Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Design of a FPGA Mezzanine Card compatible with VITA 57 standard

TÍTOL DEL TFG: Design of a FPGA Mezzanine Card compatible with VITA 57 standard

TITULACIÓ: Grau en Enginyeria de Sistemes de Telecomunicació

AUTOR: Jesús Álvarez Alonso

DIRECTOR: Pere Lluís Gilabert

SUPERVISOR: José Ávila Abellán

DATA: 16 de septiembre de 2018

Resumen

En base a la necesidad existente en el sincrotrón *ALBA* de sustituir tarjetas adquisidoras que previsiblemente quedarán fuera de producción, así como la necesidad de una mejor integración y control con sistemas ya existentes, se ha realizado el desarrollo de un prototipo de FMC, VITA-57 compatible basada en la FPGA Carrier SPEC, desarrollada por la *European Organization for Nuclear Research* (*CERN*) originariamente para el sistema de timing del parque de aceleradores de partículas y a su vez basado en el estándar de sincronismo de alta precisión WhiteRabbit. Dicho estándar desarrollado por el *CERN* y el *GSI Helmholtz Centre for Heavy Ion Research*. La FMC resumida en esta memoria consta de 4 canales y un ADC por canal con resolución de 18bits, y velocidad máxima de muestreo 5MSPS.

Dedicado a mi familia, por su paciencia.

Agradecimientos

Deseo expresar mi agradecimiento a José Ávila y Xavi Serra, supervisores de este TFG. Por sus consejos y recomendaciones, ya que sin su ayuda no hubiese sido posible el correcto desarrollo del mismo. Aprecio enormemente todo el valioso conocimiento adquirido durante el diseño y realización del mismo.

Doy las gracias asimismo a Oscar Matilla, jefe de la sección de Electrónica del sincrotrón ALBA, por darme esta oportunidad y haberme permitido desarrollar esta tesis de grado dentro de las tareas de desarrollo de equipos científicos del sincrotrón ALBA.

Asimismo, agradezco de igual forma a toda la sección de Computing, en especial a la sección de Electrónica, por el soporte, ayuda y trato recibido durante mi estancia en el sincrotrón ALBA.

INTRODUCCIÓN

El objetivo del presente proyecto es el diseño, fabricación y caracterización de una FMC (FPGA Mezzanine Card) basada en el estándar VITA-57 y diseñada para funcionar con la FPGA Carrier (SPEC), constará de 4 canales con un ADC de tipo SAR por cada uno además de una entrada adicional para disparador externo. Las especificaciones y rendimiento de la misma se irán viendo en el desarrollo de esta memoria.

La FPGA Carrier que ejercerá de host para la FMC es un diseño del *CERN*, creado inicialmente con el propósito de ser usado como parte del sistema de sincronización para el proyecto White Rabbit.

White Rabbit es un sistema desarrollado por el *CERN* y el GSI Helmholtz Centre for Heavy Ion Research como solución para el sistema de timing en aceleradores de partículas, consistente en una red completamente determinista basada en Ethernet para transferencia de datos de propósito general y transferencia de tiempo con precisión de sub nanosegundos. Su uso inicial fue como una red de distribución de tiempo para control y adquisición de datos.

Aunque el propósito general de la SPEC es el sistema White Rabbit, sus características, así como la licencia OHL sobre la que se adscribe, la hacen perfecta para otro tipo de proyectos como el tratado aquí, obteniendo además la ventaja de trabajar sobre una arquitectura SBC-SPEC-FMC ya antes probada y conocida en otros proyectos realizados anteriormente en el sincrotrón ALBA, lugar de desarrollo de este TFG.

Las funciones que realizará esta FMC en sustitución de las actuales tarjetas adquisidoras serán, por ejemplo, digitalizar la salida de los electrómetros conectados a los BPM (Beam Position Monitor) de los Front Ends del acelerador hacia las beamlines. [1]

1. REQUERIMIENTOS Y SELECCIÓN DE COMPONENTES

1.1 Requerimientos

Teniendo en cuenta que lo que se va a desarrollar es una placa con 4 ADCs, se van a establecer unos parámetros mínimos necesarios por la aplicación para cubrir las necesidades mínimas del sistema. Se intentará subir en cada uno de los parámetros tanto como sea posible siempre y cuando otros parámetros importantes no se vean comprometidos por ello.

Los parámetros mínimos necesarios para las aplicaciones en las que se usará el ADC son los siguientes:

- Frecuencia de muestreo: 1 MSPS
- Resolución: 16 bits
- Entrada de señal: Single ended
- Rango de entrada: ±10V
- INL < 2 LSB
- VITA-57 compatible
- Máximo número de líneas de control FMC-FPGA libres

1.2 Tecnologías de ADC disponibles

Una de las partes más críticas para conseguir un diseño que cumpla con las especificaciones es la selección del chip ADC más adecuado, por lo que se ha puesto énfasis en investigar y seleccionar el chip idóneo que mejor se ajusta a las necesidades. [1]

