diferencia de precio es mucho mayor, costando entre 4 y 5 veces más fabricar un PCB de 6 capas que uno de 4 capas, en este fabricante en particular.

Esta diferencia en el precio se debe sobre todo a la diferencia en el volumen de fabricación. Los fabricantes de prototipos de bajo coste combinan los diseños de varios clientes en un único panel más grande a la hora de fabricarlo. Estos fabricantes reciben muchos más pedidos de 4 capas que de 6, lo que contribuye a reducir el coste.

5.2.3.2. Diferencias en la integridad de señal

El mayor inconveniente que tiene el stackup de 4 capas es la mayor degradación de las señales del bus de la memoria RAM debido a las discontinuidades en el camino de retorno de las corrientes al cambiar de plano de referencia.

Sin embargo, se ha comprobado que en la práctica existen en el mercado dispositivos que funcionan correctamente utilizando este tipo de stackup con este tipo de SoCs y memorias. Las guías de desarrollo de NXP [10] y ST [15] asumen un stackup de 4 capas y NXP lo utiliza en la placa de desarrollo MCIMX6ULL-EVK [18].

Es el caso también de la BeagleBone Black. Aunque la BeagleBone utiliza un stackup de 6 capas, solo dispone de 2 planos de referencia, uno de masa y otro de alimentación. La referencia de algunas de las pistas de la interfaz de la memoria DDR3L cambia entre estos dos planos y a pesar de ello también funciona correctamente [19].

De aquí podemos concluir que para el tipo de SoC y memoria utilizados en el trabajo podemos esperar que el resultado obtenido con el stackup de 4 capas sea suficientemente bueno. En caso de haber utilizado un SoC más potente con una interfaz de memoria más rápida, probablemente el stackup de 4 capas no habría sido suficiente.

5.2.3.3. Compatibilidad con ambas propuestas

Finalmente el diseño se ha preparado para poder ser fabricado tanto en 4 como en 6 capas. Para ello se han tenido en cuenta los siguientes aspectos:

- Solo se utilizan las dos capas más externas como capas de señal.
- El espesor final del PCB es el mismo en ambos stackups (1.2mm), por lo que en caso de cambiar de stackup el *length-matching* realizado seguiría siendo válido y no sería necesario reajustar la longitud de las pistas.
- Los prepregs más exteriores son idénticos en ambos stackups (0.0889mm) por lo que en caso de cambiar de stackup los perfiles de impedancia calculados seguirían siendo válidos y no sería necesario reajustar la anchura de las pistas.
- Los cambios de capa de las pistas del bus de la memoria RAM se realizan cerca de condensadores de desacoplo y cerca de vías de masa. De esta forma se le da continuidad a las corrientes de retorno en los planos de referencia con ambos stackups, aunque cabe esperar que con el de 4 capas se obtenga peor resultado.

Inicialmente se utilizará el stackup de 4 capas. En caso de que posteriormente se detectase algún problema de integridad de señal o EMC, sería posible migrar el diseño al stackup de 6 capas con cambios mínimos en el PCB, ya que solo sería necesario añadir dos planos extra de masa y cambiar el orden de los planos.

5.2.4. Introducción del stackup en Altium Designer

El stackup se ha introducido en Altium Designer de la siguiente forma:

#	Name	Material	Type	Weight	Thickness	Dk	Df
	Top Overlay		Overlay				
	Top Solder Mask	Solder Mask	Solder Mask		0.0203mm	3.8	
1	L1	Copper	Signal	1oz	0.035mm		
	Prepreg 1	2313	Prepreg		0.0889mm	4.05	0.02
2	L2 (GND)	Copper	Plane	1/2oz	0.0175mm		
	Core	FR-4	Core		0.865mm	4.5	
3	L3 (PWR)	Copper	Plane	1/2oz	0.0175mm		
	Prepreg 2	2313	Prepreg		0.0889mm	4.05	0.02
4	L4	Copper	Signal	1oz	0.035mm		
	Bottom Solder Mask	Solder Mask	Solder Mask		0.0203mm	3.8	
	Bottom Overlay		Overlay				

Se ha introducido también el espesor y constante dieléctrica del *solder mask*, ya que se ha comprobado que su efecto era significativo. Para una determinada anchura de pista, al considerar el solder mask aumentaba en 1-2 Ω la impedancia calculada.

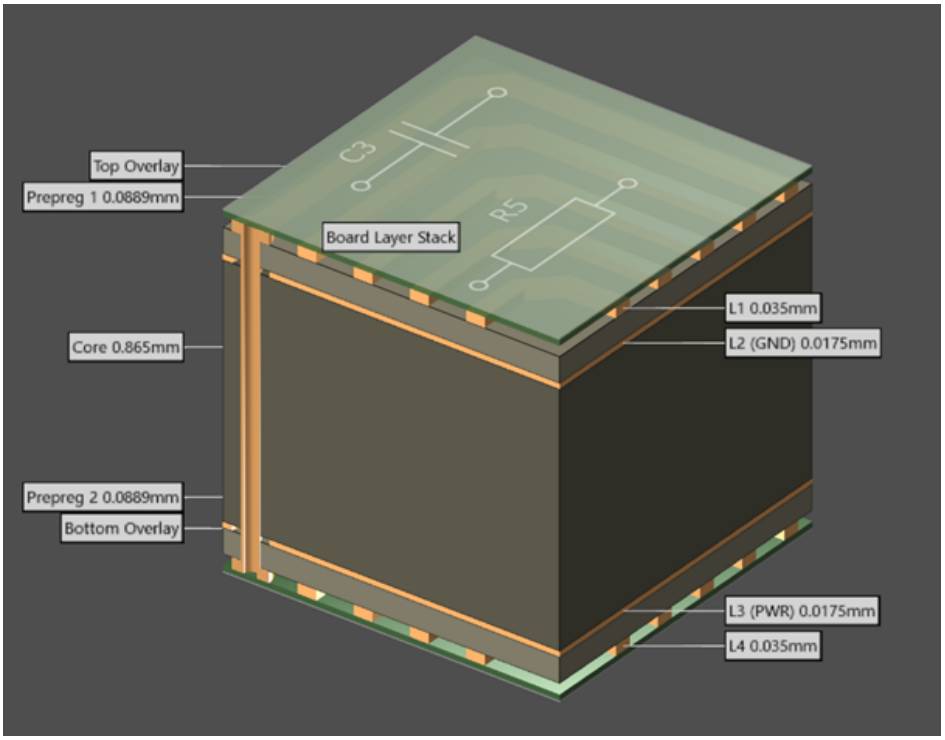


Figura 5.5: Vista 3D del stackup utilizado en el trabajo.

En cuanto a los planos internos, finalmente se han configurado en Altium como tipo *Signal* en lugar de como tipo *Plane*. Esto es debido a que si se configuran como Plane, los *gerbers* de dichas capas son exportados por Altium en negativo, y JLCPCB indica que todos los gerbers deben ser proporcionados en positivo.

En cuanto a las vías, JLCPCB no admite microvías. Se utilizará únicamente un tipo de vía pasante de 0.2mm de diámetro de agujero y 0.45mm de diámetro exterior.

5.3. Obtención de los perfiles de impedancias

Una vez introducidos todos los parámetros del stackup en Altium Designer, podemos crear los distintos perfiles de impedancias necesarios en Altium para obtener cuál es la anchura de pista necesaria para conseguir una impedancia determinada.

JLCPCB también ofrece una calculadora para obtener estos datos [16]. En la siguiente tabla se muestra la anchura de pista necesaria en función del valor de impedancia característica deseado, según las calculadoras de Altium y de JLCPCB:

Impedancia	**Anchura de pista (Altium)**	**Anchura de pista (JLCPCB)**
40 Ω	0.220 mm	0.222 mm
45 Ω	0.178 mm	0.180 mm
50 Ω	0.144 mm	0.147 mm
55 Ω	0.117 mm	0.120 mm
60 Ω	0.094 mm	0.098 mm

Se observa como ambas calculadoras arrojan valores muy similares. Se ha probado también la calculadora de Saturn PCB pero este software arrojaba valores ligeramente distintos. Esto es debido a que Saturn PCB no tiene en cuenta el solder mask, algo que sí que tienen en cuenta las otras dos herramientas.

Idealmente, para reducir las reflexiones y mejorar la integridad de señal, queremos que la impedancia característica de la pista coincida lo mejor posible con la impedancia de salida de la fuente y con la impedancia de entrada de la carga.

A continuación, estudiaremos los requisitos que tienen en este aspecto las distintas pistas de alta velocidad que consideraremos en el diseño.

5.3.1. Pistas single-ended

5.3.1.1. Líneas bidireccionales

En el caso de la interfaz DDR3L, la impedancia de salida de los drivers es configurable tanto en el SoC como en la RAM. El SoC permite seleccionar entre distintos valores según un divisor de 240 Ω, entre los que se incluyen 80 Ω, 60 Ω, 48 Ω, 40 Ω o 34 Ω. La RAM solo permite seleccionar entre dos valores, 40 Ω o 34 Ω.

En cuanto a la impedancia de entrada, en las líneas de datos, tanto el SoC como la RAM disponen de resistencias de terminación configurables *On-Die Termination* (ODT). Estas resistencias solo aplican a las líneas bidireccionales y solo se activan cuando el pin está funcionando como receptor. Ambos chips permiten configurar estas resistencias entre distintos valores: 20 Ω, 30 Ω, 40 Ω, 60 Ω o 120 Ω.

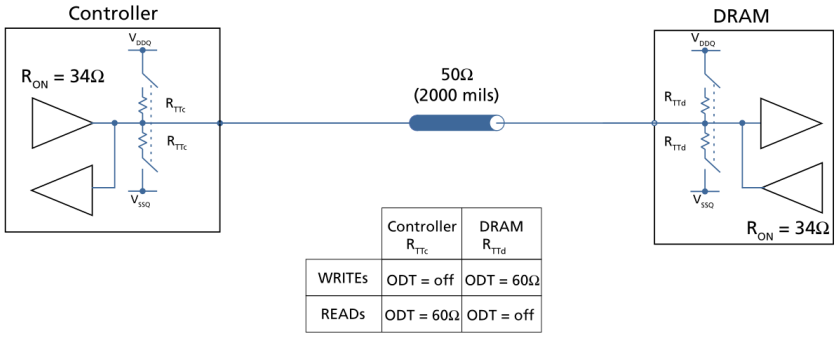


Figura 5.6: Conexión entre SoC y RAM en líneas bidireccionales [7].

En este tipo de líneas, configuraremos la impedancia de salida de los drivers tanto del SoC como de la RAM en 40 Ω, ya que es el máximo valor común a ambos chips. Las resistencias de terminación ODT se configurarán en 60 Ω en ambos chips.

Dada esta configuración, lo ideal sería utilizar una línea de transmisión de 40 Ω. Sin embargo, de acuerdo con las guías y notas de aplicación de los fabricantes, cabe esperar tener un resultado suficientemente bueno si se utilizan pistas entre 45 y 60 Ω.

5.3.1.2. Líneas unidireccionales

En las líneas unidireccionales como las de dirección y control, el SoC siempre actúa como transmisor y la RAM como receptor. En estas líneas, la RAM no dispone de terminaciones configurables a su entrada. En sistemas en los que estas líneas unidireccionales son largas y se disponen de varios chips de RAM conectados a ellas, es necesario terminarlas a VTT (un voltaje derivado de VDDQ/2) con resistencias situadas cerca del último chip. Este esquema es conocido como *fly-by topology*.

Sin embargo, para el caso de este trabajo en el que solo se dispone de un único chip de memoria situado muy cerca del SoC, es habitual no colocar las terminaciones a VTT y en su lugar conectar las líneas directamente de un chip a otro, lo que se conoce como *point-to-point topology*. Algunos fabricantes recomiendan en este caso introducir una resistencia en serie en cada una de las líneas, con el fin de reducir las reflexiones.

A menudo no es posible añadir estas resistencias por falta de espacio en el PCB. En este caso, no se pondrán estas resistencias. Será posible aumentar la impedancia de salida del driver del SoC cambiando su configuración, de forma que coincida lo mejor posible con la impedancia de la línea.

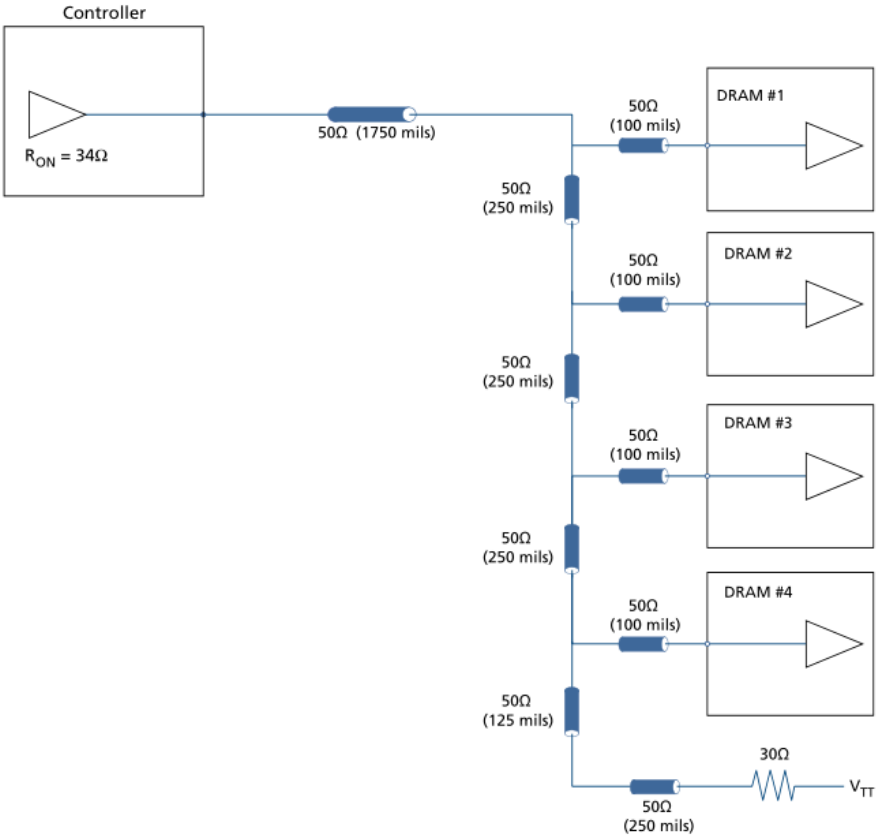


Figura 5.7: Topología *fly-by* en líneas unidireccionales [7].

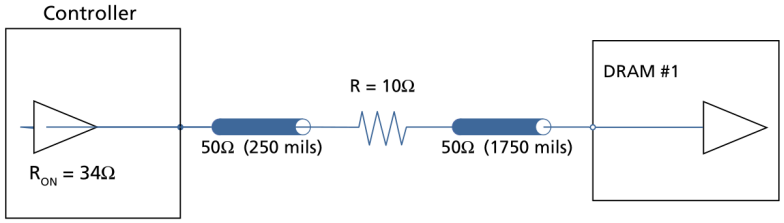


Figura 5.8: Topología *point-to-point* en líneas unidireccionales.

5.3.2. Pares diferenciales

En cuanto a los pares diferenciales, la impedancia a utilizar depende de la interfaz. Siguiendo lo indicado por la especificación de cada interfaz, utilizaremos una impedancia diferencial de 90 Ω en el USB y de 100 Ω en DDR3L y Ethernet.

5.3.3. Perfiles de impedancias

Dado el stackup utilizado, no hay mucho margen a la hora de seleccionar la impedancia de pista. Lo ideal sería utilizar pistas de 40 Ω con el fin de que la impedancia de la pista coincida lo mejor posible con la impedancia de salida de los drivers.

Sin embargo, con el stackup de bajo coste utilizado, las pistas de 40 Ω son demasiado anchas: 0.220 mm. Esto hace que sea inviable rutar el diseño, al no poder escapar los BGA adecuadamente y necesitar demasiado espacio para el rutado.

Finalmente, como compromiso, se utilizará una anchura de 0.12 mm para todas las pistas, ya que es la máxima anchura para la que es factible rutar el diseño.

Z_{o-obj}	Z_{o-real}	**Dev.**	**Trace Width**	**Trace Gap**	t_{p0} **ns/m**	L_0 **nH/m**	C_0 **pF/m**
50 Ω S.E.	54.3 Ω S.E.	8.6 %	0.12 mm	-	6.03	327.6	111.1
90 Ω Diff.	90.2 Ω Diff.	0.2 %	0.12 mm	0.12 mm	5.80	522.6	64.3
100 Ω Diff.	99.5 Ω Diff.	0.6 %	0.12 mm	0.21 mm	5.85	581.5	58.8

Con una anchura de pista de 0.12 mm se obtiene una impedancia de 54.3 Ω según las calculadoras de Altium y JLCPCB. Este valor está dentro del margen de tolerancia de $50\Omega \pm 10\,\%$ que recomienda utilizar el fabricante del SoC [10], por lo que se espera que el resultado sea suficientemente bueno en un sistema de este tipo.

En el capítulo 6 se realizarán simulaciones de integridad de señal para comprobar si las asunciones hechas en este apartado son correctas. En caso de detectar algún problema, será posible modificar la configuración de los drivers y resistencias ODT, así como reducir la anchura de pista si es necesario, con el fin de encontrar la combinación de parámetros con la que se obtenga el mejor resultado.

5.4. Diseño del esquemático y BOM

El diseño del esquemático y PCB se ha realizado utilizando el software Altium Designer. En el apéndice A se ha incluido el esquemático completo realizado.

El *Bill of Materials* (BOM) se ha generado a partir del esquemático utilizando un *Output Job File* de Altium. En el apéndice B se ha incluido el BOM completo.

5.5. Diseño del PCB layout

5.5.1. Reglas de diseño

Al contrario que otros fabricantes, JLCPCB no diferencia entre varias clases, sino que aplican las mismas reglas a todo lo que fabrican. Sí que hay distintas reglas según la capa, las capas externas tienen mejores especificaciones que las internas [20]. Para facilitar la fabricación y conseguir un mejor *yield*, se han evitado apurar las reglas:

- Se han usado vías de 0.2mm de diámetro de agujero y 0.45mm de diámetro exterior, a pesar de que admiten 0.40mm de diámetro exterior.
- En capas externas, se ha usado una anchura mínima y una separación mínima de 0.12mm/0.12mm, a pesar de que admiten 0.09mm/0.09mm.
- En capas internas, se ha usado una anchura mínima y una separación mínima de 0.15mm/0.20mm, a pesar de que admiten 0.127mm/0.127mm.
- Se han mantenido 0.4mm de separación entre el cobre y el borde exterior de la placa. Esto permite panelizar la placa utilizando V-Cut.

El resto de reglas se han configurado como indican los límites del fabricante.

5.5.1.1. Consideraciones especiales bajo BGAs en capas externas

Dado que el pitch de los componentes BGA utilizados es de 0.8mm, para conseguir escapar todas las pistas adecuadamente es necesario colocar vías bajo el componente separadas con una distancia entre centros de 0.8mm. Además, se necesario que las pistas puedan pasar entre dos vías separadas dicha distancia.

La separación mínima entre el pad de la vía y la pista se ha fijado en 0.115mm, esto permite a una pista de 0.12mm pasar entre dos vías bajo el BGA:

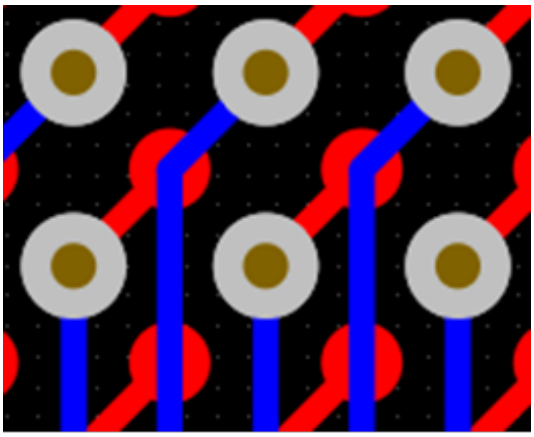


Figura 5.9: Detalle del rutado bajo el SoC.

Por el mismo motivo, la separación mínima entre el agujero de la vía y la pista se ha fijado en 0.24mm, permitiendo así el paso de una pista de 0.12mm entre dos vías. Esto incumple la regla de JLCPCB que indica que este parámetro debe ser de 0.254mm como mínimo, pero no debería ser un problema ya que la diferencia es muy pequeña. En caso de que finalmente el fabricante no lo admita, será necesario reducir la anchura de pista a 0.092mm en los segmentos que pasen entre vías bajo estos componentes.

5.5.1.2. Consideraciones especiales bajo BGAs en capas internas

Otro aspecto a tener en cuenta bajo los componentes BGA es mantener la continuidad de los planos en las capas internas, asegurando que los planos de alimentación y masa sean capaces de llegar a la zona central bajo el BGA y alcanzar las vías correspondientes.

En capas internas, la mayor anchura mínima y mayor separación mínima impedían que el cobre pasase entre los pads de las vías bajo el BGA, partiendo completamente los planos en las zonas donde hay varias vías contiguas.

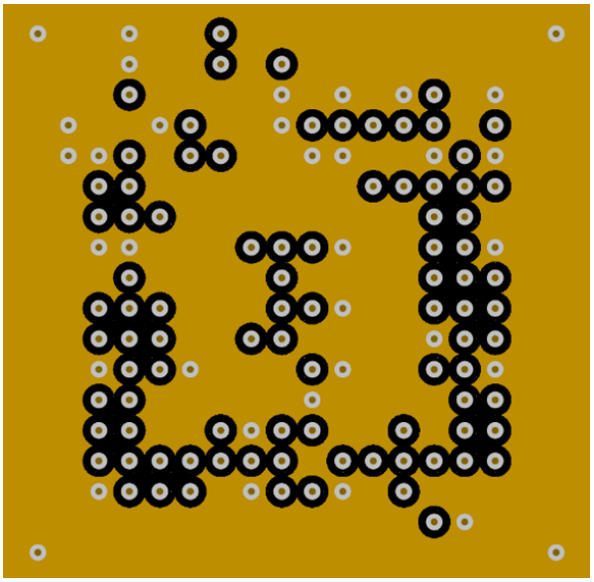


Figura 5.10: Plano de masa inicial bajo el SoC.

La solución ha sido utilizar la opción de Altium para eliminar los pads de las vías sin conectar en capas internas, esto permite que se reduzca el diámetro del antipad alrededor de la vía, permitiendo así que el cobre pase entre las vías bajo el BGA.

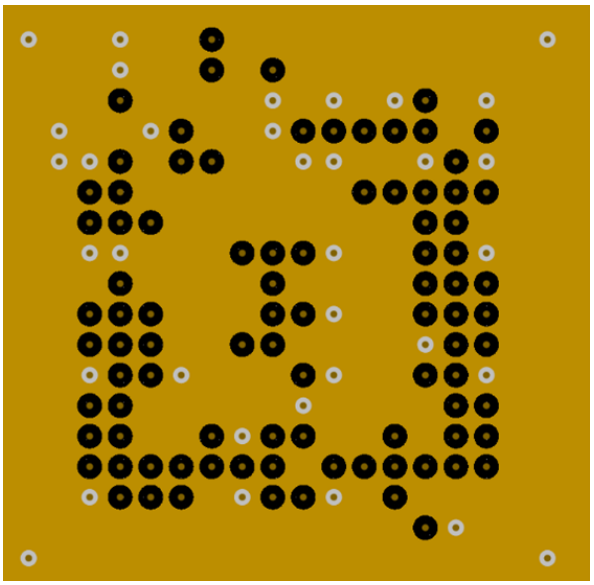


Figura 5.11: Plano de masa final bajo el SoC.

5.5.2. Rutado de la memoria DDR3L

La interfaz DDR3L distingue entre varios grupos de señales. Los dos grupos de bytes de datos son intercambiables entre sí, es decir, es posible conectar el byte 0 del SoC al byte 1 de la RAM y viceversa. También es posible intercambiar los bits dentro de cada byte. Para facilitar el rutado del PCB y reducir los cruces entre pistas, en este caso se ha tenido que realizar el swap de los dos bytes de datos, y también se ha realizado el swap de algunos bits dentro de cada byte.

Chip signals	Group	Length (mils) Min		Recommendations
DRAM_SDCLK0 DRAM_SDCLK0_B	Clock	Short as possible	2 inches	Match the signals ± 5 mils.
DRAM_A[15:0] DRAM_SDBA[2:0] DRAM_RAS DRAM_CAS DRAM_SDWE	Address and Command	Clock (min) – 200	Clock (min)1	Match the signals ± 25 mils.
DRAM_D[7:0] DRAM_DQM0 DRAM_SDQS0 DRAM_SDQS0_B	Byte Group 1	—	Clock (min)	Match the signals of each byte group ± 25 mils. Match the differential signals of DQS ± 10 mils.
DRAM_D[15:8] DRAM_DQM1 DRAM_SDQS1 DRAM_SDQS1_B	Byte Group 2	—	Clock (min)	—
DRAM_CS[1:0] DRAM_SDCKE[1:0] DRAM_SDODT[1:0]	Control signals	Clock (min) – 200	Clock (min)	Match the signals ± 50 mils.

1. Clock (min)—the shortest length of the clock group signals because this group has a ±5 mil matching tolerance. Finally, the impedance for the signals must be 50 Ω for the single-ended and 100 Ω for the differential pairs.

Figura 5.12: Reglas para el rutado DDR3L por grupos de señales [10].

Lo más recomendable es que todas las pistas de un mismo grupo sean rutadas de la misma manera, esto es, que sean rutadas en la misma capa, que usen los mismos planos de referencia, y que tengan el mismo número de vías y cambios de capa, así como la misma longitud en cada capa. Esto facilita que todas las señales de un mismo grupo lleguen a la vez a su destino, ya que se elimina el efecto de la posible variación en el retardo de propagación entre las distintas capas y al pasar por las vías.

Este enfoque además tiene la ventaja de que así el rutado es independiente del espesor del stackup. Sería posible aumentar o reducir el espesor de las capas del PCB sin que se viese afectado el length matching, ya que todas las pistas de un mismo grupo pasarían por el mismo número de vías. En este caso, dado que solo se disponía de dos capas para el rutado, no ha sido posible respetar estas recomendaciones. Las pistas de los distintos grupos se han rutado indistintamente por ambas capas.

Sí que se ha procurado rutar casi la totalidad de cada pista o bien por la capa superior, o bien por la capa inferior, evitando los cambios de capa a mitad del camino. En algunos casos no ha sido posible evitar cruces entre pistas y por tanto cambios de plano de referencia. En estos casos, para evitar añadir condensadores adicionales, la vía se ha colocado debajo del chip, lo más cerca posible del pad o de un condensador de desacoplo, tratando así de reducir los bucles de las corrientes de retorno.

Se ha procurado también que haya vías a masa lo más cerca posible de estos cambios de capa de forma que, en caso de pasar de 4 a 6 capas, estas vías proporcionarían un camino a las corrientes de retorno entre los planos de masa superior e inferior. En el caso del stackup de 4 capas, se depende de los condensadores de desacoplo cercanos para dar continuidad a las corrientes de retorno.

También se ha tratado de mantener suficiente separación entre las pistas para reducir el *crosstalk*. Los fabricantes recomiendan mantener una separación entre pistas de al menos 3S (3 veces la distancia entre la pista y su plano de referencia). Se ha tratado de respetar esta regla en medida de lo posible, aunque en algunos segmentos por falta de espacio se ha tenido que reducir a 2S o 1S, especialmente bajo los BGAs.

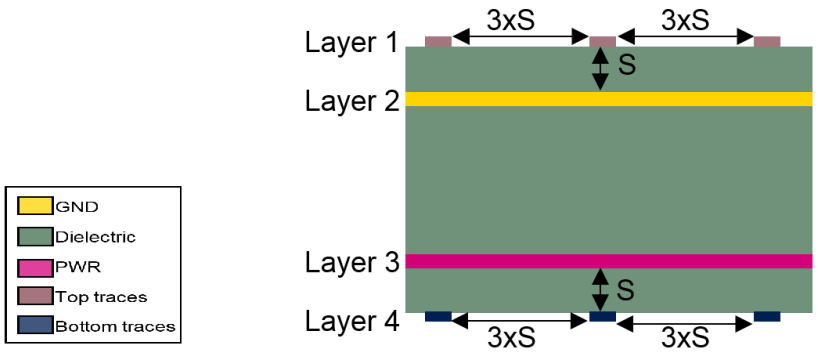


Figura 5.13: Distancia mínima entre pistas para minimizar el *crosstalk* [15].

5.5.3. Length matching

El *length matching* consiste en igualar la longitud de las distintas pistas que forman parte de un mismo bus, con el fin de que el retardo de propagación no provoque diferencias en el tiempo de llegada de las distintas señales a su destino. En este trabajo el length matching se ha aplicado a las señales relativas al USB y a la memoria RAM.

Por defecto Altium solo es capaz de calcular la longitud total rutada de la red. Esto no es un inconveniente al ajustar redes que solo disponen de dos pads (por ejemplo, una pista que viaja desde un pad del SoC hasta un pad de la RAM), ya que la longitud que se pretende ajustar coincide con la longitud total rutada de la red.

Sin embargo, es un problema a la hora de ajustar redes que tienen más de dos pads, por ejemplo si hay terminaciones o resistencias pull-down. También puede llevar a equivocación si hay segmentos de una pista rutados encima de otros por error.

En el caso del USB las redes tienen más de dos pads. Esto es debido a que el conector USB-C es reversible, por lo que las mismas pistas van conectadas a distintos pines del conector que actúan en función de la orientación del cable conectado.

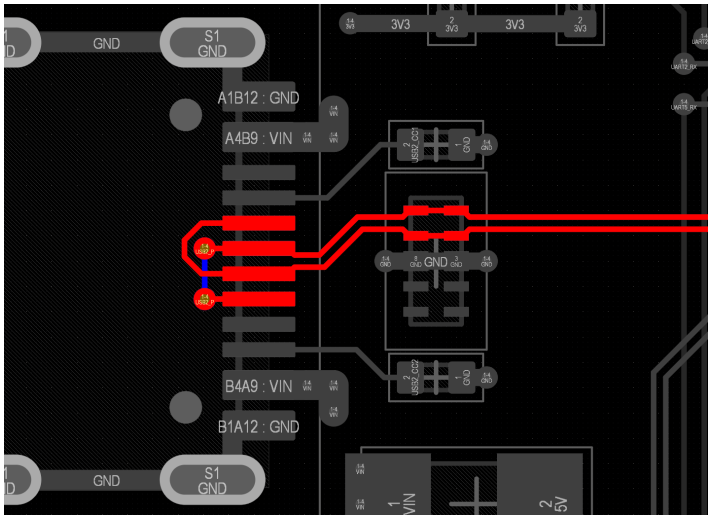


Figura 5.14: Conexión de las pistas USB al conector USB-C.

Es también el caso de algunas redes de la RAM que disponen de resistencias de terminación o pull-down que hacen que haya más de dos pads en la misma red.

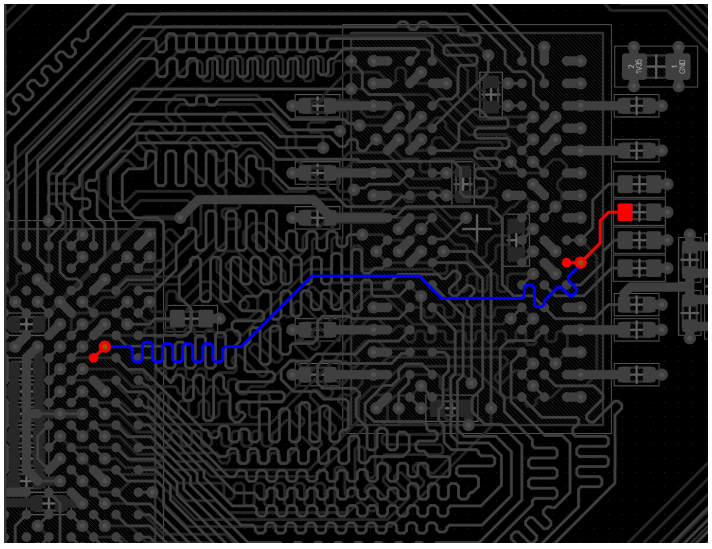


Figura 5.15: Conexión de la pista DRAM_SDCKE0 con resistencia pull-down.

En todos estos casos el ajuste es necesario realizarlo teniendo en cuenta únicamente la longitud entre dos pads determinados. Esto es posible realizarlo con la funcionalidad *xSignals* de Altium. Las xSignals son pistas virtuales que conectan únicamente los pads entre los que se quiere hacer el length matching. Se han creado xSignals para todas las señales del USB y RAM. De esta forma los ajustes se han realizado considerando las longitudes de las xSignals en lugar de la totalidad de la red.

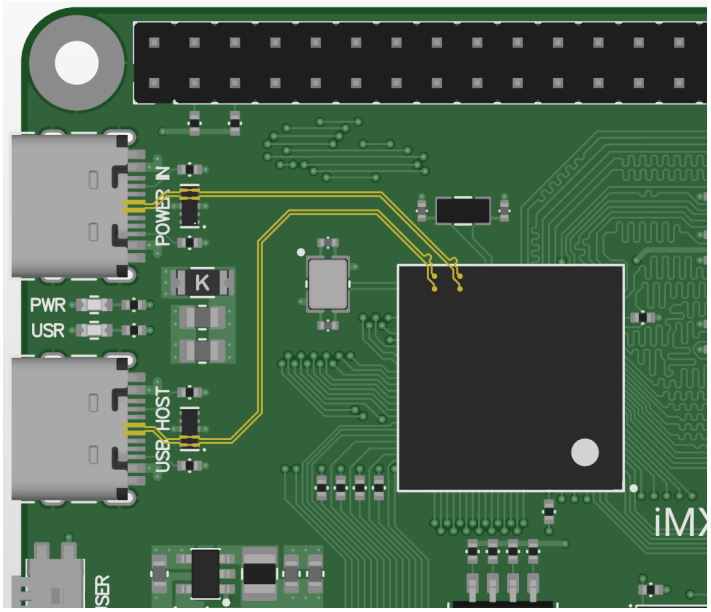


Figura 5.16: xSignals del USB tras rutado y length matching.

En las pistas del bus de la RAM, dado que no ha sido posible que todas las pistas de un mismo grupo pasen por el mismo número de vías, para calcular la longitud total de la pista es necesario incluir la longitud de las vías. Esto lo tiene en cuenta Altium automáticamente al haber introducido los datos relativos al stackup.

Signals	Total length	Recommendations
Address and Bank	Clock length	Match the signals ±25 mils of the value specified in the length column.
Data and Buffer	Clock length	Match the signals ±25 mils of the value specified in the length column.
Control signals	Clock length	Match the signals ±25 mils of the value specified in the length column.
Clock DRAM_SDCLK[1:0]	Longest trace less than or equal to 2 inches	Match the signals of clocks signals ±5 mils. Each differential clock pair.
DRAM_SDQS[1:0] and DRAM_SDQS[1:0]_B	Clock length	Match the signals of the DQS signals ±10 mils of the value specified in the length column.

Figura 5.17: Reglas para el rutado DDR3L con igual longitud [10].

Aunque en la interfaz DDR3L es posible ajustar la longitud de manera independiente para cada grupo de señales, en este caso se ha comprobado que la pista más larga era de 30 mm y se han ajustado el resto de señales a la misma longitud.

En línea con las reglas indicadas por el fabricante, las dos pistas de cada par diferencial se han ajustado con una tolerancia de ±0.127 mm (±5 mil) y el resto de señales se han ajustado con una tolerancia de ±0.635 mm (±25 mil).

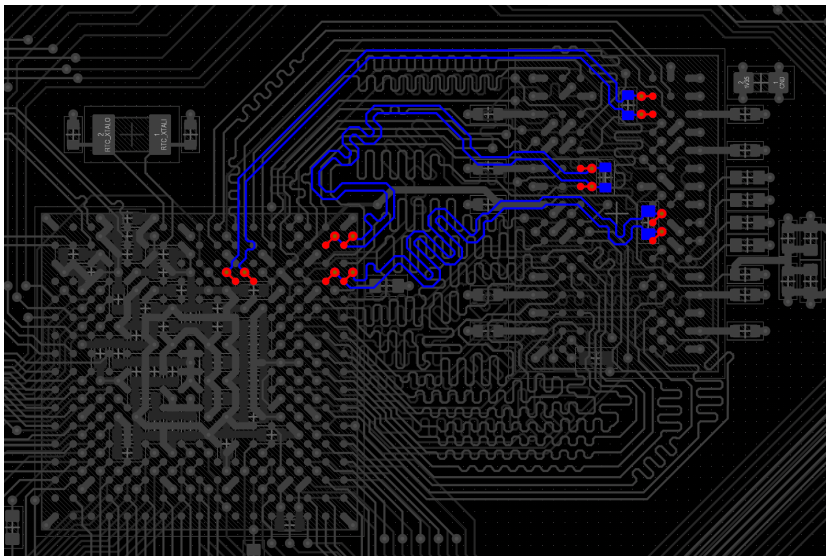


Figura 5.18: Rutado de la interfaz DDR3L. Pares diferenciales.

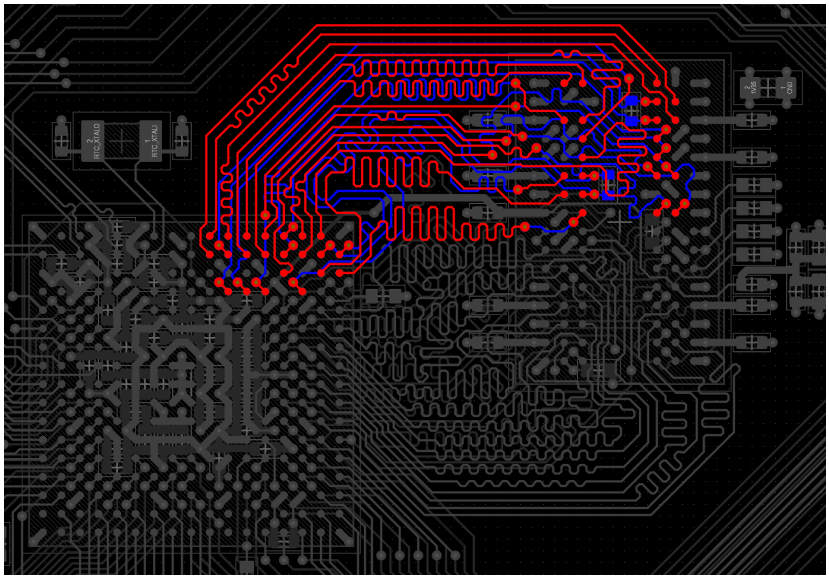


Figura 5.19: Rutado de la interfaz DDR3L. Líneas de datos.

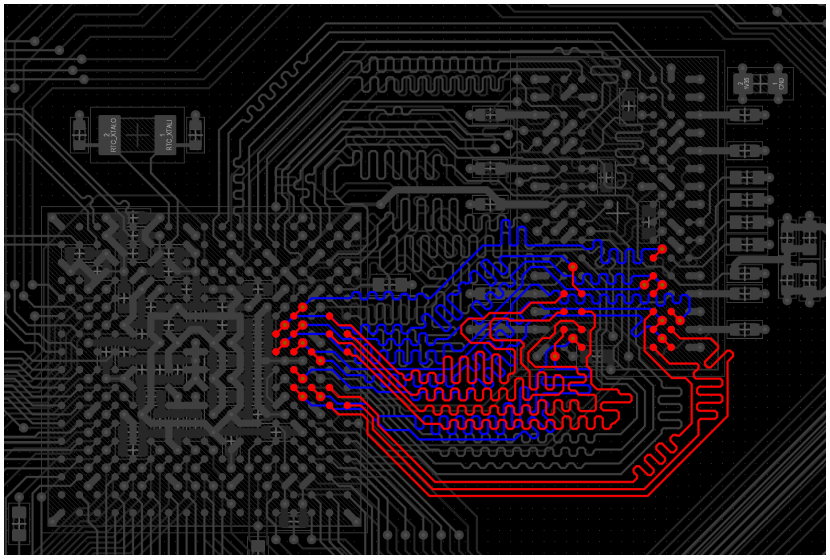


Figura 5.20: Rutado de la interfaz DDR3L. Líneas de dirección.

5.6. Generación de los archivos de salida

Por último, se han generado todos los archivos de salida necesarios para la fabricación del PCB. Siguiendo las recomendaciones del fabricante, se ha creado una capa mecánica (.GM1) en la que se ha copiado el borde exterior de la placa. Se ha creado un fichero *OutJob* de Altium que genera los siguientes archivos de salida:

- Esquemático en formato PDF.
- BOM en formato Excel y PDF.
- Modelo 3D del PCB en formato STEP.
- Gerbers y ficheros de taladros, necesarios para la fabricación del PCB y stencil.
- Capas de ensamblaje en formato PDF con las referencias de los componentes, necesarios para realizar el ensamblaje manualmente.
- Fichero con las posiciones de los componentes, necesario para realizar el ensamblaje mediante máquina *Pick and Place*.

5.6.1. Capas de cobre

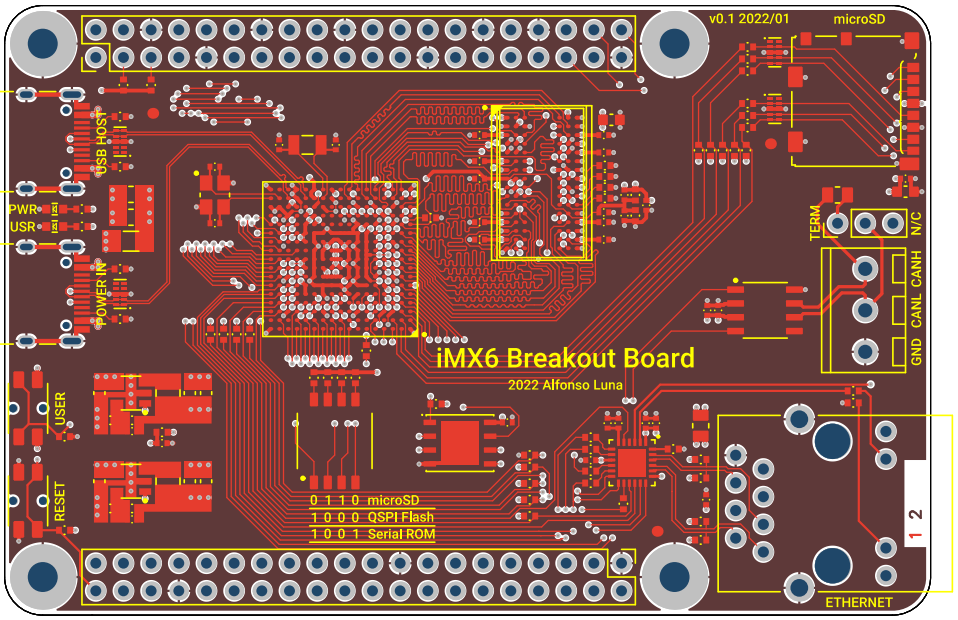


Figura 5.21: Capa de cobre superior y plano de masa.