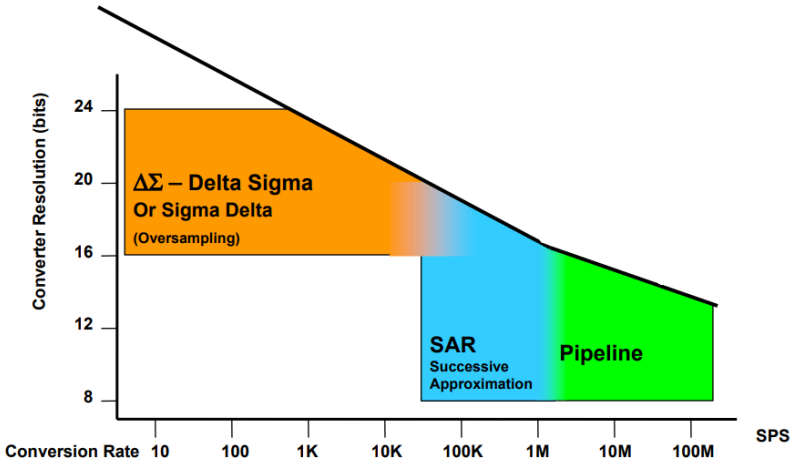


Figura 1 - Comparación entre arquitecturas disponibles en cuanto a resolución y velocidad

A continuación, se hará una breve descripción de cada una de las 3 principales arquitecturas de ADCs disponibles para ver más en detalle porque unas son más adecuadas que otras para la aplicación requerida.

- Sigma-Delta: Un modulador sigma-delta clásico de primer orden está compuesto por un integrador, un muestreador, un cuantificador uniforme de un bit y un convertidor D/A en el camino de realimentación. Es el tipo de ADC más común y lleva usándose desde los años 60. La forma de digitalizar la señal es perfectamente válida para el propósito, pero existen tecnologías que permiten alcanzar velocidades mayores cumpliendo con los criterios de resolución.

- SAR (*Successive aproximation*): Este tipo de ADC está formado por un circuito sample and hold para adquirir el voltaje de entrada. A continuación, la señal pasa a un comparador de voltaje analógico que compara Vin con la salida del DAC interno y envía el resultado de la comparación al registro de aproximación sucesiva (SAR). Si este voltaje analógico excede a Vin, el comparador hace que el SAR restablezca este bit; de lo contrario, el bit queda en 1. Luego, el siguiente bit se establece en 1 y se realiza la misma prueba, continuando esta búsqueda binaria hasta que se haya probado cada bit en el SAR. El código resultante es la aproximación digital del voltaje de entrada muestreado que finalmente es enviado por el SAR al final de la conversión (EOC).

- Flash: También conocidos como ADCs de conversión directa o *pipelined* ADCs, basan su funcionamiento en una escalera de voltaje lineal con un comparador en cada nivel de la escalera para comparar el voltaje de entrada con voltajes de referencia sucesivos. Suelen alcanzar velocidades muy altas y son adecuados cuando la aplicación no requiere de un *delay* bajo.

Teniendo en cuenta que para esta aplicación es crítico que exista un sampleo simultáneo de los 4 canales y que se necesitará al menos 1MSPS@16bit de frecuencia de muestreo la opción más adecuada será un ADC de tipo SAR por lo que la búsqueda se ha centrado principalmente en este tipo de ADCs. Se descartan los Flash porque no se necesita tanta velocidad y si más resolución si es posible además de una muy buena linealidad. Los SAR son los más lineales de todos.

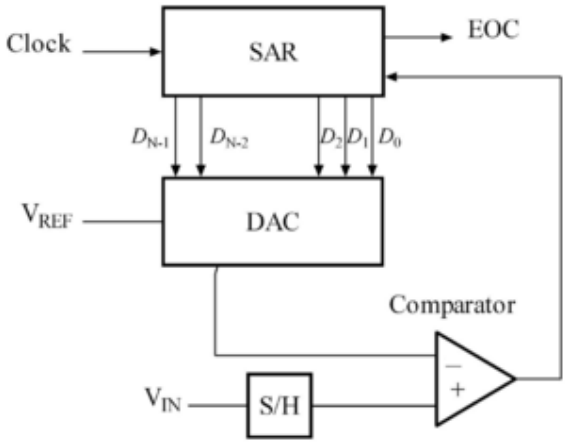


Figura 2 - Arquitectura de un ADC de aproximación sucesiva o SAR.

1.3 Selección y estudio de componentes

1.3.1 Selección de ADC

Teniendo en cuenta el análisis hecho en la sección anterior y tras una profunda búsqueda entre todos los fabricantes se decidió que el chip más adecuado era el AD7960 de la serie PulSAR de Analog Devices, el cuál es un conocido fabricante de electrónica especialmente reputado en conversores AD. Este chip tiene como características principales las siguientes:

Throughput: 5 MSPS
Resolución: 18 bits
INL: ±0.8 LSB (typical), ±2 LSB (maximum)
DNL: ±0.5 LSB (typical), ±0.99 LSB (maximum)
Rango dinámico: ±4.096 V o ±5 V
1 canal de entradas diferenciales
Salida de datos serial LVDS

Otras características resaltables son la posibilidad de funcionar en dos modos diferentes de reloj:

-*Self-clocked*: Este modo de funcionamiento está diseñado para minimizar el número de líneas entre el ADC y el host digital, de tal manera que solo son necesarios 3 pares diferenciales para controlar por completo el dispositivo. Estos son el reloj (CLK), la salida de datos (D), y la señal de conversión (CNV)

-*Echoed-clocked*: Este modo se diferencia principalmente del anterior en que además se añade otro par diferencial de salida (DCO) hacia la FPGA que contiene una copia del reloj, de este modo la FPGA, dispone de una copia de reloj de retorno para la evaluación de los datos sin tener en cuenta la problemática impuesta por el retardo añadido debido a la longitud de las pistas, consiguiendo de esta forma una mejor sincronización entre la señal de datos serie y el reloj que si se adquiriesen los datos utilizando como referencia el mismo reloj generado por la propia FPGA.

Se trabajará en modo echoed-clocked pero utilizando el clock de la FPGA para evaluar los datos en vez de la copia de clock de retorno, como se justificará en el capítulo referido a la programación del VHDL propio.

El chip dispone además de 4 pines de configuración estática y otro de referencia de entrada para establecer el margen dinámico y otras opciones que se detallan en la Tabla 1. En este caso la opción que más interesa es utilizar el modo X0010 para hacerlo funcionar con margen dinámico de ±5 V, ya que según el fabricante es la mejor opción para tener el mejor rendimiento de margen dinámico y la mejor SNR. La mejora en SNR respecto de usar el modo ±4.096 V es de 1.7 dB.

EN3	EN2	EN1	EN0	REFIN	Reference Mode Description
X[1]	0	0	0	X[1]	Power-down mode. Everything is powered down, including the LVDS interface.
X[1]	0	0	1	0 V	Interface powered up. Reference buffer disabled. An external 5 V reference is applied to the REF pin. Connect REFIN to 0 V in this mode. The bandwidth of the input sampling network is set to 28 MHz.
X[1]	0	0	1	2.048 V	Internal reference buffer enabled. An external 2.048 V reference applied to REFIN pin is required. A buffered 4.096 V reference is available on the REF pin. The bandwidth of the input sampling network is set to 28 MHz.
X[1]	0	1	0	0 V	Internal reference buffer disabled. Drive the REF pins with a 4.096 V external reference. Connect REFIN to 0 V in this mode. The bandwidth of the input sampling network is set to 28 MHz.
X[1]	0	1	1	0 V	Snooze mode.[2] LVDS powers down. The chip is unresponsive to CNV± start pulses. The wake-up time is fast (5 µs) when EN3 to EN0 are set to XX01 or XX10. Ensure that the CNV± start pulse is low when transitioning in and out of this mode.
0	1	0	0	X[1]	Test patterns output on LVDS. The ADC output is not available on the interface.
1	1	0	0	X[1]	Invalid mode.
X[1]	1	0	1	0 V	Reference buffer disabled. Drive the REF pins with a 5 V external reference. The bandwidth of the input sampling network is set to narrow (9 MHz).
X[1]	1	0	1	2.048 V	Internal reference buffer enabled and driving REF pin to 4.096 V. The bandwidth of the input sampling network is set to narrow (9 MHz).
X[1]	1	1	0	0 V	Reference buffer disabled. Drive the REF pins with a 4.096 V external reference. The bandwidth of the input sampling network is set to narrow (9 MHz).
X[1]	1	1	1	0 V	Snooze mode.[2] LVDS powers down. The chip is unresponsive to CNV± start pulses. The wake-up time is fast (5 µs) when EN3 to EN0 are set to XX01 or XX10.

[1] X = don't care.
[2] The snooze mode is not useful when the internal reference buffer is used because the fast wake-up is not possible due to the settling of the internal reference buffer.

Tabla 1 – Configuraciones de pines posibles para cambio de rango dinámico y otras opciones.

Aquí cabe comentar que para el caso que corresponde, es decir, segunda fila de la Tabla 1, aparece el pin EN3 marcado con una nota que dice: "no importa" respecto a donde conectar dicho PIN. Esto es algo confuso que no es del todo cierto. Debido a ello más adelante se tuvieron problemas hasta que se localizó en otra sección del *datasheet* que el PIN EN3 habría de estar conectado a 1 para activar la salida de voltaje en modo común del ADC. Fue necesario hacer por ello un *rework* del que más adelante se hablará.

Dado que la señal de entrada en el conector de la FMC es de ±10 V se ha de establecer una ganancia de ½ en el lazo de realimentación del driver para ajustar la señal de entrada al margen dinámico del ADC. En la siguiente sección referida al driver se mostrará como ajustar esta ganancia.

El AD7960 precisa además de 2 fuentes de alimentación diferenciadas que son conectadas a la placa a través de múltiples pines para reducir los efectos inductivos. Una primera fuente de 5 V alimenta la parte analógica y otra de 1.8 V se encarga de alimentar la parte digital. Ambas tensiones son generadas por 2 reguladores que convierten los 12 V y 3.3 V provenientes de la SPEC atendiendo a la especificación VITA-57 a 5 V y 1.8 V respectivamente.

También será necesario proveer al ADC con una referencia de tensión lo más estable posible para tener el menor ruido de cuantificación posible. Además, se genera una tensión en modo común a la mitad del rango dinámico del ADC, utilizada por el driver a través de una línea entre ambos con un seguidor de tensión entre medias. Siempre que se hable de algo referido al ADC y su driver esto se ha de tener en cuenta que es por cada uno de los 4 canales que tiene la FMC.