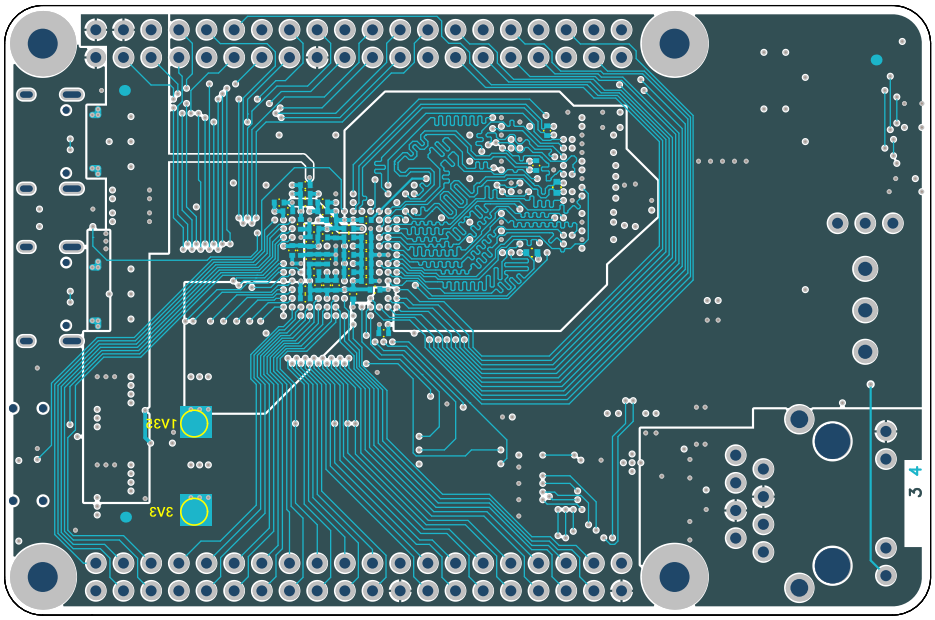


Figura 5.22: Capa de cobre inferior y plano de alimentación.

5.6.2. Capas de ensamblaje

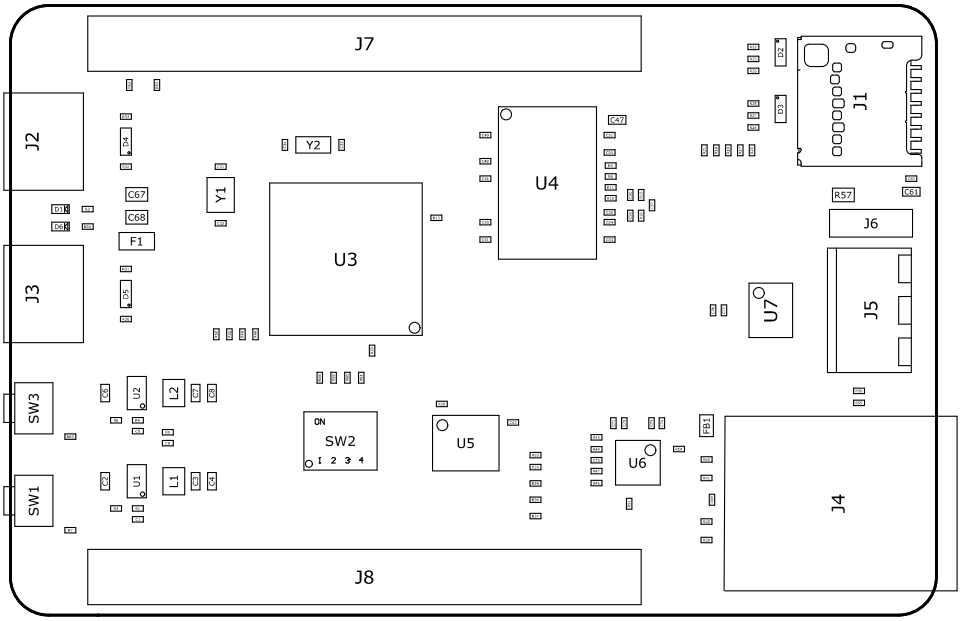


Figura 5.23: Capa de ensamblaje superior.

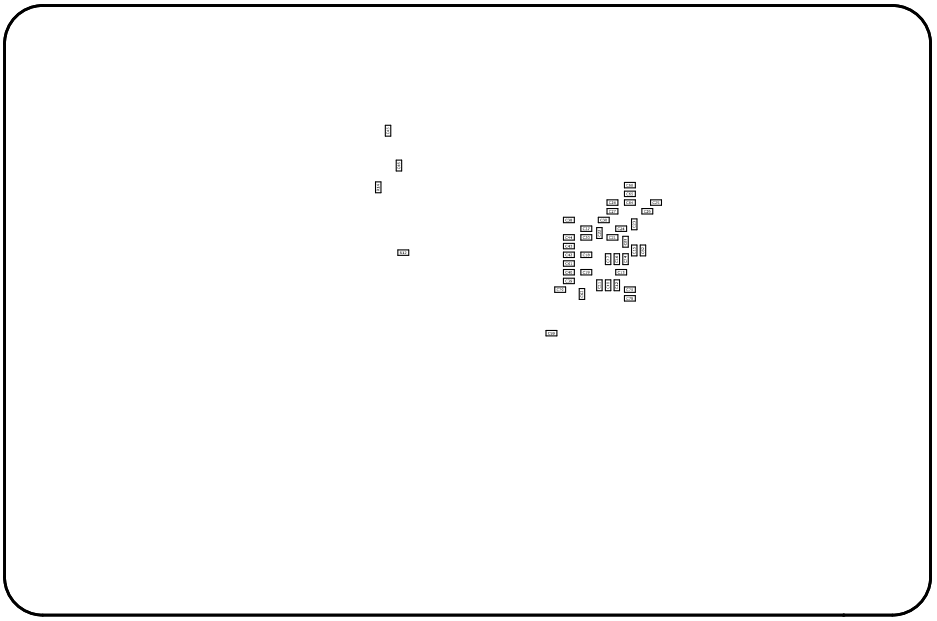


Figura 5.24: Capa de ensamblaje inferior.

Capítulo 6

Estudio de integridad de señal

6.1. Introducción a la integridad de señal

En sistemas digitales, especialmente cuando las señales son de alta velocidad o eléctricamente largas, distintos fenómenos pueden causar efectos indeseados en la señal degradando su forma hasta el punto de que el sistema deje de funcionar correctamente.

El estudio de la integridad de señal consiste en analizar y mitigar estos efectos, reduciendo la degradación de la señal a lo largo del camino que recorre de manera que la señal que llegue al receptor sea lo más cercana posible a la que sale del emisor.

6.1.1. Fenómenos que degradan la señal

Entre los fenómenos más comunes que degradan la señal se encuentran las reflexiones y el *crosstalk*.

6.1.1.1. Reflexiones

Las reflexiones son debidas a cambios en la impedancia de la línea por la que circula la señal. Cuando una señal se encuentra con un cambio de impedancia en su camino, parte de esta señal se refleja hacia atrás de vuelta hacia la fuente.

Cuando la señal reflejada llega a la fuente, se encuentra de nuevo con un cambio de impedancia que hace que parte de esta señal se refleje de vuelta hacia la carga, y así sucesivamente hasta que las reflexiones quedan suficientemente atenuadas.

Estas reflexiones afectan a la forma de onda de las señales, provocando sobreimpulsos y subimpulsos que pueden llegar a cambiar el nivel lógico de la señal, hacer que la señal quede en un nivel intermedio en el que el sistema entre en metaestabilidad, o dañar las protecciones de sobretensión de los circuitos integrados.

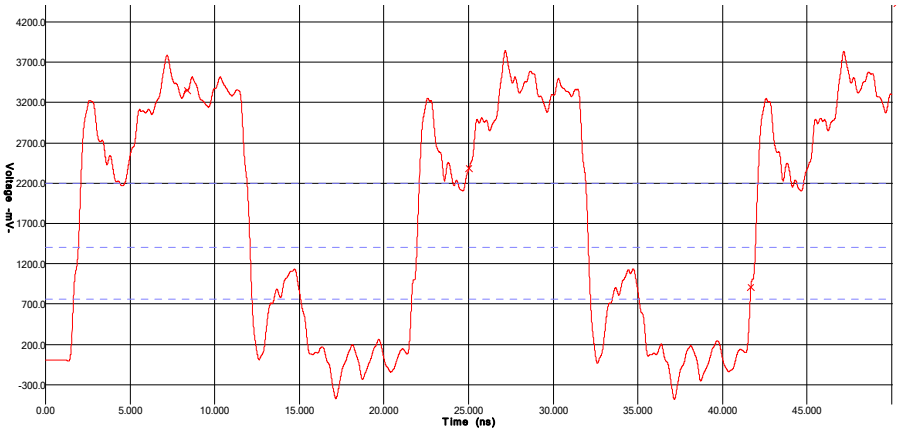


Figura 6.1: Señal de reloj deteriorada debido a las reflexiones [2].

La forma de atenuar las reflexiones consiste en reducir todo lo posible las discontinuidades en la impedancia a lo largo de toda la línea:

- Es importante utilizar un stackup apropiado y pistas con la impedancia controlada, así como dar continuidad a las corrientes de retorno si hay cambios de plano de referencia, tal y como se ha estudiado en el capítulo 5.
- La impedancia característica de la pista debe coincidir lo mejor posible con la impedancia de salida de la fuente y con la impedancia de entrada de la carga.
- Si el driver y/o el receptor son configurables, es posible ajustar sus configuraciones de forma que sus impedancias coincidan lo mejor posible con la de la pista.
- Lo más habitual es que la impedancia de salida del driver no sea configurable y tenga un valor de unos 15-18 Ω. Esto hace que a menudo sea muy efectivo añadir una resistencia de terminación en serie junto al driver con un valor de 22-37 Ω, de forma que la impedancia de salida quede adaptada a los 40-60 Ω de la pista.

6.1.1.2. Crosstalk

El *crosstalk* o diafonía consiste en el acoplamiento de una señal de una pista a otra. Ocurre cuando en uno de los conductores existe una diferencia de potencial variable entre este y un plano de masa, de forma que actúa como fuente de interferencia mientras que otro es el receptor de la interferencia. En sistemas digitales, son los flancos de la línea agresora los que generan un acoplamiento en la víctima.

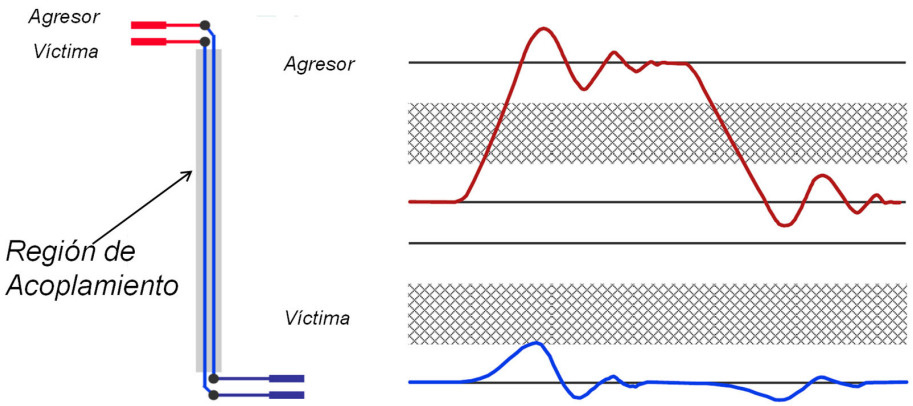


Figura 6.2: Efecto del crosstalk entre dos pistas acopladas [2].

Algunas medidas para reducir el crosstalk son:

- Reducir la separación entre las pistas y su plano de referencia. De esta forma se consigue reducir el área de bucle y por tanto el acoplamiento entre ambas señales.
- Aumentar la separación entre pistas. A partir de una separación de 3S (3 veces la distancia entre la pista y su plano de referencia) se reduce en gran medida.
- Relajar los tiempos de subida y de bajada de los flancos, de forma que la señal varíe tan lentamente como sea posible. Las componentes de alta frecuencia son más susceptibles de acoplarse a otras pistas y de empeorar los resultados de EMC.
- Evitar tener varias pistas que discurran de forma paralela durante largas distancias. Utilizar planos de masa entre capas de señal. Cuidar especialmente las señales más peligrosas como los relojes y las más sensibles como las de reset.

6.1.2. Objetivo del estudio

Cualquier señal tendrá siempre cierta degradación debido a los fenómenos comentados anteriormente, entre otros. No es posible eliminar estos fenómenos completamente. El objetivo será asegurarse de que los efectos son suficientemente pequeños como para que no afecten al correcto funcionamiento del circuito.

Para comprobarlo se realizarán simulaciones de integridad de señal. A diferencia de las simulaciones clásicas con SPICE, en este tipo de simulaciones las conexiones entre componentes no se consideran ideales, sino que se modela a nivel electromagnético el comportamiento de la línea de transmisión, así como el acoplamiento entre distintas líneas, obteniendo el efecto de estos fenómenos en la señal.

La forma de onda de la señal a menudo se analiza utilizando los denominados diagramas de ojos. Estos diagramas consisten en la superposición de varias partes de la señal en un rango de tiempo o bits determinado y alineadas con los flancos de reloj.

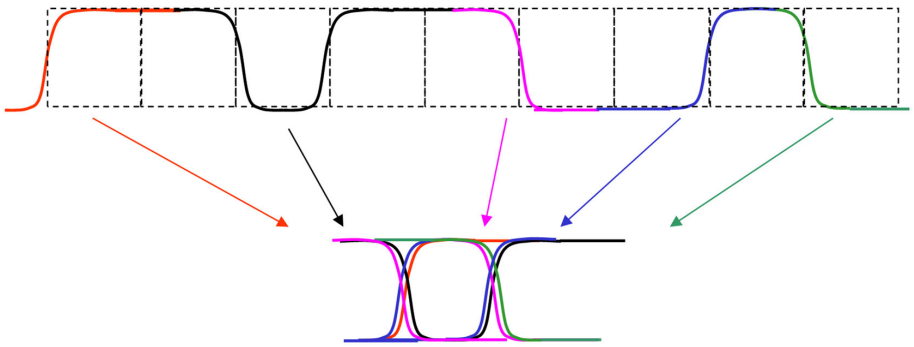


Figura 6.3: Diagrama de ojo formado a partir de varias partes de una señal [2].

Para comprobar si la integridad de una señal es suficientemente buena, será necesario comprobar si se respetan las especificaciones que impone el estándar de la interfaz utilizada así como los límites de los circuitos integrados involucrados:

- Los valores de tensión máxima y mínima de la señal deben estar dentro de unos límites para evitar dañar las protecciones de los circuitos integrados.
- Se deben respetar las tensiones que representan los niveles lógicos de la señal, evitando que la señal quede en un nivel intermedio en el momento de muestrearla.
- Los flancos deben ser monótonamente crecientes o decrecientes, es decir, deben evitarse los máximos o mínimos durante las transiciones entre niveles lógicos.

En las interfaces de alta velocidad es habitual que el estándar especifique lo que se conoce como máscaras de ojos. Estas máscaras definen áreas prohibidas en las que la señal del diagrama de ojo no debe entrar para garantizar el buen funcionamiento.

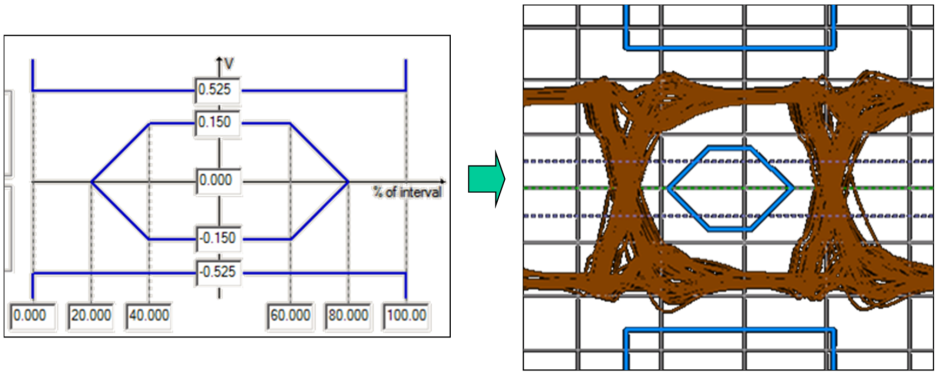


Figura 6.4: Máscara de ojo para USB 2.0 High-Speed [2].

También cabe destacar que a menudo mejorar la integridad de señal tiene un efecto positivo a la hora de reducir la radiación en el PCB. Es decir, aunque la calidad de la señal sea suficientemente buena para que el circuito funcione adecuadamente, mejorarla aún más podría ser de ayuda para mejorar los resultados en EMC.

6.1.3. Modelos IBIS

Los modelos IBIS (*Input/Output Buffer Information Specification*) son una forma de que los fabricantes de circuitos integrados proporcionen información sobre los buffers de entrada y salida de sus dispositivos sin necesidad de revelar información propietaria sobre cómo están construidos internamente.

A diferencia de los modelos SPICE, los modelos IBIS no están formados por componentes que representan la estructura interna del dispositivo sino que solamente proporcionan información sobre los parásitos del encapsulado y las tablas de corriente-tensión y tensión-tiempo para modelar la conmutación de cada pin.

Esto también tiene la ventaja de que estos modelos pueden crearse para componentes complejos como microprocesadores o memorias, ya que solo modelan la conmutación de los pines de entrada y salida, sin necesidad de simular la funcionalidad del microprocesador en su totalidad. Será necesario por tanto indicarle al simulador la secuencia a aplicar en cada pin de salida para simular el comportamiento de la interfaz.

6.1.4. Programas de simulación

En este trabajo se utilizará HyperLynx para realizar las simulaciones de integridad de señal, aunque en el mercado también hay disponibles otras alternativas tales como Cadence Sigrity, Ansys SIwave o Keysight ADS. Todos ellos permiten importar el PCB desde Altium y disponen de herramientas para simular interfaces DDRx como la utilizada en este trabajo, aunque a un precio bastante elevado.

Una alternativa más económica es utilizar las simulaciones de integridad de señal que han sido incluidas recientemente en Altium, aunque tienen algunas limitaciones. Es suficiente para realizar las simulaciones más sencillas, pero no es posible representar diagramas de ojos o simular interfaces específicas como DDRx.

Por otro lado, también es posible realizar simulaciones de integridad de señal con programas gratuitos basados en SPICE como LTspice o Micro-Cap [21]. Sin embargo, estos programas tienen aún más limitaciones, ya que las simulaciones no se realizan a nivel electromagnético sino a nivel eléctrico. Además, es necesario convertir los modelos IBIS a modelos SPICE equivalentes e introducir manualmente todos los parámetros de la pista, ya que no es posible importar el stackup o el PCB layout.

6.1.5. Limitaciones del estudio

Para asegurarse de que la calidad de la señal sea la adecuada sería necesario tomar medidas en el PCB una vez fabricado. La simulación sirve para tener una primera aproximación pero será necesario dejar cierto margen para poder asumir posibles errores en la simulación así como pequeñas variaciones durante la fabricación. Si el sistema no funciona en la simulación probablemente tampoco lo hará en la realidad, pero que funcione en la simulación no garantiza que funcione en la práctica.

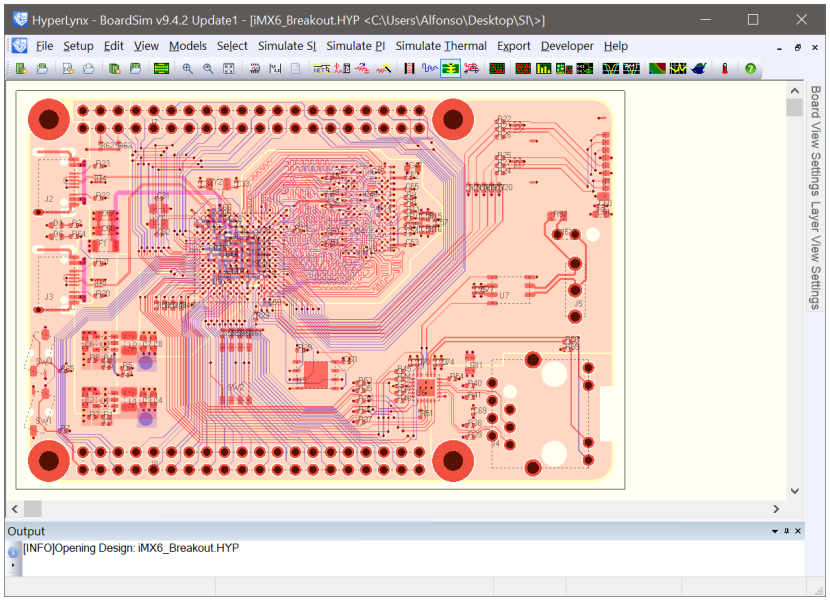
Otro aspecto a tener en cuenta es que el simulador utilizado considera que todos los caminos de retorno son ideales. Con el stackup de 6 capas en el que todas las pistas usan un plano de masa como referencia, si se usan vías para unir ambos planos en los cambios de capa el resultado de la simulación sería muy cercano al de la realidad.

En cambio, con el stackup de 4 capas se tendrá una degradación adicional que no será posible comprobar en simulación. En las últimas versiones de HyperLynx se puede realizar una simulación *power-aware* que permite asignar modelos a los condensadores y simular las discontinuidades de los caminos de retorno. No se ha realizado este tipo de simulación al no tener acceso a una licencia de esta nueva versión.

6.2. Configuración de HyperLynx

6.2.1. Creación del proyecto

Es posible exportar el PCB en formato HyperLynx directamente desde Altium haciendo click en *File* → *Export* → *HyperLynx*. Esto generará un archivo .HYP que contiene toda la información relativa al PCB incluyendo el layout y el stackup.



6.2.2. Descarga de los modelos IBIS

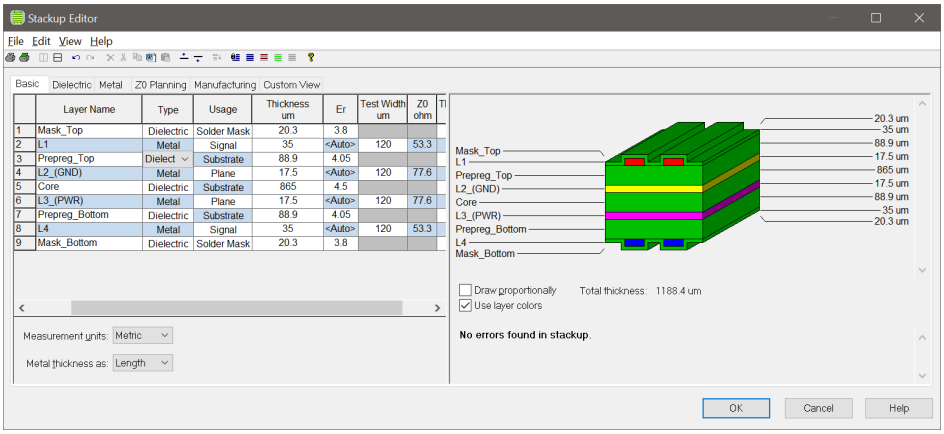
Los siguientes circuitos integrados utilizados en el trabajo disponen de modelos IBIS, disponibles para descargar en la página web de cada fabricante:

- CPU NXP i.MX6ULL MCIMX6Y2DVM09AB
- Memoria RAM MT41K256M16TW-107:P
- Memoria Flash MX25L12835FZNI
- Transceiver Ethernet LAN8720A
- Transceiver CAN SN65HVD230

Una vez descargados los modelos, es necesario indicarle a HyperLynx dónde se encuentra la carpeta con los modelos en *Setup* → *Options* → *Directories* → *Model-library file path(s)*.

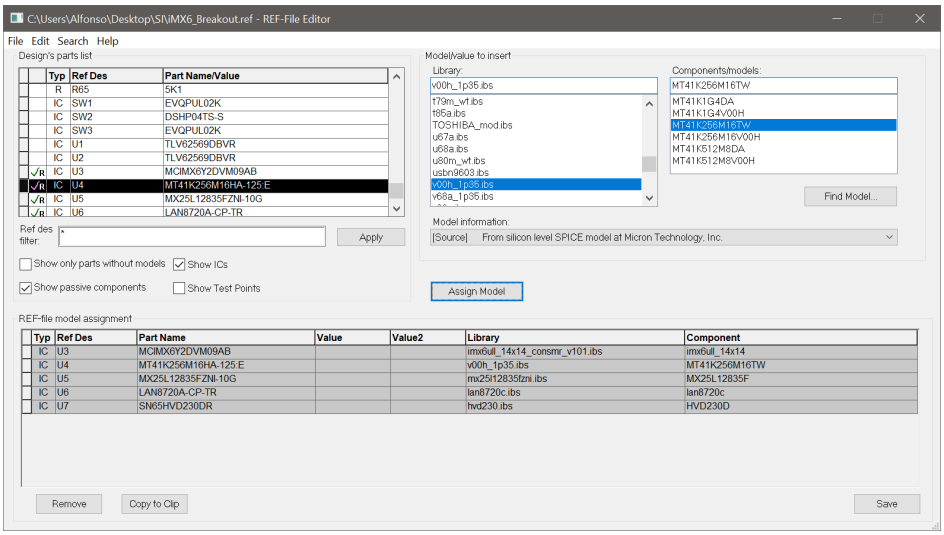
6.2.3. Configuración del stackup

A pesar de que el archivo .HYP exportado desde Altium ya incluye la información relativa al stackup, ha sido necesario terminar de configurarlo correctamente desde *Setup* → *Stackup* → *Edit*. Se ha especificado si las capas eran de señal o de plano. También se han añadido manualmente las capas de máscara de soldadura. Tras estos cambios los valores de impedancia que calcula HyperLynx son muy similares a los calculados por Altium.



6.2.4. Asignación de los modelos

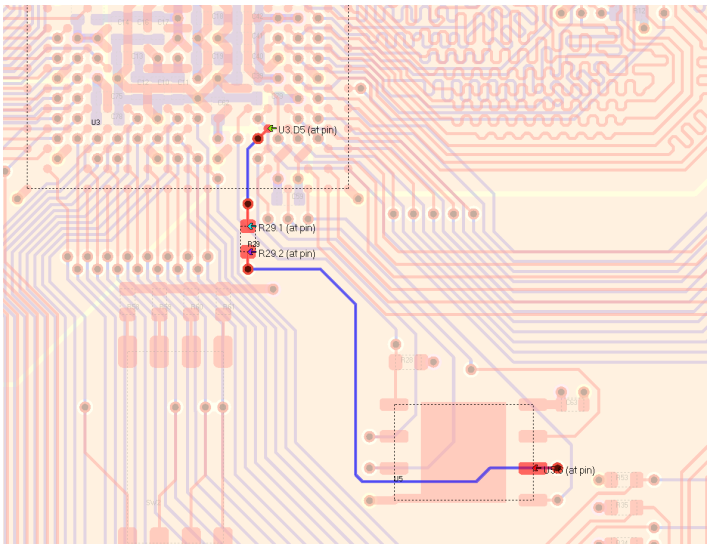
El siguiente paso consiste en asignar a cada componente su modelo IBIS correspondiente desde *Models* → *Assign Models/Values by Reference Designator*.



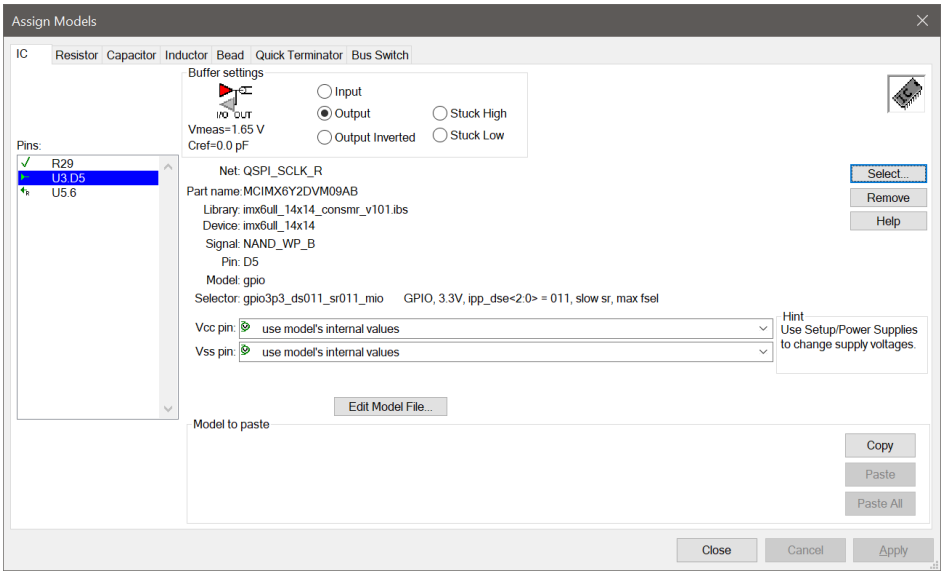
6.3. Simulación de las señales de reloj

6.3.1. Simulación de la memoria Flash

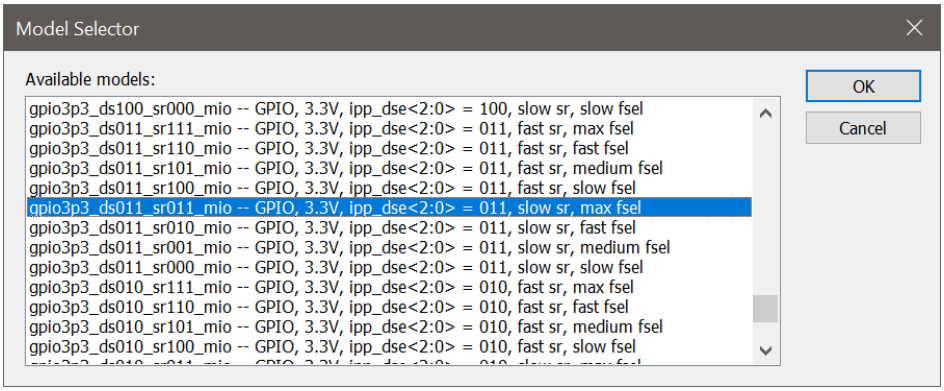
En primer lugar se selecciona desde *Select* → *Net by Name for SI Analysis* la señal a simular, en este caso la formada por las pistas *QSPI_SCLK* y *QSPI_SCLK_R*. Seleccionando cualquiera de ellas, HyperLynx reconoce automáticamente ambas pistas como un único camino en el que en medio se encuentra la resistencia de terminación:



Desde *Models* → *Assign Models/Values by Net* es necesario indicar para cada pin si es de entrada o de salida, así como seleccionar el pin correspondiente del modelo IBIS y la configuración del pin en caso de que hayan varias entre las que elegir:

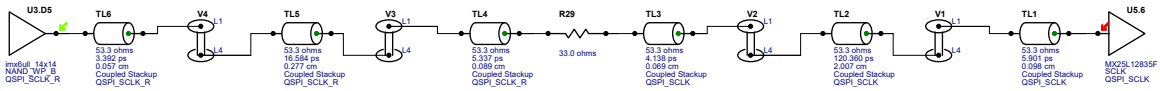


En el lado del SoC es posible elegir entre distintos modelos para cada pin, en función de la tensión de alimentación y de la configuración del pin. Tal y como se analizó en el apartado 5.1.1.4, la frecuencia máxima de reloj de esta señal es de 104 MHz por lo que se seleccionará la configuración *Slew Rate = Slow* y *Drive = Max*, ya que es la configuración más lenta que permite el funcionamiento a dicha frecuencia:



En el lado de la memoria Flash hay un único modelo, por lo que no se puede configurar. En cuanto a la resistencia de terminación, es posible asignarle cualquier valor de forma numérica, lo que permitirá comprobar fácilmente cómo varía el resultado de la simulación para distintos valores de la resistencia.

También es posible visualizar la interpretación que hace HyperLynx de la pista desde *Export* → *Net To* → *Free-Form Schematic*:



Por último, se puede ejecutar y visualizar la simulación haciendo click en *Simulate SI* → *Run Interactive Simulation*. Como estímulo se ha configurado un oscilador a 104MHz con 50 % de Duty Cycle.

A continuación, se muestran los resultados obtenidos para el caso *IC Modeling = Typical*, *Slew Rate = Slow* y *Drive = Max*. En azul se representa la señal medida a la salida del SoC, mientras que en rojo se representa la señal medida a la entrada de la memoria Flash. La simulación se ha realizado para R = 0Ω y R = 33Ω.

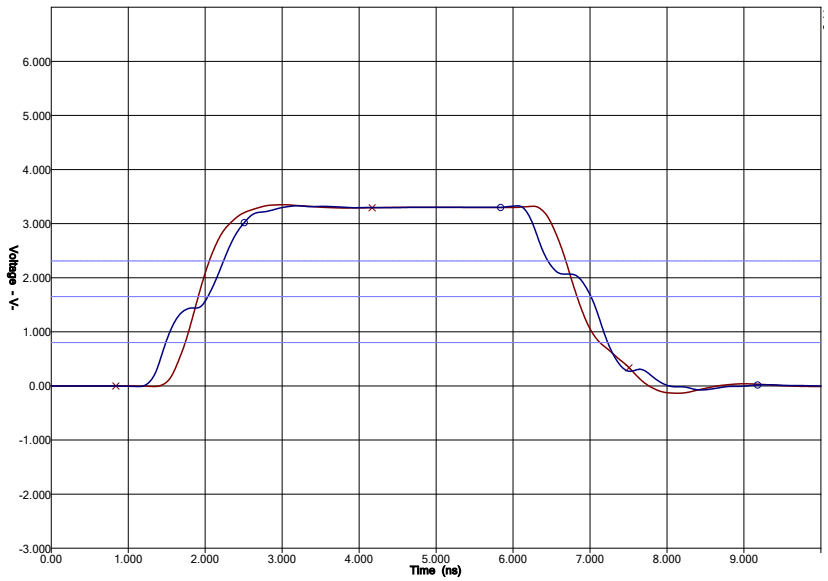


Figura 6.5: Señal de reloj de la memoria Flash para el caso típico. R = 0Ω.

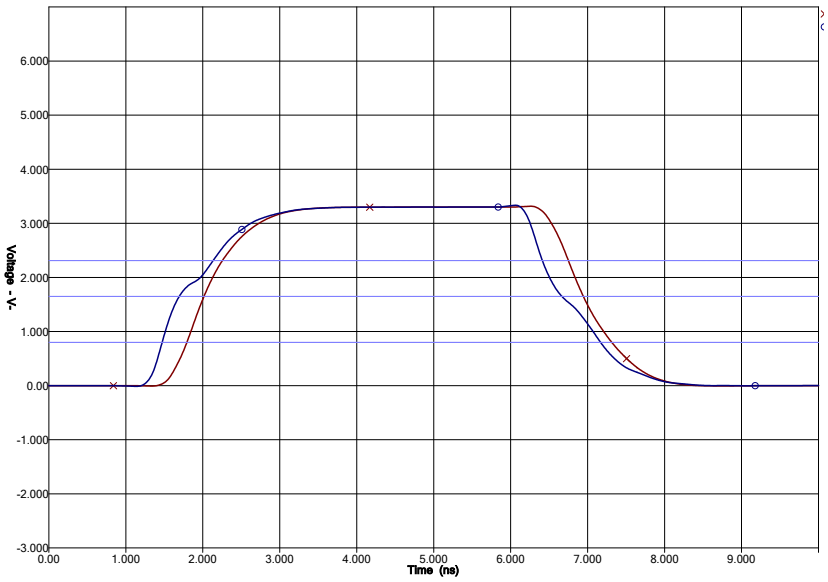


Figura 6.6: Señal de reloj de la memoria Flash para el caso típico. R = 33Ω.

Se observa que en ambas gráficas la señal de reloj que llega a la Flash es muy limpia. No hay una gran mejora al añadir la resistencia de 33Ω ya que la pista está muy por debajo de la longitud crítica por lo que apenas se ve afectada por las reflexiones.

Este resultado se puede extrapolar al resto de señales de la interfaz que también funcionarán correctamente cuando el SoC sea el emisor y la Flash el receptor, aunque no haya resistencia de terminación en dichas líneas.

También se ha realizado la simulación para el caso más rápido con la configuración *IC Modeling = Fast-Strong*, *Slew Rate = Fast* y *Drive = Max*. En este caso el tiempo de subida es de alrededor de 1ns y la longitud crítica de unos 3cm, muy cercana a la longitud de la pista. De nuevo se ha realizado para R = 0 y R = 33Ω.

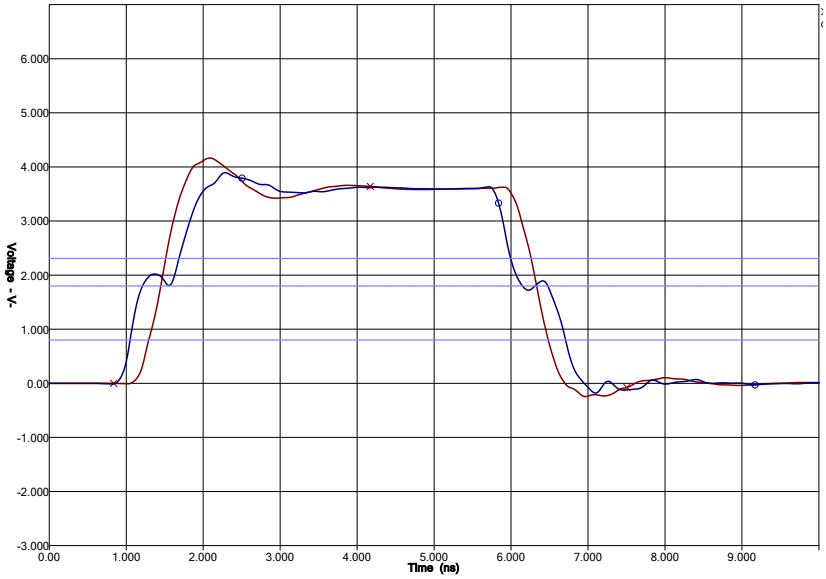


Figura 6.7: Señal de reloj de la memoria Flash para el caso más rápido. R = 0Ω.

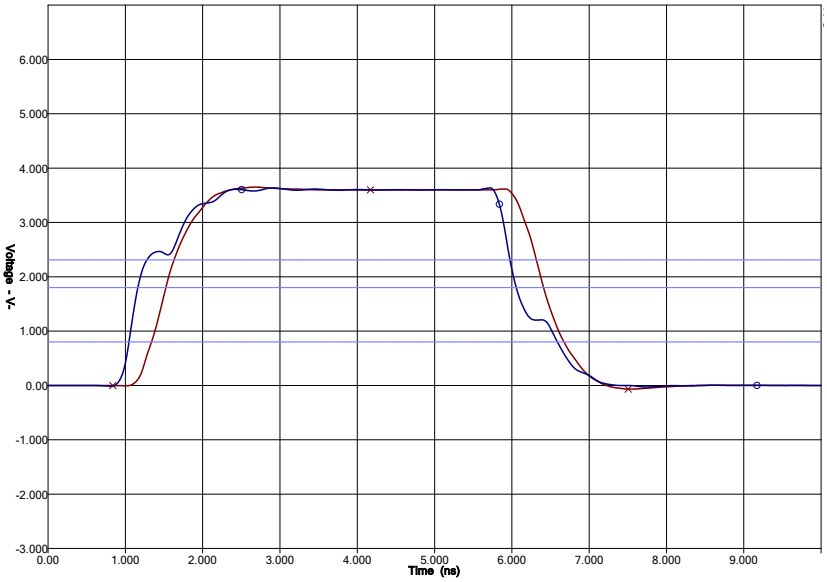
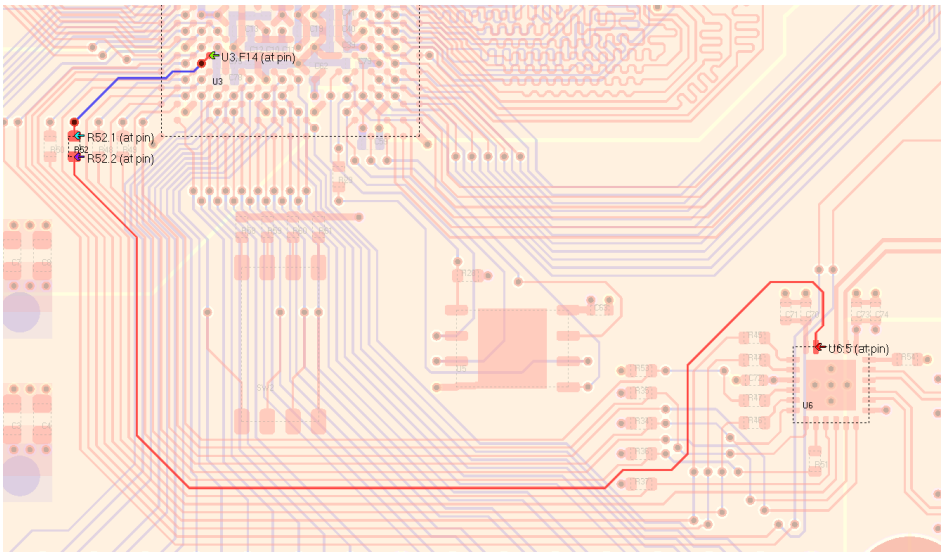


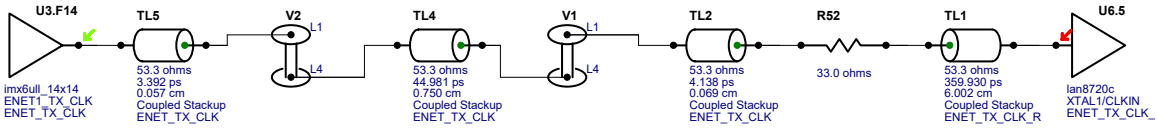
Figura 6.8: Señal de reloj de la memoria Flash para el caso más rápido. R = 33Ω.

En este caso se observa que la señal se ve más afectada por las reflexiones con un sobrepico en el flanco de subida de 557mV que provoca que la señal supere los 4V. Al añadir la resistencia de 33Ω las reflexiones se reducen considerablemente.

6.3.2. Simulación del transceiver Ethernet

En esta sección se ha realizado la simulación de la señal *ENET_TX_CLK*.





Como estímulo se ha configurado un oscilador a 50MHz con 50 % de Duty Cycle. A continuación, se muestran los resultados obtenidos para el caso *IC Modeling* = *Typical*, *Slew Rate* = *Slow* y *Drive* = *Max*. En azul se representa la señal medida a la salida del SoC, mientras que en rojo se representa la señal medida a la entrada del transceiver Ethernet. La simulación se ha realizado para R = 0 y R = 33Ω.

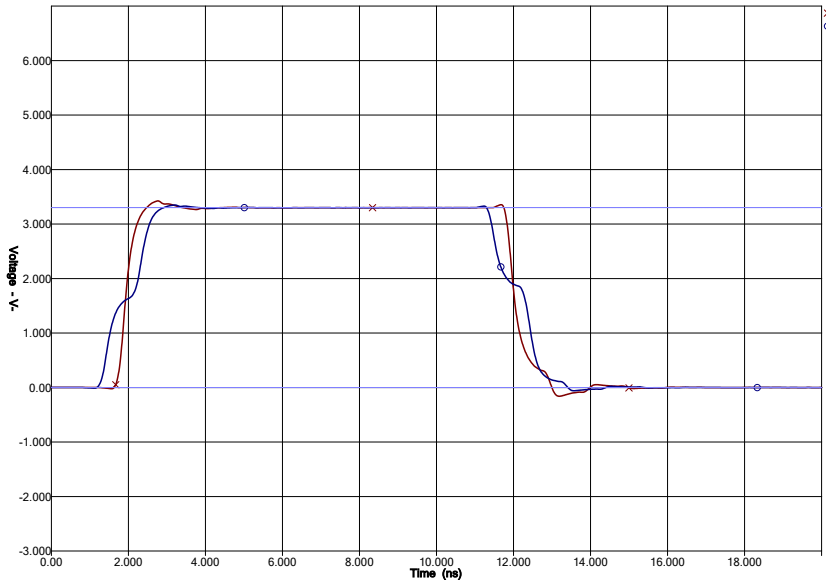


Figura 6.9: Señal de reloj del Ethernet para el caso típico. R = 0Ω.

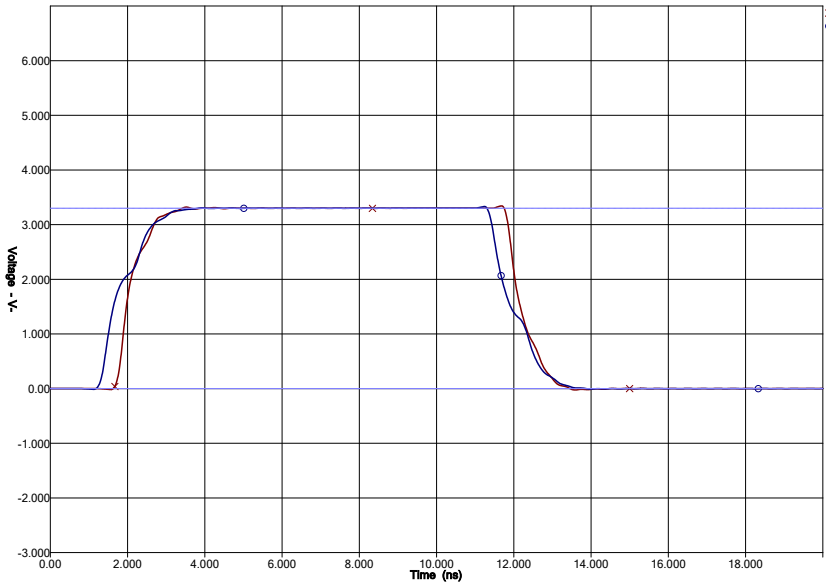


Figura 6.10: Señal de reloj del Ethernet para el caso típico. R = 33Ω.

Se observa que en ambas gráficas la señal de reloj que llega al transceiver Ethernet es aceptable. El tiempo de subida y bajada de los flancos es menor que 2ns por lo que la longitud crítica es menor que 6cm. Las pistas del Ethernet son ligeramente más largas lo que hace que se vean afectadas levemente por las reflexiones. Al añadir la resistencia de 33Ω se consigue disminuir los pequeños sobrepicos de la señal.

Este resultado se puede extrapolar al resto de señales de la interfaz que también funcionarán correctamente cuando el SoC sea el emisor y el transceiver Ethernet el receptor, ya que se ha colocado una resistencia de 33Ω en todas las líneas.

También se ha realizado la simulación para el caso más rápido con la configuración *IC Modeling* = *Fast-Strong*, *Slew Rate* = *Fast* y *Drive* = *Max*.

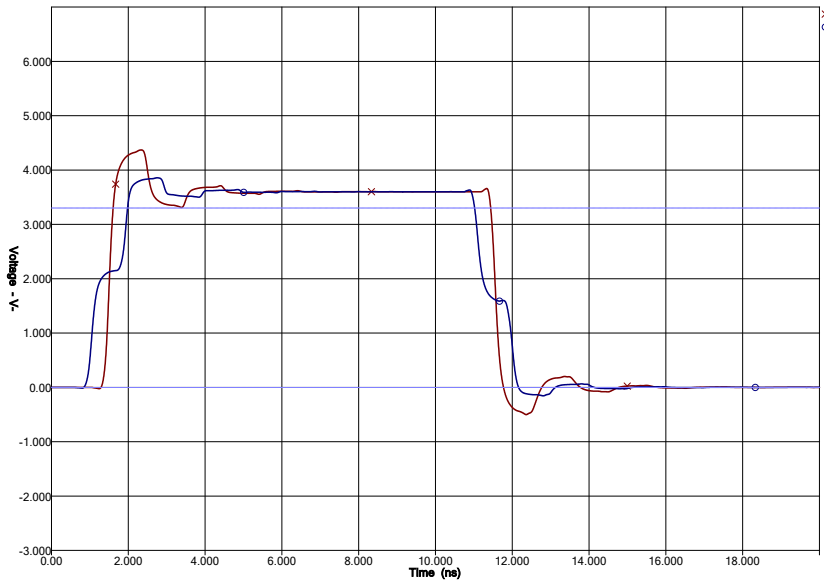


Figura 6.11: Señal de reloj del Ethernet para el caso más rápido. R = 0Ω.

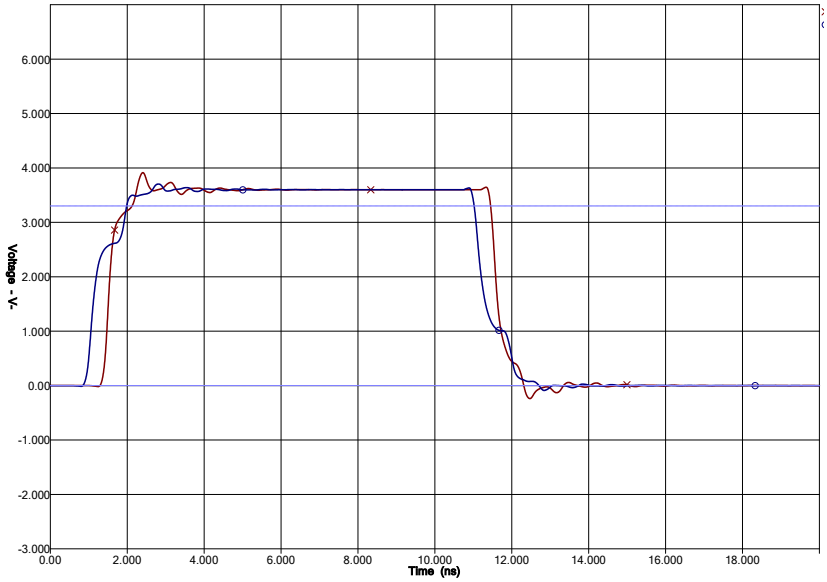


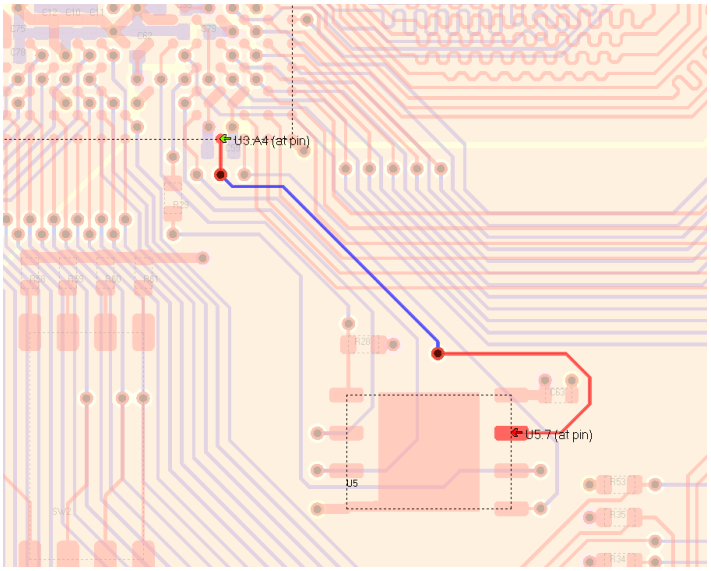
Figura 6.12: Señal de reloj del Ethernet para el caso más rápido. R = 33Ω.

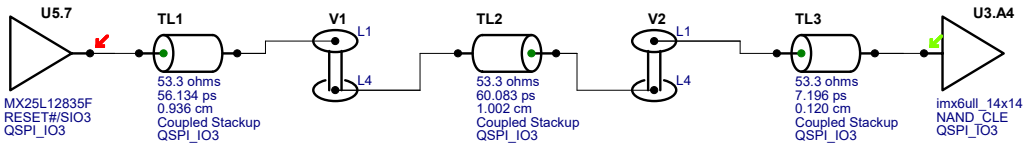
En este caso la señal se ve bastante más afectada por las reflexiones que en el caso de la Flash, debido a la mayor longitud de la pista en relación a la longitud crítica. Con la resistencia de 33Ω se consigue atenuar el sobrepico de 767mV a 311mV, a pesar de que esta se encuentra bastante alejada de la fuente debido a la falta de espacio.

6.4. Simulación de las señales de datos

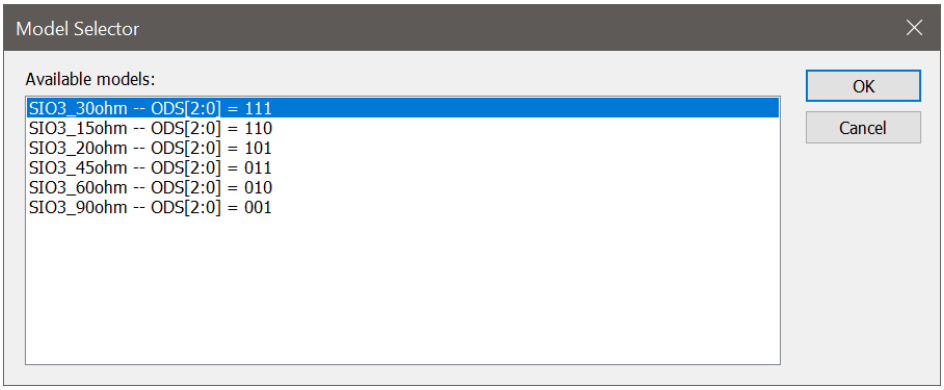
6.4.1. Simulación de la memoria Flash

En esta sección se ha realizado la simulación de la señal *QSPI_IO3* cuando la memoria Flash es el emisor y el SoC es el receptor.

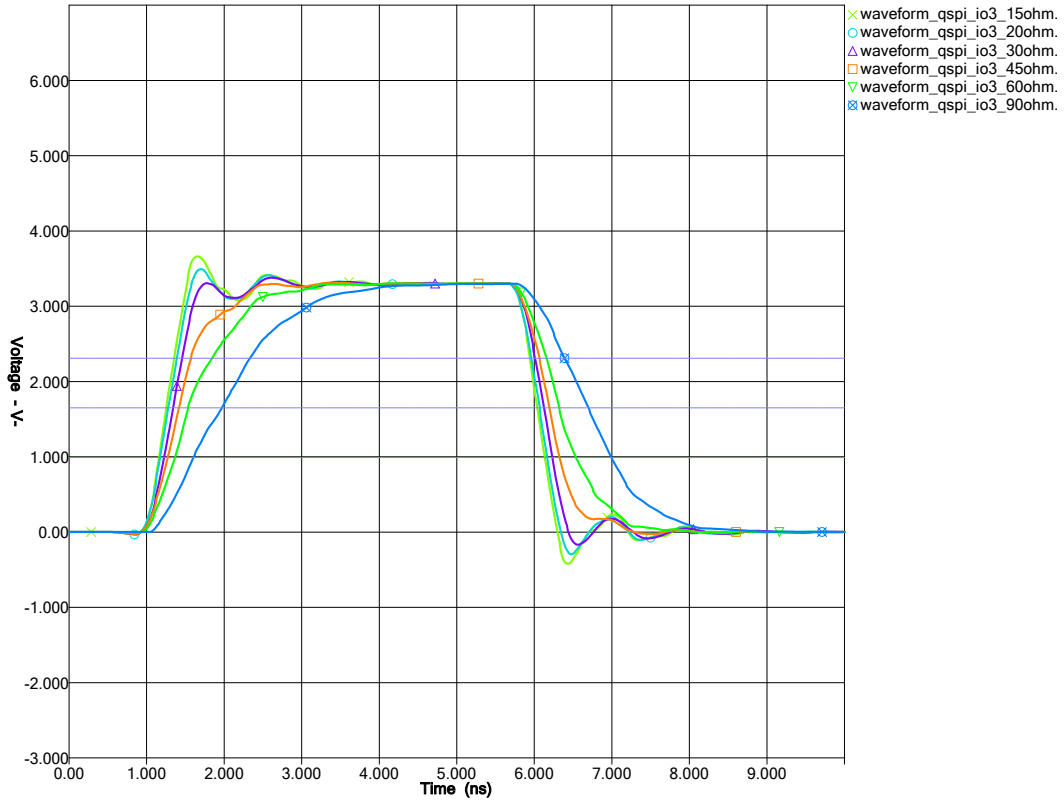




La memoria Flash permite configurar la impedancia de salida del driver entre 15Ω y 90Ω modificando un registro de configuración:



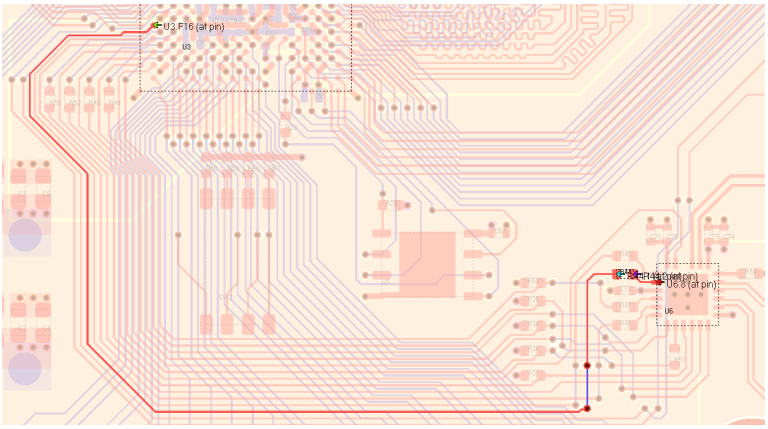
A continuación, se representa la señal medida en el receptor con *IC Modeling = Typical* y para cada uno de los valores de impedancia de salida configurables:

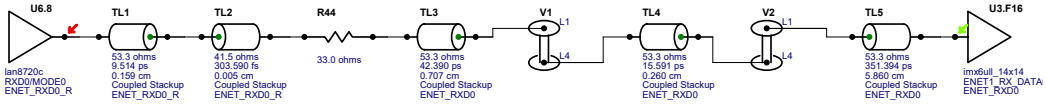


Se observa que con el valor por defecto de 30Ω se consigue un resultado aceptable, aunque el mejor resultado se consigue con 45Ω o 60Ω ya que de esta forma la impedancia de salida del driver queda mejor adaptada a la impedancia de la pista. Con 90Ω los flancos son bastante más lentos aunque siguen estando dentro de la especificación para 104MHz por lo que podría ser de utilidad para reducir la radiación.

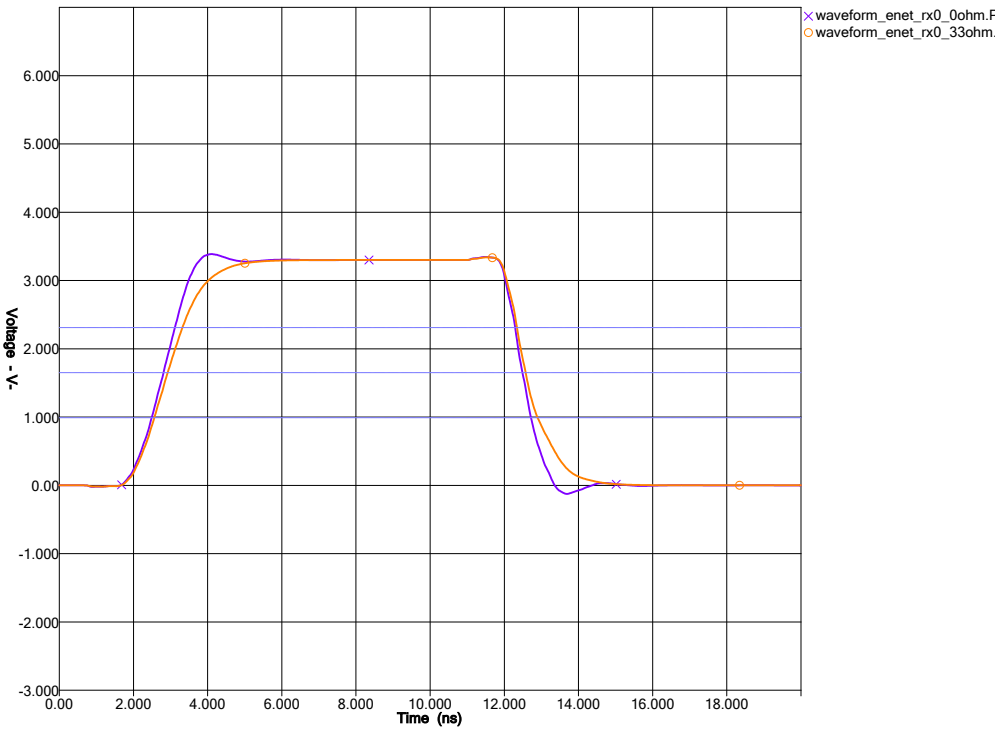
6.4.2. Simulación del transceiver Ethernet

En esta sección se ha realizado la simulación de la señal *ENET_RXD0* cuando el transceiver Ethernet es el emisor y el SoC es el receptor.





El transceiver no permite configurar la impedancia de salida por lo que en todas las líneas se ha colocado una resistencia de 33Ω. A continuación, se representa la señal medida en el receptor con *IC Modeling = Typical* con y sin resistencia:



6.5. Simulación de la memoria DDR3L

6.5.1. Obtención de la máscara de ojo

El estándar de la memoria DDR3L no especifica una máscara de ojo de forma explícita, aunque se puede construir a partir de los límites para los niveles lógicos de la señal y los tiempos de *setup* y *hold*.

6.5.1.1. Niveles lógicos de la señal

Se aplicarán las especificaciones AC160 y DC90 que son las más restrictivas para DDR3L-800. Estos valores fijan los límites para los niveles lógicos de la señal. Están referidos a $V_{REF} = 0.675$V y aplican a todas las señales, tanto a las de dirección y control como a las de datos:

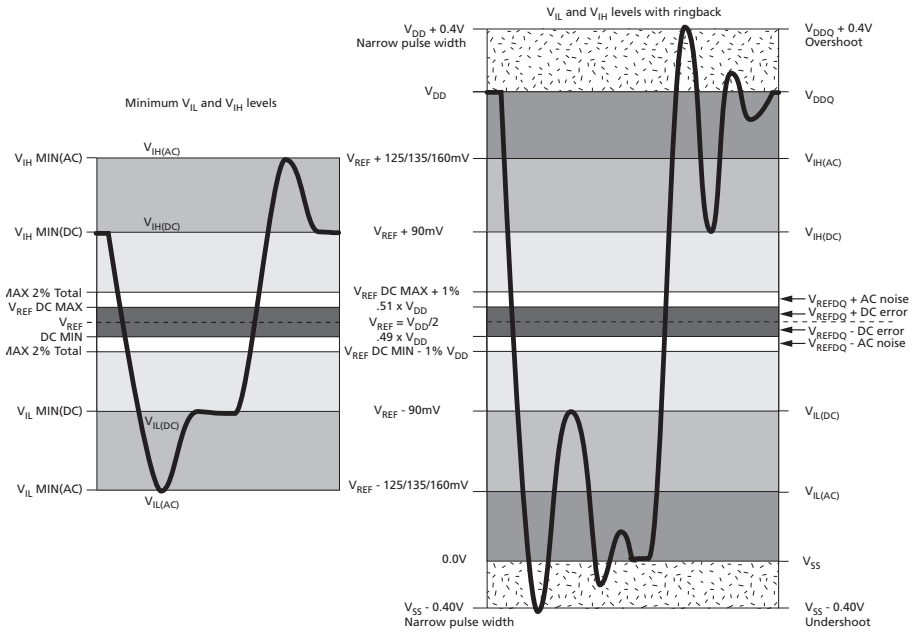


Figura 6.13: Definición de los niveles lógicos de la señal [22].

- $V_{IH(AC160)} = 160$mV
- $V_{IL(AC160)} = -160$mV
- $V_{IH(DC90)} = 90$mV
- $V_{IL(DC90)} = -90$mV

Como HyperLynx no permite especificar distintos niveles para el lado izquierdo y derecho de la máscara, se usarán los de 160mV y -160mV ya que son los más restrictivos. Además, el datasheet especifica los valores máximos de *overshoot* y *undershoot*, aunque estos valores de pico solo se pueden mantener durante un tiempo limitado, no durante todo el periodo de la señal:

- $V_{max} = V_{DD}$ + 0.4V = 1.75V
- $V_{min} = V_{SS}$ - 0.4V = -0.4V

6.5.1.2. Tiempos de setup y hold

El tiempo de setup es el tiempo mínimo que la señal debe permanecer estable antes de un flanco de reloj, mientras que el tiempo de hold es el tiempo mínimo que la señal debe permanecer estable tras el flanco. Los tiempos de setup y hold se especifican en el datasheet en función de la especificación seleccionada, en este caso AC160 y DC90.

Además es necesario sumar al tiempo base un término de *derating* en función del *slew rate* de la señal. Se ha comprobado que el slew rate es mayor que 2V/ns por lo que los tiempos se han calculado de la siguiente forma:

Para señales de dirección y control:

- Setup time: $t_{IS} = t_{IS(base,AC160)} + \Delta t_{IS} = 215 + 80 = 295$ps
- Hold time: $t_{IH} = t_{IH(base,DC90)} + \Delta t_{IH} = 285 + 45 = 330$ps

Para señales de datos:

- Setup time: $t_{DS} = t_{DS(base,AC160)} + \Delta t_{DS} = 90 + 80 = 170$ps
- Hold time: $t_{DH} = t_{DH(base,DC90)} + \Delta t_{DH} = 160 + 45 = 205$ps

6.5.1.3. Configuración de la máscara en HyperLynx

En DDR3L-800 la frecuencia de reloj es de 400MHz. Las líneas de dirección y control se muestrean en los flancos de subida del reloj, por lo que el intervalo de un bit es de 1/400MHz = 2500ps. Las líneas de datos se muestrean en los flancos tanto de subida como de bajada del reloj, por lo que el intervalo de un bit es de 1250ps.

De aquí se obtiene que para señales de dirección y control, la anchura de la máscara estará entre los siguientes porcentajes del periodo:

$$\%_{min} = \frac{(\frac{2500}{2} - 295) \cdot 100}{2500} = 38{,}2\,\%$$

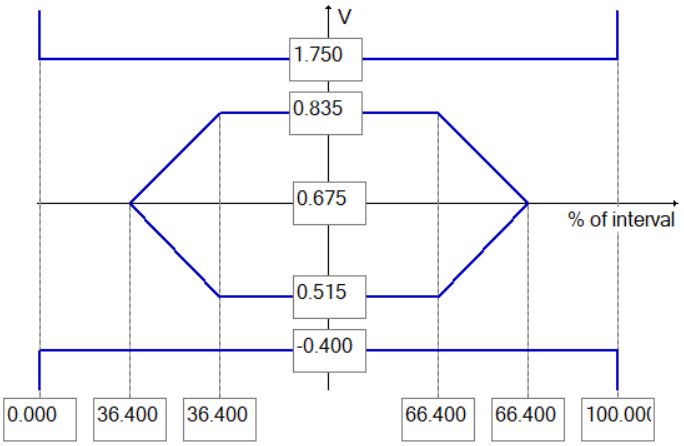
$$\%_{max} = \frac{(\frac{2500}{2} + 330) \cdot 100}{2500} = 63{,}2\,\%$$

Y para las señales de datos, la anchura de la máscara estará entre los siguientes porcentajes del periodo:

$$\%_{min} = \frac{(\frac{1250}{2} - 170) \cdot 100}{1250} = 36{,}4\,\%$$

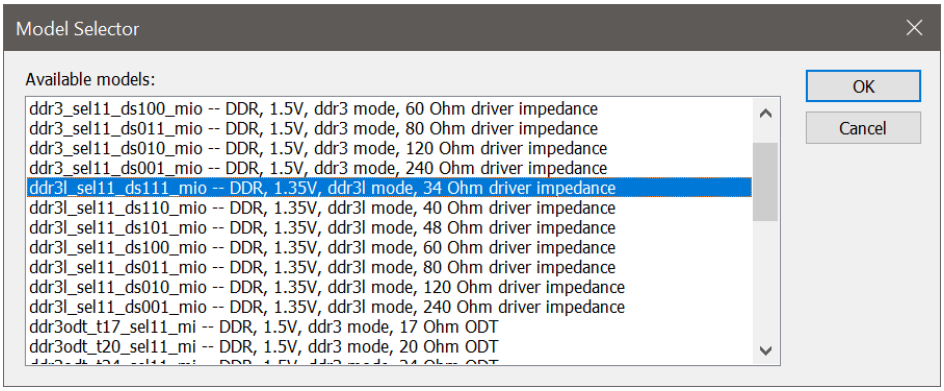
$$\%_{max} = \frac{(\frac{1250}{2} + 205) \cdot 100}{1250} = 66{,}4\,\%$$

Como los valores son muy similares, por sencillez se utilizará la misma máscara en todas las simulaciones, quedándose con los valores más restrictivos:

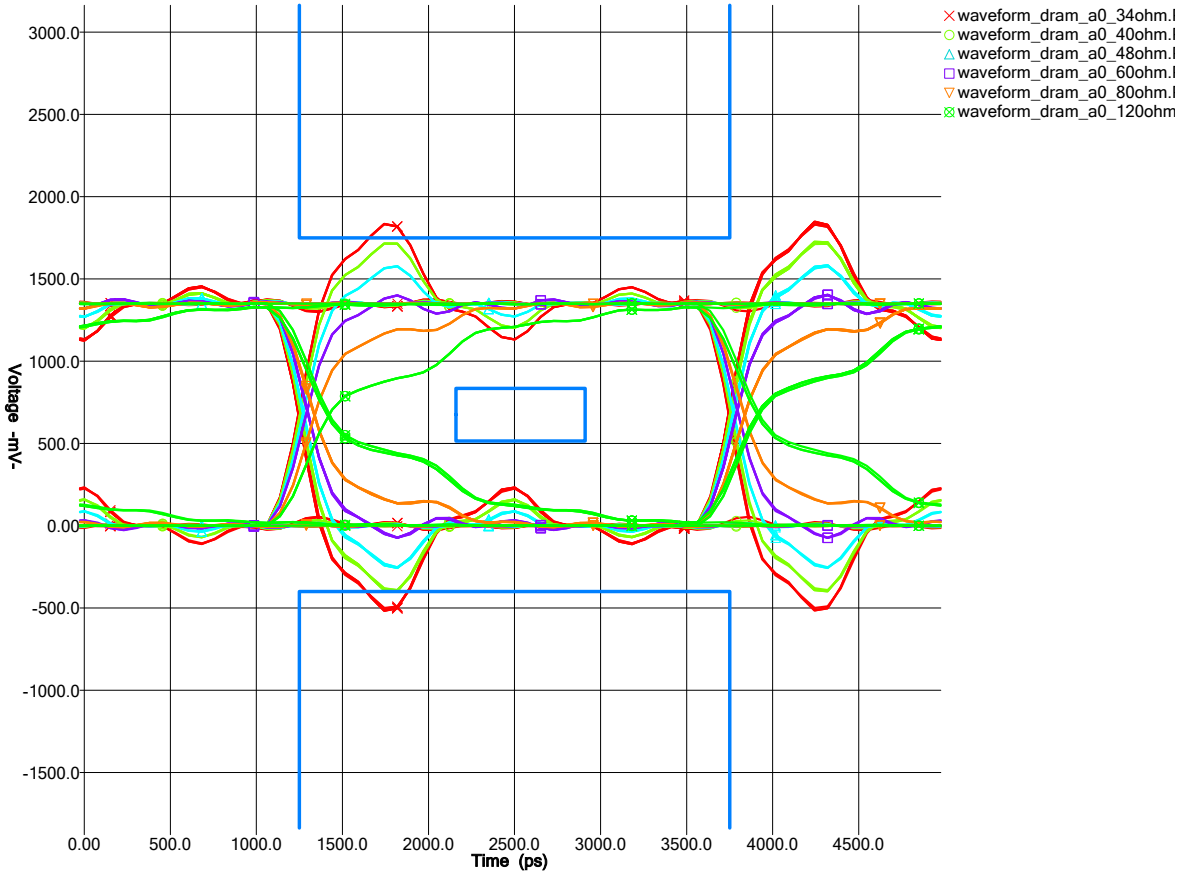


6.5.2. Líneas unidireccionales

Las líneas de dirección y control son unidireccionales, en estas líneas el SoC siempre actúa como el emisor y la RAM como el receptor. La RAM no dispone de terminaciones configurables en estas entradas, pero el SoC sí que permite configurar la impedancia de salida entre 34Ω y 240Ω:

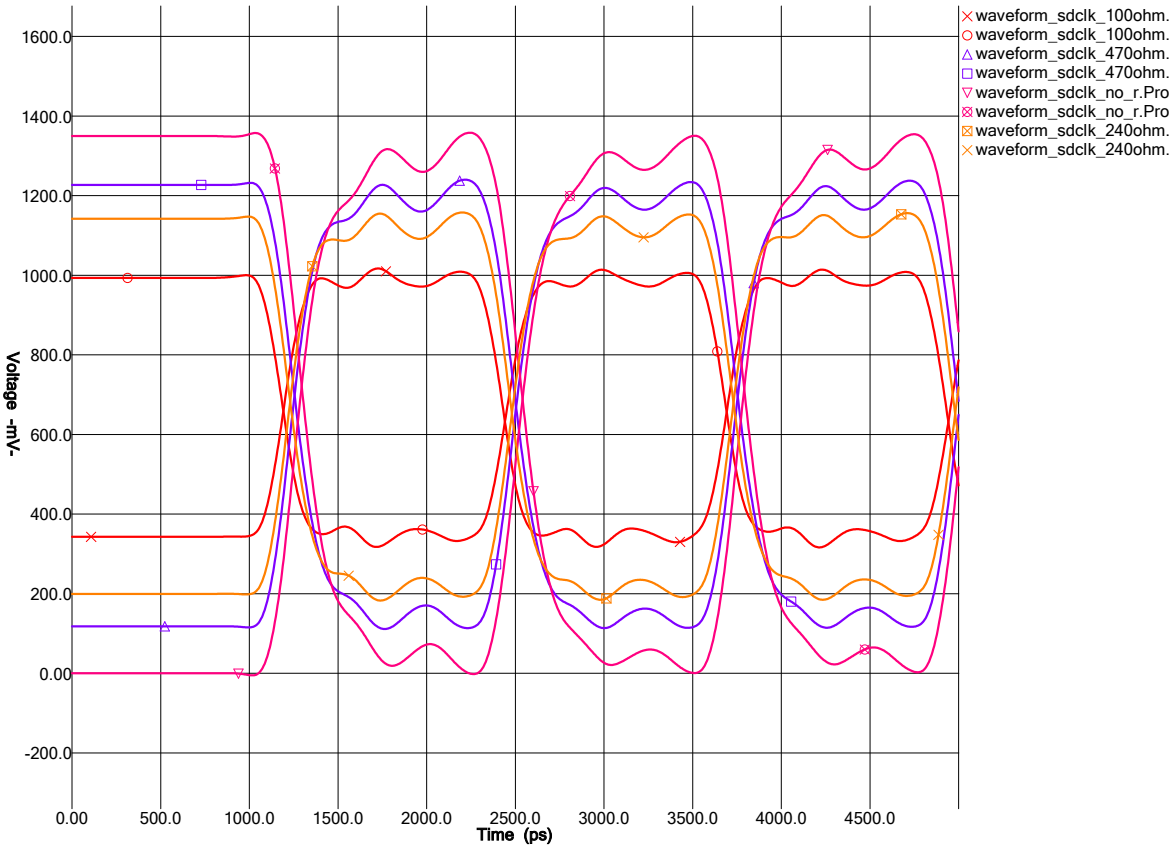


A continuación, se representa la señal *DRAM_A0* medida en el receptor para cada uno de los valores de impedancia de salida configurables:



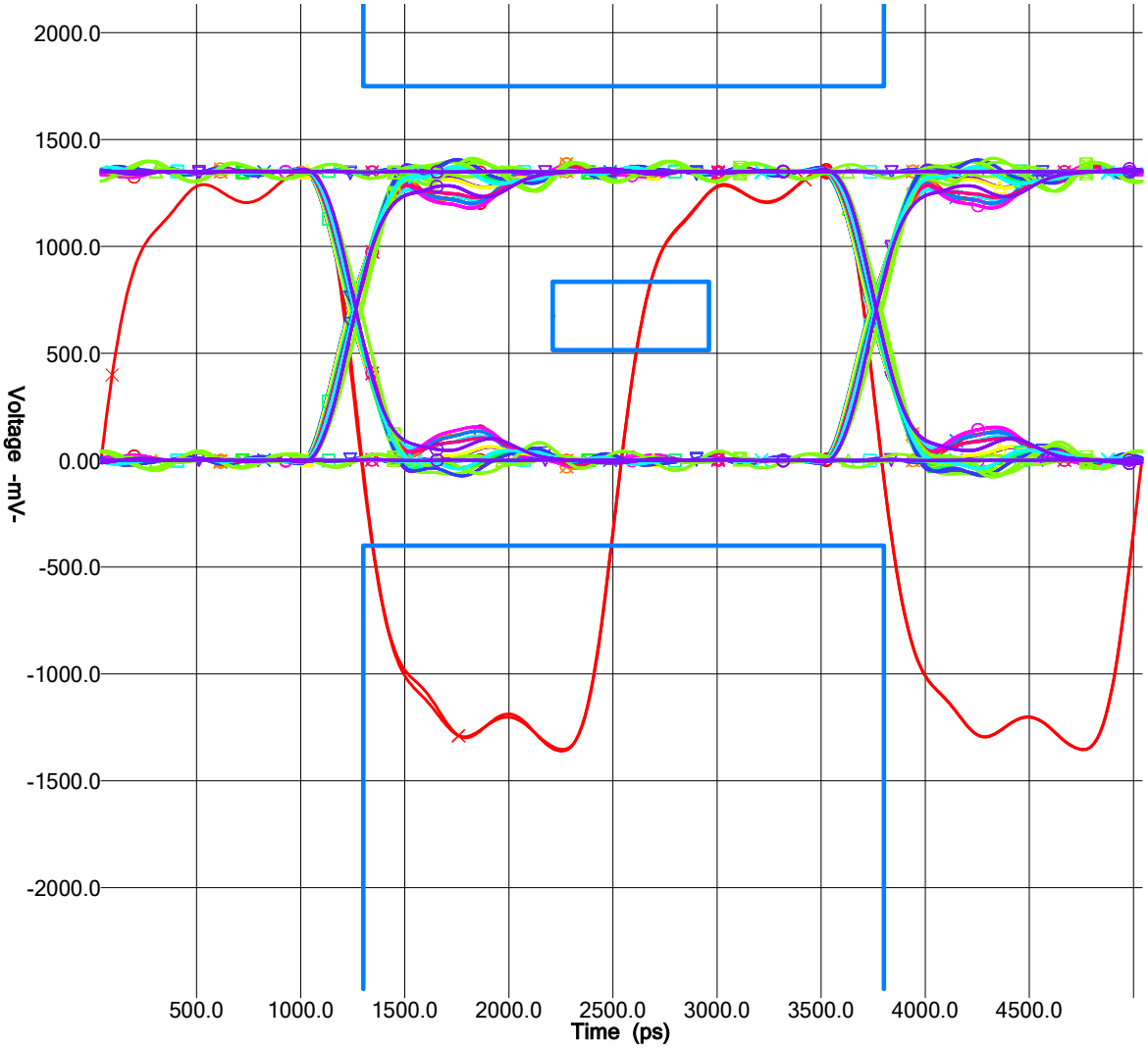
Se observa que salvo con los valores extremos de 34Ω y 240Ω, cualquier otra impedancia de salida entre 40Ω y 120Ω cumple con la máscara de ojo. El mejor resultado se consigue con el valor de 60Ω, con este valor es con el que el ojo queda más abierto, por lo que será el valor que se configure posteriormente.

Estas líneas se muestrean en el flanco de subida de la señal de reloj del par *DRAM_SDCLK_P* y *DRAM_SDCLK_N*. En la topología *point-to-point* en la que los dos chips se conectan directamente, es común no utilizar ninguna terminación en este par, aunque algunos fabricantes colocan una resistencia de 100Ω, 200Ω o 470Ω. Se han simulado las distintas opciones para comprobar su efecto en la señal de reloj:



En un principio no se utilizará ninguna terminación, aunque añadir una resistencia de 100Ω o 200Ω podría ser de ayuda para reducir la radiación.

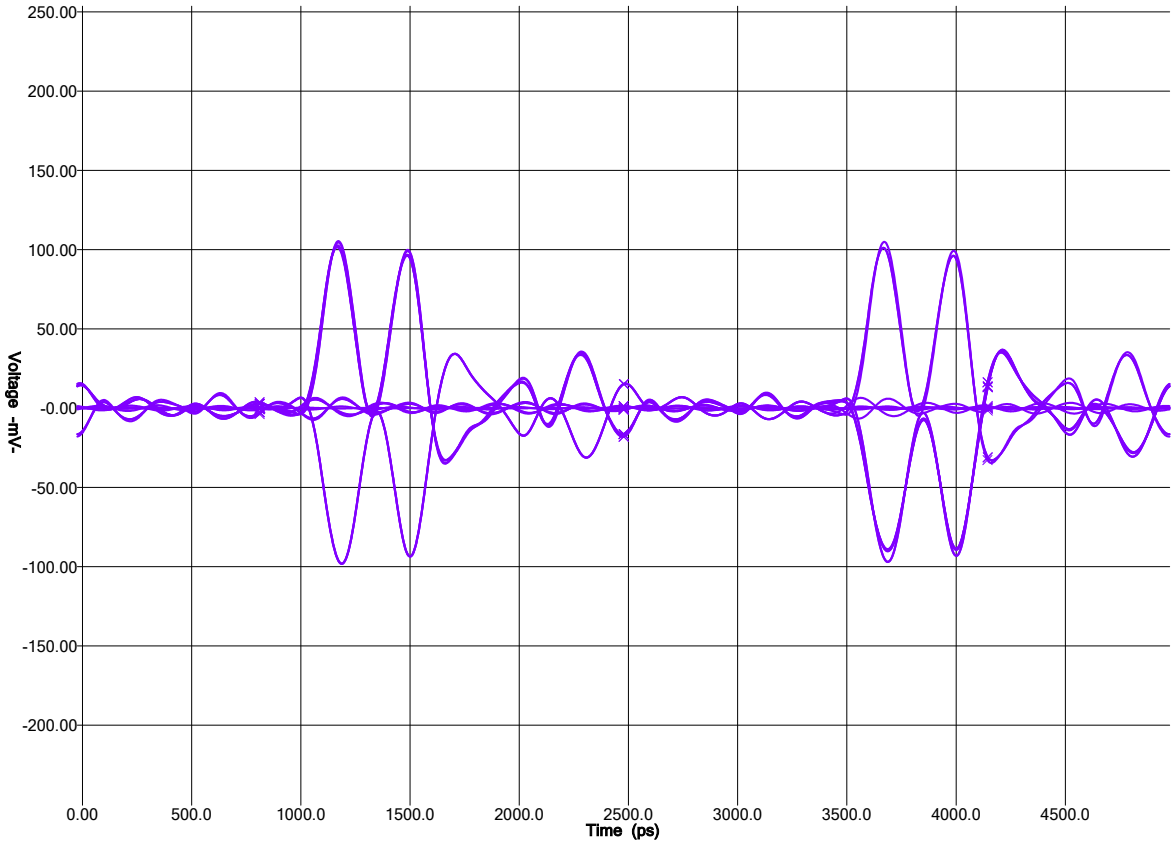
A continuación, se han simulado a la vez todas las líneas de dirección de *DRAM_A0* a *DRAM_A14* junto a la señal de reloj representada en rojo, en el receptor. La impedancia de salida se ha configurado en 60Ω para todas las señales:



Se observa que el resultado obtenido es bastante bueno. Las señales respetan la máscara del ojo y con mucho margen. Probablemente este margen sea suficiente para asumir la degradación adicional que se tendrá en la práctica debido a las discontinuidades en los caminos de retorno al cambiar de plano de referencia.

De aquí también se puede concluir que gracias al margen que hay y a las opciones de configuración del SoC, la interfaz funcionará correctamente con cualquier impedancia de pista entre 40Ω y 60Ω, por lo que el valor concreto de impedancia que se utilice para la pista no será muy importante siempre que sea un valor razonable.

Por último, se ha realizado una simulación de crosstalk para la señal *DRAM_A11*. Se ha fijado el estado de la señal a nivel bajo y se ha ejecutado la simulación para comprobar el acoplamiento en esta señal por parte de otras pistas:



El máximo sobrepico en esta señal debido al crosstalk es de 100mV lo que no provoca un cambio en el nivel lógico de la señal. Además, los mayores picos ocurren durante los flancos de subida y bajada del resto de señales.

En un bus paralelo como este, inevitablemente siempre va a haber cierto grado de crosstalk cuando las señales cambian de estado. Lo importante es que el efecto se haya disipado en el momento de muestrear el estado de las señales del bus.

6.5.3. Líneas bidireccionales

Las líneas de datos y los pares diferenciales de *strobe* son bidireccionales. Las líneas *DRAM_D0* a *DRAM_D7* se muestrean con el par de strobe *DRAM_SDQS0_P* y *DRAM_SDQS0_N*, mientras que las líneas *DRAM_D8* a *DRAM_D15* se muestrean con el par de strobe *DRAM_SDQS1_P* y *DRAM_SDQS1_N*.

Las señales de strobe son un reloj que funciona a ráfagas durante las operaciones de lectura o escritura, y es emitido por el mismo componente que emite los datos, es decir, por la RAM durante la lectura y por el SoC durante la escritura. Estos relojes funcionan a la misma frecuencia que el reloj principal *DRAM_SDCLK_P* y *DRAM_SDCLK_N*, pero no están sincronizados. Esto proporciona varias ventajas:

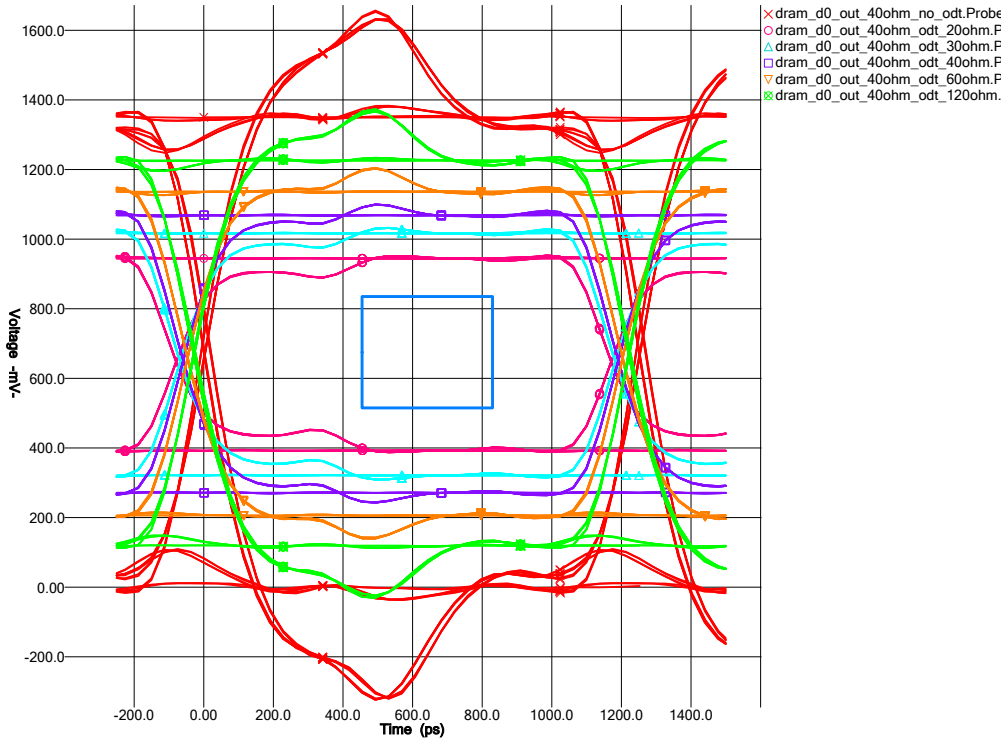
- Si se usase un único reloj emitido por el SoC, durante las lecturas el SoC transmitiría primero el reloj a la RAM, donde llegaría un tiempo más tarde. Una vez recibido, la RAM enviaría los datos al SoC, donde llegarían otro tiempo más tarde. Es decir, el retardo de propagación limitaría la velocidad máxima de la transmisión. Cuando el mismo componente que emite los datos emite el reloj, la velocidad de transmisión es independiente de la longitud de la pista.
- Al usar cada grupo de señales un reloj distinto, cada grupo puede tener longitudes de pista distintas, lo que facilita el rutado de la interfaz ya que no es necesario hacer un length matching estricto entre señales que pertenecen a distintos grupos.