1.3.2 Selección de driver

El ADC seleccionado por ser de entradas diferenciales exige la inclusión de un circuito de entrada para acondicionar el tipo de señal requerido por las

especificaciones (desbalanceada), al tipo de señal requerido por el chip ADC (balanceada). En lugar de trabajar en el diseño de una red para conseguir esto y siguiendo las recomendaciones propuestas por el fabricante, se ha decidido utilizar un driver que haga la citada función a la vez que sirva para ajustar la señal de entrada al margen dinámico del ADC mediante el ajuste de ganancia.

El chip escogido para realizar esta función es el ADA4932 también del fabricante Analog Devices. Teniendo en cuenta que será alimentado con una tensión Vs = 5V, este permitirá un voltaje de entrada de −Vs + 0.2 V a +Vs − 1.8 V, la ganancia diferencial ajustable se establece con una simple retroalimentación de cuatro resistencias externas que determinan la ganancia de bucle cerrado del amplificador.

El cálculo de las resistencias del lazo de realimentación es sencillo, siendo unos valores adecuados Rf=4,22 kΩ y Rg=8,45 kΩ. Teniendo en cuenta que se trata de establecer la ganancia en 0.5, hay muchas relaciones que podrían servir para este propósito. Se han elegido esos rangos concretos de tal manera que las corrientes que pasan por ellas sean lo suficientemente bajas como para que la potencia no exceda los 0.063W, máxima admitida por la familia SMD 0603, usada en la gran mayoría del prototipo.

Otro motivo por el que se han elegido esos rangos es porque si se pretende realizar una red de entrada para adaptar en alta impedancia, no será posible a menos que esas resistencias sean de un orden similar del que se quiere adaptar. El cálculo para establecer dichas resistencias viene determinado por:

$$\frac{V_{out}}{V_{in}} = \frac{R_F}{R_G}$$

1.3.3 Selección de reguladores

Como se dijo anteriormente para alimentar todos los elementos que conforman la FMC serán necesarias 3 tensiones distintas. 12V, 5V, y 1.8V. Los 12V vienen directamente de la Spec, pero las tensiones de 5V y 1.8V será necesario generarlas a partir de los 12V y los 3.3V que vienen de la Spec. Debido a que VITA-57 establece un tamaño máximo de PCB y la placa tiene una gran cantidad de componentes debido a los 4 canales, se buscaron reguladores específicos con entradas-salidas ya prestablecidas para no ocupar más espacio mediante el uso de redes de realimentación para fijar la tensión de salida, si bien es cierto que al final se necesitaron condensadores adicionales para hacer estables dichos reguladores y el ahorro en espacio no fue tanto.
Para regular a 5V se decidió utilizar el ADM7150ACPZ-5 y para regular a 1V8 el ADM7170ACPZ-1.8 con 800mA y 500mA de corriente máxima de salida respectivamente, suficiente para alimentar todos los dispositivos a los que alimentan en condiciones simultaneas de máximo consumo.

1.3.4 Selección de referencia de tensión y seguidores de tensión

Otros componentes importantes que juegan papeles bastante importantes en la precisión de los ADC serán los dedicados a la generación de una referencia de tensión de 5V de alta estabilidad, así como todos los seguidores de tensión que se pongan para evitar los posibles efectos de carga en partes susceptibles de ello.

En cuanto a la referencia de tensión se buscó una lo más precisa y estable posible a diferentes temperaturas y se decidió que la ADR4520 sería una opción acertada, con un coeficiente de temperatura máximo de 2ppm/ºC y mostrando un comportamiento muy poco cambiante en todo el eje de temperaturas. Así mismo la regulación de línea de dicha referencia de tensión es excelente, con un error muy pequeño, teniendo en cuenta que la corriente máxima de salida es de tan solo 10mA y que se desconoce la corriente que se consume para ser utilizada como referencia por cada ADC, se añadió un seguidor de tensión para cada uno de los 4 ADC que conforman la placa a modo de evitar posibles efectos de carga que serían críticos en la precisión de las adquisiciones.

También se han añadido siguiendo las recomendaciones del fabricante seguidores de tensión para las salidas de voltaje en modo común de los ADC hacia los drivers.

1.3.5 Relés, drivers de relés y port expander.

La placa se ha diseñado de tal forma que es posible cambiar entre 2 impedancias de entrada distintas a través de la activación de unos relés que activan dos resistencias de entrada de baja impedancia para conseguirlo. De esta manera se puede establecer la FMC en alta o baja impedancia y minimizar según sea conveniente el efecto de carga al realizar alguna medición.

Los relés se han escogido de tal forma que sea posible alimentarlos con 12V y que además el tamaño no sobrepase 9.5mm de alto. VITA-57 establece una altura máxima de componentes de 9.5mm.

La razón para decidir que se han de alimentar a 12V es doble, por una parte, se separan las líneas de voltaje sobre las que se alimentan y que previsiblemente serán ruidosas debido a la naturaleza de los relés en el momento de conmutación, de otras que serán utilizadas para alimentar elementos más sensibles a la estabilidad en la alimentación, en este caso los reguladores actúan como estabilizadores ante una eventual activación.