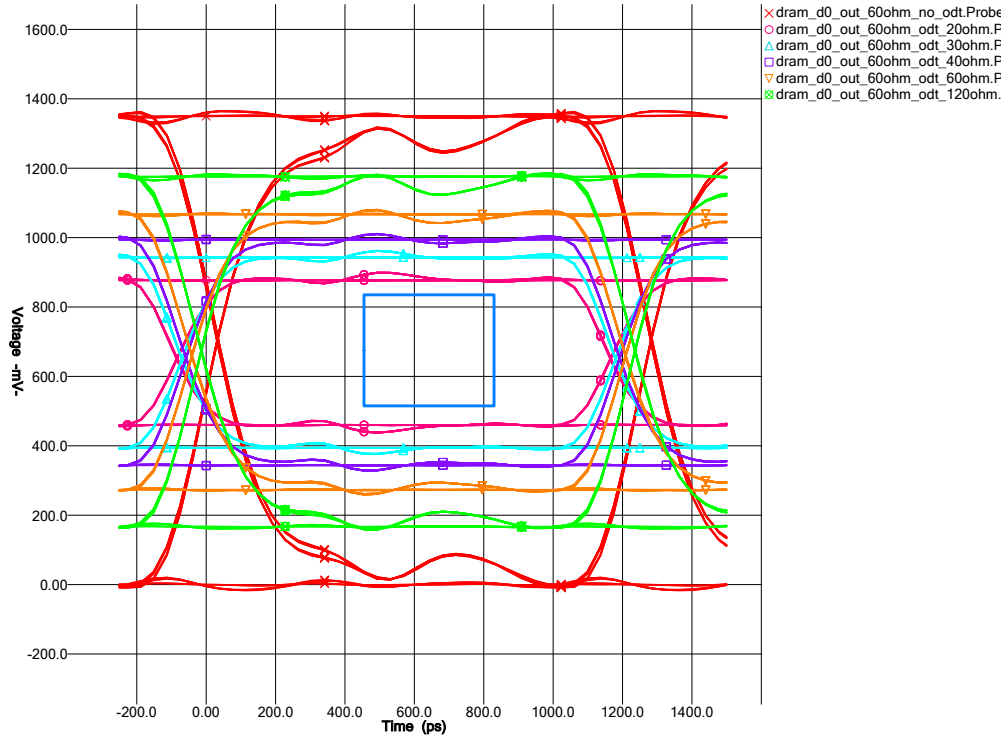
6.5.3.1. SoC como emisor y RAM como receptor

El SoC permite configurar la impedancia de salida del driver entre 34Ω y 240Ω, de forma similar a las líneas unidireccionales. La RAM permite configurar resistencias de terminación ODT a la entrada del receptor con valores entre 20Ω y 120Ω.

En las siguientes gráficas se muestra la señal *DRAM_D0* medida en el receptor, para una impedancia de salida de 40Ω o 60Ω en el driver y para cada uno de los valores de resistencia ODT configurables a la entrada del receptor.

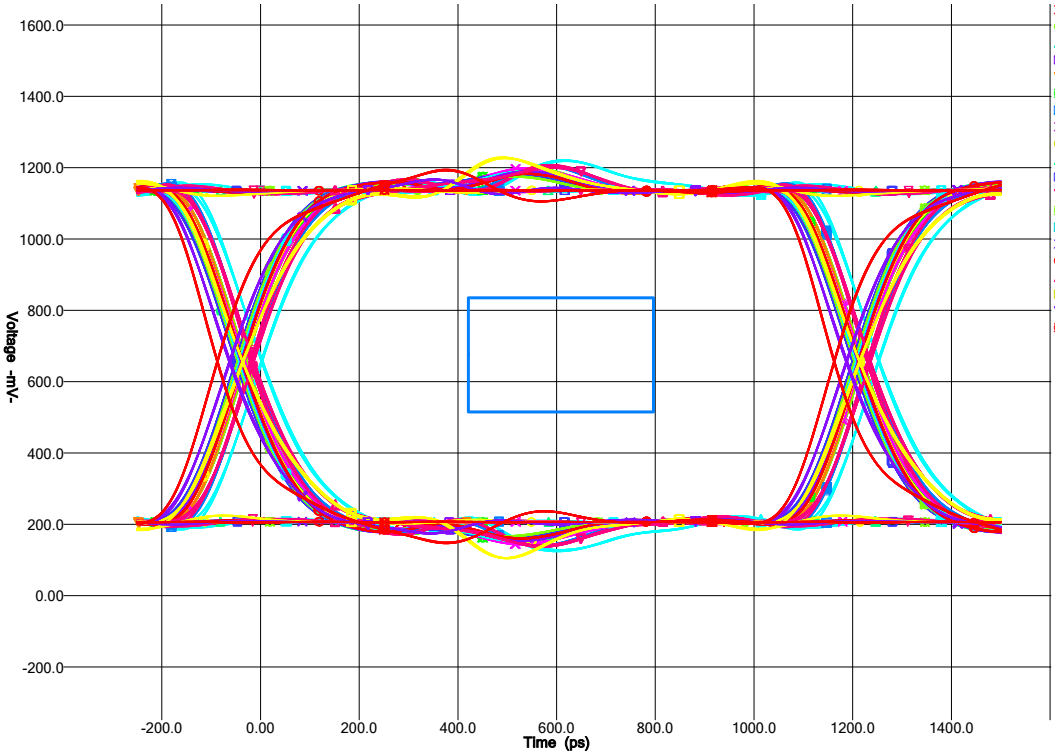
En todos los casos se respeta la máscara del ojo, aunque en función de la configuración hay mucha diferencia en los sobrepicos y en el margen de la señal. Al disminuir la impedancia de salida aumenta el margen pero aumentan los sobrepicos. Al disminuir las resistencias ODT disminuyen los sobrepicos pero disminuye el margen.





Como compromiso, se utilizará una impedancia de salida de 40Ω y una resistencia ODT de 60Ω, en línea con lo que recomienda el fabricante. De esta forma se consigue un buen equilibrio entre la apertura del ojo y la reducción de las reflexiones.

Se han simulado a la vez todas las líneas de datos de *DRAM_D0* a *DRAM_D15* junto a los dos pares de strobe en el receptor. La impedancia de salida se ha configurado en 40Ω para todas las señales y las resistencias ODT en 60Ω:

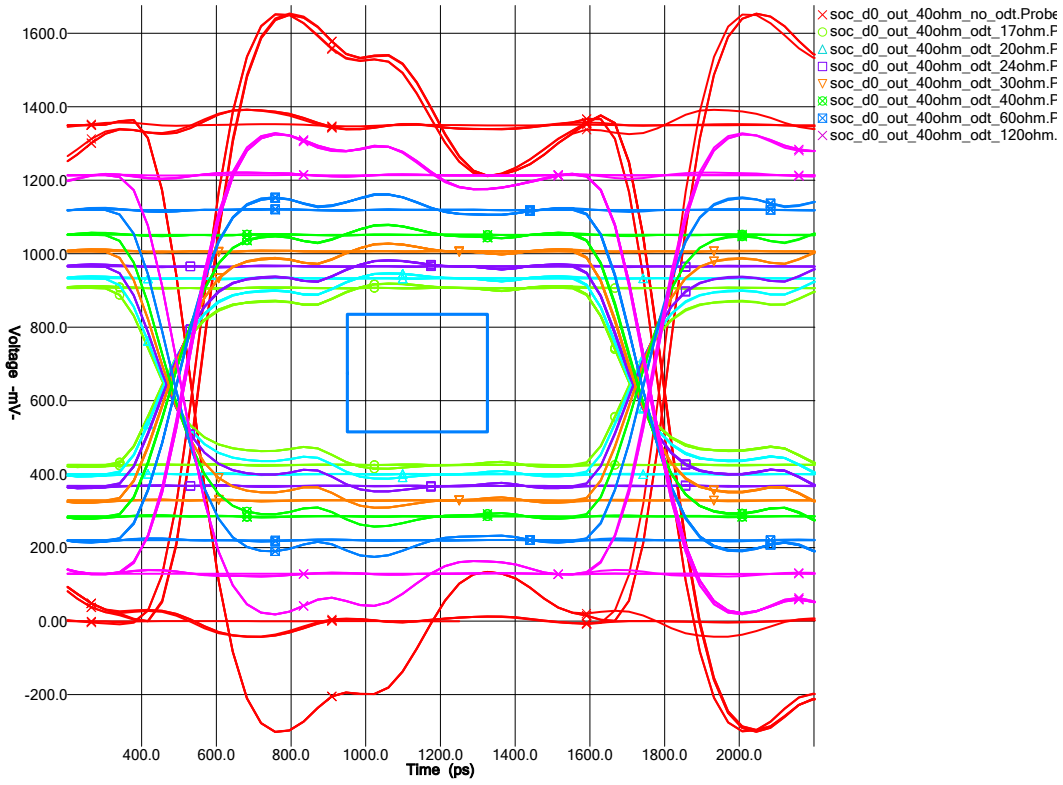


Se observa que el resultado obtenido es bastante bueno aunque los márgenes son menores que en las líneas unidireccionales, tanto en tiempo como en tensión.

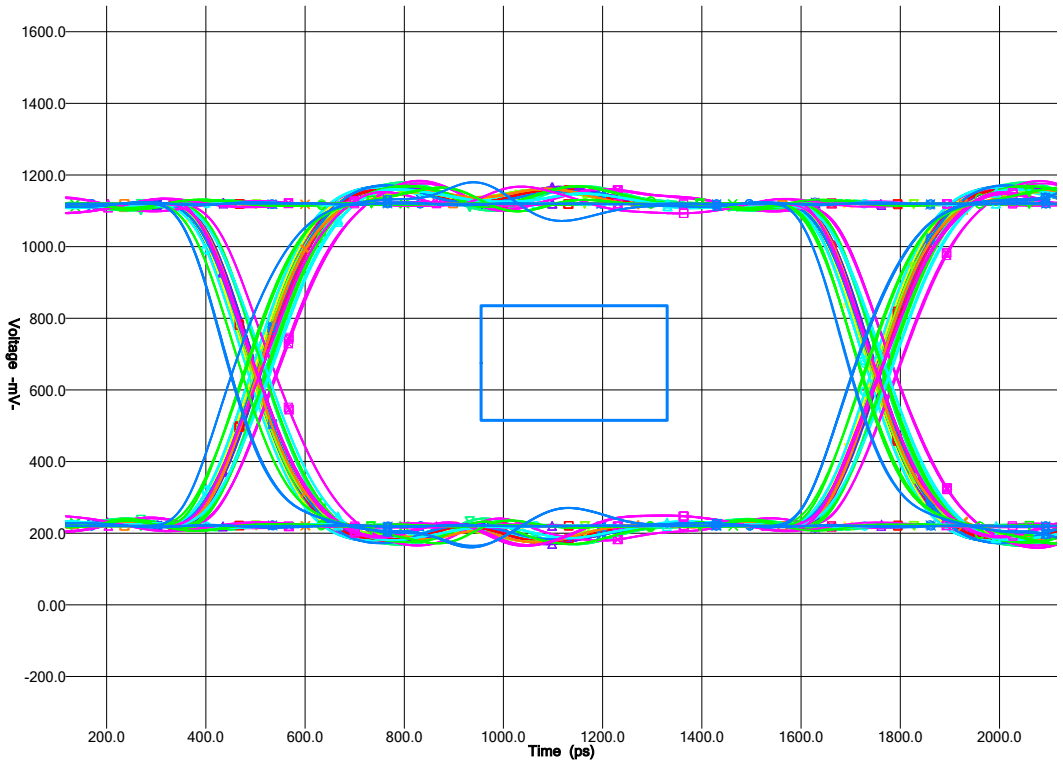
6.5.3.2. RAM como emisor y SoC como receptor

La RAM permite configurar la impedancia de salida del driver a 34Ω o 40Ω, no es posible configurarla a 60Ω como en el SoC. El SoC permite configurar resistencias de terminación ODT a la entrada del receptor con valores entre 17Ω y 120Ω.

En la siguiente gráfica se muestra la señal *DRAM_D0* medida en el receptor, para una impedancia de salida de 40Ω y para cada uno de los valores de resistencia ODT configurables a la entrada del receptor. Se observa que al igual que en la otra dirección, en todos los casos se respeta la máscara del ojo, aunque con la impedancia de salida de 40Ω y las resistencias ODT de 60Ω se consigue el resultado más equilibrado.

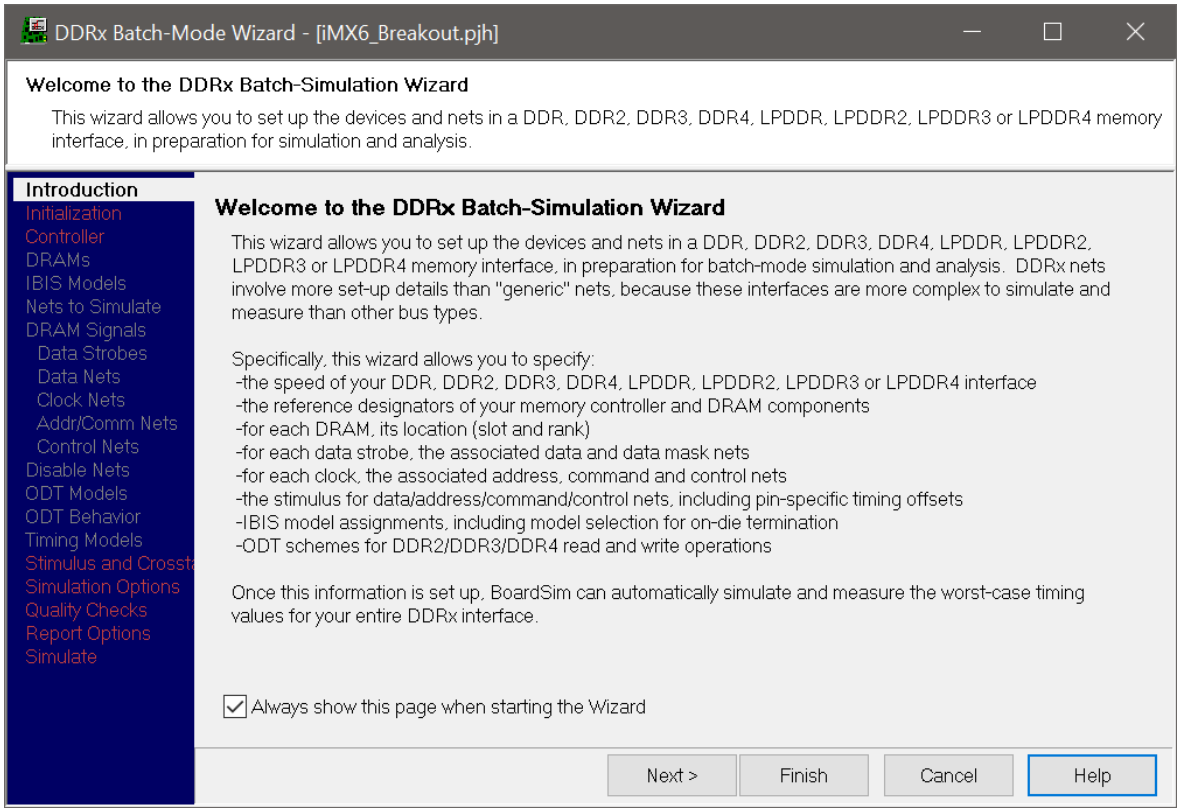


Con dicha configuración se han representado todas las líneas de datos de *DRAM_D0* a *DRAM_D15* junto a los dos pares de strobe en el receptor:



6.5.4. Simulación batch

La simulación *batch* permite fácilmente simular por lotes todas las líneas de la interfaz simultáneamente, simulando transacciones reales como operaciones de lectura o de escritura. HyperLynx dispone de un *wizard* que facilita su configuración:



En cada paso del wizard se ha ido configurando cada parámetro de la simulación, indicando cuáles son las señales de la interfaz a simular y asignando a cada señal los modelos con la impedancia de salida o resistencias ODT correspondientes que se ha decidido utilizar en los apartados anteriores.

En la pestaña *Timing Model* ha sido necesario modificar el modelo del controlador, en este caso el SoC. Los parámetros por defecto de $t_{DS} = 150ps$ y $t_{DH} = 250ps$ se han cambiado por $t_{DS} = 170ps$ y $t_{DH} = 205ps$, los mismos que indica la especificación de la RAM incluyendo el derating, ya que el datasheet del SoC no especifica esta información. El resto de parámetros se han mantenido por defecto.

Una vez completada la configuración del wizard, se ha ejecutado la simulación tras la que se ha obtenido un informe con los resultados para cada operación y para cada pista. Todas las señales superan el test con suficiente margen.

En las siguientes imágenes se muestran los extractos para varias operaciones. Para cada señal se indica el margen disponible tanto en tiempo como en tensión.

	Signal/Dram		Status	Corner	Setup	Hold	Overshoot	Undershoot	Overshoot Area	Undershoot Area	tVAC	VIH/L	
#	Signal	Accessed DRAM	Pass/Fail	Case	Margin [ps]	Margin [ps]	Margin [mV]	Margin [mV]	Margin [V*ns]	Margin [V*ns]	Margin [ps]	VIH/L(AC)	VIH/L(DC)
	Filter	Filter		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	DRAM_A0	U4.N3	Pass	Typ	603.1	597.2	358.5	338.9	0.664	0.660	N/A	Pass	Pass
2	DRAM_A1	U4.P7	Pass	Typ	648.9	553.6	389.3	388.0	0.668	0.668	N/A	Pass	Pass
3	DRAM_A10	U4.L7	Pass	Typ	637.0	566.7	386.1	385.3	0.668	0.668	N/A	Pass	Pass
4	DRAM_A11	U4.R7	Pass	Typ	628.1	570.1	388.4	388.6	0.669	0.668	N/A	Pass	Pass
5	DRAM_A12	U4.N7	Pass	Typ	635.1	565.7	389.3	389.6	0.669	0.668	N/A	Pass	Pass
6	DRAM_A13	U4.T3	Pass	Typ	646.7	554.4	389.3	391.4	0.669	0.669	N/A	Pass	Pass
7	DRAM_A14	U4.T7	Pass	Typ	635.7	566.9	382.2	377.3	0.668	0.667	N/A	Pass	Pass
8	DRAM_A2	U4.P3	Pass	Typ	641.4	562.3	381.2	374.5	0.668	0.667	N/A	Pass	Pass
9	DRAM_A3	U4.N2	Pass	Typ	639.8	564.8	380.7	369.3	0.668	0.666	N/A	Pass	Pass
10	DRAM_A4	U4.P8	Pass	Typ	639.1	559.3	388.4	387.5	0.669	0.668	N/A	Pass	Pass
11	DRAM_A5	U4.P2	Pass	Typ	633.1	569.0	370.0	348.0	0.666	0.661	N/A	Pass	Pass
12	DRAM_A6	U4.R8	Pass	Typ	643.3	557.9	389.2	388.3	0.669	0.668	N/A	Pass	Pass
13	DRAM_A7	U4.R2	Pass	Typ	646.8	552.7	388.5	391.2	0.669	0.668	N/A	Pass	Pass
14	DRAM_A8	U4.T8	Pass	Typ	635.2	562.8	388.5	390.3	0.669	0.668	N/A	Pass	Pass
15	DRAM_A9	U4.R3	Pass	Typ	637.4	564.9	383.6	383.0	0.668	0.668	N/A	Pass	Pass
16	DRAM_BA0	U4.M2	Pass	Typ	634.7	569.1	349.5	328.4	0.663	0.658	N/A	Pass	Pass
17	DRAM_BA1	U4.N8	Pass	Typ	649.7	554.9	383.8	382.8	0.668	0.668	N/A	Pass	Pass
18	DRAM_BA2	U4.M3	Pass	Typ	654.6	550.2	381.9	382.2	0.668	0.668	N/A	Pass	Pass
19	DRAM_CAS	U4.K3	Pass	Typ	659.2	545.4	385.5	385.0	0.668	0.668	N/A	Pass	Pass
20	DRAM_CS0	U4.L2	Pass	Typ	638.8	565.7	372.5	350.7	0.667	0.663	N/A	Pass	Pass

Figura 6.14: Tabla de resultados para líneas de dirección y control.

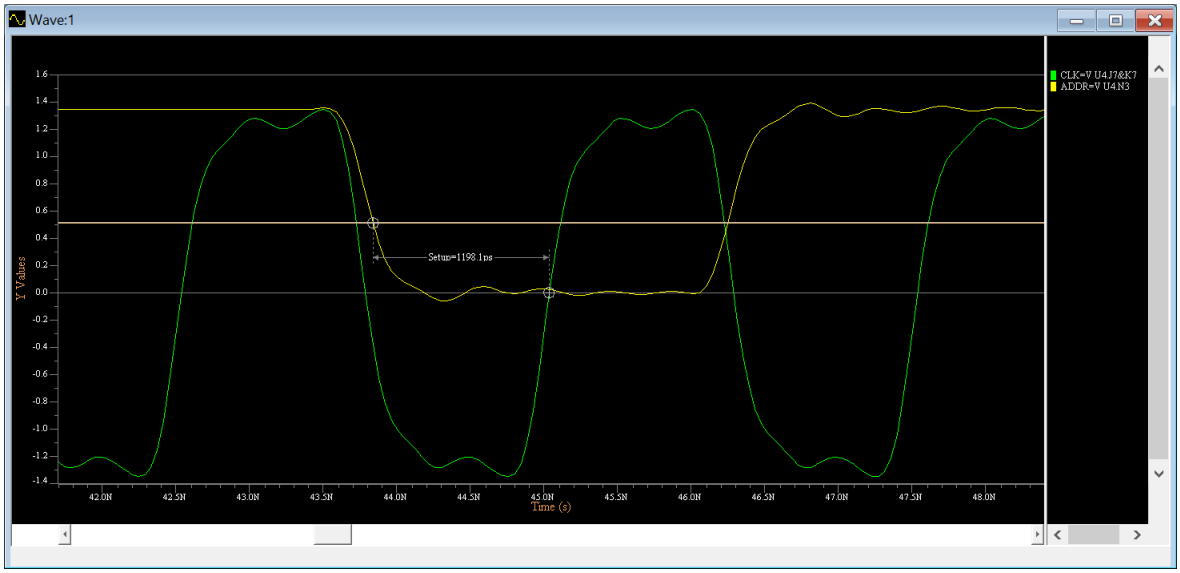


Figura 6.15: Peor caso de setup time para la señal DRAM_A0.

	Signal/DRAM/Controller	Status	Corner	Setup				Hold			
#	Signal	Pass/Fail	Case	Measurement [ps]	Output Variation [ps]	Base Requirement [ps]	Margin [ps]	Measurement [ps]	Output Variation [ps]	Base Requirement [ps]	Margin [ps]
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	DRAM_D0	Pass	Typ	644.3	-250.0	-170.0	224.3	506.9	-250.0	-205.0	51.9
2	DRAM_D1	Pass	Typ	658.9	-250.0	-170.0	238.9	499.1	-250.0	-205.0	44.1
3	DRAM_D10	Pass	Typ	601.7	-250.0	-170.0	181.7	553.5	-250.0	-205.0	98.5
4	DRAM_D11	Pass	Typ	575.1	-250.0	-170.0	155.1	559.8	-250.0	-205.0	104.8
5	DRAM_D12	Pass	Typ	525.5	-250.0	-170.0	105.5	587.1	-250.0	-205.0	132.1
6	DRAM_D13	Pass	Typ	620.8	-250.0	-170.0	200.8	555.3	-250.0	-205.0	100.3
7	DRAM_D14	Pass	Typ	639.2	-250.0	-170.0	219.2	536.8	-250.0	-205.0	81.8
8	DRAM_D15	Pass	Typ	637.5	-250.0	-170.0	217.5	552.5	-250.0	-205.0	97.5
9	DRAM_D2	Pass	Typ	662.5	-250.0	-170.0	242.5	495.1	-250.0	-205.0	40.1
10	DRAM_D3	Pass	Typ	663.3	-250.0	-170.0	243.3	497.8	-250.0	-205.0	42.8
11	DRAM_D4	Pass	Typ	666.0	-250.0	-170.0	246.0	497.1	-250.0	-205.0	42.1
12	DRAM_D5	Pass	Typ	625.4	-250.0	-170.0	205.4	509.2	-250.0	-205.0	54.2
13	DRAM_D6	Pass	Typ	629.1	-250.0	-170.0	209.1	511.5	-250.0	-205.0	56.5
14	DRAM_D7	Pass	Typ	664.4	-250.0	-170.0	244.4	507.9	-250.0	-205.0	52.9
15	DRAM_D8	Pass	Typ	642.0	-250.0	-170.0	222.0	536.7	-250.0	-205.0	81.7
16	DRAM_D9	Pass	Typ	607.8	-250.0	-170.0	187.8	553.1	-250.0	-205.0	98.1

Figura 6.16: Tabla de resultados para líneas de datos durante la lectura.

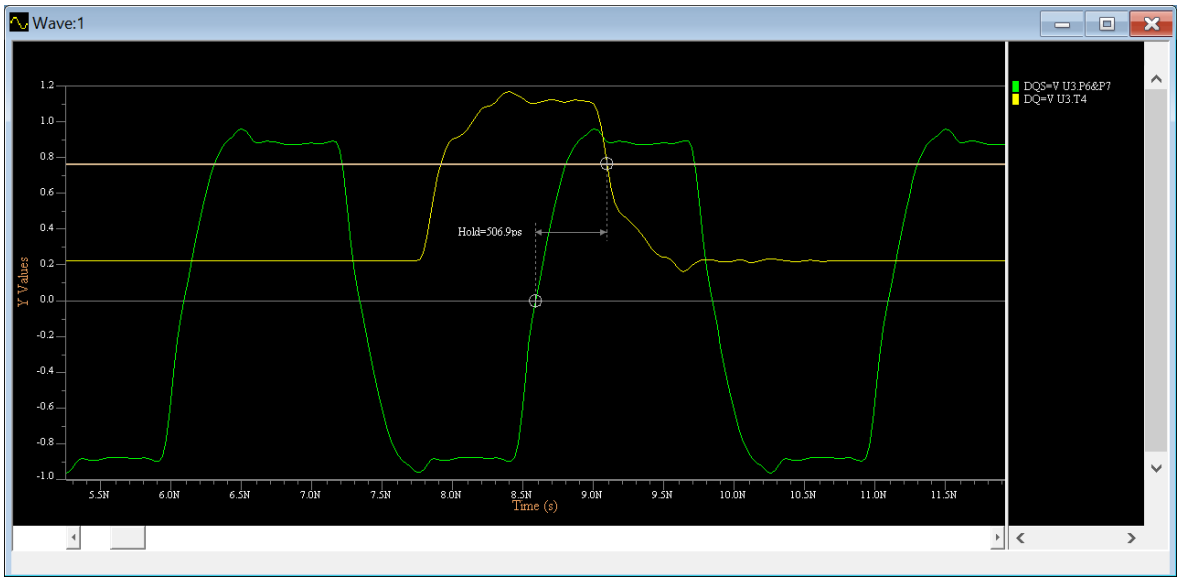


Figura 6.17: Peor caso de hold time para la señal DRAM_D0 durante la lectura.

	Signal/Controller/DRAM	Status	Corner	Setup					Hold					Overshoot	Undershoot
#	Signal	Pass/Fail	Case	Measurement [ps]	Output Variation [ps]	Base Requirement [ps]	Derating [ps]	Margin [ps]	Measurement [ps]	Output Variation [ps]	Base Requirement [ps]	Derating [ps]	Margin [ps]	Margin [mV]	Margin [mV]
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	DRAM_D0	Pass	Typ	545.8	-150.0	-90.0	-80.0	225.8	618.9	-150.0	-160.0	-45.0	263.9	400.0	400.0
2	DRAM_D1	Pass	Typ	555.3	-150.0	-90.0	-80.0	235.3	601.4	-150.0	-160.0	-45.0	246.4	400.0	400.0
3	DRAM_D10	Pass	Typ	595.1	-150.0	-90.0	-80.0	275.1	568.9	-150.0	-160.0	-45.0	213.9	400.0	400.0
4	DRAM_D11	Pass	Typ	604.6	-150.0	-90.0	-80.0	284.6	561.0	-150.0	-160.0	-45.0	206.0	400.0	400.0
5	DRAM_D12	Pass	Typ	598.7	-150.0	-90.0	-80.0	278.7	565.5	-150.0	-160.0	-45.0	210.5	400.0	400.0
6	DRAM_D13	Pass	Typ	579.3	-150.0	-90.0	-80.0	259.3	583.3	-150.0	-160.0	-45.0	228.3	400.0	400.0
7	DRAM_D14	Pass	Typ	599.8	-150.0	-90.0	-80.0	279.8	561.2	-150.0	-160.0	-45.0	206.2	400.0	400.0
8	DRAM_D15	Pass	Typ	552.3	-150.0	-90.0	-80.0	232.3	602.8	-150.0	-160.0	-45.0	247.8	400.0	400.0
9	DRAM_D2	Pass	Typ	563.3	-150.0	-90.0	-80.0	243.3	597.7	-150.0	-160.0	-45.0	242.7	400.0	400.0
10	DRAM_D3	Pass	Typ	563.5	-150.0	-90.0	-80.0	243.5	594.2	-150.0	-160.0	-45.0	239.2	400.0	400.0
11	DRAM_D4	Pass	Typ	556.7	-150.0	-90.0	-80.0	236.7	601.3	-150.0	-160.0	-45.0	246.3	400.0	400.0
12	DRAM_D5	Pass	Typ	565.3	-150.0	-90.0	-80.0	245.3	589.6	-150.0	-160.0	-45.0	234.6	400.0	400.0
13	DRAM_D6	Pass	Typ	560.2	-150.0	-90.0	-80.0	240.2	592.4	-150.0	-160.0	-45.0	237.4	400.0	400.0
14	DRAM_D7	Pass	Typ	533.0	-150.0	-90.0	-80.0	213.0	622.6	-150.0	-160.0	-45.0	267.6	400.0	400.0
15	DRAM_D8	Pass	Typ	592.4	-150.0	-90.0	-80.0	272.4	571.8	-150.0	-160.0	-45.0	216.8	400.0	400.0
16	DRAM_D9	Pass	Typ	594.5	-150.0	-90.0	-80.0	274.5	567.6	-150.0	-160.0	-45.0	212.6	400.0	400.0
17	DRAM_DQM0	Pass	Typ	553.6	-150.0	-90.0	-80.0	233.6	610.0	-150.0	-160.0	-45.0	255.0	400.0	400.0
18	DRAM_DQM1	Pass	Typ	592.7	-150.0	-90.0	-80.0	272.7	573.3	-150.0	-160.0	-45.0	218.3	400.0	400.0

Figura 6.18: Tabla de resultados para líneas de datos durante la escritura.

	Signal/Driver/Receicer				Status	Corner	Overshoot	Undershoot	Overshoot Area	Undershoot Area	tDVAC	Vix			
#	Signal	Driver.Pins	Receiver.Pins	Operation	Pass/Fail	Case	Margin [mV]	Margin [mV]	Margin [V*ns]	Margin [V*ns]	Margin [ps]	Margin [mV]	VID(AC) [mV]	VID(DC)	VSEH/L
	Filter	Filter		Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	DRAM_SDCLK_P	U3.P1&P2	U4.J7&K7	Write rank(1,1)	Pass	Typ	395.2	400.0	0.250	0.250	N/A	103.9	1026.2	Pass	Pass
2	DRAM_SDQS0_P	U4.C7&B7	U3.P6&P7	Read rank(1,1)	Pass	Typ	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
3	DRAM_SDQS0_P	U3.P6&P7	U4.C7&B7	Write rank(1,1)	Pass	Typ	400.0	400.0	0.250	0.250	N/A	146.7	664.7	Pass	Pass
4	DRAM_SDQS1_P	U4.F3&G3	U3.T1&T2	Read rank(1,1)	Pass	Typ	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
5	DRAM_SDQS1_P	U3.T1&T2	U4.F3&G3	Write rank(1,1)	Pass	Typ	400.0	400.0	0.250	0.250	N/A	136.1	727.6	Pass	Pass
6	DRAM_SDCLK_P	U3.P1&P2	U4.J7&K7	Write rank(1,1)	Pass	Slow	388.4	391.6	0.249	0.249	N/A	111.6	936.8	Pass	Pass
7	DRAM_SDQS0_P	U4.C7&B7	U3.P6&P7	Read rank(1,1)	Pass	Slow	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
8	DRAM_SDQS0_P	U3.P6&P7	U4.C7&B7	Write rank(1,1)	Pass	Slow	400.0	400.0	0.250	0.250	N/A	137.7	592.8	Pass	Pass
9	DRAM_SDQS1_P	U4.F3&G3	U3.T1&T2	Read rank(1,1)	Pass	Slow	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
10	DRAM_SDQS1_P	U3.T1&T2	U4.F3&G3	Write rank(1,1)	Pass	Slow	400.0	400.0	0.250	0.250	N/A	141.3	673.0	Pass	Pass
11	DRAM_SDCLK_P	U3.P1&P2	U4.J7&K7	Write rank(1,1)	Pass	Fast	373.7	400.0	0.247	0.250	N/A	115.5	1117.4	Pass	Pass
12	DRAM_SDQS0_P	U4.C7&B7	U3.P6&P7	Read rank(1,1)	Pass	Fast	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
13	DRAM_SDQS0_P	U3.P6&P7	U4.C7&B7	Write rank(1,1)	Pass	Fast	400.0	400.0	0.250	0.250	N/A	141.8	641.6	Pass	Pass
14	DRAM_SDQS1_P	U4.F3&G3	U3.T1&T2	Read rank(1,1)	Pass	Fast	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
15	DRAM_SDQS1_P	U3.T1&T2	U4.F3&G3	Write rank(1,1)	Pass	Fast	400.0	400.0	0.250	0.250	N/A	128.8	691.6	Pass	Pass

Figura 6.19: Tabla de resultados para pares diferenciales.

Capítulo 7

Estudio de compatibilidad electromagnética

7.1. Introducción

Todo equipo electrónico emite radiación electromagnética a su entorno, a la vez que está expuesto a recibir radiación por parte del resto de equipos del entorno. Si no hubiese ningún tipo de control sobre esta radiación que emiten los equipos, estos se interferirían entre sí, dando lugar a todo tipo de problemas de funcionamiento.

El objetivo de la compatibilidad electromagnética consiste en ponerle unos límites a la radiación que puede emitir cada equipo, a la vez que se le exige a estos equipos que sean capaces de tolerar la radiación que reciben del resto de su entorno.

De esta forma se consigue que distintos equipos y de distintos fabricantes puedan convivir en un mismo entorno electromagnético sin que se perjudiquen entre ellos. Esto hace necesario que haya alguna normativa que regule el campo de juego estableciendo cuáles son estos límites y los ensayos para verificar que se cumplen.

7.2. Marco normativo

7.2.1. Marco normativo en Europa

Para poder comercializarse dentro de la Unión Europea, muchos tipos de productos electrónicos deben llevar obligatoriamente el marcado CE. Este marcado demuestra que el producto cumple con las directivas europeas que le sean de aplicación.

Antes de colocar el marcado CE a un producto, es necesario que el fabricante redacte y firme una declaración de conformidad, con la que el fabricante se responsabiliza de la conformidad del producto con los requisitos de estas directivas.

Estas directivas solo fijan los requisitos más esenciales. Para comprobar si se cumple con estas directivas, es habitual recurrir a estándares armonizados en los que se especifica con mayor detalle estos requisitos en función de distintos tipos de producto.

En la declaración de conformidad el fabricante indica las directivas europeas que son de aplicación al producto, así como los estándares armonizados que se han utilizado en los ensayos con los que se respalda la declaración de conformidad.

Las directivas aplicables dependen en gran medida del tipo de producto y la aplicación a la que se destine. A continuación se indican las directivas más comunes en productos electrónicos, aunque en función de la aplicación a la que vayan destinados, algunos productos pueden estar sujetos a otras directivas más específicas:

- Directiva 2014/30/UE (EMC) sobre compatibilidad electromagnética [23].
- Directiva 2014/53/UE (RED) sobre equipos radioeléctricos [24].
- Directiva 2014/35/UE (LVD) sobre baja tensión [25].
- Directiva 2011/65/UE (RoHS) sobre restricciones a sustancias peligrosas [26].
- Directiva 2012/19/UE (WEEE) sobre residuos de aparatos electrónicos [27].

7.2.2. Marco normativo en Estados Unidos

En Estados Unidos los productos electrónicos deben llevar el marcado FCC regulado por la *Federal Communications Commission*, para el cual también se exigen requisitos de compatibilidad electromagnética. Los requisitos de emisiones no son muy distintos de los que marca la Unión Europea, sin embargo, en el caso de Estados Unidos no hay requisitos de inmunidad.

7.2.3. Marco normativo en otros países

Otros países tienen su propia normativa, aunque a menudo se basa en la de los anteriores, o en los mismos estándares internacionales. Si el equipo cumple con los requisitos para CE y FCC, es habitual que también cumpla con los requisitos que imponen gran parte del resto de países. Las mayores diferencias entre países suelen estar en las bandas de frecuencia permitidas en el caso de equipos de radiofrecuencia. Además, algunos países requieren de burocracia adicional y no admiten la auto-certificación, necesitando revisar y aprobar cada producto individualmente.

7.3. Normativa aplicable en Europa

En el caso de este SBC estamos ante un producto bastante genérico que no tiene una utilidad claramente definida, por lo que la normativa aplicable dependerá en gran parte del mercado o la aplicación a la que se decida orientar el producto.

El producto no está orientado a los sectores de la automoción, el ferroviario, el de la aviación, el médico, ni a ningún otro sector que requiera de normativa específica. Tampoco dispone de radios ni de antenas, no emite ni recibe por radiofrecuencia de manera intencionada, por lo que no le aplica la directiva RED (2014/53/UE).

El producto no se conecta directamente a la red eléctrica y tiene una tensión de funcionamiento DC menor a 75V, por lo que tampoco le aplica la directiva de baja tensión (2014/35/UE). Será necesario alimentar el dispositivo con una fuente de alimentación que sí que tenga certificado CE y cumpla con dicha directiva.

Como a todo producto electrónico, sí que le sería de aplicación la directiva RoHS (2011/65/EU). Para cumplir con esta directiva es necesario utilizar componentes electrónicos libres de plomo, así como un proceso de fabricación y ensamblado también libre de plomo y compatible con RoHS, de acuerdo al siguiente estándar armonizado:

- EN IEC 63000:2018. Documentación técnica para la evaluación de los productos eléctricos y electrónicos con respecto a la restricción de sustancias peligrosas.

En función del tipo de usuario al que se destine el producto, también le podría ser de aplicación la directiva general de EMC (2014/30/UE). Dado que el producto se trata de una placa de desarrollo o evaluación, sus emisiones e inmunidad dependerán en gran medida del software que desarrolle y le cargue el usuario, así como de los distintos y variados periféricos que el usuario conecte a los headers de expansión de la placa.

Por estas razones, el producto podría estar exento de cumplir con la directiva, al no considerarse un aparato acabado destinado al usuario final, sino un componente o kit de evaluación destinado a profesionales con fines de investigación y desarrollo.

En el caso de que el producto se comercializase como un aparato acabado destinado a usuarios finales, la directiva de EMC (2014/30/UE) sí que le sería de aplicación. En este caso, se podría considerar el producto como un equipo informático o multimedia, por lo que debería cumplir con los siguientes estándares armonizados:

- EN 55032:2015 + A11:2020 (Clase B). Compatibilidad electromagnética de equipos multimedia. Requisitos de emisión. (Anteriormente EN55022).
- EN 55035:2017 + A11:2020. Compatibilidad electromagnética de equipos multimedia. Requisitos de inmunidad. (Anteriormente EN55024).
- EN IEC 61000-3-2:2019 + A1:2021. Compatibilidad electromagnética (CEM). Parte 3-2: Límites para las emisiones de corriente armónica.
- EN 61000-3-3:2013 + A1:2019. Compatibilidad electromagnética (CEM). Parte 3-3: Límites de las variaciones de tensión, fluctuaciones de tensión y flicker.

7.4. Ensayos aplicables al producto

7.4.1. Emisiones radiadas

La norma EN 55032 establece los límites para las emisiones radiadas, así como la forma en la que se deben realizar estas mediciones. Actualmente la norma establece rangos de medición distintos en función de la máxima frecuencia interna del equipo. En este caso son 900MHz por lo que la medición se realizaría desde 30MHz hasta 5GHz.

La norma diferencia entre equipos de Clase A y de Clase B, siendo los de Clase A los que se utilizan en un entorno industrial y los de Clase B en un entorno residencial. En este caso supondremos que es de Clase B. La norma también establece límites distintos en función de la distancia de medición, según sea de 3 o 10 metros.

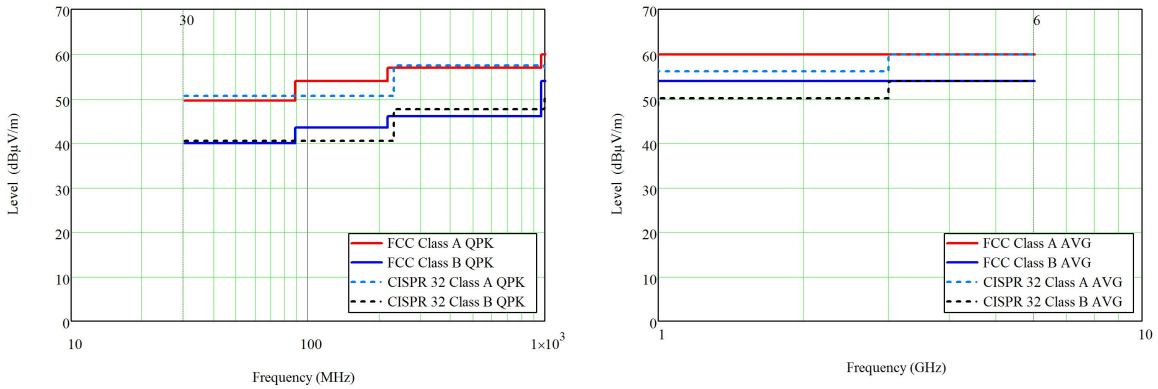


Figura 7.1: Límites para las emisiones radiadas a 3 metros [28].