Aunque es cierto que la impedancia de entrada no es algo que se necesite cambiar durante la adquisición, normalmente y aunque depende del escenario, no suelen tener mucho sentido las mediciones que se realizan en una adquisición en el momento de cambiar dicha impedancia, por lo que no es crítico que los relés estén separados de las líneas de 5V por ejemplo, pues el ruido de los relés solo afectará a las líneas de tensión en el momento de activación de éstos. Aun así, la razón de más peso ha sido que el consumo de dichos relés se carga sobre

la tarjeta portadora reduciendo con ello las exigencias de los reguladores de la FMC y el ruido térmico adicional generado por ello.

Los relés tampoco pueden ser excesivamente grandes en anchura ya que el espacio en la PCB es bastante limitado, han de soportar la corriente de conmutación y la tensión de activación tiene que ser lo suficiente como para que el driver del *port expander* la pueda proporcionar. Los que mejor se adaptan a las limitaciones son los MS12-1A87-75D del fabricante MEDER, con 8.75V de tensión de conmutación, tensión de alimentación a 12V y corriente máxima conmutada de 500mA. Para activar los relés se hace necesario el uso de los transistores NPN BC817DS, los cuales amplifican la tensión de salida que da el *port expander* para alcanzar las tensiones de conmutación y corrientes exigidas por éstos.

Respecto al *port expander* se ha utilizado el MCP23008-E/SS del fabricante Microchip, existente en 2 versiones: con control SPI y con control I2C. Por simplicidad se utilizará I2C para ser controlado desde la FPGA. Las señales SCL y SDA, ya están incluidas en el estándar VITA-57, sin ocupar con ello pistas libres adicionales será esta la variante utilizada. El *port expander* es de 8 bits y solo se hará uso de 4 canales de salida para activar los 4 grupos de 2 relés que crean cada una de las 2 posibles configuraciones de impedancia de entrada en cada canal.

1.3.6 Selección de conectores

Para los conectores de las entradas, así como del disparador, se deben seleccionar aquellos que sean compatibles con cable coaxial, teniendo en cuenta que la velocidad máxima de muestreo del ADC es de 5MSPS para cumplir el factor de Nyquist la frecuencia de la señal máxima que se puede muestrear a la entrada será de 2.5 MHz, si bien cuanto mayor sea este margen tanto en cable como en conector, menos abrupta será la caída por atenuación de las frecuencias más altas. Se estudió el uso de SMA, el ancho de banda es más que suficiente y cumple perfectamente con las prestaciones necesarias, pero a pesar de ello al final se optó por utilizar conectores de tipo Lemo, que tienen un coste bastante más elevado que los SMA, pero ofrecen una versatilidad y unas prestaciones más altas, dicho tipo de conectores son muy utilizados en el ámbito científico.

Así mismo dado que la placa se basa en el estándar VITA-57 esto impone que el conector de enlace entre la placa portadora y la FMC sea el SAMTEC ASP-134604-01, conector con 160pines y estándar VITA-57, aunque se ha de indicar que en este prototipo se usará la versión de 80pines (LPC) dado que la Carrier es LPC también.

El estándar VITA-57 define dos maneras de uso de la interfaz según el número de líneas que se utilicen entre la FMC y el host digital. Estas son: HPC, o *High Pin-Count* y LPC o *Low Pin-Count*.

El conector LPC proporciona 68 señales definidas por el usuario, de terminación única o 34 pares diferenciales definidos por el usuario. El conector HPC

proporciona 160 señales definidas por el usuario, de terminación única (u 80 pares diferenciales definidos por el usuario), 10 pares de transceptores en serie y relojes adicionales. Los conectores HPC y LPC usan el mismo conector mecánico.

Para poder acceder externamente a las líneas sobrantes que se utilizan en esta configuración LPC, se han añadido también dos conectores FFC de 40 pines cada uno.

1.3.7 Otros componentes

Se han añadido condensadores de desacoplo en las alimentaciones, referencias, líneas de tensión común y sitios potencialmente sensibles de los componentes.

Los condensadores de desacoplo se ponen cerca de los circuitos integrados para evitar generar ruidos en los nodos de alimentación cuando estos circuitos integrados producen picos de corriente de consumo debido a su propio funcionamiento. De esta forma la carga instantánea demandada por el integrado es proporcionada por el condensador evitando así la fluctuación de voltaje en el nodo.

Todos los componentes pasivos pertenecen a la familia SMD 0603, y en mucha menor medida SMD 0805. Así mismo se han utilizado diodos MMBZ12VDL en la entrada para proteger las entradas de descargas electroestáticas, estos diodos limitan entre ±11.4V y ±12.6V la entrada de las señales analógicas.

Inmediatamente a la entrada del *trigger* también se ha puesto un driver LVDS para transmitir la señal de *trigger* hacia la FPGA, el driver es el SN65LVDS1DBV del fabricante Texas Instruments.

2. DISEÑO DE LA PLACA

Para el diseño de la placa tanto en la parte del esquemático como en el diseño de la PCB, se hizo uso del software Altium, un software muy potente que permite tanto plasmar el desarrollo en forma de esquemático, para tener una mejor perspectiva de la parte funcional de la PCB, como trasladar posteriormente estos componentes a un modo para el diseño de la propia PCB.

2.1 Diseño del esquemático

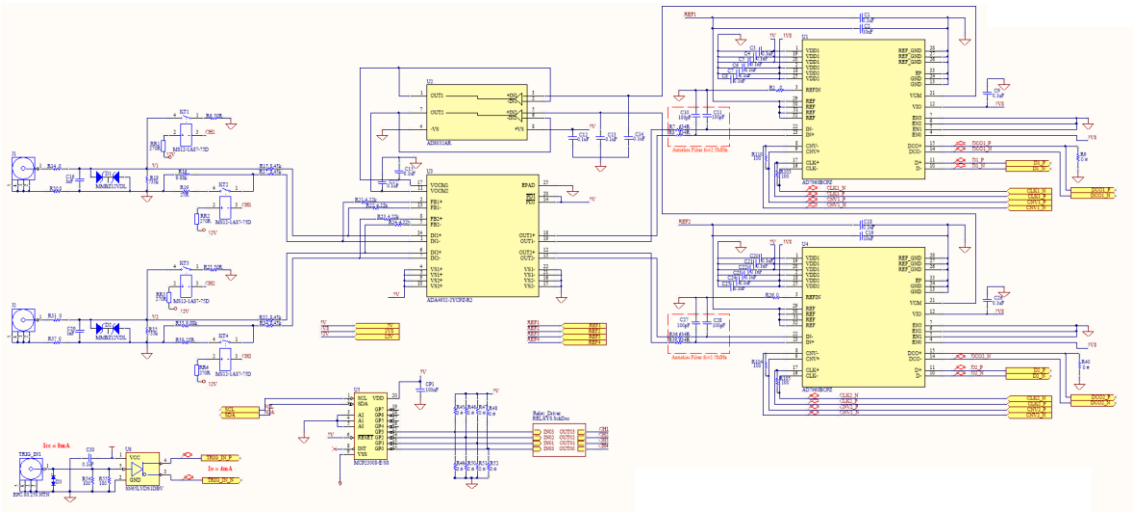


Figura 3 – Ramas pertenecientes a entradas 1 y 2, entrada de trigger y port-expander.

En la Figura 3 se pueden apreciar los canales 1 y 2 en el esquemático, con sus correspondientes redes de entrada, su driver, para el cual se ha utilizado la versión que integra 2 amplificadores operacionales en un mismo chip, además de los ADC y un filtro *antialiasing* entre driver y ADC para eliminar las réplicas generadas en el proceso de muestreo. También se pueden observar los seguidores de tensión colocados a la salida de la tensión en modo común de los ADCs. En la parte inferior se puede ver la red de entrada para el disparo, junto con su diodo de protección para electricidad estática y su driver LVDS para enviar la señal de disparo externa directa a la FPGA. En la parte inferior también se puede observar el *port expander*, con alimentación a 5 V y con 4 de sus salidas conectadas a los transistores que harán de driver de los relés (estos transistores puestos en otra hoja de esquemático y representados como una caja blanca a la salida del driver de la Figura 3). También se pueden observar un conjunto de resistencias de 0 Ω, las cuales algunas serán montadas en una primera fase para manejar manualmente los relés conectando a 0 V o 5 V mediante un puente, la resistencia de base de los transistores. Los canales 3 y 4 son idénticos al 1 y 2, pero se han obviado en la Figura 3 para la mejor visualización de esa hoja del esquemático.

2.1.1 Red de entrada

Para controlar las 2 posibles configuraciones de impedancia de entrada se hace uso de un *port expander* que controla unos relés que activan las resistencias de baja impedancia.

Para adaptar la señal de entrada se ha diseñado una red para controlar la impedancia de entrada mediante la selección de diferentes resistencias a fin de mejorar la adaptación de la señal. Para controlar las 2 posibles configuraciones de impedancia de entrada se hace uso de un port expander que controla unos relés que activan las resistencias de baja impedancia. Existen dos configuraciones: baja impedancia (50 Ω) y alta impedancia (10 kΩ). En la Figura 4 se puede ver la red de entrada con el diodo de protección electrostática, así como con los 2 relés que al ser activados conectan las resistencias de baja impedancia, siendo por estas y no por las de mucha más alta impedancia que seguirán conectadas, por las que circule la casi totalidad de la corriente al ser las de alta impedancia poco significativas respecto a las de baja.