7.4.2. Emisiones conducidas

Por debajo de los 30MHz, la longitud de onda es mayor a 10 metros. Los equipos bajo ensayo normalmente tienen dimensiones muy pequeñas en comparación, lo que hace que sean muy malos radiadores a tan bajas frecuencias. Por debajo de los 30MHz, solo los cables podrían radiar de manera importante, debido a su mayor longitud. Por otro lado, la medición con una antena en campo lejano de estas frecuencias no sería práctica debido a las dimensiones tan grandes que debería tener la instalación.

Por estos motivos, la radiación por debajo de esta frecuencia se mide indirectamente de manera conducida en los cables. La norma EN 55032 también establece la forma en la que se deben realizar estas mediciones, habitualmente a través de un LISN conectado a un analizador de espectros, aunque el procedimiento puede variar en función del tipo de los cables y de la señal.

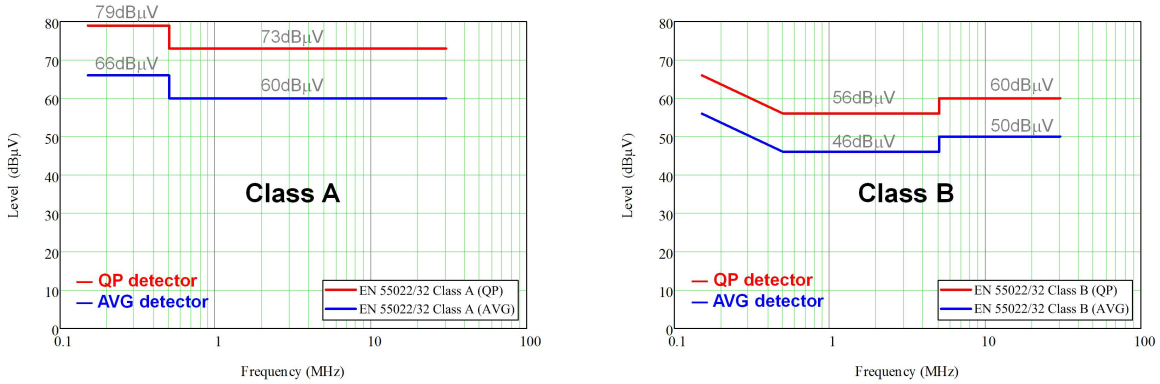


Figura 7.2: Límites para las emisiones conducidas en cables de alimentación [29].

7.4.3. Inmunidad a la radiación

La norma EN 55035 establece los requisitos de inmunidad, haciendo referencia a las normas básicas IEC 61000-4-3 y IEC 61000-4-6. Estas normas indican que debe comprobarse la inmunidad a un barrido con una señal de 1kHz modulada en AM al 80 %, de 150kHz a 80MHz para las conducidas y de 80MHz a 1GHz para las radiadas.

7.4.4. Inmunidad a ESD

La norma EN 55035 también establece los requisitos de inmunidad a las descargas electrostáticas, haciendo referencia a la norma básica IEC 61000-4-2. Las descargas solo se aplican a los puntos y superficies que se espera tocar durante el uso normal del equipo. El equipo debe soportar descargas de 4kV en los disparos por contacto o 8kV en los disparos por aire, siguiendo la forma de onda de la norma.

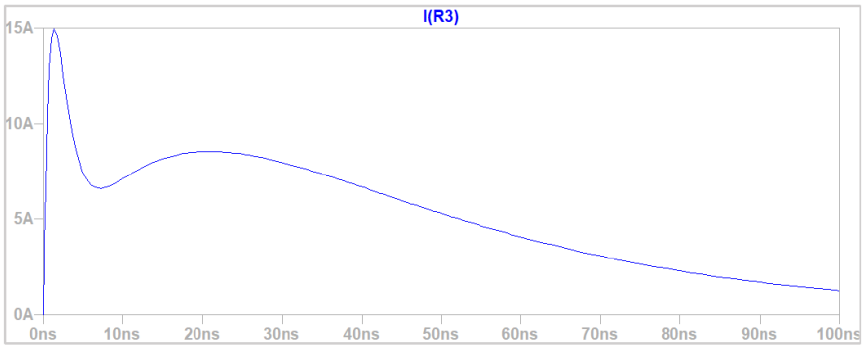


Figura 7.3: Forma de onda del disparo según IEC 61000-4-2 para 4kV.

En los siguientes apartados se explicarán los aspectos que se han tenido en cuenta para tratar de superar estos ensayos con mayores garantías. Por un lado, se verán los aspectos para reducir la radiación emitida por el PCB, los cuales también servirán para mejorar la inmunidad ante la radiación que el PCB recibe de su entorno. Por otro lado, también se verán los aspectos para mejorar la inmunidad ante ESD.

7.5. Radiación en el PCB

La radiación emitida por un SBC es un fenómeno bastante impredecible, ya que depende de la geometría de una gran cantidad de pistas, vías y otros componentes, así como de las señales que circulan por estas estructuras, las cuales están en constante cambio en función del software que se ejecute y el uso que se le de al dispositivo.

Si bien sería posible simular algunas partes críticas del diseño, es prácticamente imposible estimar con precisión cuál será la radiación emitida por todo el conjunto, aun disponiendo de los más avanzados programas de simulación. Esto implica que durante la etapa de diseño, antes de fabricar el PCB, no hay forma de garantizar si el circuito, una vez fabricado, acabará cumpliendo o no con los límites de la normativa.

Para comprobar el cumplimiento de estos límites se hace necesario por tanto fabricar el equipo y someterlo a ensayo en un laboratorio. En caso de no superar el ensayo, será necesario realizar cambios en el diseño, volver a fabricarlo y volver a someterlo a ensayo, y así sucesivamente hasta que el ensayo se supere con éxito. Dado el alto coste y el tiempo que supone cada una de estas iteraciones, es conveniente tomar todo tipo de medidas para reducir la radiación en el PCB desde el principio.

La forma de trabajar consistirá en entender, de forma general, cuáles son las causas más comunes que aumentan la radiación en el PCB y aplicar desde el comienzo del desarrollo una serie de reglas o buenas prácticas para tratar de reducirla todo lo posible. Hasta que no se fabrique el PCB y se lleve a laboratorio no será posible cuantificar esta radiación y asegurar que está por debajo de los límites, pero sí que habrá muchas más posibilidades de éxito en el primer intento.

7.5.1. Radiación debida a las pistas

Todas las pistas del PCB, así como las interconexiones dentro de los circuitos integrados, radiarán irremediablemente. Se pueden diferenciar dos tipos de radiación en función del rutado de la pista: radiación por modo diferencial o por modo común.

7.5.1.1. Radiación por modo diferencial

La radiación por modo diferencial no hace referencia a los pares diferenciales sino al hecho de que la corriente de retorno vuelva por un camino paralelo y opuesto al de la corriente de ida. Es lo que se consigue cuando se dispone de un plano de referencia continuo y cercano a la pista. De esta forma ambos caminos radian en sentido opuesto y se cancelan parcialmente entre ellos, reduciendo así la radiación total generada.

La radiación generada por una pista por modo diferencial se puede estimar utilizando la siguiente expresión [30]:

$$E_d = 1{,}316 \cdot 10^{-14} \cdot \frac{I \cdot f^2 \cdot L \cdot s}{d}$$

Donde:

- E_d = Campo eléctrico en la dirección de máxima radiación expresado en V/m
- d = Distancia desde la que se mide el campo eléctrico expresada en m
- I = Corriente expresada en A
- f = Frecuencia de la señal expresada en Hz
- L = Longitud de la pista expresada en m
- s = Distancia entre el camino de ida y el de retorno expresada en m

Por lo general, la radiación por modo diferencial no se puede evitar completamente, pero suele ser bastante pequeña por lo que normalmente no será problemática. La radiación por modo diferencial suele ser menor y por tanto preferible a la radiación por modo común. Atendiendo a la fórmula, se puede tratar de minimizarla:

- Reduciendo la corriente que circula por la pista tanto como sea posible.
- Reduciendo la frecuencia de la señal o aumentando la duración de los flancos. Será recomendable usar una frecuencia tan lenta como permita la aplicación, así como relajar los flancos utilizando drivers tan lentos como sea posible.
- Reduciendo el área del bucle, es decir, reduciendo la longitud de la pista y acercando entre sí los caminos de ida y retorno. Será recomendable por tanto usar stackups en los que la pista de ida esté lo más cerca posible del plano de retorno. Se deberá tratar de evitar especialmente el uso de stackups de 2 capas en los que los caminos de ida y retorno estarán muy separados, aumentando en gran medida el área de bucle o provocando la conversión a modo común.
- Mediante apantallamiento, rutando las señales más críticas en capas internas.

7.5.1.2. Radiación por modo común

La radiación por modo común ocurre cuando, debido a una asimetría, en alguna sección de la línea la corriente de retorno no se corresponde con el opuesto de la de ida, sino que ambas tienen el mismo sentido y por tanto la radiación que generan se suma en lugar de cancelarse. Es lo que ocurre cuando, por ejemplo, hay alguna discontinuidad en el plano de retorno que provoca que la corriente de vuelta recorra un camino más largo que la de ida, provocando que ambas dejen de estar en fase.

La radiación generada por una pista por modo común se puede estimar utilizando la siguiente expresión [30]:

$$E_c = 1{,}257 \cdot 10^{-6} \cdot \frac{I \cdot f \cdot L}{d}$$

Donde:

- E_c = Campo eléctrico en la dirección de máxima radiación expresado en V/m
- d = Distancia desde la que se mide el campo eléctrico expresada en m
- I = Corriente expresada en A
- f = Frecuencia de la señal expresada en Hz
- L = Longitud de la pista expresada en m

Se puede comprobar que a igualdad de corriente y frecuencia, la radiación provocada por el modo común es considerablemente mayor a la provocada por el modo diferencial. Será conveniente por tanto tratar de rutar todas las pistas en modo diferencial, evitando en medida de lo posible cualquier conversión al modo común.

Para reducir la radiación generada por modo común, será recomendable aplicar las mismas medidas que se pueden utilizar para reducir la radiación por modo diferencial, tales como reducir la corriente, la frecuencia y la longitud de la pista.

También cabe destacar que las medidas tomadas para mejorar la integridad de la señal y reducir las reflexiones, también tendrán un efecto beneficioso a la hora de reducir la radiación del PCB, por lo que aplicar estas reglas será muy conveniente:

- En términos de EMC, evitando las discontinuidades en los caminos de retorno se reduce el área de bucle y por tanto la radiación en modo diferencial, así como la conversión a modo común, reduciendo por tanto la radiación total en el PCB.
- En términos de integridad de señal, evitando las discontinuidades en los caminos de retorno se reducen los cambios bruscos en la impedancia de la línea y por tanto las reflexiones. Al reducirse las reflexiones se está reduciendo el contenido de alta frecuencia de la señal, lo que también contribuye a reducir la radiación.

7.5.2. Radiación debida a los cables

Los cables que interconecten la placa con otros elementos radiarán siguiendo los mismos principios que las pistas, por modo diferencial o modo común. La radiación de los cables suele ser más peligrosa que la de las pistas, debido a su gran longitud.

Para reducir la radiación de los cables, se aplicarán medidas similares a las que se utilizan para reducir la radiación de las pistas:

- Reducir la corriente y frecuencia de la señal. Es habitual utilizar filtros en los extremos de los cables para atenuar las componentes de alta frecuencia de la corriente que circula por ellos, reduciendo así la radiación que emite el cable, así como evitando que la radiación exterior que se acopla al cable afecte a la placa.
- Reducir el área de bucle utilizando cables tan cortos como sea posible y acercando los caminos de ida y retorno. En los cables paralelos es recomendable añadir varias líneas de masa intercaladas con las líneas de señal, idealmente un cable de masa por cada cable de señal. Así se consigue reducir el área de los bucles y por tanto la radiación, así como el crosstalk al evitar el solapamiento de los bucles.
- Utilizar cables apantallados conectados a masa en ambos extremos.

7.5.3. Radiación debida a antenas parásitas

Algunas estructuras en el PCB pueden comportarse como antenas parásitas que radien de forma no intencionada. Algunos ejemplos de antenas parásitas son [2]:

- Áreas de cobre flotantes o conectadas a masa en un único punto, que pueden comportarse como antenas de tipo parche. Será necesario conectar estas áreas a masa utilizando múltiples vías, separadas una distancia inferior a $\lambda/10$.
- Otros elementos conductores tales como los disipadores, que no estén conectados a masa o estén conectados en un único punto.
- Tarjetas de expansión conectadas a la placa principal. En este caso es recomendable añadir muchos pines de masa a los conectores para reducir la inductancia entre las masas de ambas placas, reduciendo así la radiación.

7.5.4. Radiación debida a los bordes del PCB

Los planos de alimentación y masa del PCB pueden comportarse como una guía biplaca, propagando el ruido de estos planos hasta que es radiado al exterior por los bordes del PCB. Algunas formas de reducir esta radiación son [2]:

- Reducir el ruido producido por la conmutación de los circuitos digitales en estos planos, añadiendo condensadores para mejorar la red de desacoplo.
- Aplicar la regla *20H*: Consiste en reducir el tamaño del plano de alimentación de forma que sea más pequeño que el de masa, manteniendo una distancia de 20H entre los bordes de cada plano, siendo H la separación entre ambos planos.
- Aplicar el *via stitching*: Consiste en unir los planos de masa superior e inferior por medio de vías alrededor de todo el PCB. De esta forma se consigue confinar el campo en el interior del PCB, sin embargo, puede aumentar el crosstalk.

7.6. Inmunidad a ESD

Como se ha visto anteriormente, la radiación en el PCB es difícil de cuantificar sin realizar físicamente un ensayo. Sin embargo, la inmunidad a ESD es más sencillo simularla y de forma bastante certera. El objetivo de este apartado es simular las protecciones frente a transitorios ESD para cada uno de los conectores que podrían ser tocados directamente por el usuario durante el funcionamiento normal del dispositivo.

7.6.1. Conector USB

7.6.1.1. Líneas de alimentación

Las líneas de 5V de los conectores USB van conectadas a las entradas de los reguladores TLV62569, será necesario comprobar si estos reguladores soportan los pulsos ESD que indica la normativa. En este caso, el datasheet indica que cualquiera de sus pines es capaz de soportar ±2000V según el *Human Body Model* (HBM).

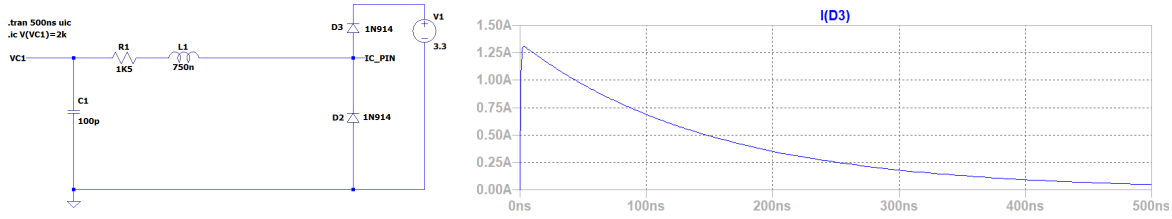


Figura 7.4: Curva de la corriente para HBM 2kV soportada por los reguladores.

Por tanto, el objetivo será comprobar si con las protecciones que lleva actualmente la placa la corriente se mantiene por debajo de dicha curva, y en caso contrario, añadir las protecciones que sean necesarias para conseguirlo.

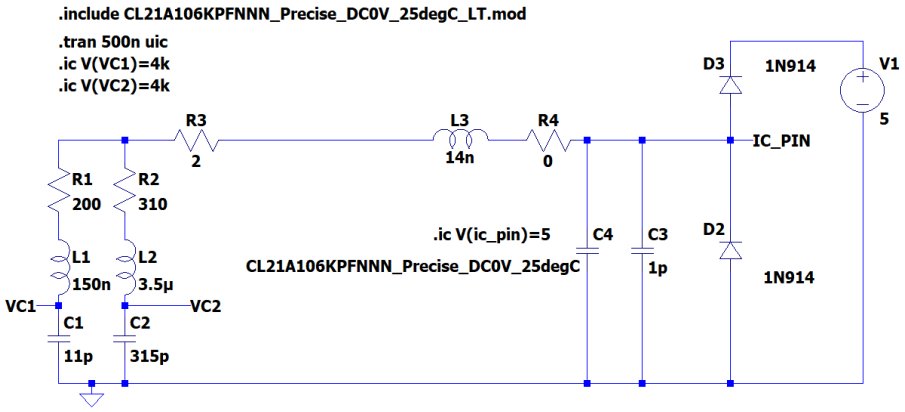


Figura 7.5: Simulación del pulso IEC 61000-4-2 para 4kV con un condensador 805.

Se han realizado 4 simulaciones, según estuviese el condensador inicialmente descargado o cargado, y según el pulso de ESD fuese positivo o negativo. Si el pulso es positivo, las peores condiciones se obtienen con el condensador cargado, mientras que si es negativo, las peores condiciones se obtienen con el condensador descargado. A continuación se muestran los resultados para ambos casos:

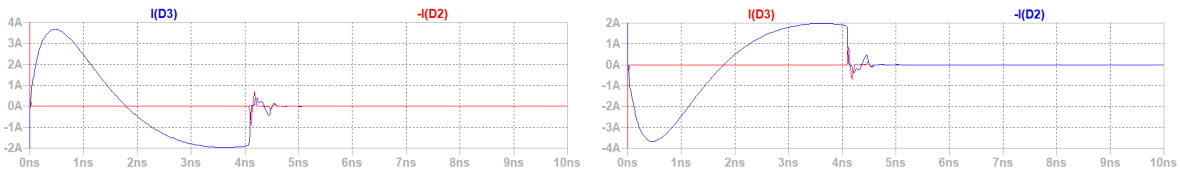


Figura 7.6: Curvas de la corriente en el pin del regulador para pulso IEC 61000-4-2.

Durante la mayor parte del tiempo que duran los transitorios ESD podemos observar que los valores están dentro de los límites recomendados de operación. Hay momentos en los que los límites se superan, pero se superan por poco y durante apenas un nanosegundo, por lo que podemos concluir que estos pulsos no serán problemáticos y el dispositivo podría pasar el ensayo sin dañarse con mucha probabilidad.

De aquí también podemos concluir que el uso de un condensador cerámico (con baja ESR) puede ser una buena opción de cara a mitigar los efectos de la ESD, siempre y cuando la señal sea suficientemente lenta para poder colocar un condensador suficientemente grande. En este caso es bastante adecuado ya que se trata de una línea de alimentación que queremos mantener lo más estable posible, para lo que ponemos varios condensadores de gran capacidad. Estos mismos condensadores son capaces de absorber los pulsos de ESD manteniendo en la entrada valores bastante razonables.

7.6.1.2. Líneas de datos

En las líneas de datos, el datasheet del SoC también indica que como máximo soporta un pulso de ±2000V según el HBM. En caso de no colocar ninguna protección adicional, el pulso IEC de 4kV aplicado directamente es claramente muy superior al HBM de 2kV, lo que podría provocar que se dañasen estos puertos del SoC.

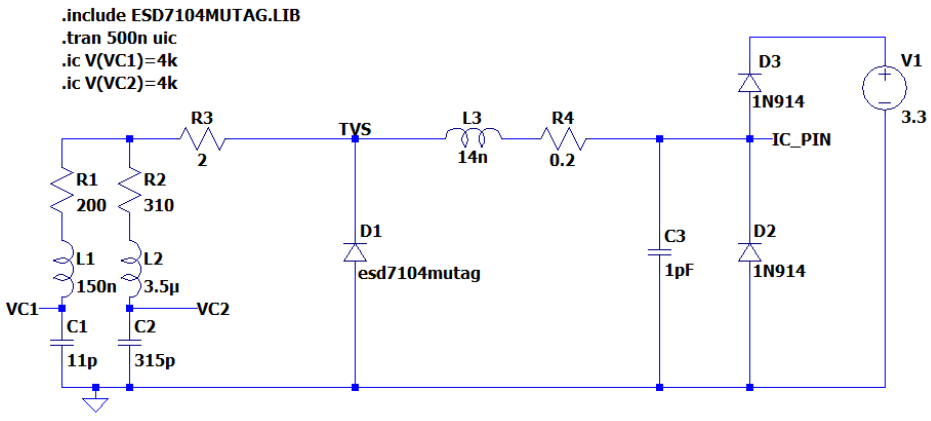


Figura 7.7: Simulación del pulso IEC 61000-4-2 para 4kV con el TVS ESD7104MUTAG.

Como protección adicional se ha colocado un diodo TVS ESD7104MUTAG, obteniendo los siguientes resultados para pulsos positivos o negativos:

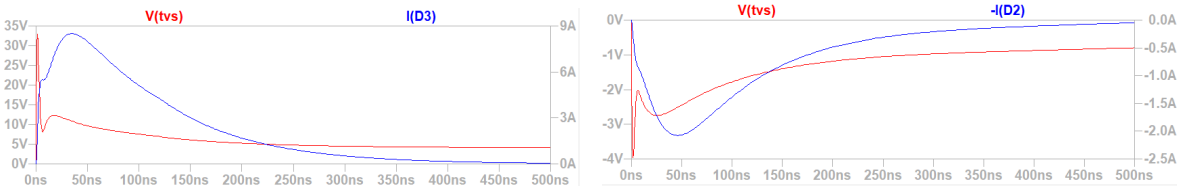


Figura 7.8: Curvas de corriente en el SoC y tensión en el TVS con el ESD7104MUTAG.

Se observa que únicamente con el ESD7104MUTAG utilizado los resultados no son suficientemente buenos. Esto es debido a que el TVS no conduce lo suficiente antes de los 5V, lo que hace que la corriente en el pin del SoC siga estando por encima de la soportada. Una posible solución sería incorporar una resistencia o ferrita tras el diodo TVS, pero es una solución delicada en este caso ya que podría tener un efecto indeseado en la impedancia de la pista USB la cual es de alta velocidad.

Otra opción es utilizar un TVS de menor tensión de trabajo como el ESD8004MUTAG el cual comienza a conducir a partir de 3.3V. Este TVS no se podría utilizar en la línea de alimentación ya que esta funciona a 5V, pero sí que podría utilizarse en las líneas de datos ya que según el estándar USB este par diferencial trabaja entre 0 y 3.3V. Sin embargo, con esta solución en caso de que algún dispositivo no respete completamente el estándar el diodo TVS podría conducir de manera indeseada.

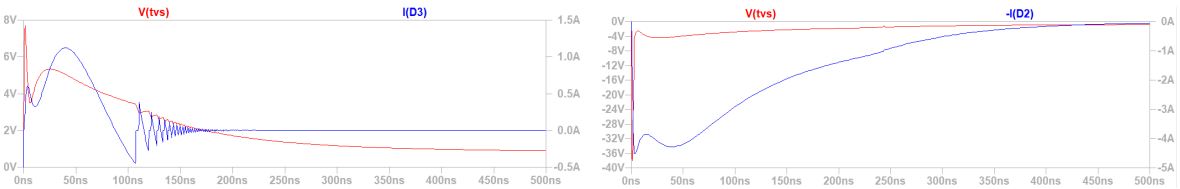


Figura 7.9: Curvas de corriente en el SoC y tensión en el TVS con el ESD8004MUTAG.

Se observa que únicamente con el ESD8004MUTAG se consigue reducir la corriente en el pin del SoC considerablemente en el pulso positivo, quedando por debajo de la soportada por el SoC según la curva HBM 2kV. En cambio, con el pulso negativo la corriente se ve incrementada respecto al ESD7104MUTAG quedando por encima del HBM. Sería recomendable añadir una resistencia o ferrita para reducir esta corriente.

7.6.2. Conector microSD

En el caso de la microSD, en la línea de 3.3V no se dispone de TVS pero sí de condensadores cerámicos muy cerca del conector. Las mismas simulaciones realizadas para el USB también le serían de aplicación. En las líneas de datos se consiguen buenos resultados con el ESD7104MUTAG debido a las resistencias de 33Ω que se colocaron en estas líneas para mejorar la integridad de señal. También se podría utilizar el ESD8004MUTAG ya que la tensión de funcionamiento de estas líneas es de 3.3V.

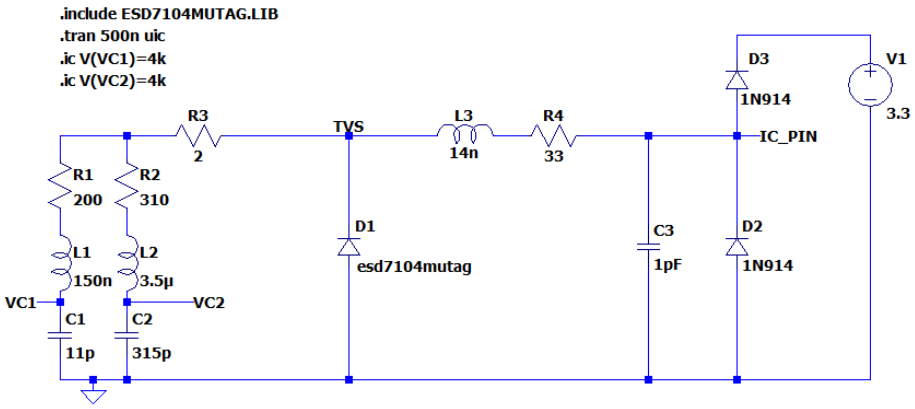


Figura 7.10: Simulación del pulso IEC 61000-4-2 para 4kV en el conector microSD.

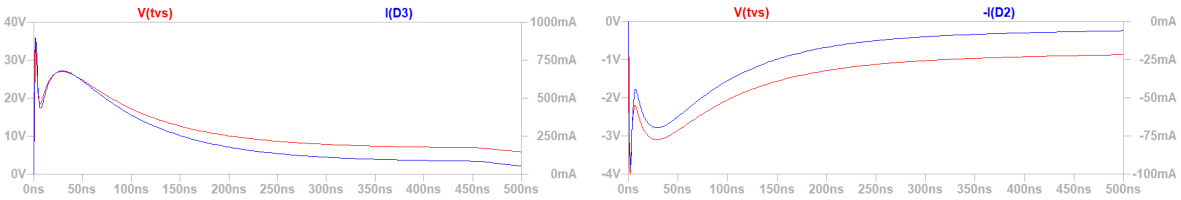


Figura 7.11: Curvas de corriente en el SoC y tensión en el TVS con el ESD7104MUTAG.

7.6.3. Conector CAN

Los transceivers CAN son por lo general bastante robustos dadas las aplicaciones típicas a las que se destinan este tipo de buses. En este caso el transceiver SN65HVD230 utilizado soporta pulsos de ±16000V según el Human Body Model. Se observa que esta curva es similar o inferior a la del pulso IEC 61000-4-2 para 4kV, por lo que la protección incluida en el transceiver ya es bastante adecuada. El pulso inicial de la curva IEC sí que supera los valores de corriente de la curva HBM, pero la duración de este pulso es inferior a unos 5ns por lo no se espera que sea problemática, siendo el resto del pulso el que contiene la mayor parte de la energía a disipar.

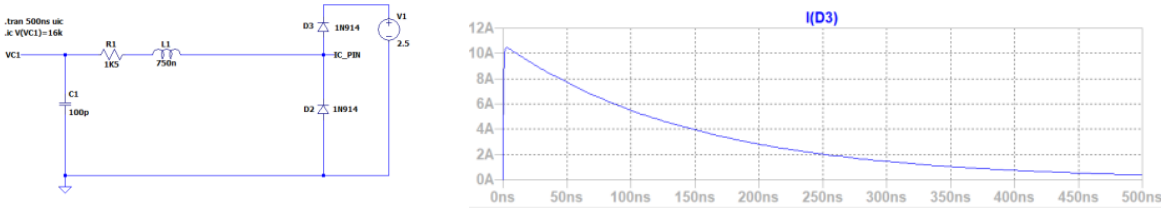


Figura 7.12: Curva de la corriente para HBM 16kV soportada por el transceiver CAN.

7.6.4. Conector Ethernet

El datasheet del transceiver LAN8720A utilizado indica que soporta pulsos IEC 61000-4-2 para ±8kV, por lo que las protecciones integradas en el propio transceiver son más que suficientes, no será necesaria ninguna protección adicional. En la práctica, los magnetics incorporados en el conector también ayudarán a atenuar la curva que llega al transceiver, aumentando aún más la protección ante este tipo de transitorios.

7.6.5. Headers de expansión

Los pines de los headers de expansión solo soportan pulsos HBM de 2kV, al igual que el resto de pines del SoC. En estos pines no se dispone de ningún tipo de protección adicional, por lo fallarían el ensayo con total seguridad. Será necesario indicar que estos pines son sensibles a las descargas ESD o ensayar el producto dentro de una envolvente de plástico con la que estos pines no queden expuestos.

Parte III

FABRICACIÓN Y TESTEO

Capítulo 8

Ensamblado y bring-up del PCB

8.1. Introducción

El objetivo de este capítulo consiste en llevar a cabo la fabricación del PCB, así como el posterior ensamblado de los componentes. También se realizarán unas primeras pruebas para asegurarse de que el diseño funciona correctamente, al menos hasta cierto punto, antes de proceder a la instalación de Linux en el siguiente capítulo.

8.2. Pedido del PCB y stencil

8.2.0.1. Especificaciones del PCB

La fabricación tanto del PCB como del *stencil* se ha encargado al fabricante asiático JLCPCB. Tal y como se decidió en el capítulo 5, el PCB se ha pedido con la impedancia controlada con el stackup JLC2313 de 4 capas y 1.2mm de espesor.

JLCPCB ofrece dos calidades de materiales con distinta temperatura de transición vítrea (Tg), uno de 130-140°C y otro de 155°C. En línea con las recomendaciones del fabricante, como en este caso el espacio entre las pistas es muy pequeño y se usarán altas temperaturas debido a la soldadura sin plomo, se ha elegido el material de 155°C para reducir posibles defectos por deformaciones en el PCB.

También ofrecen distintos acabados para la superficie tales como HASL o ENIG. Aunque el HASL es más económico, se ha elegido el ENIG ya que consigue un acabado más plano de las superficies, lo que reduce los defectos con componentes BGA y QFN de bajo pitch. Con ENIG también es mayor la vida útil del acabado por lo que será posible almacenar durante más tiempo los PCBs antes de ensamblarlos.

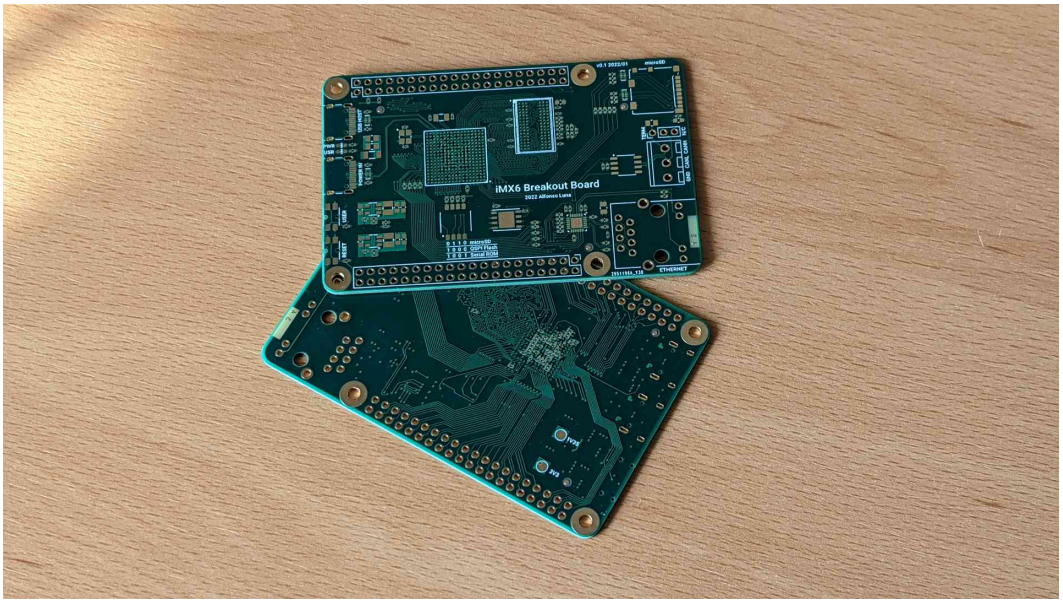


Figura 8.1: PCBs recibidos desde el fabricante JLCPCB.

En cuanto a la máscara de soldadura se ha elegido el color verde ya que es el que permite un menor espaciado entre los pads de los componentes, de 0.2mm en este caso. Con el resto de colores es necesario mantener una distancia entre pads de al menos 0.25mm para que permanezca la máscara de soldadura, lo que era insuficiente para algunos de los componentes del diseño. En caso de eliminar la máscara de soldadura entre pads, podrían aumentar también los defectos al unirse los pads con la soldadura.

8.2.0.2. Especificaciones del stencil

El stencil también se ha pedido en JLCPCB utilizando la capa paste de los gerbers y con 0.1mm de espesor. El stencil se ha pedido con electropulido ya que de esta forma se consigue un mejor acabado en el corte de los orificios lo que será de ayuda al aplicar la pasta de soldadura en los pads de tamaño más pequeño.

8.3. Ensamblado de los componentes

8.3.1. Componentes SMD de la cara superior

8.3.1.1. Aplicación de la pasta de soldadura

El primer paso ha consistido en aplicar la pasta de soldadura con ayuda del stencil y una espátula. En este paso lo más crítico es aplicar la cantidad exacta de pasta, si se aplica demasiada o demasiada poca podrían ocurrir defectos en la fabricación.

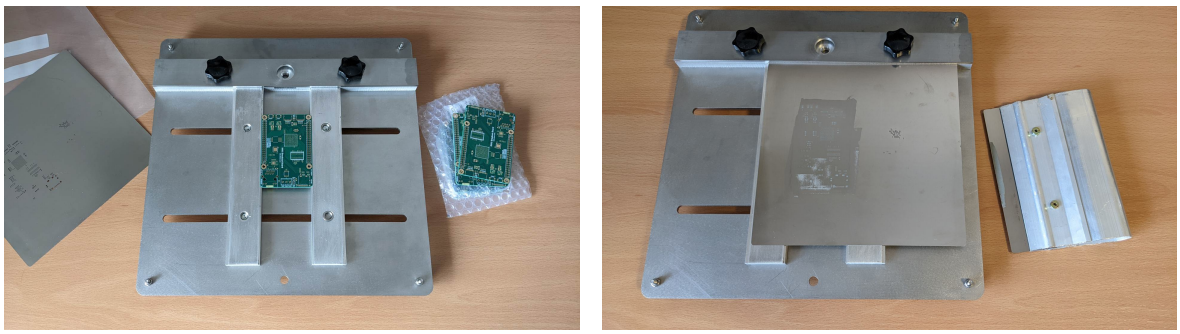


Figura 8.2: Aplicación de la pasta de soldadura mediante stencil.

La cantidad de pasta se puede controlar con el espesor del stencil y la apertura de los pads en la capa paste. En este caso se ha observado que en los pads 603 había demasiada pasta, por lo que habría sido conveniente reducir la apertura en dichos pads.

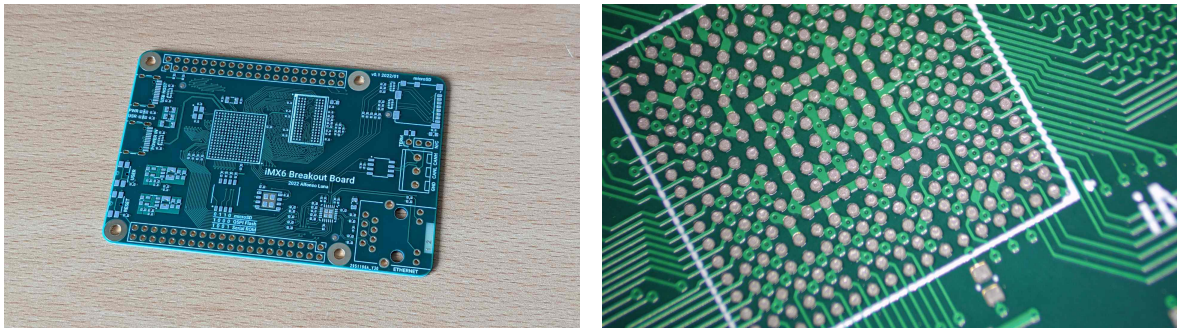


Figura 8.3: Detalle de la pasta de soldadura una vez aplicada.

8.3.1.2. Colocación de los componentes

El siguiente paso consiste en colocar todos los componentes sobre sus correspondientes huellas. En una línea automatizada lo habitual sería realizar este paso con una máquina *Pick and Place*. En este caso los componentes se han tenido que colocar de manera manual, bajo un microscopio y con la ayuda de unas pinzas.

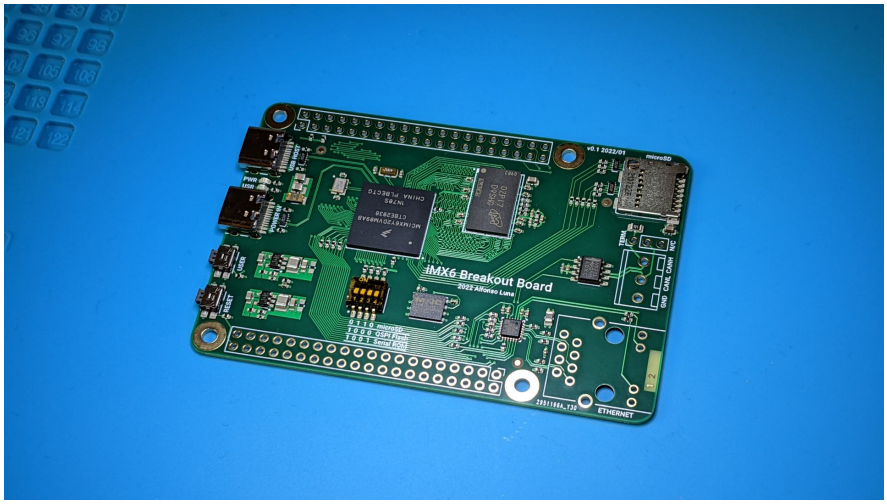


Figura 8.4: PCB tras colocar manualmente todos los componentes.

8.3.1.3. Soldadura por reflow

Una vez colocados todos los componentes, se ha realizado la soldadura por *reflow*. Para la soldadura se ha utilizado una cama caliente junto con un controlador que permite que la cama siga la curva de temperatura especificada. Para la soldadura de la cara superior se ha utilizado pasta sin plomo de composición Sn96.5/Ag3.0/Cu0.5, ya que es la misma que utilizan las bolas de los encapsulados BGA.

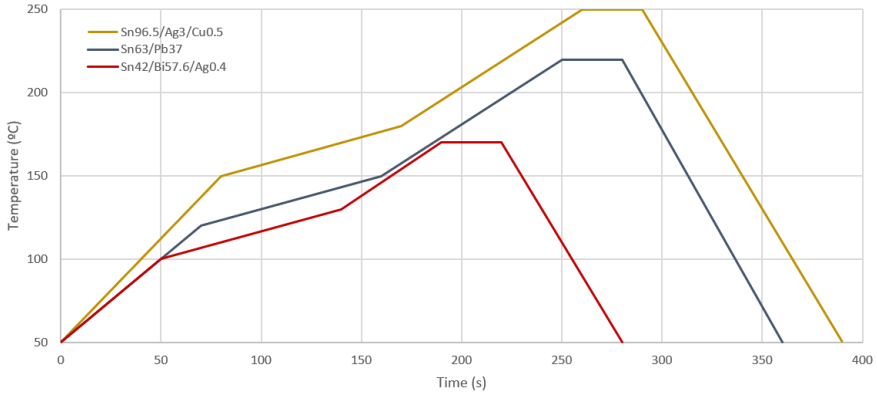


Figura 8.5: Curvas de temperatura configuradas en el controlador.

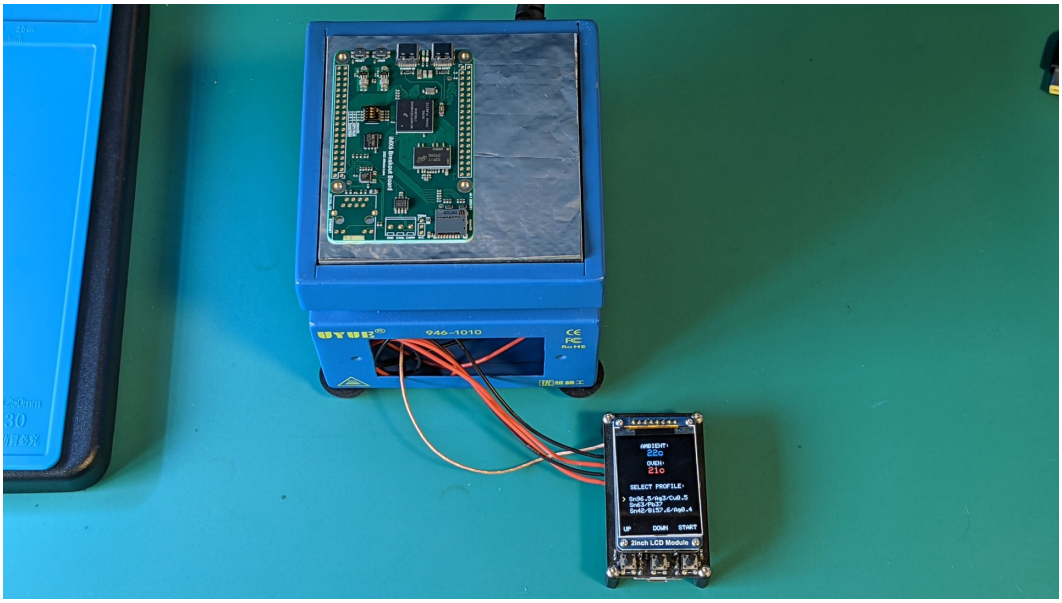


Figura 8.6: Soldadura por reflow mediante cama caliente.

8.3.1.4. Inspección visual

Una vez completado el ciclo de reflow, se ha comprobado con ayuda de un microscopio que no hubiese puentes ni otros defectos de soldadura.

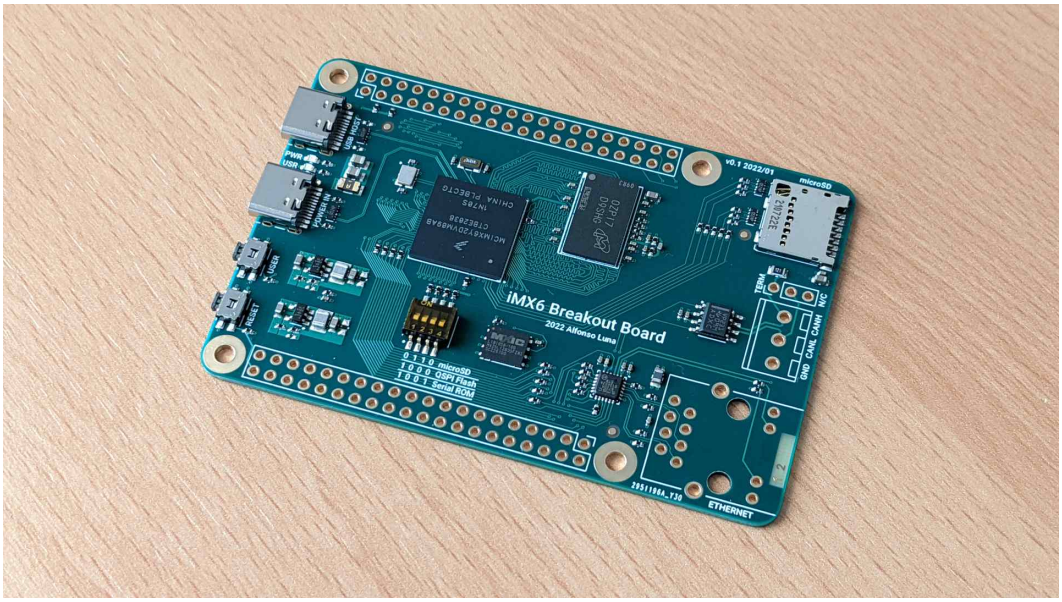


Figura 8.7: PCB tras soldar los componentes SMD de la cara superior.