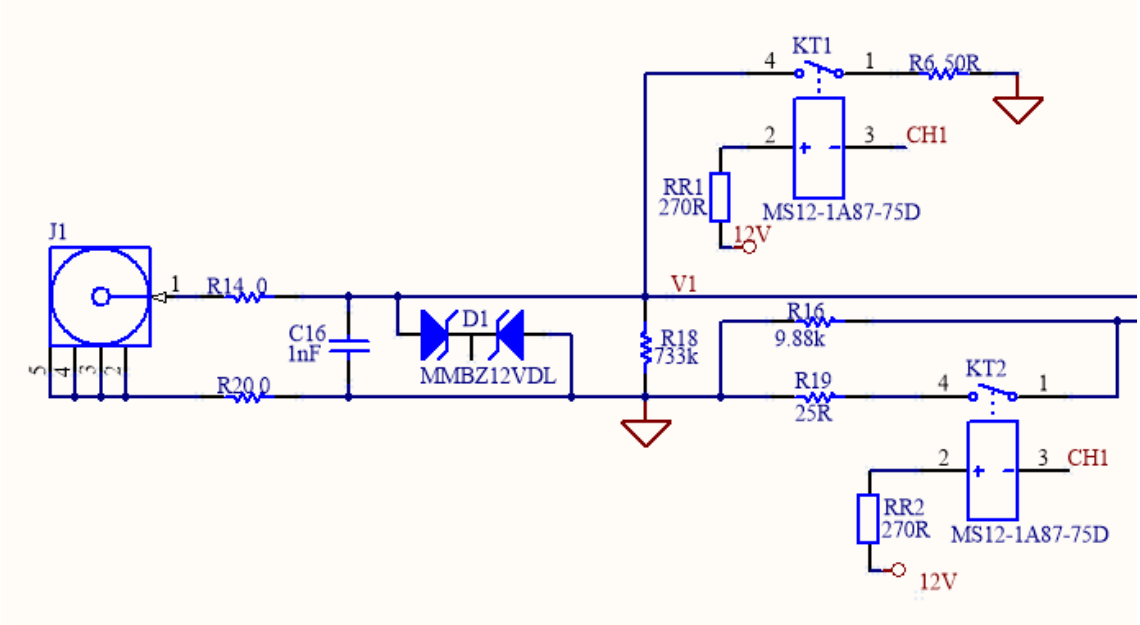


Figura 4 – Red de adaptación de entrada del canal 1

A continuación, se harán los cálculos propuestos por Analog Devices para ambos casos para terminar una entrada *single-ended* correctamente.

2.1.1.1 Cálculo de resistencias para entrada en baja impedancia

Método propuesto en el *datasheet* del fabricante del driver para realizar la correcta adaptación de la entrada a 50 Ω haciendo referencia al esquemático de la Figura 5.

1. Cálculo de la impedancia de entrada.

$$R_{in} = \left(\frac{R_G}{1 - \frac{R_F}{2(R_G + R_F)}} \right) = \left(\frac{8{,}45 * 10^3}{1 - \frac{4{,}22 * 10^3}{2(8{,}45 * 10^3 + 4{,}22 * 10^3)}} \right)$$

$$= 10141{,}6\Omega \approx 10k\Omega$$

2. Para adaptar a 50Ω se utilizará la siguiente expresión para hallar la resistencia de terminación:

$$R_t||10141.6\Omega = 50\ \Omega$$

$$R_t = \frac{50 * R_{in}}{R_{in} - 50} = 50{,}24\ \Omega$$

$$R_t = 50{,}24\ \Omega$$

3. Según el *datasheet* del driver:

   "Para compensar por el desequilibrio de las resistencias de ganancia, agregue una resistencia de corrección (Rts) en serie con Rg en el ciclo inferior. Rts es el Thevenin equivalente de la resistencia de origen, Rs y la terminación resistencia, Rt, y es igual a Rs || Rt."

   Por tanto:
   $R_{ts=}R_t||R_s$
   siendo $R_s = 50\Omega$ y $R_t = 50.24\ \Omega \approx\ 50\ \Omega$

$$R_{ts} = \frac{R_t * R_s}{R_t + R_{ts}} = \frac{50 * 50}{50 + 50} = 25\ \Omega$$

4. La resistencia de Thevenin equivalente será:

   $R_{th} = R_{ts} = R_t||R_s = 25\ \Omega$

5. Por tanto, los ramales de entrada para el caso de impedancia de entrada de 50 Ω quedarían como en la Figura 5:

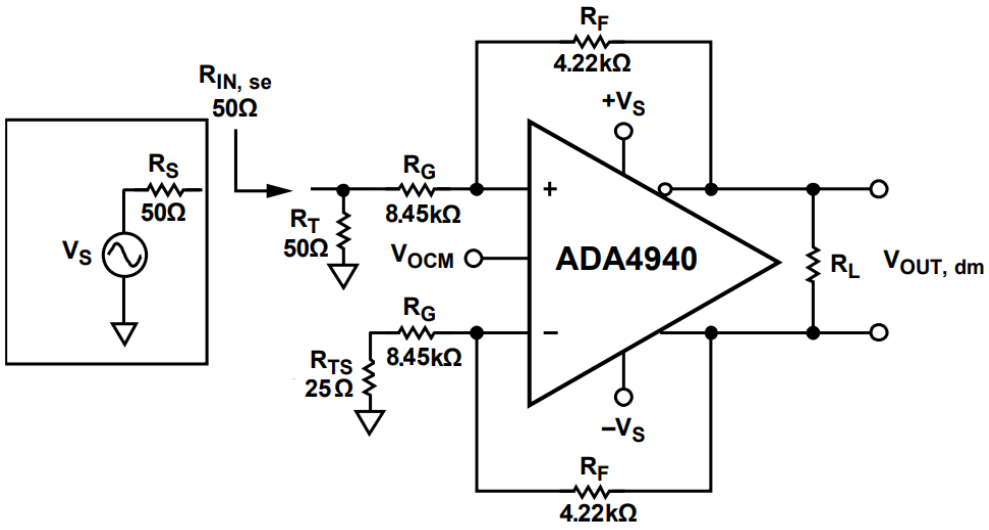


Figura 5 – Modelo para red de entrada en baja impedancia.