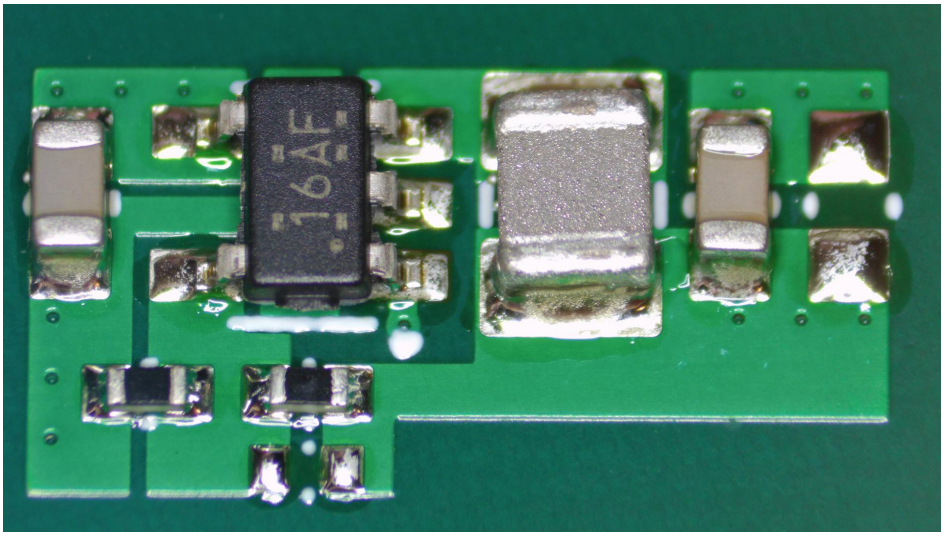


Figura 8.8: Detalle de la soldadura de uno de los reguladores.

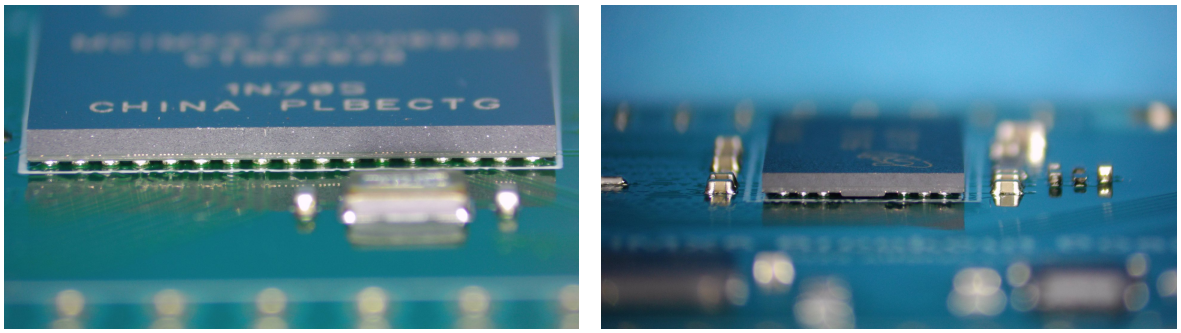


Figura 8.9: Detalle de la soldadura de los componentes BGA.

8.3.2. Componentes SMD de la cara inferior

Para soldar la cara inferior se ha repetido de nuevo el proceso pero con el lado contrario. En este caso la soldadura se ha realizado con una pistola de aire caliente, ya que no era posible utilizar la cama caliente al tener componentes por el otro lado.

En una línea automatizada con un horno lo ideal sería realizar el proceso a la inversa, primero se soldarían los componentes de la cara inferior, y posteriormente los de la superior. Durante la soldadura de la cara superior, los componentes de la cara inferior al ser muy ligeros se mantendrían sujetos al PCB por la tensión superficial.

En este caso se ha tenido que soldar primero la cara superior. Para soldar la cara inferior se ha utilizado pasta sin plomo de composición Sn42/Bi57.6/Ag0.4, la cual tiene un punto de fusión considerablemente menor que la usada para la cara superior. De esta forma ha sido posible soldar los componentes de la cara inferior con la pistola de aire caliente sin riesgo de que se desoldasen los componentes de la otra cara.

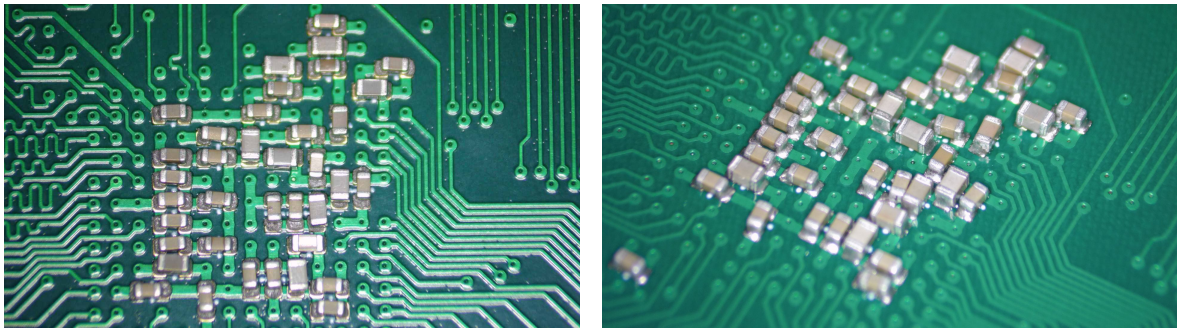


Figura 8.10: Soldadura de los componentes de la cara inferior.

8.3.3. Componentes THT o de agujero pasante

Por último, se han soldado manualmente el resto de componentes de agujero pasante tales como los conectores del puerto Ethernet y puerto CAN.

8.4. Bring-up del PCB

8.4.1. Testeo de los reguladores

El primer paso ha consistido en comprobar el correcto funcionamiento de los reguladores. Tras comprobar con un multímetro que no habían cortocircuitos en las líneas de alimentación, se ha procedido a alimentar la placa con una fuente de laboratorio, comprobando que los valores de todas las tensiones eran correctos.

Se ha limitado la tensión y la corriente de la fuente de laboratorio a lo mínimo necesario para que la placa arranque, en este caso la placa se ha alimentado a 4V y 100mA. El objetivo es que si hay algún defecto en la placa (como un cortocircuito o un componente colocado con polaridad u orientación incorrecta), que no circule una corriente excesiva por el PCB que podría dañar las pistas o algunos de los componentes.

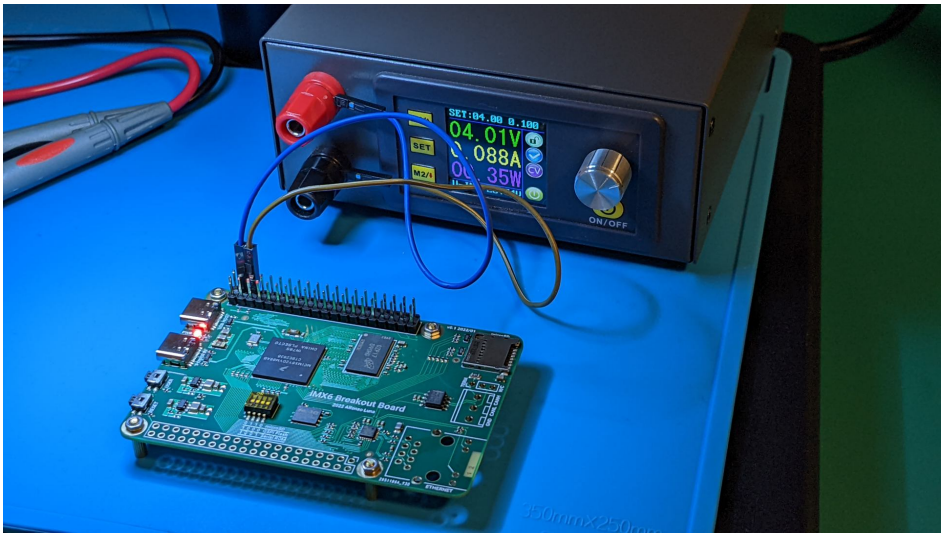
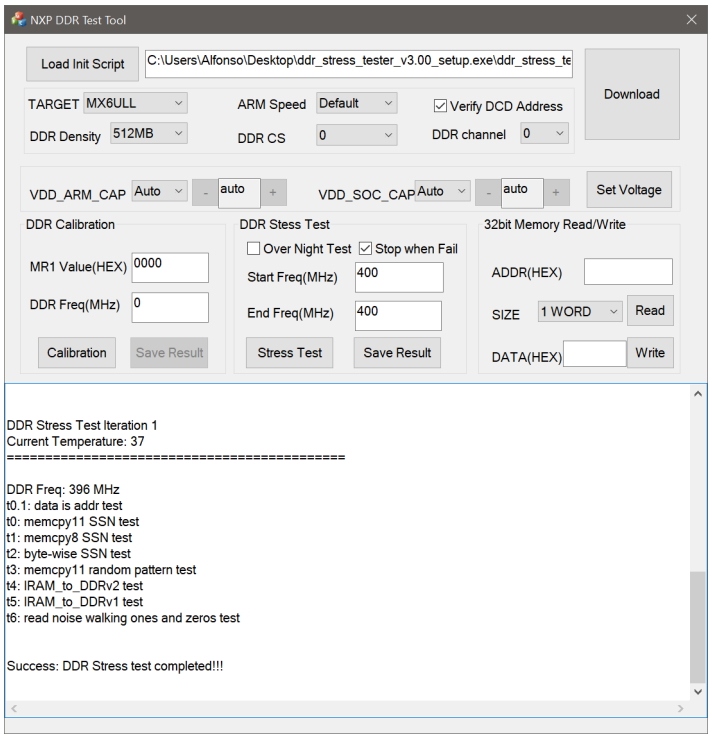


Figura 8.11: Alimentación de la placa mediante la fuente de laboratorio.

8.4.2. Testeo del procesador y RAM

Una vez comprobada la alimentación, se ha conectado la placa a un PC por USB, observando que la Boot ROM del SoC arrancaba y aparecía como un nuevo dispositivo USB en el PC. De esta forma ha sido posible cargarle al SoC un programa de prueba que proporciona NXP. Así ha sido posible comprobar de forma rápida y sencilla que tanto el SoC como la memoria RAM funcionaban correctamente.



Capítulo 9

Instalación de Linux embebido

9.1. Introducción

Los Single-Board Computer a menudo se utilizan con una distribución de Linux embebido como sistema operativo. El objetivo de este capítulo consiste en realizar una instalación de Linux en la placa, realizando las configuraciones y modificaciones necesarias tanto al bootloader como al kernel, así como comprobando posteriormente que todos los periféricos funcionan correctamente y son accesibles desde Linux.

9.1.1. Proceso de arranque

9.1.1.1. Boot ROM

El proceso de arranque comienza tras un *Power-On Reset* (POR) con el que el hardware reinicia toda la lógica del SoC y el microprocesador comienza a ejecutar un programa llamado *Boot ROM*. Esta Boot ROM es un programa que viene de fábrica dentro del SoC, almacenado en una memoria de solo lectura.

La Boot ROM se encarga de realizar la inicialización más básica del SoC, para a continuación cargar otro programa (en este caso será U-Boot) desde alguno de los medios de almacenamiento soportados, al que posteriormente cede el control.

El dispositivo desde el que la Boot ROM carga el programa se puede seleccionar y configurar en función del estado de algunas entradas GPIO durante el arranque. También se puede grabar en el SoC una configuración en lugar de utilizar las entradas GPIO. En el caso de esta placa, se dispone de un interruptor con el que se puede elegir entre tres dispositivos de arranque: la tarjeta microSD, la memoria NOR Flash, o un modo *Serial Downloader* de descarga por el puerto serie o USB. La configuración completa de la Boot ROM se indica en la página 13 del esquemático.

En el caso de seleccionar el arranque desde la microSD o la NOR Flash, la Boot ROM se encargará de configurar los periféricos correspondientes y de copiar el binario de U-Boot desde la memoria seleccionada a la memoria RAM, para posteriormente cederle el control. En caso de seleccionar el modo *Serial Downloader* o por fallo de otro modo, este binario puede enviarse al SoC por medio del puerto serie o puerto USB.

El binario completo de U-Boot no cabe en la memoria SRAM del SoC, que es de solo 128KB, por lo que es necesario copiarlo a la memoria RAM DDR3 antes de poder ejecutarlo. Esto significa que, antes de copiar el binario de U-Boot a la memoria DDR3, la Boot ROM debe realizar la inicialización del controlador de memoria DDRx en función del tipo de memoria de la que se disponga (DDR2, DDR3...) y de la configuración necesaria, incluyendo la configuración de las resistencias ODT.

En la cabecera al comienzo del binario de U-Boot, se incluyen las tablas *Device Configuration Data* (DCD). En estas tablas se indica la configuración para el controlador de memoria DDRx. La Boot ROM es capaz de leer esta tabla e inicializar el controlador de memoria con los parámetros indicados en dicha tabla, antes de proceder a copiar el resto del binario de U-Boot a la memoria DDR3.

9.1.1.2. U-Boot

U-Boot es el bootloader más utilizado en este tipo de dispositivos. Una vez cargado y ejecutado por la Boot ROM, U-Boot puede realizar una configuración más completa del SoC y del resto de periféricos, y proporcionar una consola de comandos desde la que se puede realizar todo tipo de operaciones relacionadas con el arranque. Las operaciones a realizar en el arranque se pueden introducir manualmente en esta consola o automatizarse con un comando de arranque como se verá posteriormente.

Al contrario que la Boot ROM, que realiza una inicialización básica de la memoria y solo puede cargar el binario desde un sector determinado, U-Boot es capaz de acceder y navegar por el sistema de archivos de la memoria, ya sea FAT32 o ext4, entre otros. Incluso es capaz de conectarse a Internet y obtener el programa a ejecutar a través de la red, lo que puede ser muy conveniente durante el desarrollo.

En dispositivos en los que la funcionalidad de la Boot ROM es más limitada, U-Boot puede dividirse en dos partes. La primera parte (SPL) es una versión muy limitada de U-Boot de pequeño tamaño de manera que quepa en la memoria SRAM del SoC. El SPL realiza la inicialización del controlador de memoria RAM DDRx del SoC, para posteriormente copiar el resto de U-Boot a dicha memoria. En el caso del SoC iMX6ULL utilizado en esta placa, el SPL no es necesario, ya que esta configuración inicial de la memoria RAM DDR3 la realiza la Boot ROM en función de la tabla DCD.

Aunque también es posible utilizarlo para otros fines, U-Boot está especialmente preparado para el arranque del kernel Linux. U-Boot se encargará de copiar el binario del kernel desde la tarjeta microSD (aunque también podría realizarse desde la memoria NOR Flash o desde el Ethernet) a la memoria RAM. También copiará otros archivos adicionales necesarios por el kernel como el *device tree* o el *RAM Disk*.

Por último, una vez todos los binarios necesarios están en la memoria RAM, U-Boot cede el control al kernel, pudiendo además pasarle al kernel una lista de argumentos o parámetros que el kernel tendrá en cuenta durante el arranque.

9.1.1.3. Kernel Linux

Tras recibir el control por parte de U-Boot, el kernel comienza su proceso de arranque. Durante el arranque, el kernel se encarga de cargar e inicializar los drivers de todos los periféricos del SoC y de la placa según se le haya indicado en el device tree, entre otras tareas. Al final del proceso, el kernel accede al sistema de archivos de la unidad que U-Boot le ha pasado como argumento, y ejecuta el programa `/sbin/init`.

Este programa `/sbin/init` puede encontrarse directamente dentro del sistema de archivos de la microSD, o en el RAM Disk también cargado por U-Boot.

9.1.1.4. Proceso Init en el RAM Disk

El RAM Disk es utilizado por distribuciones tipo *Live CD* que funcionan únicamente desde la memoria RAM. Consiste en una imagen de un sistema de archivos que U-Boot copia junto al kernel a la memoria RAM. El kernel puede ejecutar el proceso `/sbin/init` de esta imagen, de forma que todo el sistema puede funcionar únicamente en la memoria RAM, sin necesidad de disponer de otro almacenamiento externo. Esto puede ser especialmente interesante en caso de no disponer de tarjeta microSD y querer utilizar la memoria NOR Flash como única memoria de almacenamiento.

El RAM Disk también puede ser de utilidad incluso cuando el sistema de archivos definitivo se encuentra en otra unidad. Es necesario en configuraciones exóticas, por ejemplo, en caso de que el `/sbin/init` definitivo se encuentre en una unidad cifrada o en un RAID configurado por software [31]. En estos casos, el kernel no es capaz de acceder al `/sbin/init` definitivo por sus propios medios, por lo que el RAM Disk se utiliza como un paso intermedio durante el arranque, e incluye todas las herramientas necesarias para configurar y preparar el acceso a dicha unidad.

U-Boot soporta RAM Disks con formato *initrd* o *initramfs*. La distribución que usaremos utiliza un RAM Disk initramfs únicamente como paso extra antes de llamar al `/sbin/init` definitivo que estará en una partición de la tarjeta microSD.

9.1.1.5. Proceso Init en la microSD

Por último, el kernel ejecuta el proceso `/sbin/init` definitivo. Este proceso es el primero que se lanza en el espacio de usuario, y se mantiene siempre en ejecución, siendo el último en detenerse al apagar el equipo. Normalmente se trata de un programa específico que se encarga de lanzar y gestionar el resto de servicios y aplicaciones del sistema, necesarias para disponer de un sistema Linux completamente funcional.

Entre los Init más populares se encuentran *systemd* y *OpenRC*. La distribución que usaremos utiliza systemd como Init y gestor de servicios. En sistemas embebidos, también es muy común utilizar *BusyBox* como alternativa mucho más ligera.

9.1.2. Prerrequisitos para la instalación

9.1.2.1. Prerrequisitos de hardware

El interruptor de la placa se ha configurado para el arranque se realice desde la tarjeta microSD. También se ha conectado externamente un conversor de puerto serie a USB conectado a los pines *UART1_TX* y *UART1_RX* de la placa. De esta forma será posible tener acceso en el PC a la consola tanto de U-Boot como de Linux.

Posteriormente, se podría configurar en Linux un puerto serie virtual en uno de los conectores USB, de forma que por medio de un único cable USB conectado al PC se podría tanto alimentar la placa como acceder a la consola de Linux.

9.1.2.2. Prerrequisitos de software

Para poder realizar en los siguientes pasos la configuración y compilación de U-Boot y de Linux, ha sido necesario realizar previamente una instalación de Ubuntu como sistema operativo en el PC. También ha sido necesario instalar los siguientes paquetes, con el fin de poder ejecutar todos los comandos utilizados posteriormente:

```
1 sudo apt update
2 sudo apt upgrade
3 sudo apt install bison device-tree-compiler flex gcc gcc-arm-
    linux-gnueabihf git gparted libncurses5-dev libssl-dev lzop
    make u-boot-tools
```

También ha sido necesario descargar la última versión del código fuente tanto de U-Boot como del kernel Linux, clonando ambos repositorios:

```
1 git clone -b v2022.01 https://github.com/u-boot/u-boot --depth
    =1
2 git clone -b v5.16 https://github.com/torvalds/linux --depth=1
```

9.1.3. Flujo de trabajo

Crear un port de U-Boot y de Linux para una nueva placa desde cero es complicado, ya que hay que crear una gran cantidad de archivos de configuración [32] [33]. Para facilitar este proceso, es habitual que el fabricante del SoC proporcione cierto soporte en este aspecto, suministrando parte de estos archivos o alguna herramienta para generarlos, lo que a menudo se conoce como el *Board Support Package* (BSP).

A menudo, los fabricantes de SoCs mantienen sus propios repositorios de U-Boot y de Linux, incluyendo drivers, parches y demás archivos de configuración necesarios para que el sistema operativo funcione correctamente con sus dispositivos.

Algunos fabricantes consiguen que estas modificaciones acaben incluyéndose en la rama principal de estos proyectos. Sin embargo, en otros casos esto no sucede, por lo que es necesario acudir a los repositorios parcheados del fabricante, especialmente si alguno de los drivers necesarios es de código cerrado. Esto suele ser un inconveniente, ya que a menudo estos repositorios modificados por los fabricantes no se van actualizando a las últimas versiones de U-Boot y de Linux, quedando rápidamente obsoletos.

El SoC iMX6ULL utilizado en este trabajo está soportado en los repositorios principales, tanto de U-Boot como de Linux, por lo que ha sido posible clonar y trabajar sobre la última versión de estos repositorios. No ha sido necesario por tanto descargar ninguna herramienta específica del fabricante ni usar repositorios parcheados.

A nivel de SoC, se dispone de buen soporte. Todos los drivers y archivos de configuración necesarios ya van incluidos en dichos repositorios. Todos estos archivos podrán reutilizarse tal cual, sin necesidad de hacer ningún cambio en ellos.

A nivel de placa, sí que será necesaria cierta personalización. En dichos repositorios se incluyen los archivos de configuración para distintas placas basadas en esta misma familia de SoCs, entre las que se incluye la placa de evaluación *MCIMX6ULL-EVK* de NXP, que es la que tiene más similitudes con la nuestra.

Para evitar crear todos estos archivos de configuración de cero para la nueva placa, se partirá de la configuración de la placa *MCIMX6ULL-EVK* y se realizarán los mínimos cambios en sus archivos para adaptarlos a la nuestra. En cuanto a los drivers, no será necesario crear ninguno, ya que el resto de periféricos incluidos en la placa también están bien soportados y ya disponen de drivers en U-Boot y en Linux.

9.2. Preparación del device tree

9.2.1. Introducción al device tree

En los PCs convencionales con procesadores x86 el arranque está muy estandarizado. La BIOS o UEFI se encarga de la configuración inicial, proporcionando una interfaz estandarizada al sistema operativo. Posteriormente el kernel, durante el arranque, es capaz de descubrir los periféricos de los que dispone el equipo, gracias a que están conectados mediante interfaces con auto descubrimiento tales como ACPI, PCI-Express o USB. Los periféricos conectados a este tipo de buses son capaces de identificarse a sí mismos, y de esta forma el kernel puede cargar y asignarles sus drivers correspondientes. De este modo, una misma compilación e instalación de Linux se puede utilizar en PCs con configuraciones de componentes muy distintas.

En los sistemas embebidos con procesadores ARM, sin embargo, no existe o no está tan extendida esta estandarización. El proceso de inicialización y arranque es distinto para cada fabricante y para cada SoC. Cada SoC y cada placa dispone de periféricos distintos, a menudo conectados mediante buses más simples tales como I2C o SPI que no permiten el auto descubrimiento, o directamente se encuentran mapeados a direcciones de memoria. Esto implica que el kernel por sus propios medios no sea capaz de detectar los periféricos disponibles. Es complicado por tanto reutilizar una misma distribución en distintos SBCs, siendo necesaria cierta personalización para cada placa.

Como solución a este problema se crearon los *device tree* o árboles de dispositivos. Un device tree consiste en una serie de archivos en los que, usando una sintaxis determinada, se describen los periféricos hardware de los que dispone el sistema, las relaciones entre ellos, y una indicación de qué driver de los incluidos en el kernel es compatible con cada periférico [34]. Haciendo uso del device tree, tanto U-Boot como Linux son capaces de reconocer, inicializar y utilizar todos los periféricos de los que dispone el sistema, tanto los que están incluidos dentro del mismo SoC como el resto de componentes disponibles en la placa.

Aunque tanto U-Boot como Linux utilizan el mismo formato de device tree, cada proyecto es independiente y mantiene en su repositorio device trees distintos para cada placa. A menudo estos device tree son muy similares, aunque los de U-Boot tienden a ser más sencillos, ya que U-Boot no necesita disponer de acceso a todo el hardware sino únicamente al implicado en el arranque del sistema. A continuación, se indican los cambios que ha sido necesario realizar en los device trees de la placa *MCIMX6ULL-EVK* para adaptarlos a las particularidades de la nuestra.

9.2.2. Modificación del device tree de U-Boot

En el archivo `u-boot/arch/arm/dts/imx6ull-14x14-evk.dts`, así como en el resto de archivos `.dtsi` incluidos en él, se encuentra el device tree de la placa EVK. En este archivo se ha modificado el nombre de la placa:

```
11      model = "iMX6 Breakout Board";
```

9.2.2.1. Tarjeta microSD

En el archivo `u-boot/arch/arm/dts/imx6ul-14x14-evk.dtsi` se encuentra el resto del device tree. Se han eliminado las siguientes líneas, que se encargaban en la placa EVK de habilitar o deshabilitar, así como de configurar el regulador de voltaje de la SD entre 3.3V y 1.8V. En este caso la tarjeta SD está siempre alimentada a 3.3V y el pin GPIO1_IO9 se utiliza para otros fines como entrada del pulsador, por lo que estas líneas no son necesarias:

```
28      reg_sd1_vmmc: regulator-sd1-vmmc {
29          compatible = "regulator-fixed";
30          regulator-name = "VSD_3V3";
31          regulator-min-microvolt = <3300000>;
32          regulator-max-microvolt = <3300000>;
33          gpio = <&gpio1 9 GPIO_ACTIVE_HIGH>;
34          enable-active-high;
35      };
```

```
487             MX6UL_PAD_UART1_RTS_B__GPIO1_IO19       0x17059 /*
    SD1 CD */
488             MX6UL_PAD_GPIO1_IO05__USDHC1_VSELECT    0x17059 /*
    SD1 VSELECT */
489             MX6UL_PAD_GPIO1_IO09__GPIO1_IO09        0x17059 /*
    SD1 RESET */
```

Al principio la tarjeta SD no era detectada por U-Boot, debido a que el pin *Card Detect* no estaba a nivel bajo. En este caso el conector de la tarjeta SD no dispone del pin Card Detect, por lo que ha sido necesario cambiar las siguientes líneas:

```
255 &usdhc1 {
256     pinctrl-names = "default", "state_100mhz", "state_200mhz";
257     pinctrl-0 = <&pinctrl_usdhc1>;
258     pinctrl-1 = <&pinctrl_usdhc1_100mhz>;
259     pinctrl-2 = <&pinctrl_usdhc1_200mhz>;
260     cd-gpios = <&gpio1 19 GPIO_ACTIVE_LOW>;
261     keep-power-in-suspend;
262     wakeup-source;
263     vmmc-supply = <&reg_sd1_vmmc>;
264     status = "okay";
265 };
```

A por las siguientes:

```
218 &usdhc1 {
219     pinctrl-names = "default", "state_100mhz", "state_200mhz";
220     pinctrl-0 = <&pinctrl_usdhc1>;
221     pinctrl-1 = <&pinctrl_usdhc1_100mhz>;
222     pinctrl-2 = <&pinctrl_usdhc1_200mhz>;
223     no-1-8-v;
224     broken-cd;
225     keep-power-in-suspend;
226     wakeup-source;
227     status = "okay";
228 };
```

Tras este cambio U-Boot ya no detecta la tarjeta comprobando si el pin Card Detect está a nivel bajo, sino que lo hace por polling de la interfaz.

9.2.2.2. Memoria QSPI NOR Flash

La documentación del driver SPI NOR de Linux se encuentra en el archivo `linux/Documentation/devicetree/bindings/mtd/jedec,spi-nor.yaml`. Además, en el archivo `linux/drivers/mtd/spi-nor/macronix.c` se encuentra el código del driver específico para las memorias del fabricante Macronix.

Consultando estos archivos ha sido posible averiguar los modelos de memoria Flash soportados por dicho driver. La memoria Macronix MX25L12835F de la que se dispone en la placa no está soportada expresamente, sin embargo, sí que está soportada la memoria Macronix MX25L12805D que es muy similar.

En el device tree, se han cambiado las siguientes líneas:

```
183 &qspi {
184     pinctrl-names = "default";
185     pinctrl-0 = <&pinctrl_qspi>;
186     status = "okay";
187 
188     flash0: n25q256a@0 {
189         #address-cells = <1>;
190         #size-cells = <1>;
191         compatible = "micron,n25q256a";
192         spi-max-frequency = <29000000>;
193         spi-rx-bus-width = <4>;
194         spi-tx-bus-width = <4>;
195         reg = <0>;
196     };
197 };
```

A por las siguientes:

```
&qspi {
    pinctrl-names = "default";
    pinctrl-0 = <&pinctrl_qspi>;
    status = "okay";

    flash0: mx25l12805d@0 {
        #address-cells = <1>;
        #size-cells = <1>;
        compatible = "macronix,mx25l12805d", "jedec,spi-nor";
        spi-max-frequency = <133000000>;
        spi-rx-bus-width = <4>;
        spi-tx-bus-width = <4>;
        reg = <0>;
    };
};
```

El modelo *jedec,spi-nor* se utiliza como *fallback*. Las memorias NOR Flash disponen de un registro en el que se indica de forma estandarizada el ID del fabricante, tipo de memoria y densidad. De esta forma, el driver es capaz de leer dicho registro y conocer el modelo y características de la memoria en función de estos valores. La memoria MX25L12835F utilizada tiene el mismo ID (0xC22018) que la memoria MX25L12805D que soporta el driver, lo que normalmente indica que son compatibles.

9.2.2.3. Transceiver Ethernet

El transceiver Ethernet también es distinto en ambas placas. Se han eliminado las líneas relativas al segundo controlador. Las relativas al primer controlador se han modificado, añadiendo las señales de interrupción y reset, entre otras modificaciones:

```
&fec1 {
    pinctrl-names = "default";
    pinctrl-0 = <&pinctrl_enet1>;
    phy-mode = "rmii";
    phy-handle = <&ethphy0>;
    phy-reset-gpios = <&gpio2 14 GPIO_ACTIVE_LOW>;
    phy-reset-duration = <26>;
    status = "okay";

```

```
89      mdio {
90          #address-cells = <1>;
91          #size-cells = <0>;
92
93          ethphy0: ethernet-phy@0 {
94              compatible = "ethernet-phy-ieee802.3-c22";
95              reg = <0>;
96              smsc,disable-energy-detect;
97              interrupt-parent = <&gpio2>;
98              interrupts = <15 IRQ_TYPE_LEVEL_LOW>;
99              status = "okay";
100         };
101     };
102 };
```

```
266     pinctrl_enet1: enet1grp {
267         fsl,pins = <
268             MX6UL_PAD_ENET1_RX_EN__ENET1_RX_EN  0x1b0b0
269             MX6UL_PAD_ENET1_RX_ER__ENET1_RX_ER  0x1b0b0
270             MX6UL_PAD_ENET1_RX_DATA0__ENET1_RDATA00 0x1b0b0
271             MX6UL_PAD_ENET1_RX_DATA1__ENET1_RDATA01 0x1b0b0
272             MX6UL_PAD_ENET1_TX_EN__ENET1_TX_EN  0x1b0b0
273             MX6UL_PAD_ENET1_TX_DATA0__ENET1_TDATA00 0x1b0b0
274             MX6UL_PAD_ENET1_TX_DATA1__ENET1_TDATA01 0x1b0b0
275             MX6UL_PAD_ENET1_TX_CLK__ENET1_REF_CLK1  0x40017051
276             MX6UL_PAD_ENET2_RX_DATA1__ENET1_MDC 0x1b0b0
277             MX6UL_PAD_ENET2_RX_DATA0__ENET1_MDIO    0x1b0b0
278             MX6UL_PAD_ENET2_TX_CLK__GPIO2_IO14  0x0b0b0
279             MX6UL_PAD_ENET2_RX_ER__GPIO2_IO15   0x1b0b0
280         >;
281     };
```

9.2.2.4. Transceiver CAN

Únicamente se han eliminado las líneas relativas al segundo transceiver, manteniendo las relativas al primero:

```
104 &can1 {
105     pinctrl-names = "default";
106     pinctrl-0 = <&pinctrl_flexcan1>;
107     status = "okay";
108 };
```

```
283     pinctrl_flexcan1: flexcan1grp{
284         fsl,pins = <
285             MX6UL_PAD_UART3_RTS_B__FLEXCAN1_RX  0x1b020
286             MX6UL_PAD_UART3_CTS_B__FLEXCAN1_TX  0x1b020
287         >;
288     };
```

9.2.3. Modificación del device tree de Linux

Los dos archivos modificados en el repositorio de U-Boot tienen sus homónimos en el repositorio de Linux:

- `linux/arch/arm/boot/dts/imx6ull-14x14-evk.dts`
- `linux/arch/arm/boot/dts/imx6ul-14x14-evk.dtsi`

Los archivos de ambos repositorios no son idénticos pero sí muy similares. En los de Linux se han realizado exactamente los mismos cambios que en los de U-Boot.

9.3. Configuración y compilación de U-Boot

Al igual que con el device tree, para evitar crear de cero todos los archivos de configuración, se aprovechará todo lo posible de la placa *MCIMX6ULL-EVK*. La inicialización básica del SoC, RAM y periférico UART es válida para ambas placas. También se aprovecharán tal cual las tablas DCD para la configuración del controlador de memoria RAM, las cuales se encuentran ubicadas en los siguientes archivos:

- `u-boot/board/freescale/mx6ullevk/imximage.cfg`
- `u-boot/board/freescale/mx6ullevk/plugin.S`

No ocurre así con el periférico SD/MMC. La configuración de la placa *MCIMX6ULL-EVK* está pensada para que Linux arranque desde el periférico SD2, mientras que en este caso el conector de la tarjeta microSD está conectado al periférico SD1. También ha sido necesario realizar cambios en la configuración del Ethernet.

9.3.1. Configuración de U-Boot

En el archivo `u-boot/configs/mx6ull_14x14_evk_defconfig` ha sido necesario eliminar las siguientes líneas, ya que las variables de entorno de U-Boot no se cargarán desde la tarjeta SD. Tampoco se utilizará un transceiver de Micrel:

```
1 CONFIG_ENV_IS_IN_MMC=y
2 CONFIG_SYS_MMC_ENV_DEV=1
3 CONFIG_PHY_MICREL=y
4 CONFIG_PHY_MICREL_KSZ8XXX=y
```

Y ha sido necesario añadir las siguientes líneas, relacionadas con la memoria Flash y con el transceiver Ethernet, que son distintos respecto a la placa EVK:

```
1 CONFIG_SPI_FLASH_MACRONIX=y
2 CONFIG_CMD_MII=y
3 CONFIG_CMD_MDIO=y
4 CONFIG_PHY_SMSC=y
5 CONFIG_DM_MDIO=y
6 CONFIG_DM_MDIO_MUX=y
7 CONFIG_NET_RANDOM_ETHADDR=y
```

También, en el archivo `u-boot/include/configs/mx6ullevk.h` ha sido necesario cambiar las siguientes líneas:

```
#define CONFIG_SYS_FSL_ESDHC_ADDR USDHC2_BASE_ADDR
...
#define CONFIG_MMCROOT "/dev/mmcblk1p2" /* USDHC2 */
...
#define CONFIG_FEC_ENET_DEV     1
```

A por las siguientes:

```
#define CONFIG_SYS_FSL_ESDHC_ADDR USDHC1_BASE_ADDR
...
"bootdelay=1\0"
...
#define CONFIG_MMCROOT "/dev/mmcblk0p2" /* USDHC1 */
...
#define CONFIG_FEC_XCV_TYPE RMII
...
#define CONFIG_FEC_ENET_DEV     0
```

9.3.2. Comando de arranque

En el archivo `u-boot/include/configs/mx6ullevk.h` también se ha modificado el comando de arranque a por el siguiente:

```
echo \"Booting Arch Linux...\";
setenv kernel_addr 0x82000000;
setenv dtb_addr 0x83000000;
setenv ramdisk_addr 0x83080000;
setenv bootargs \"console=ttymxc0,115200n81 root=/dev/mmcblk0p1
    rootwait rw rootfstype=ext4\";
ext4load mmc 0:1 ${kernel_addr} \"/boot/zImage\";
ext4load mmc 0:1 ${dtb_addr} \"/boot/dtbs/imx6-breakout-board.
   dtb\";
ext4load mmc 0:1 ${ramdisk_addr} \"/boot/initramfs-linux.img\";
bootz ${kernel_addr} ${ramdisk_addr}:${filesize} ${dtb_addr};
```

De esta forma, se le indica a U-Boot los pasos a realizar durante el arranque:

- Se especifican los argumentos de entrada del kernel, indicando así al kernel el puerto serie que debe usar para mostrar la consola, así como la partición de la tarjeta SD en la que se dispone del sistema de archivos de la distribución.
- Se copia la imagen del kernel `zImage` desde la ruta correspondiente de la tarjeta SD a la dirección 0x82000000 de la memoria RAM.
- Se copia el device tree del kernel `imx6-breakout-board.dtb` desde la ruta correspondiente de la tarjeta SD a la dirección 0x83000000 de la memoria RAM.
- Se copia la imagen del initramfs `initramfs-linux.img` desde la ruta correspondiente de la tarjeta SD a la dirección 0x83080000 de la memoria RAM.
- Por último, U-Boot cede el control al kernel, pasándole todos los parámetros especificados anteriormente.

9.3.3. Compilación de U-Boot

Tras estos cambios, se aplica la configuración y se compila U-Boot:

```
1 cd u-boot
2 export ARCH=arm
3 export CROSS_COMPILE=arm-linux-gnueabihf-
4 make distclean
5 make mx6ull_14x14_evk_defconfig
6 make
```

Al final del proceso se genera el binario `u-boot-dtb.imx` que será el que se utilice posteriormente. En este archivo ya va incluido el device tree de U-Boot, así como la cabecera DCD que leerá la BootROM para configurar el controlador de la RAM.

9.4. Configuración y compilación del kernel

Con los siguientes comandos se configura el kernel con la configuración por defecto de NXP para el SoC iMX6ULL utilizado en este trabajo:

```
1 cd linux
2 export ARCH=arm
3 export CROSS_COMPILE=arm-linux-gnueabihf-
4 make distclean
5 make imx_v6_v7_defconfig
```

Para configurar más opciones manualmente, es posible acceder a un menú gráfico de configuración utilizando el siguiente comando:

```
1 make menuconfig
```

En este caso no ha sido necesario realizar ninguna modificación adicional. Una vez completada la configuración, se compila el kernel:

```
1 make
```

En función de la configuración, algunos drivers van incluidos en el kernel, mientras que otros se compilan aparte en forma de módulos. Para compilar los módulos:

```
1 make modules
```

Una vez compilados, sería necesario copiarlos al directorio correspondiente:

```
1 make modules_install INSTALL_MOD_PATH=/path/to/install/modules/
```

El proceso de compilación de kernel también se encarga de compilar el device tree de la placa, generando el archivo `imx6ull-14x14-evk.dtb`. Este archivo se ha renombrado a `imx6-breakout-board.dtb` y será el que se utilice posteriormente.

Finalmente, el kernel y módulos compilados en este apartado no se han utilizado, ya que ha resultado más conveniente aprovechar el kernel y módulos que van incluidos con la distribución. De esta forma, al actualizar el sistema con el gestor de paquetes de la distribución, también se actualiza el kernel, evitando así tener que volver a compilar y copiar el kernel y módulos manualmente tras cada actualización.

9.5. Obtención del sistema de archivos

El objetivo de esta sección es obtener el resto de la distribución Linux, también conocido como el sistema de archivos *root*, es decir, la estructura de carpetas, archivos y todos los programas necesarios para tener un sistema Linux completamente funcional.

Para ello, una opción es generar una distribución a medida con Buildroot o Yocto. Estas herramientas permiten seleccionar los paquetes a utilizar, configurarlos y compilarlos a medida de la arquitectura. De esta manera se pueden personalizar los paquetes a medida de la aplicación, obteniendo una distribución más ligera y optimizada, más adecuada para aplicaciones embebidas. En caso de querer arrancar desde la NOR Flash de 16MB, sería necesario preparar una distribución de este tipo.

También es posible utilizar el sistema de archivos de una distribución Linux que tenga versión para ARMv7 tales como Debian, Arch Linux o Alpine Linux. Esto nos permitirá tener un sistema operativo funcional de manera más rápida y sencilla, así como un gestor de paquetes que nos permitirá instalar y actualizar los programas como si de una distribución de escritorio se tratase. Aunque estas distribuciones son más pesadas, finalmente se ha optado por esta opción, ya que el sistema arrancará desde la tarjeta SD por lo que el espacio no es un problema en este caso.

Finalmente se ha optado por utilizar la distribución Arch Linux ARM, la cual se encuentra disponible para las arquitecturas ARMv7 y ARMv8. Desde la página de la distribución se ha descargado el paquete genérico `ArchLinuxARM-armv7-latest.tar.gz` para ARMv7. En este paquete se incluye toda la estructura de carpetas, el kernel y el resto de programas necesarios, a falta de proporcionar únicamente el bootloader y el device tree en función de la placa utilizada.

9.6. Preparación de la tarjeta SD

Llegado a este punto ya se dispone de todos los archivos necesarios. A continuación, se han realizado los siguientes pasos para preparar la tarjeta SD, teniendo en cuenta que `/dev/sda` era el dispositivo asignado a la tarjeta en este caso [35].

Se elimina la tabla de particiones:

```
1 sudo dd if=/dev/zero of=/dev/sda bs=1M count=10
```

El binario de U-Boot es necesario copiarlo a un sector específico de la tarjeta SD, en el que es buscado por la Boot ROM durante el arranque:

```
1 sudo dd if=./u-boot/u-boot-dtb.imx of=/dev/sda seek=2 bs=512
```

Se crea una nueva partición, donde irá el sistema de archivos:

```
1 sudo sfdisk /dev/sda <<-__EOF__
2 1M,,L,*
3 __EOF__
```

Se formatea la nueva partición en formato ext4:

```
1 sudo mkfs.ext4 -L rootfs /dev/sda1
```

Se copian todos los archivos de la distribución a la nueva partición:

```
1 sudo tar xfvp ./ArchLinuxARM-armv7-latest.tar.gz -C /run/media/
     alfonso/rootfs
2 sync
```

Por último, se copia también el device tree de la placa para el kernel:

```
1 sudo cp ./imx6-breakout-board.dtb /run/media/alfonso/rootfs/
     boot/dtbs/imx6-breakout-board.dtb
2 sync
```

9.7. Testeo de la instalación y los periféricos

Una vez terminada la instalación, se ha comprobado que el diseño funciona correctamente. Para ello, se han testeado distintas funciones del sistema operativo y de los periféricos, sin ánimo de ser tests muy exhaustivos, pero comprobando que todos los componentes de la placa son accesibles desde Linux y funcionan correctamente.

9.7.1. Testeo del puerto serie

El puerto serie es uno de los periféricos más críticos y que más interesa que funcionen desde el principio. Es la forma más sencilla de tener acceso a la consola tanto de U-Boot como de Linux. Esto ha sido imprescindible durante las primeras pruebas para poder observar los mensajes durante el arranque e ir solucionando los errores que aparecían. Para ello se ha utilizado el programa *minicom* desde Linux en el PC.

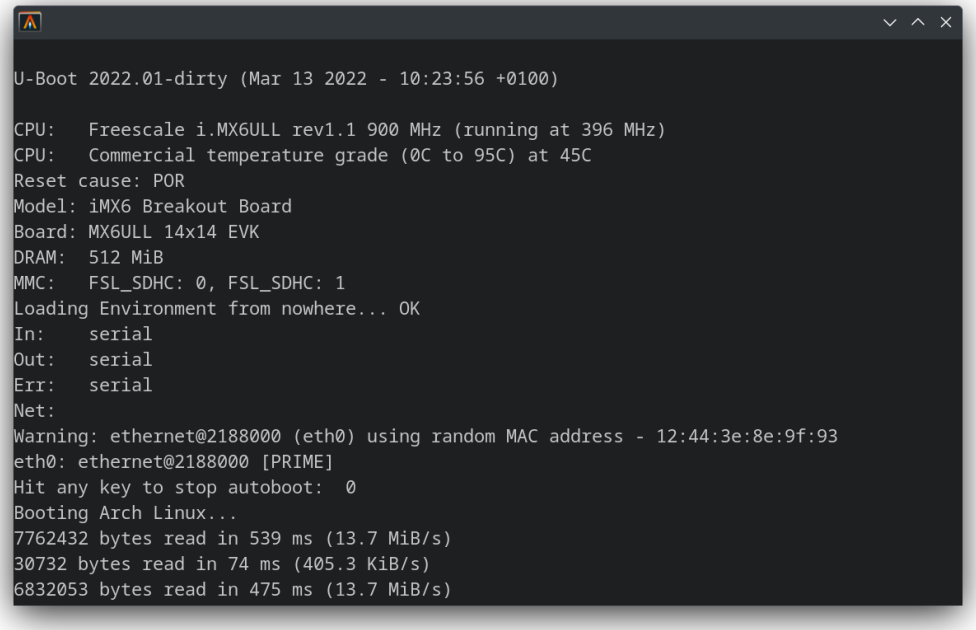


Figura 9.1: Mensajes mostrados en consola durante el arranque de U-Boot.

9.7.2. Testeo del procesador

La frecuencia configurada actualmente se puede comprobar con este comando:

```
1 cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq
```

La frecuencia del procesador se puede configurar desde el espacio de usuario cambiando el *governor* al modo *userspace*:

```
1 echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/
    scaling_governor
```

Una vez realizado este cambio, es posible cambiar manualmente la frecuencia del procesador entre los distintos puntos de operación indicados en el device tree:

```
1 echo 198000 > /sys/devices/system/cpu/cpu0/cpufreq/
     scaling_setspeed
2 echo 396000 > /sys/devices/system/cpu/cpu0/cpufreq/
     scaling_setspeed
3 echo 528000 > /sys/devices/system/cpu/cpu0/cpufreq/
     scaling_setspeed
4 echo 792000 > /sys/devices/system/cpu/cpu0/cpufreq/
     scaling_setspeed
5 echo 900000 > /sys/devices/system/cpu/cpu0/cpufreq/
     scaling_setspeed
```

También es posible obtener la temperatura del procesador con este comando:

```
1 cat /sys/devices/virtual/thermal/thermal_zone0/temp
```

Se ha ejecutado el *benchmark* UnixBench para estresar el procesador y comprobar que funciona de manera estable. La temperatura máxima del procesador durante el test ha sido de 56°C con una temperatura ambiente de 20°C, lo cual coincide con los resultados de la simulación obtenidos anteriormente en el capítulo 3.

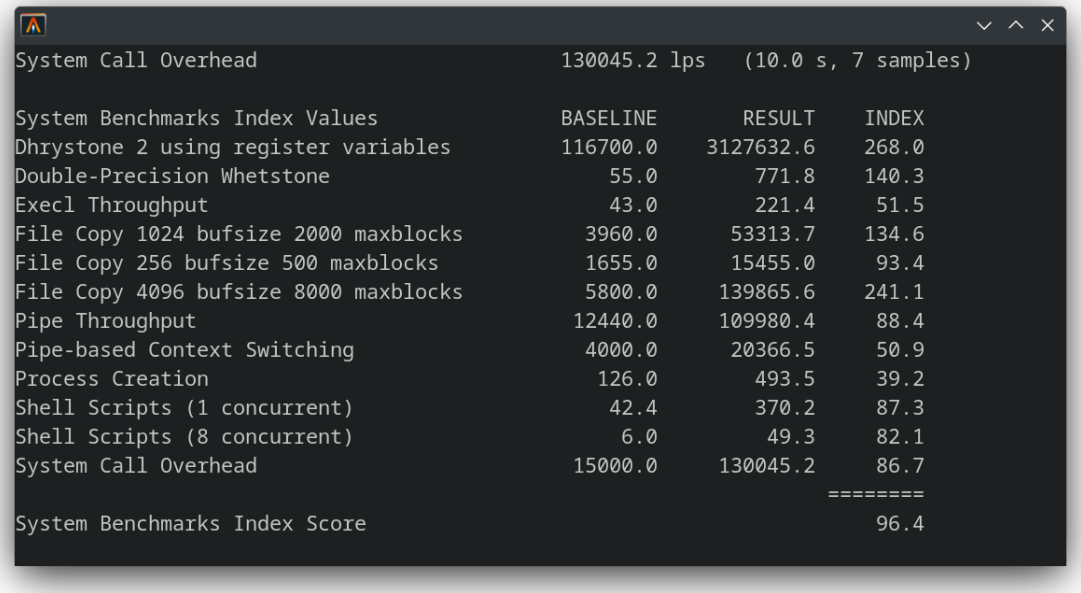


Figura 9.2: Resultados del test UnixBench v5.1.3.

9.7.3. Testeo de la memoria RAM

La memoria RAM ya ha sido testeada en el capítulo anterior. Desde Linux no es posible testearla en su totalidad, ya que parte de la memoria está ocupada por el sistema operativo y las aplicaciones. Se han completado satisfactoriamente dos tests distintos que estaban disponibles en los repositorios de la distribución. Ha sido necesario configurarles un máximo de 400MB de memoria ya que el resto no estaba disponible:

```
memtester 400m 1
```

```
stress --cpu 1 --vm 1 --vm-bytes 400MB --timeout 120s
```

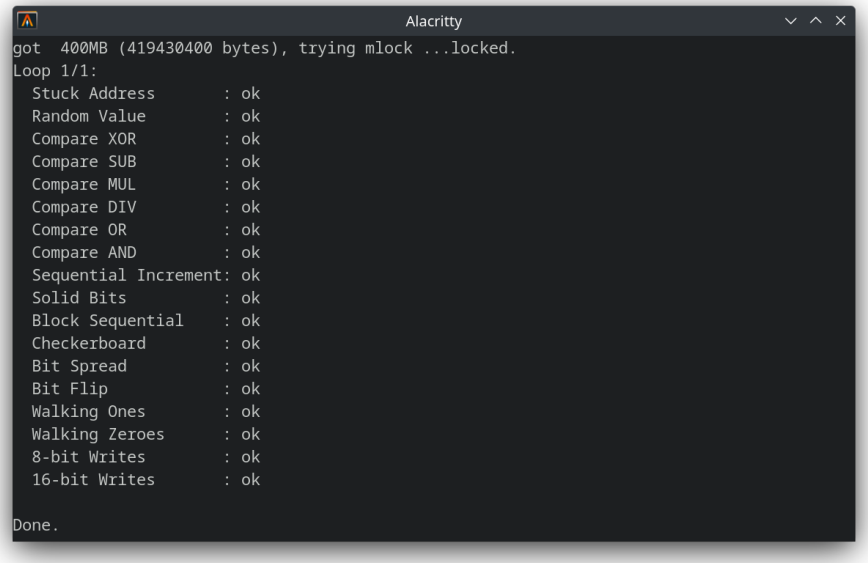


Figura 9.3: Resultados del test memtester v4.5.1.

9.7.4. Testeo del pulsador y LED

Es necesario configurar los puertos GPIO correspondientes. Con estos comandos se ha configurado el del LED como salida y el del pulsador como entrada:

```
echo 5 > /sys/class/gpio/export
echo 9 > /sys/class/gpio/export
echo "out" > /sys/class/gpio/gpio5/direction
echo "in" > /sys/class/gpio/gpio9/direction
```

El LED se puede encender o apagar utilizando estos comandos:

```
1 echo 1 > /sys/class/gpio/gpio5/value
2 echo 0 > /sys/class/gpio/gpio5/value
```

El estado del pulsador se puede leer con el siguiente comando:

```
1 cat /sys/class/gpio/gpio9/value
```

9.7.5. Testeo de la memoria NOR Flash

Con los siguientes comandos se ha comprobado que la memoria funciona, escribiendo una cadena de texto y leyéndola de vuelta para mostrarla por consola:

```
1 echo "Hola" > /dev/mtd0
2 dd if=/dev/mtd0 of=/dev/tty bs=1 count=5
```

Con el programa *dd* se ha realizado un test leyendo todo el contenido de la memoria, consiguiendo una velocidad media de lectura de 12.8MB/s:

```
1 dd if=/dev/mtd0 of=/dev/null bs=64k count=256
```

De igual forma se ha realizado un test de escritura rellenando con contenido aleatorio toda la memoria, consiguiendo una velocidad media de escritura de 841KB/s:

```
1 dd if=/dev/urandom of=/dev/mtd0 bs=64k count=256
```

9.7.6. Testeo de los puertos USB

Se ha comprobado que los puertos USB funcionan correctamente conectando distintos periféricos como una memoria USB. También se ha probado el modo *USB Gadget* con el conector USB desde el que se alimenta la placa. Con este comando se carga el módulo del kernel que crea el puerto serie virtual `/dev/ttyGS0`:

```
1 modprobe g_serial
```

Desde el PC, se puede leer el contenido que llega por el puerto serie:

```
cat /dev/ttyACM0
```

Desde la placa, se pueden enviar mensajes al PC:

```
echo "Hola" > /dev/ttyGS0
```

También sería posible configurar la consola de Linux en este puerto serie, evitando así el uso de un conversor externo.

9.7.7. Testeo del puerto Ethernet

El Ethernet es otro de los periféricos que interesa tener funcionando desde el principio. De esta forma sería posible acceder a la placa por SSH, facilitando mucho el desarrollo de los programas. Una forma sencilla de comprobar que el Ethernet funciona correctamente es realizar un ping a los servidores de Google:

```
ping 8.8.8.8
```

Una vez estaba funcionando el periférico Ethernet, ha sido posible realizar una actualización de todos los paquetes de la distribución e instalar otros nuevos.

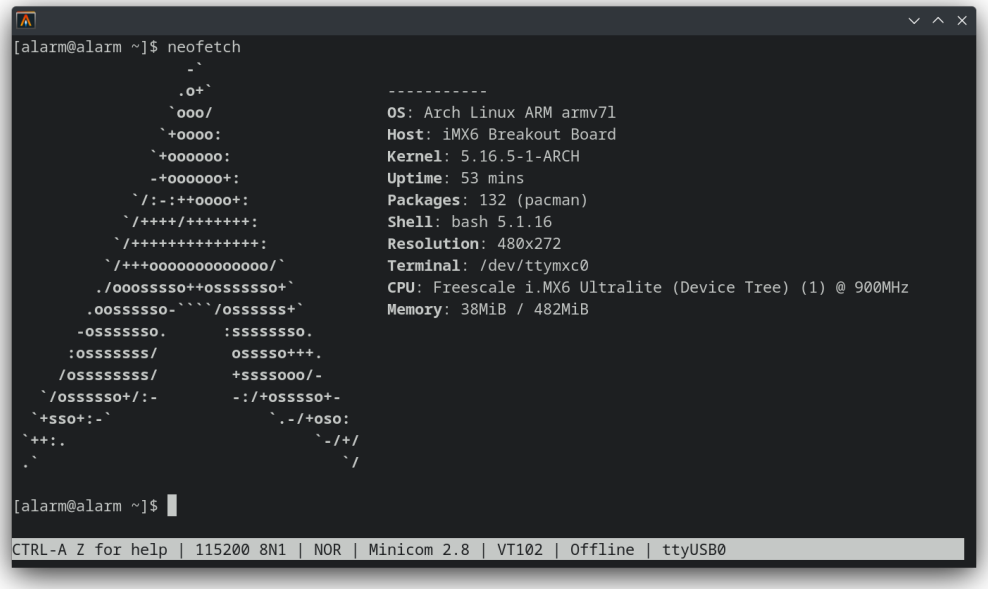


Figura 9.4: Programa *neofetch* mostrando información acerca del sistema.

9.7.8. Testeo del puerto CAN

9.7.8.1. Configuración del CAN en el PC

Para testear el puerto CAN se ha realizado una comunicación por CAN entre el PC y la placa. En el lado del PC, se ha utilizado un adaptador *CANable* [36]. Este adaptador se conecta al PC por USB e incluye un microcontrolador STM32 junto a un transceiver CAN que permiten dotar al PC de comunicación por CAN.

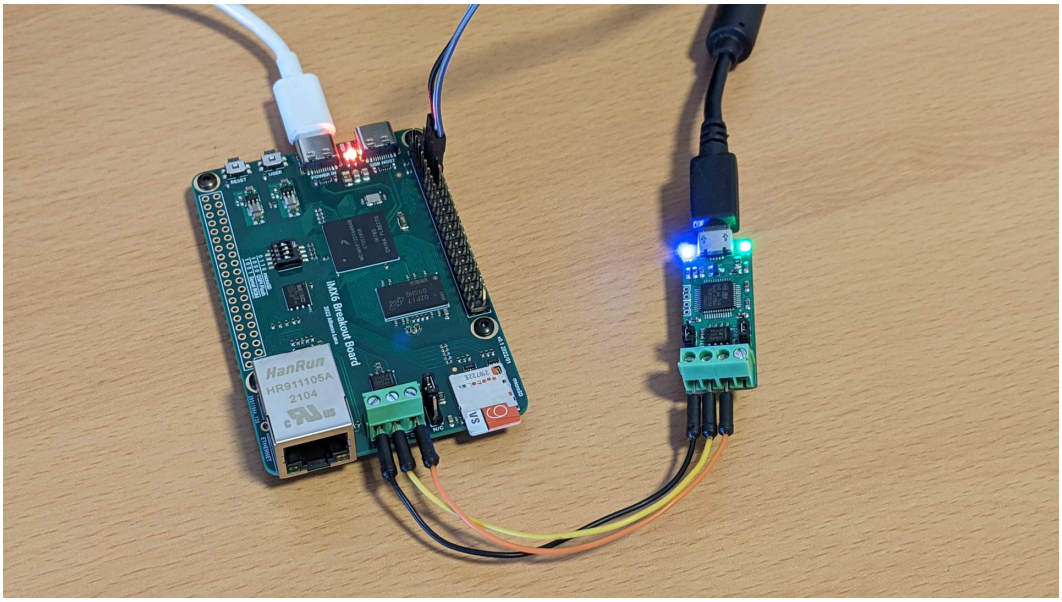


Figura 9.5: Conexión CAN entre el PC y la placa.

Una vez conectado el adaptador CANable al PC, automáticamente aparece en el Linux del PC un nuevo dispositivo de red llamado *can0*. Con los siguientes comandos se ha habilitado el adaptador tras configurarlo con un bitrate de 125k:

```
1 ip link set can0 type can bitrate 125000
2 ip link set can0 up
```

Los siguientes comandos también han sido de utilidad para ver los detalles de la configuración actual, así como las estadísticas sobre la conexión:

```
1 ip -details link show can0
2 ip -details -statistics link show can0
```

Una vez habilitado el puerto CAN en el PC, es necesario algún programa que se encargue de enviar o recibir los mensajes. Para ello se han instalado en el PC las herramientas *can-utils* que incluyen distintos programas para la comunicación CAN:

```
1 git clone https://aur.archlinux.org/can-utils.git
2 cd can-utils
3 makepkg
4 pacman -U can-utils-2021.08.0-1-x86_64.pkg.tar.zst
```

Una vez instaladas, se ha ejecutado este comando el cual queda a la escucha de todos los mensajes que se vayan recibiendo y los va mostrando por la pantalla:

```
1 candump can0
```

9.7.8.2. Configuración del CAN en la placa

En el lado de la placa, ha sido necesario habilitar el adaptador de red CAN exactamente de la misma forma que en el PC:

```
1 ip link set can0 type can bitrate 125000
2 ip link set can0 up
```

De igual forma, también son necesarias las herramientas *can-utils* en la placa:

```
1 git clone https://aur.archlinux.org/can-utils.git
2 cd can-utils
```

En este caso antes de compilar el paquete ha sido necesario modificar la arquitectura en el archivo `PKGBUILD` a por la del SoC, cambiando esta línea:

```
1 arch=('x86_64' 'armv6h')
```

A por la siguiente:

```
1 arch=('x86_64' 'armv7h' 'armv6h')
```

Una vez hecho este cambio se ha compilado e instalado el paquete:

```
makepkg
pacman -U can-utils-2021.08.0-1-armv7h.pkg.tar.xz
```

Una vez instaladas estas herramientas, en el lado de la placa se ha ejecutado este otro comando el cual se encarga de generar mensajes aleatorios en el bus CAN:

```
cangen can0
```

Se ha comprobado que los mensajes generados por la placa se recibían correctamente en el PC y viceversa.

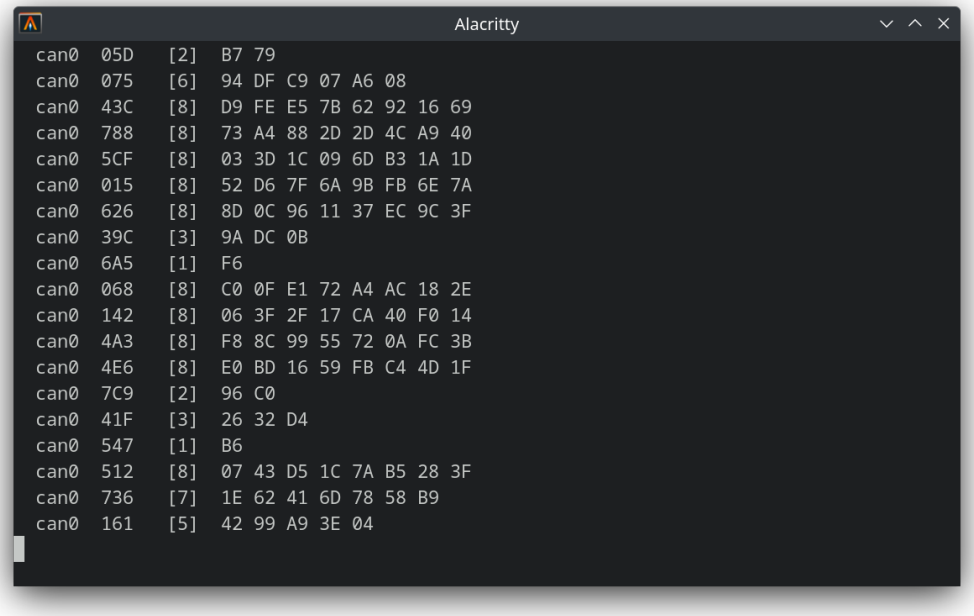


Figura 9.6: Mensajes CAN recibidos en el PC desde la placa.

Parte IV

CONCLUSIONES

Capítulo 10

Conclusiones

10.1. Resultados obtenidos

En este Trabajo Fin de Máster se ha realizado un recorrido por todas las etapas implicadas en el diseño, desarrollo y fabricación de un Single Board Computer, obteniendo como resultado un dispositivo fabricado y completamente funcional.

Se ha comenzado realizando un estudio previo, en el que tras analizar las distintas alternativas disponibles en el mercado, se ha fijado el concepto y objetivo del nuevo diseño y se han seleccionado los principales componentes que lo forman. Como parte de este estudio previo, también se han determinado cuáles son los aspectos que más influyen en el coste de fabricación del dispositivo, especialmente cuando se trata de tiradas muy pequeñas. Estos aspectos han marcado bastante el resto del trabajo.

Como parte del desarrollo, se ha realizado un estudio para distintos aspectos clave del diseño, tales como el diseño térmico, la red de desacoplo, la integridad de señal y la compatibilidad electromagnética. El coste ha sido también un factor con gran peso en todos estos aspectos. Se ha tratado siempre de comprobar cuál era la opción que permitía un menor coste de fabricación, con la que el dispositivo podía seguir funcionando de manera correcta y fiable. Una vez aclarados todos estos aspectos, se ha completado el desarrollo del dispositivo, tanto a nivel de esquemático como de PCB.

Una vez finalizado el diseño, se ha procedido a fabricarlo. Gracias a todas las consideraciones anteriores, y a las limitaciones que se han ido aplicando, ha sido posible fabricar el PCB aun con un presupuesto muy reducido. También ha sido posible llevar a cabo el montaje y soldadura del dispositivo aun con unos medios bastante limitados, haciendo así viable la fabricación física del dispositivo como parte de este proyecto.

Por último, una vez se disponía físicamente del dispositivo, se ha procedido a verificar que este funcionaba correctamente. Para ello se ha comenzado realizando una serie de pruebas básicas a los reguladores, así como comprobando que la memoria RAM DDR3L funcionaba correctamente, lo cual era el punto más crítico del diseño. Tras ello se ha procedido a instalar una distribución de Linux embebido en la placa, desde la que se ha comprobado que el resto de periféricos también funcionaban correctamente.

Todo este proceso se ha ido documentando en la memoria con un estilo lo más didáctico posible, de forma que el trabajo realizado también pueda servir a futuros lectores a modo de introducción al diseño y desarrollo de este tipo de sistemas, así como a modo de guía o referencia a consultar de cara al desarrollo de futuros proyectos.

10.2. Conclusiones

El desarrollo de este tipo de dispositivos puede llegar a ser muy complicado, ya que abarca un gran abanico de disciplinas. Cada capítulo del trabajo podría considerarse un campo de estudio distinto, requiriendo de conocimientos y herramientas completamente distintas. Debido a ello, no se ha podido profundizar excesivamente en ninguno de estos campos, pero sí que ha sido necesario sumergirse en cierta medida en todos ellos, con el fin de poder completar el trabajo satisfactoriamente.

Personalmente, el trabajo ha sido una gran oportunidad para llevar a la práctica y afianzar los conocimientos adquiridos durante todo el Máster, especialmente de asignaturas como *Diseño Térmico y Compatibilidad Electromagnética*, *Diseño de Circuitos Impresos* o *Sistemas Embebidos*. También me ha servido para practicar y profundizar en el uso de los distintos programas o herramientas utilizados, tales como Altium Designer, HyperLynx o Fusion 360, así como U-Boot y Linux.

10.3. Mejoras futuras

A continuación se listan algunas ideas con las que se podría continuar el desarrollo del proyecto en un futuro, que han quedado fuera de este trabajo al no disponer de más tiempo para realizarlas:

- Desarrollar una serie de placas de expansión para dotar al diseño de más funcionalidad, tales como pantalla LCD o audio.
- Comprobar la correcta compatibilidad del diseño con placas de expansión ya existentes en el mercado en el formato de Raspberry Pi.
- Comprobar el arranque del sistema operativo desde la memoria QSPI NOR Flash.
- Preparar un *port* más elaborado de U-Boot y de Linux, creando los archivos de configuración necesarios a medida de la placa. Los cambios realizados en estos repositorios se podrían preparar en forma de parche, de forma que sería más fácil actualizar a futuras versiones, simplemente descargando la última versión de estos repositorios y aplicando el parche correspondiente.
- Preparar una distribución más optimizada utilizando Buildroot o Yocto.
- Utilizar *device tree overlays* para modificar fácilmente el device tree en función del hardware disponible. Esto es especialmente interesante para que el diseño sea compatible con varias placas de expansión con distintos periféricos.
- Configurar el puerto USB mediante el que se alimenta la placa como USB Gadget de forma predeterminada. De esta forma, conectando la placa al PC con un único cable USB se dispondría de funcionalidad como puerto serie, dispositivo de almacenamiento masivo y puerto Ethernet, además de la alimentación.

Parte V

BIBLIOGRAFÍA

Bibliografía

[1] José F. Toledo Alarcón. *Diseño térmico en equipos electrónicos: una introducción.* URL: http://personales.upv.es/jtoledo/.

[2] José F. Toledo Alarcón. *Aspectos prácticos de compatibilidad electromagnética e integridad de señal.* URL: http://personales.upv.es/jtoledo.

[3] José F. Toledo Alarcón, Germán Ramos Peinado y Jorge D. Martínez Pérez. *Diapositivas de la asignatura Diseño de Circuitos Impresos.*

[4] Jay Carlson. *So you want to build an embedded Linux system?* URL: https://jaycarlson.net/embedded-linux/.

[5] Jay Carlson. *mxiot.* URL: https://github.com/jaydcarlson/mxiot.

[6] Texas Instruments. *TLV62569 2-A High Efficiency Synchronous Buck Converter Datasheet.* URL: https://www.ti.com/product/TLV62569.

[7] Micron. *TN-41-13: DDR3 Point-to-Point Design Support.* URL: https://www.micron.com/-/media/client/global/documents/products/technical-note/dram/tn4113_ddr3_point_to_point_design.pdf.

[8] NXP Semiconductors. *i.MX 6ULL Applications Processors for Consumer Products Datasheet.* URL: https://www.nxp.com/docs/en/data-sheet/IMX6ULLCEC.pdf.

[9] NXP Semiconductors. *i.MX 6ULL Applications Processor Reference Manual.* URL: https://www.nxp.com/webapp/Download?colCode=IMX6ULLRM.

[10] NXP Semiconductors. *Hardware Development Guide for the i.MX 6ULL Applications Processor.* URL: https://www.nxp.com/files-static/soft_dev_tools/doc/user_guide/IMX6ULLHDG.pdf.

[11] Howard Johnson y Martin Graham. *High Speed Digital Design: A Handbook of Black Magic.*

[12] STMicroelectronics. *STM32F405RG Datasheet.* URL: https://www.st.com/resource/en/datasheet/stm32f405rg.pdf.

[13] JEDEC. *JESD79-3-1A.01. Addendum No. 1 to JESD79-3 - 1.35 V DDR3L-800, DDR3L-1066, DDR3L-1333, DDR3L-1600, and DDR3L-1866*. URL: https://www.jedec.org/standards-documents/docs/jesd79-3-1a01.

[14] USB Implementers Forum. *Universal Serial Bus Specification Revision 2.0*. URL: https://usb.org/document-library/usb-20-specification.

[15] STMicroelectronics. *AN5122 Application Note. STM32MP1 Series DDR memory routing guidelines*. URL: https://www.st.com/resource/en/application_note/an5122-stm32mp1-series-ddr-memory-routing-guidelines-stmicroelectronics.pdf.

[16] JLCPCB. *PCB Impedance Calculator*. URL: https://cart.jlcpcb.com/impedanceCalculation.

[17] JLCPCB. *Multilayer high precision PCBs with impedance control*. URL: https://cart.jlcpcb.com/impedance.

[18] NXP Semiconductors. *MCIMX6ULL-EVK: Evaluation kit for the i.MX 6ULL and 6ULZ Applications Processor*. URL: https://www.nxp.com/design/development-boards/i-mx-evaluation-and-development-boards/evaluation-kit-for-the-i-mx-6ull-and-6ulz-applications-processor:MCIMX6ULL-EVK.

[19] Todd Westerhoff. *Low-cost design: When best practice is too expensive*. URL: https://www.edn.com/low-cost-design-when-best-practice-is-too-expensive/.

[20] JLCPCB. *PCB Capabilities*. URL: https://jlcpcb.com/capabilities/Capabilities.

[21] Bert Simonovich. *Drivers Output Impedance From IBIS*. URL: https://blog.lamsimenterprises.com/2010/12/22/drivers-output-impedance-from-ibis/.

[22] Micron. *MT41K256M16TW-107:P Datasheet*. URL: https://media-www.micron.com/-/media/client/global/documents/products/data-sheet/dram/ddr3/4gb_ddr3l.pdf.

[23] Parlamento Europeo y del Consejo. *Directiva 2014/30/UE de 26 de febrero de 2014, sobre la armonización de las legislaciones de los Estados miembros en materia de compatibilidad electromagnética*. URL: https://eur-lex.europa.eu/eli/dir/2014/30.

[24] Parlamento Europeo y del Consejo. *Directiva 2014/53/UE de 16 de abril de 2014, relativa a la armonización de las legislaciones de los Estados miembros sobre la comercialización de equipos radioeléctricos, y por la que se deroga la Directiva 1999/5/CE.* URL: https://eur-lex.europa.eu/eli/dir/2014/53.

[25] Parlamento Europeo y del Consejo. *Directiva 2014/35/UE de 26 de febrero de 2014, sobre la armonización de las legislaciones de los Estados miembros en materia de comercialización de material eléctrico destinado a utilizarse con determinados límites de tensión.* URL: https://eur-lex.europa.eu/eli/dir/2014/35.

[26] Parlamento Europeo y del Consejo. *Directiva 2011/65/UE de 8 de junio de 2011, sobre restricciones a la utilización de determinadas sustancias peligrosas en aparatos eléctricos y electrónicos.* URL: https://eur-lex.europa.eu/eli/dir/2011/65.

[27] Parlamento Europeo y del Consejo. *Directiva 2012/19/UE de 4 de julio de 2012, sobre residuos de aparatos eléctricos y electrónicos.* URL: https://eur-lex.europa.eu/eli/dir/2012/19/.

[28] Timothy Hegarty. *A review of EMI standards, part 2 - radiated emissions.* URL: https://e2e.ti.com/blogs_/b/powerhouse/posts/a-review-of-emi-standards-part-2-radiated-emissions.

[29] Timothy Hegarty. *A review of EMI standards, part 1 - conducted emissions.* URL: https://e2e.ti.com/blogs_/b/powerhouse/posts/a-review-of-emi-standards-part-1-conducted-emissions.

[30] Clayton R. Paul. *Introduction to Electromagnetic Compatibility. 2nd Edition.*

[31] Gentoo. *Initramfs/Guide.* URL: https://wiki.gentoo.org/wiki/Initramfs/Guide.

[32] ES Raghunandan. *Porting U-Boot to a New Board.* URL: http://portinguboottoanewboard.blogspot.com/.

[33] Quentin Schulz. *Porting U-Boot and Linux on new ARM boards: a step-by-step guide.* URL: https://elinux.org/images/2/2a/Schulz-how-to-support-new-board-u-boot-linux.pdf.

[34] Linaro. *DeviceTree Specification Release v0.3.* URL: https://www.devicetree.org/specifications/.

[35] DigiKey. *Debian: Getting Started with the MCIMX6ULL-EVK.* URL: https://forum.digikey.com/t/debian-getting-started-with-the-mcimx6ull-evk/12465.

[36] CANable. *Getting Started.* URL: https://canable.io/getting-started.html.

Parte VI

APÉNDICES

Apéndice A

Esquemático

iMX6 Breakout Board

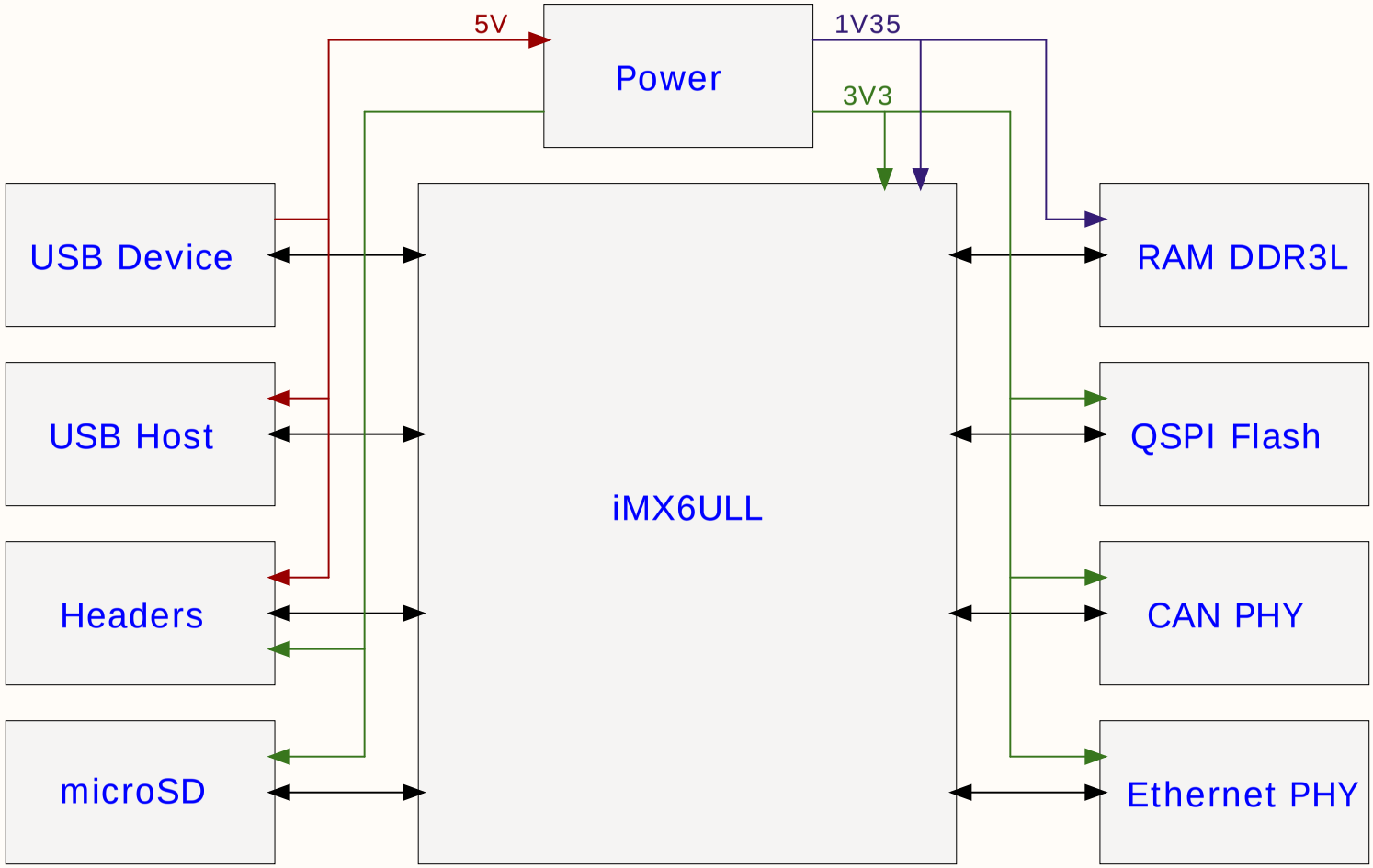
Title: iMX6 Breakout Board		
Size: A4	Page: 01_Cover_Page.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 1 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

Revision History

Revision	Date	Description
v0.1	01/18/22	Initial release

Title: iMX6 Breakout Board		
Size: A4	Page: 02_Revision_History.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 2 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

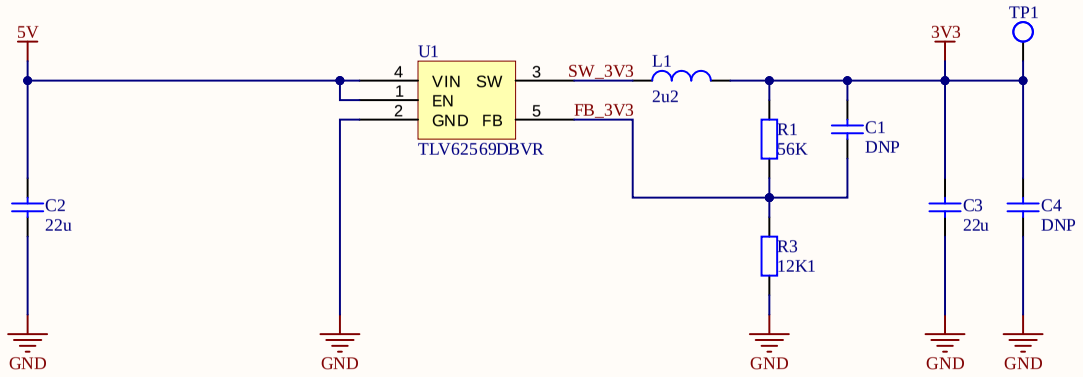
Block Diagram



Title: iMX6 Breakout Board		
Size: A4	Page: 03_Block_Diagram.SchDoc	Revision: v0.1
Date: 01/18/22	Sheet 3 of 14	
Build: Altium Designer 22.0.2.36	Designed by: Alfonso Luna	

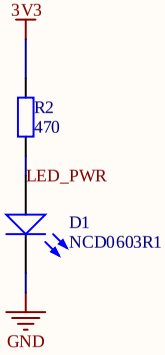
Power Converters

5V to 3V3 Converter

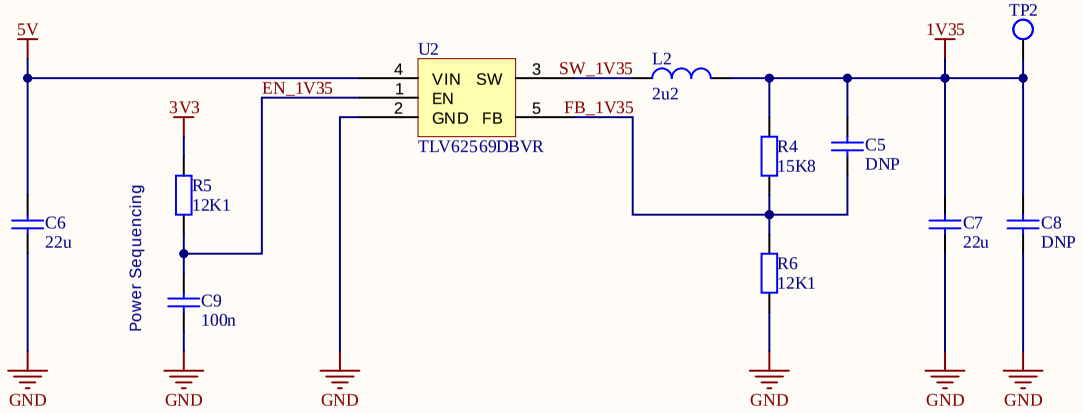


1% Resistors
3.377V Typ, 3.255-3.502V Range, 2A Max

Power LED



5V to 1V35 Converter

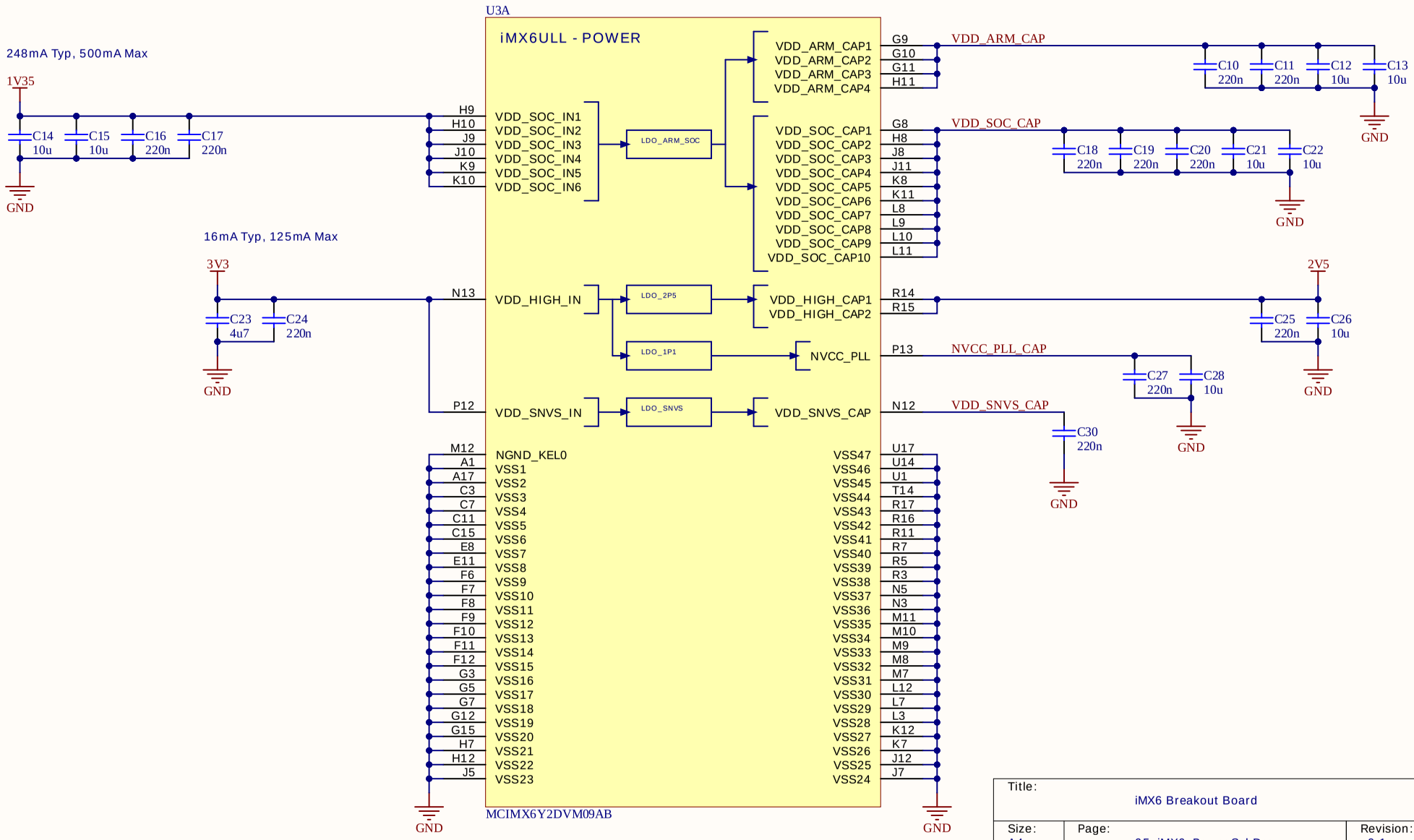


1% Resistors
1.383V Typ, 1.341-1.427V Range, 2A Max

CPU @ 900MHz, VDD_SOC_IN >= 1.375V
CPU @ 792MHz, VDD_SOC_IN >= 1.325V
CPU @ 528MHz, VDD_SOC_IN >= 1.275V

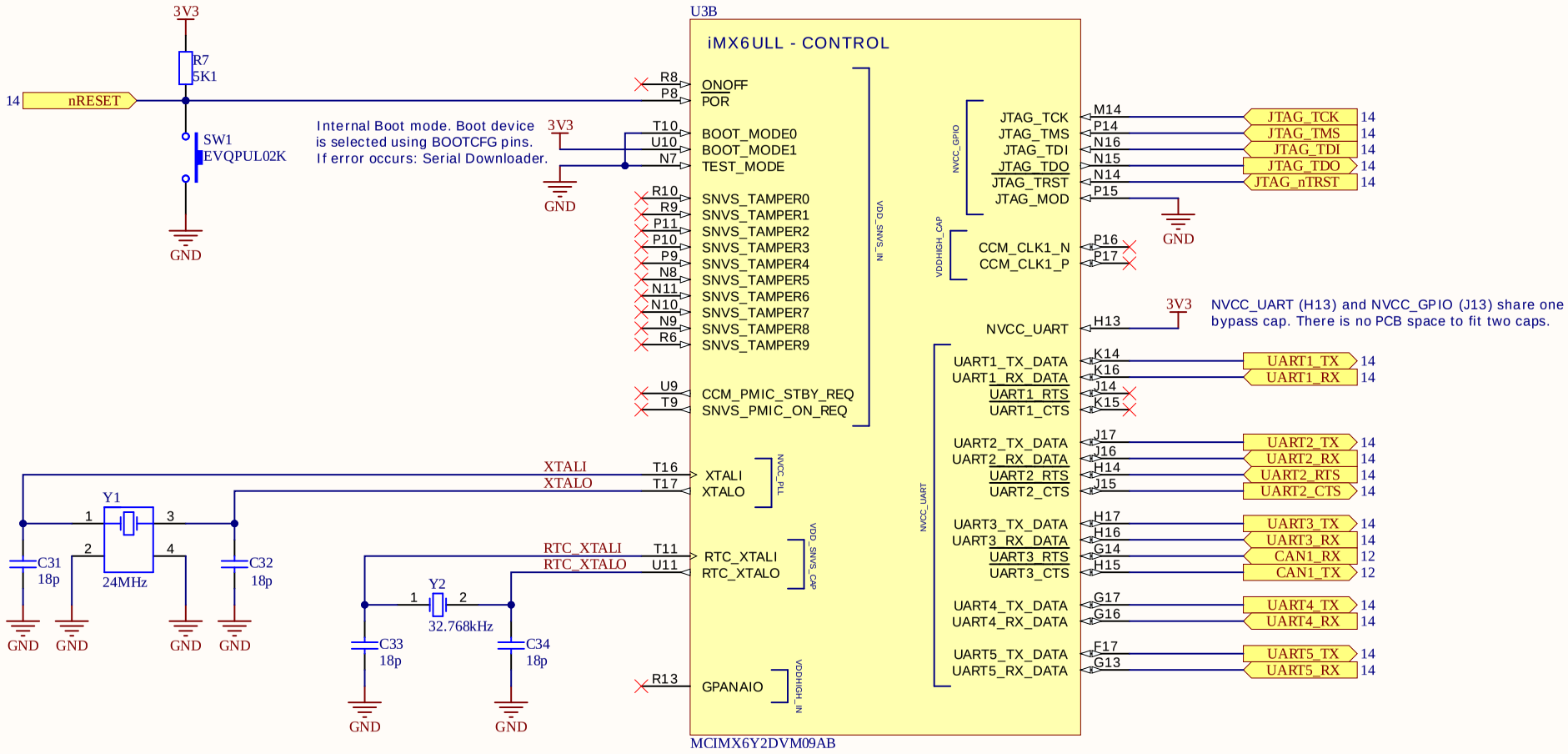
Title: iMX6 Breakout Board		
Size: A4	Page: 04_Power_Converters.SchDoc	Revision: v0.1
Date: 01/18/22	Sheet 4 of 14	
Build: Altium Designer 22.0.2.36	Designed by: Alfonso Luna	

iMX6 Power



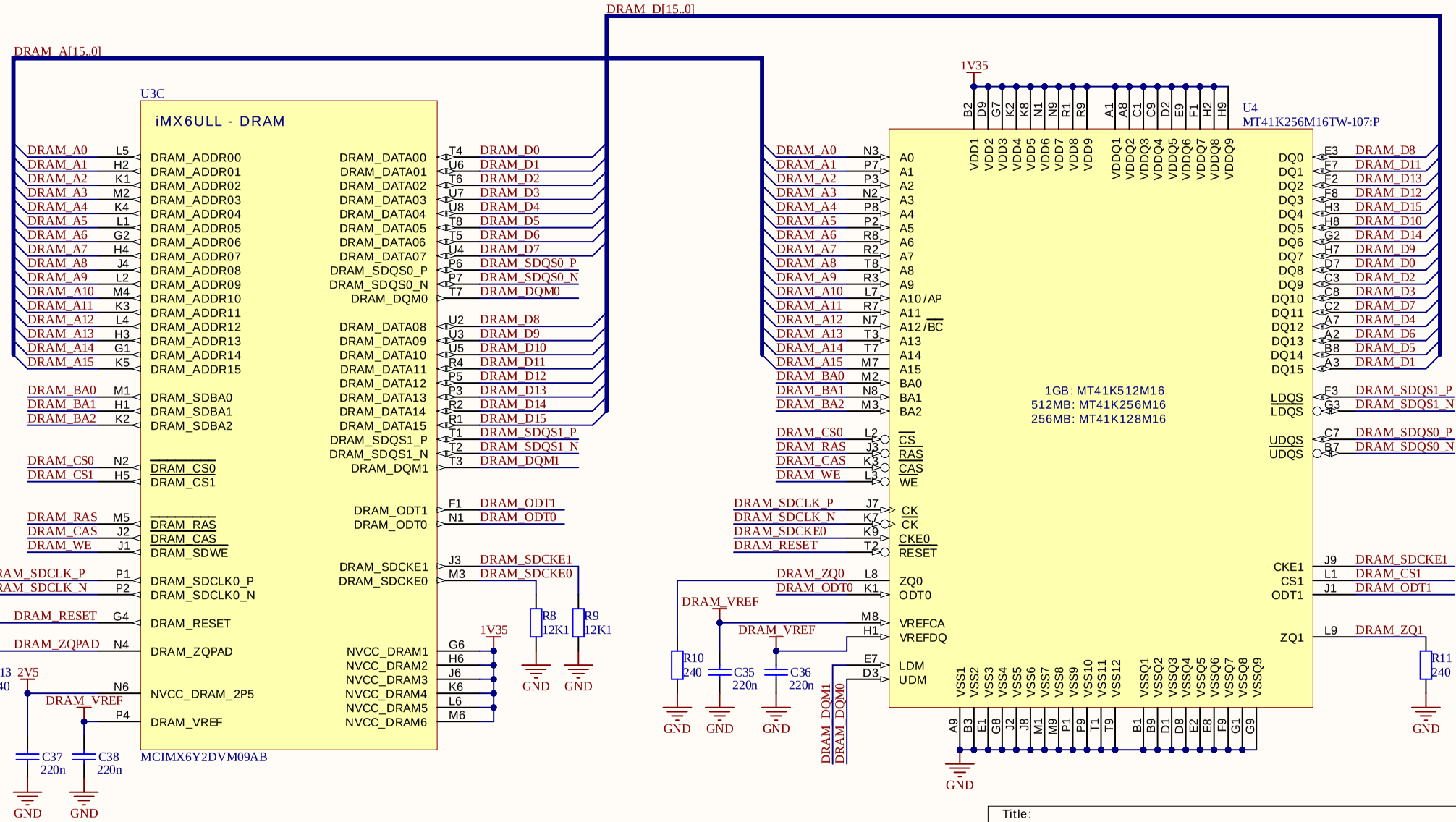
Title: iMX6 Breakout Board		
Size: A4	Page: 05_iMX6_Power.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 5 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

iMX6 Control



Title: iMX6 Breakout Board		
Size: A4	Page: 06_iMX6_Control.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 6 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

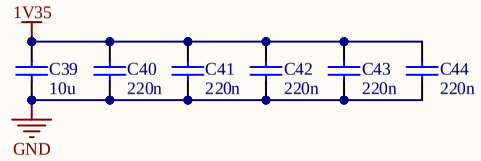
RAM DDR3L



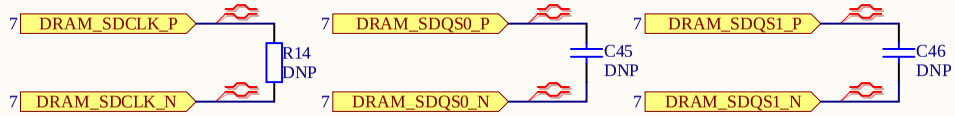
Title: iMX6 Breakout Board		
Size: A4	Page: 07_RAM_DDR3L_1.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 7 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

RAM DDR3L

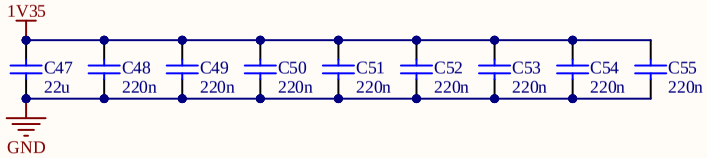
Decoupling caps for CPU (place close to NVCC_DRAMx)



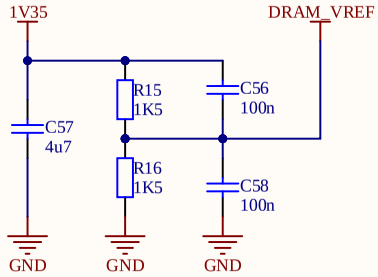
Differential pairs terminations (place close to DRAM)



Decoupling caps for DRAM (place close to VDDx, VDDQx)

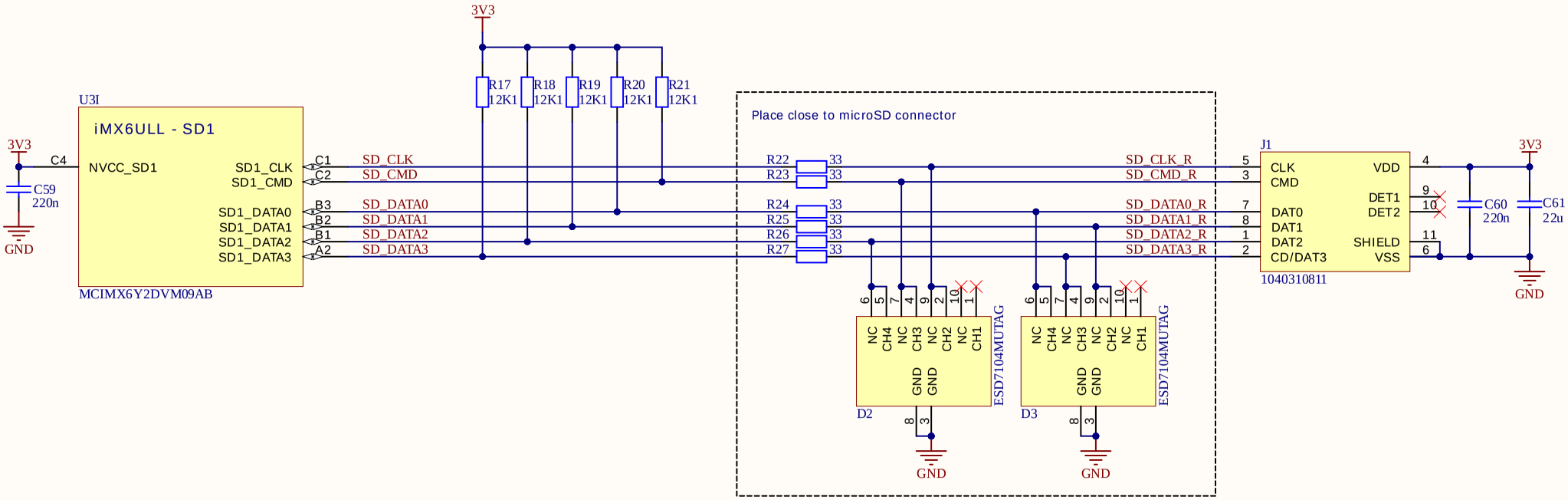


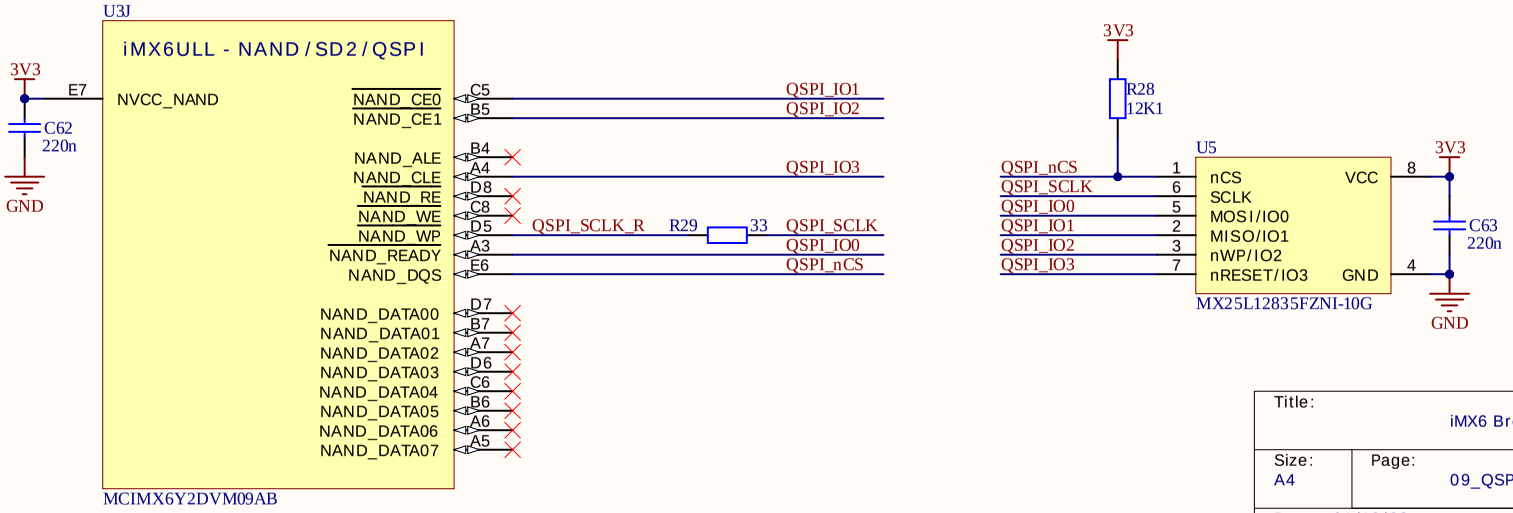
DRAM Reference Voltage



Title: iMX6 Breakout Board		
Size: A4	Page: 08_RAM_DDR3L_2.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 8 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

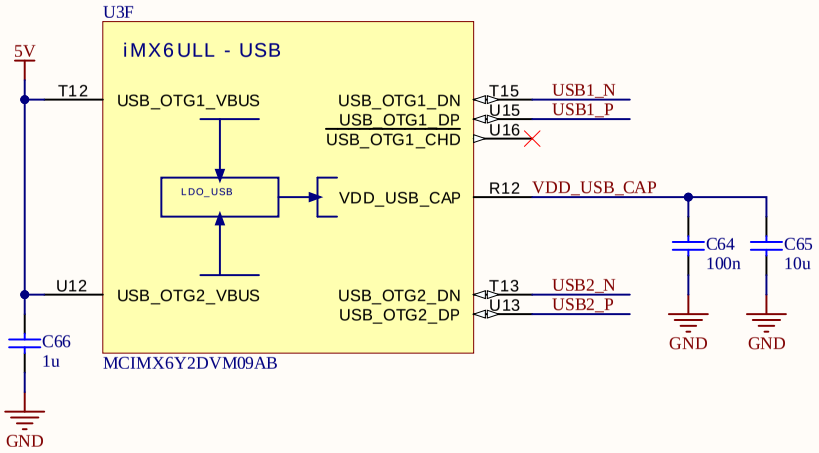
QSPI Flash, microSD

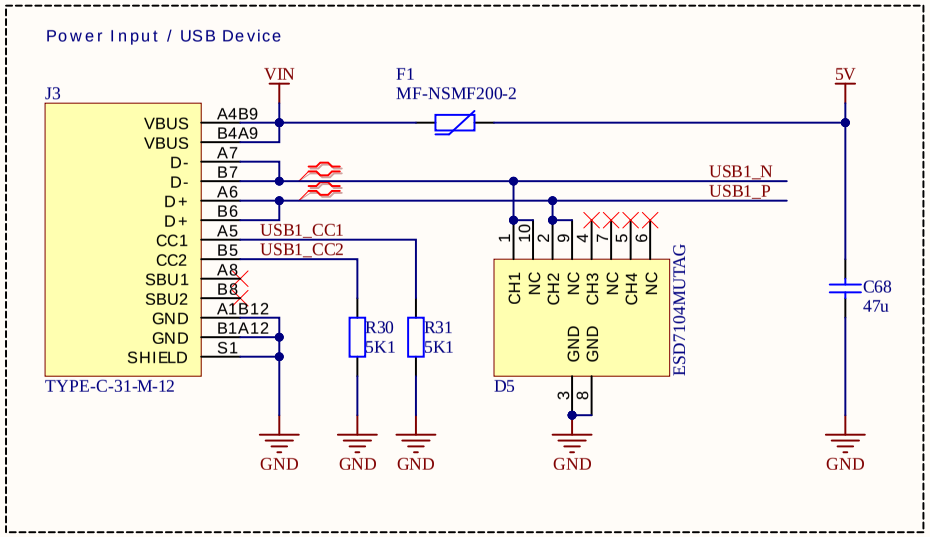


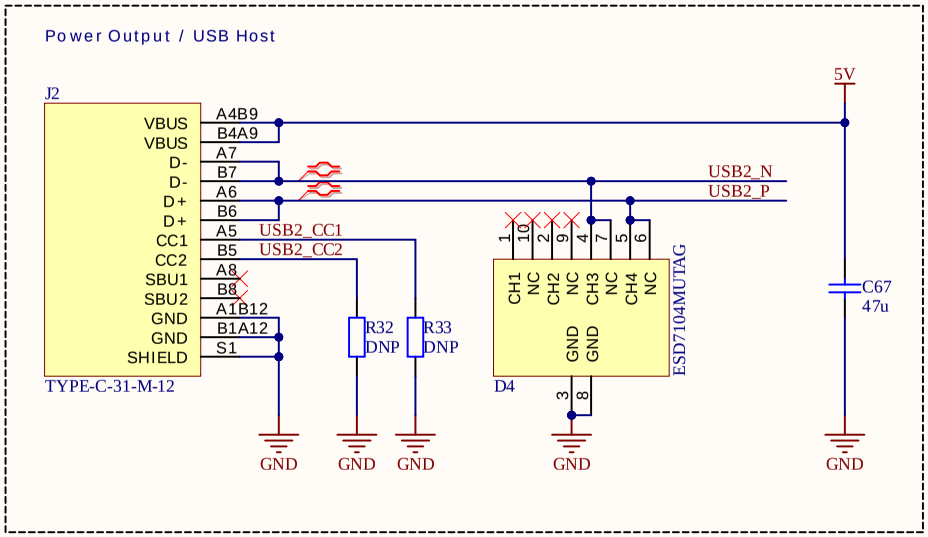


Title: iMX6 Breakout Board		
Size: A4	Page: 09_QSPI_Flash_microSD.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 9 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

USB Device, USB Host

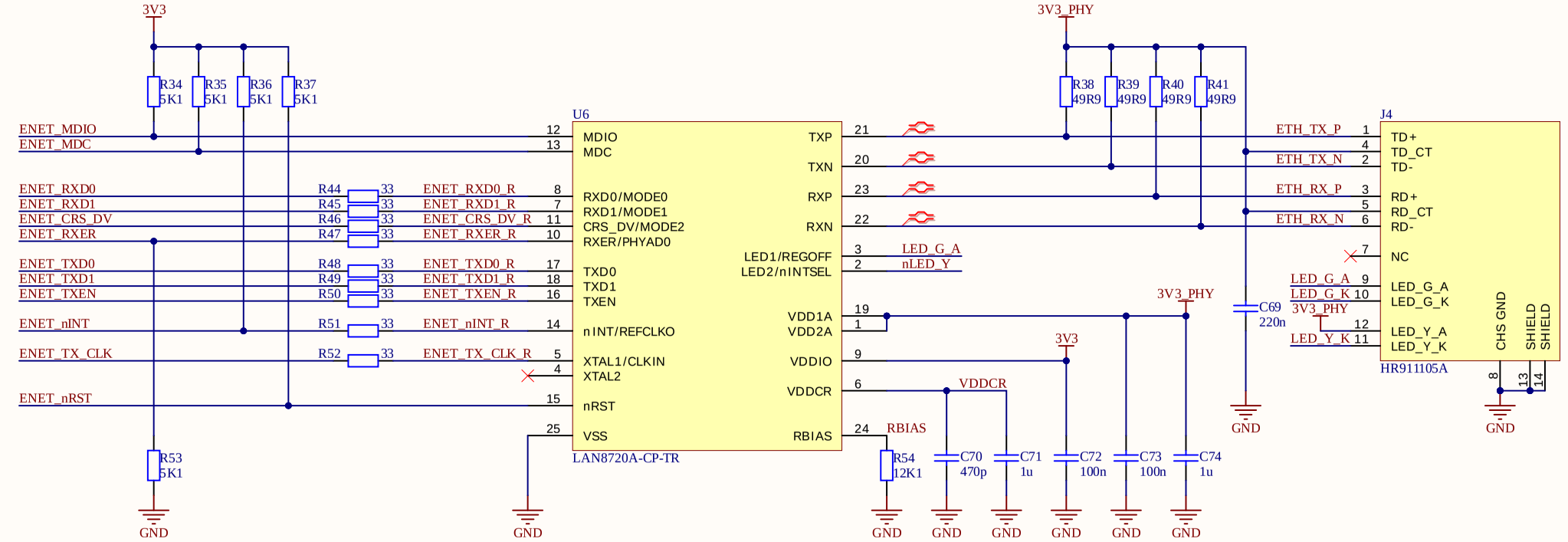


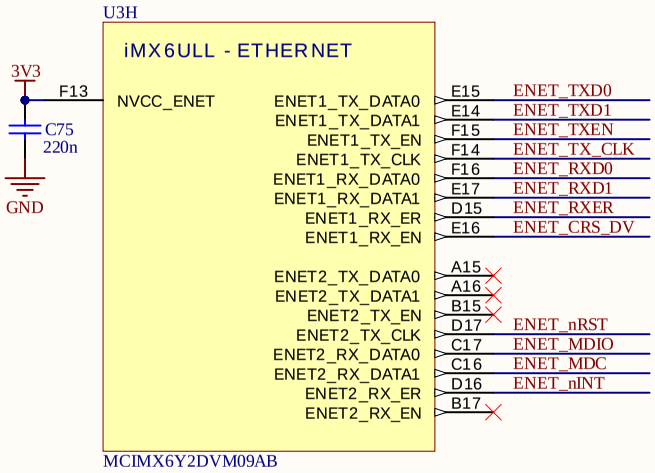




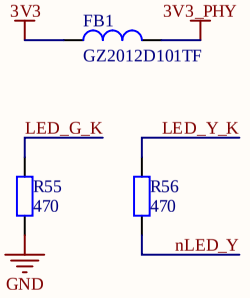
Title: iMX6 Breakout Board		
Size: A4	Page: 10_USB_Device_Host.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 10 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

Ethernet PHY



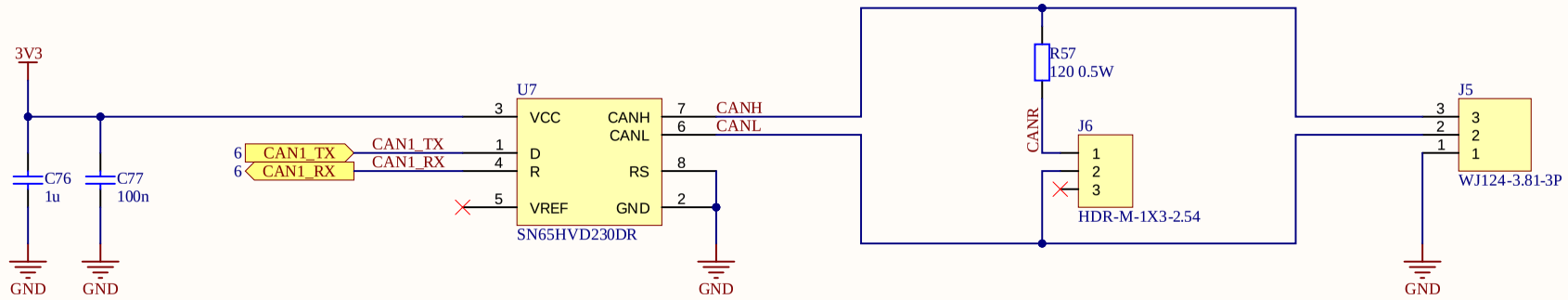


LAN8720A Configuration:
- REGOFF = GND, Internal Regulator = On
- nINTSEL = VDD2A, Internal Oscillator = Off, Pin 5 = CLKIN Input, Pin 14 = nINT Output
- MODE[2:0] = 111 (Internal Pull-Ups)
- PHYAD0 = GND



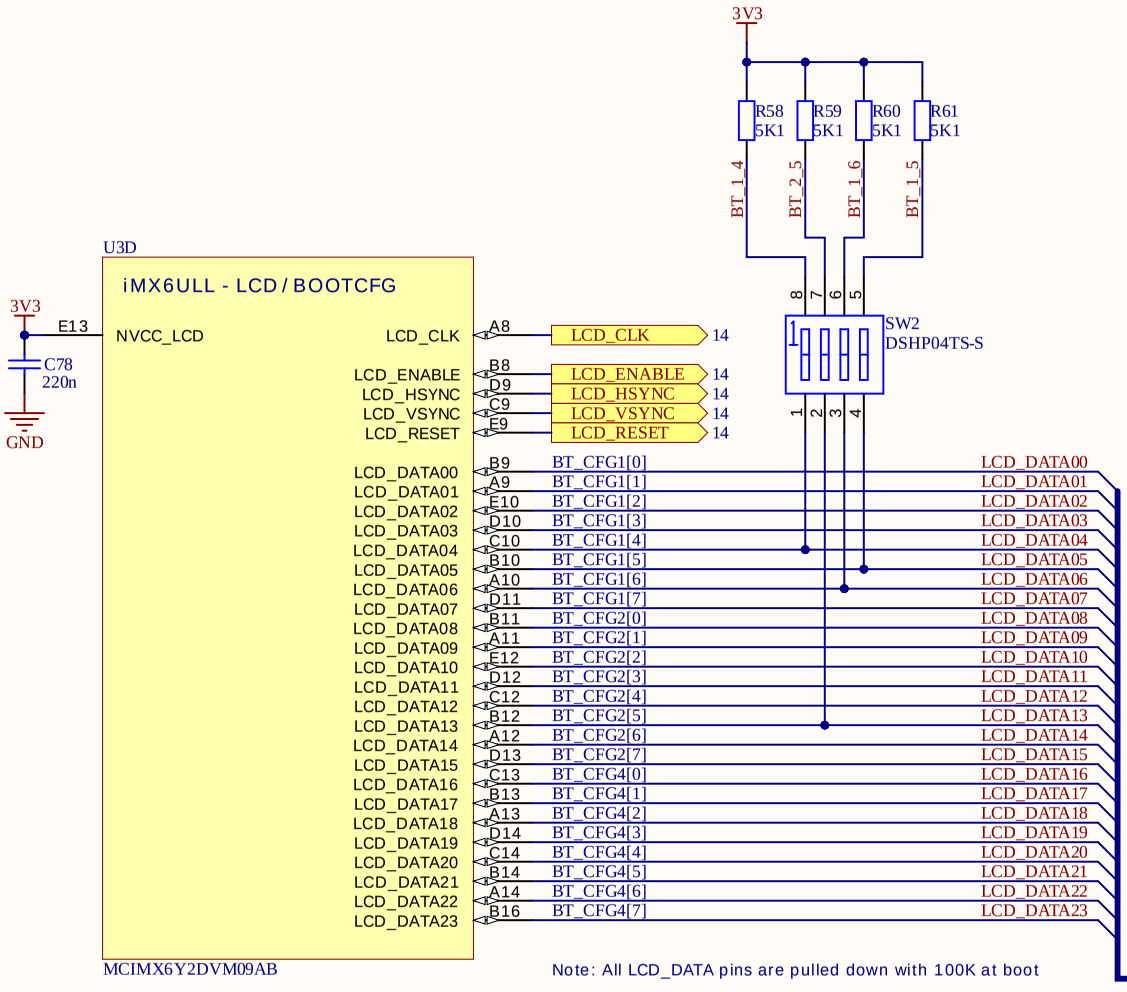
Title: iMX6 Breakout Board		
Size: A4	Page: 11_Ethernet_PHY.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 11 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

CAN PHY



Title: iMX6 Breakout Board		
Size: A4	Page: 12_CAN_PHY.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 12 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

Boot Configuration



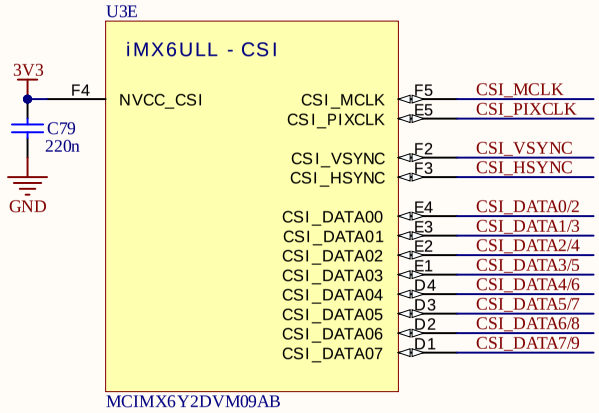
Note: All LCD_DATA pins are pulled down with 100K at boot

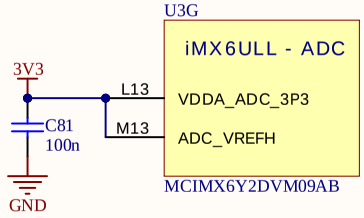
	Boot Configuration Switches			
	SW1	SW2	SW3	SW4
microSD	0	1	1	0
QSPI Flash	1	0	0	0
Serial ROM	1	0	0	1

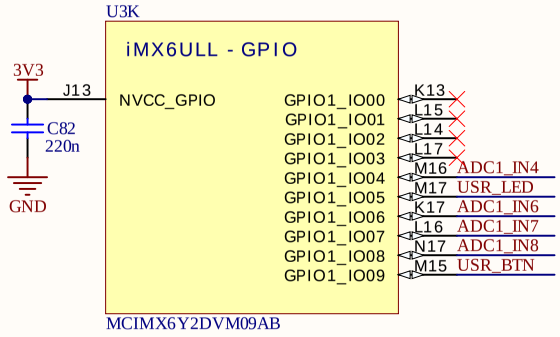
	Boot from microSD card	Boot from QSPI NOR Flash
BT_CFG1[0]	0 SD Loopback Source: SD Pad	0 DDRSMP: Default
BT_CFG1[1]	0 SD Power Cycle: Disabled	0 DDRSMP: Default
BT_CFG1[2]	0 SD Speed: Normal/SDR12	0 DDRSMP: Default
BT_CFG1[3]	0 SD Speed: Normal/SDR12	0 Interface: QSPI1
BT_CFG1[4]	0 Fast Boot: Regular	1 Boot Source: QSPI
BT_CFG1[5]	0 Boot Source: SD	0 Boot Source: QSPI
BT_CFG1[6]	1 Boot Source: SD	0 Boot Source: QSPI
BT_CFG1[7]	0 Boot Source: SD	0 Boot Source: QSPI
BT_CFG2[0]	0 Reserved	0 Reserved
BT_CFG2[1]	0 SD Voltage: 3.3V	0 Reserved
BT_CFG2[2]	0 Boot Freq ARM/DDR: 500/400MHz	0 Boot Freq ARM/DDR: 500/400MHz
BT_CFG2[3]	0 Port Select: eSDHC1	0 Full-Speed Delay: One Clock
BT_CFG2[4]	0 Port Select: eSDHC1	0 Full-Speed Phase: Non-Inverted
BT_CFG2[5]	1 Bus Width: 4-Bit	0 Half-Speed Delay: One Clock
BT_CFG2[6]	0 Calibration: 1 Delay Cell	0 Half-Speed Phase: Non-Inverted
BT_CFG2[7]	0 Calibration: 1 Delay Cell	0 Reserved
BT_CFG4[0]	0 Port Select: eCSPI1	0 Not Used
BT_CFG4[1]	0 Port Select: eCSPI1	0 Not Used
BT_CFG4[2]	0 Port Select: eCSPI1	0 Not Used
BT_CFG4[3]	0 eCSPI Addressing: 16-Bit	0 Not Used
BT_CFG4[4]	0 eCSPI Chip Select: ECPSPIx_SS0	0 Not Used
BT_CFG4[5]	0 eCSPI Chip Select: ECPSPIx_SS0	0 Not Used
BT_CFG4[6]	0 EEPROM Recovery: Disabled	0 Not Used
BT_CFG4[7]	0 Reserved	0 Not Used

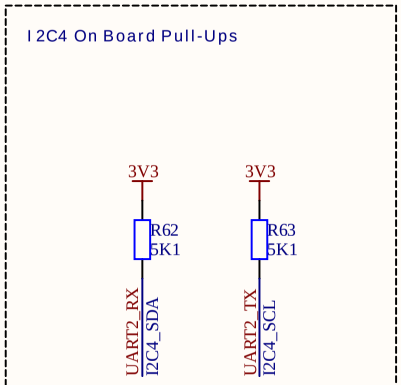
Title: iMX6 Breakout Board		
Size: A4	Page: 13_Boot_Configuration.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 13 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

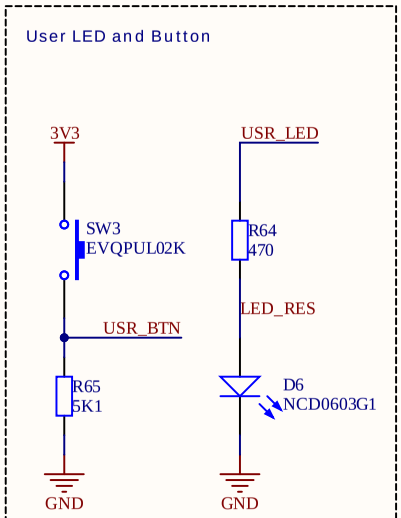
GPIO Headers

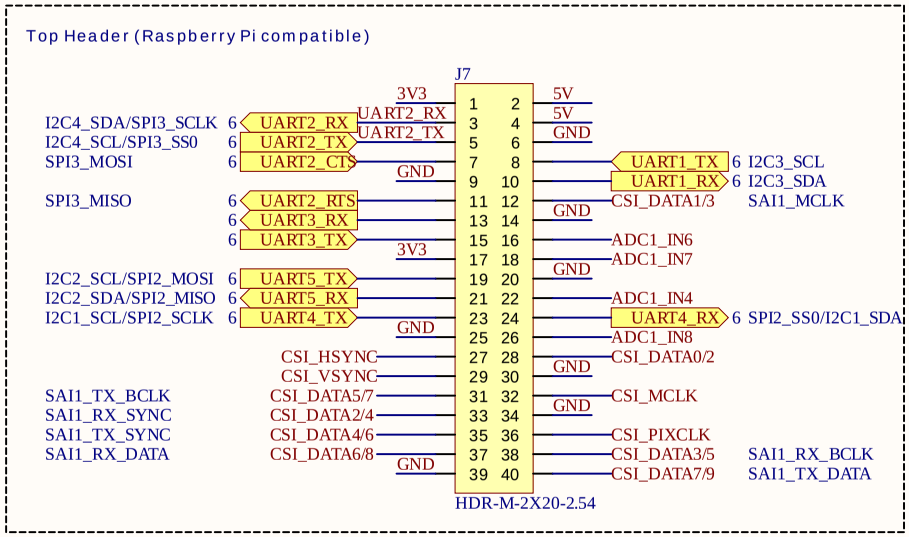


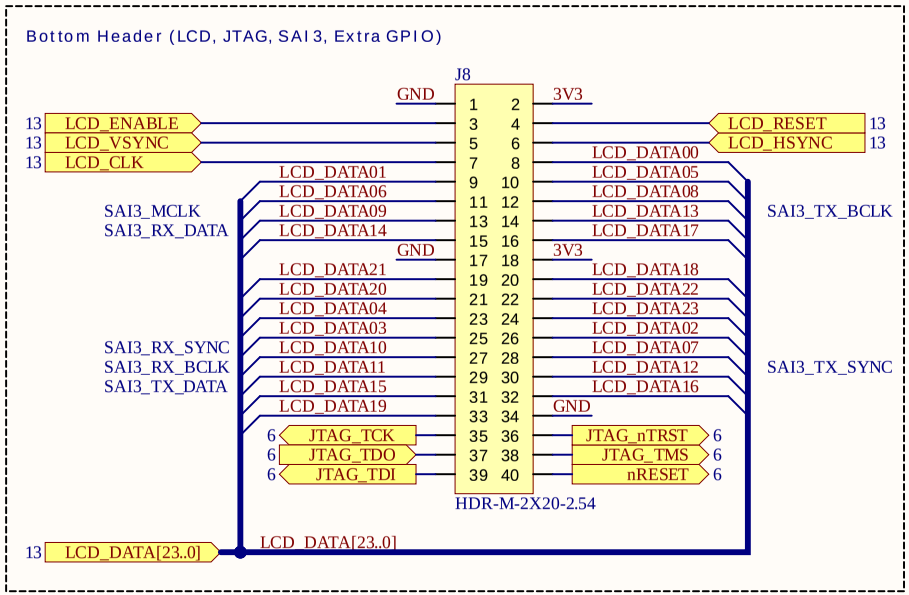












Title: iMX6 Breakout Board		
Size: A4	Page: 14_GPIO_Headers.SchDoc	Revision: v0.1
Date: 01/18/22		Sheet 14 of 14
Build: Altium Designer 22.0.2.36		Designed by: Alfonso Luna

Apéndice B

BOM - Bill of Materials

Quantity	Designator	Manufacturer	Manufacturer Part Number	Comment	Footprint	Supplier 1	Supplier Part Number 1	Supplier Stock 1	Supplier Unit Price 1
6	C2, C3, C6, C7, C47, C61	Samsung	CL10A226MQ8NRNC	22u	Capacitor_603	LCSC	C59461	24162600	
8	C9, C56, C58, C64, C72, C73, C77, C81	Samsung	CL05B104KO5NNNC	100n	Capacitor_402	LCSC	C1525	0	
37	C10, C11, C16, C17, C18, C19, C20, C24, C25, C27, C30, C35, C36, C37, C38, C40, C41, C42, C43, C44, C48, C49, C50, C51, C52, C53, C54, C55, C59, C60, C62, C63, C69, C75, C78, C79, C82	Samsung	CL05B224KO5NNNC	220n	Capacitor_402	LCSC	C16772	0	
10	C12, C13, C14, C15, C21, C22, C26, C28, C39, C65	Samsung	CL05A106MQ5NUNC	10u	Capacitor_402	LCSC	C15525	20509500	
2	C23, C57	Samsung	CL05A475KP5NRNC	4u7	Capacitor_402	LCSC	C368809	0	
4	C31, C32, C33, C34	Samsung	CL05C180JB5NNNC	18p	Capacitor_402	LCSC	C307443	0	
4	C66, C71, C74, C76	Samsung	CL05A105KO5NNNC	1u	Capacitor_402	LCSC	C29266	0	
2	C67, C68	Samsung	CL21A476MQYNNNE	47u	Capacitor_805	LCSC	C16780	47860	
1	C70	Samsung	CL05B471KB5NNNC	470p	Capacitor_402	LCSC	C26412	15000	
1	D1	Foshan NationStar Optoelectronics	NCD0603R1	NCD0603R1	LED_603	LCSC	C84263	0	
4	D2, D3, D4, D5	ON Semiconductor	ESD7104MUTAG	ESD7104MUTAG	ESD7104MUTAG	LCSC	C328026	0	
1	D6	Foshan NationStar Optoelectronics	NCD0603G1	NCD0603G1	LED_603	LCSC	C84267		
1	F1	Bourns	MF-NSMF200-2	MF-NSMF200-2	MF-NSMF200-2	LCSC	C89656	0	
1	FB1	Sunlord	GZ2012D101TF	GZ2012D101TF	Inductor_805	LCSC	C1015	0	
1	J1	Molex	1040310811	1040310811	1040310811	LCSC	C585350	0	
2	J2, J3	HRO Electronics	TYPE-C-31-M-12	TYPE-C-31-M-12	TYPE-C-31-M-12	LCSC	C165948	0	
1	J4	HanRun	HR911105A	HR911105A	HR911105A	LCSC	C12074	0	
1	J5	Kangnex	WJ124-3.81-3P	WJ124-3.81-3P	WJ124-3.81-3P	LCSC	C36780	0	
1	J6	Ckmtw(Shenzhen Cankemeng)	C124376	HDR-M-1X3-2.54	HDR-M-1X3-2.54	LCSC	C124376	0	
2	J7, J8	Ckmtw(Shenzhen Cankemeng)	C124383	HDR-M-2X20-2.54	HDR-M-2X20-2.54	LCSC	C124383		
2	L1, L2	Murata	DFE252012P-2R2M=P2	2u2	Inductor_1008	LCSC	C391305	0	
1	R1	Yageo	RC0402FR-0756KL	56K	Resistor_402	LCSC	C114756		
4	R2, R55, R56, R64	Yageo	RC0402FR-07470RL	470	Resistor_402	LCSC	C114877		
13	R3, R5, R6, R8, R9, R12, R17, R18, R19, R20, R21, R28, R54	Yageo	RC0402FR-0712K1L	12K1	Resistor_402	LCSC	C132157		
1	R4	Yageo	RC0402FR-0715K8L	15K8	Resistor_402	LCSC	C327370		
15	R7, R30, R31, R34, R35, R36, R37, R53, R58, R59, R60, R61, R62, R63, R65	Yageo	RC0402FR-075K1L	5K1	Resistor_402	LCSC	C105872		
3	R10, R11, R13	Yageo	RC0402FR-07240RL	240	Resistor_402	LCSC	C114755		
2	R15, R16	Yageo	RC0402FR-071K5L	1K5	Resistor_402	LCSC	C114759		
16	R22, R23, R24, R25, R26, R27, R29, R44, R45, R46, R47, R48, R49, R50, R51, R52	Yageo	RC0402FR-0733RL	33	Resistor_402	LCSC	C138002		
4	R38, R39, R40, R41	Yageo	RC0402FR-0749R9L	49R9	Resistor_402	LCSC	C87044		
1	R57	Panasonic	ERJ-P06F1200V	120 0.5W	Resistor_805	LCSC	C543533	0	
2	SW1, SW3	Panasonic	EVQPUL02K	EVQPUL02K	EVQPU	LCSC	C79175	0	
1	SW2	XKB Connectivity	DSHP04TS-S	DSHP04TS-S	DSHP04TS-S	LCSC	C319050	0	
2	U1, U2	Texas Instruments	TLV62569DBVR	TLV62569DBVR	SOT-23-5	LCSC	C141836	26384	
1	U3	NXP Semiconductors	MCIMX6Y2DVM09AB	MCIMX6Y2DVM09AB	NXP_BGA289_14x14_P0.8	NXP Semiconductors	MCIMX6Y2DVM09AB		
1	U4	Micron Technology	MT41K256M16TW-107:P	MT41K256M16TW-107:P	DDR3_BGA96_9x14_P0.8	LCSC	C253882	5235	
1	U5	Macronix	MX25L12835FZNI-10G	MX25L12835FZNI-10G	WSON8_6x5_P1.27_EP3.4x4.0	LCSC	C382734		
1	U6	Microchip	LAN8720A-CP-TR	LAN8720A-CP-TR	QFN24_4x4_P0.5_EP2.5x2.5	LCSC	C45223	0	
1	U7	Texas Instruments	SN65HVD230DR	SN65HVD230DR	SOIC8_3.9x4.9_P1.27	LCSC	C12084	878	
1	Y1	Seiko Epson	X1E000021012700	24MHz	Crystal_SMD4_3.2x2.5	LCSC	C89577	0	
1	Y2	Seiko Epson	Q13FC1350000300	32.768kHz	Crystal_SMD2_3.2x1.5	LCSC	C99010	0	