2.1.1.2 Cálculo de resistencias para entrada en alta impedancia

Para el caso de adaptar en alta impedancia se procederá de la misma forma que en el caso anterior, pero sustituyendo el valor de la impedancia de adaptación de 50 Ω por 10 kΩ.

Cálculo para entrada en baja impedancia

1. Cálculo de la impedancia de entrada.

$$R_{in} = \left(\frac{R_G}{1 - \frac{R_F}{2(R_G + R_F)}} \right) = \left(\frac{8.45 * 10^3}{1 - \frac{4.22 * 10^3}{2(8.45 * 10^3 + 4.22 * 10^3)}} \right)$$

$$= 10141{,}6\ \Omega \approx 10\ k\Omega$$

2. Para adaptar a 10 kΩ se utilizará la misma expresión que en el caso anterior, pero igualando a la impedancia para la que se quiere adaptar.

$$R_t || 10141.6\Omega = 10\ \text{k}\Omega$$

$$R_t = \frac{50 * R_{in}}{R_{in} - 50} = 716{,}21\ k\Omega$$

$$R_t = 716{,}21\ k\Omega$$

3. Igual que en el caso anterior y según el *datasheet* del driver:

$R_{ts=} R_t || R_s$
siendo $R_s = 10k\Omega$ y $R_t = 716.21\ k\Omega$

$$R_{ts} = R_t || R_s = \frac{R_t * R_s}{R_t + R_{ts}} = 9{,}86\ k\Omega$$

4. La resistencia de Thevenin equivalente será:

$$R_{th} = R_{ts} = R_t || R_s = 9{,}86\ k\Omega$$

5. Por tanto, los ramales de entrada para el caso de impedancia de entrada = 10 kΩ quedarían como en la Figura 6:

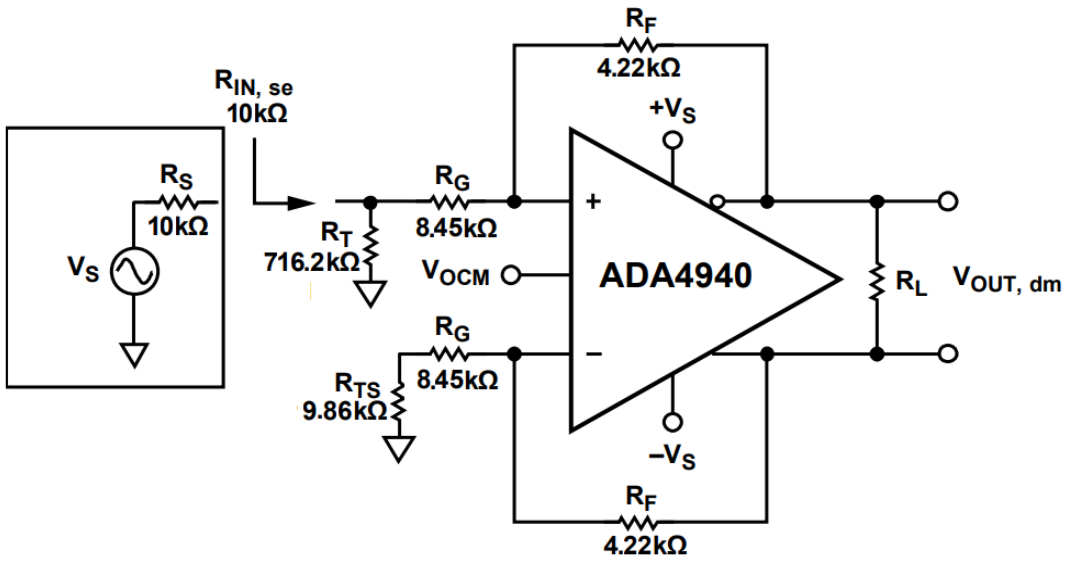


Figura 6 – Modelo para red de entrada en alta impedancia.

En la Figura 5 y Figura 6 se señalan los valores teóricos, aunque finalmente esos nos serán los que se pongan en el circuito real, pues se tendrán que buscar las impedancias comerciales que más se aproximen, siendo finalmente puestas en ambos casos resistencias de 1% de tolerancia de los siguientes valores:

$Para\ baja\ impedancia$: $R_t = 50\ \Omega$ y $R_{ts} = 25\ \Omega$ que en la Figura 4 se representan por R6 y R19 respectivamente.

$Para\ alta\ impedancia$: $R_t = 733\ k\Omega$ y $R_{ts} = 9{,}88\ k\Omega$ que del mismo modo en la Figura 4 se representan por R18 y R16 respectivamente.

2.1.2 Port expander y relés

Unos sencillos cálculos han sido necesarios para asegurar la correcta activación de los relés mediante el uso del *port expander* que es manejado a través de SPI desde la FPGA y cuyas salidas han sido conectadas a un transistor NPN que hace las veces de driver y suministra la corriente necesaria para mantener el relé activo.

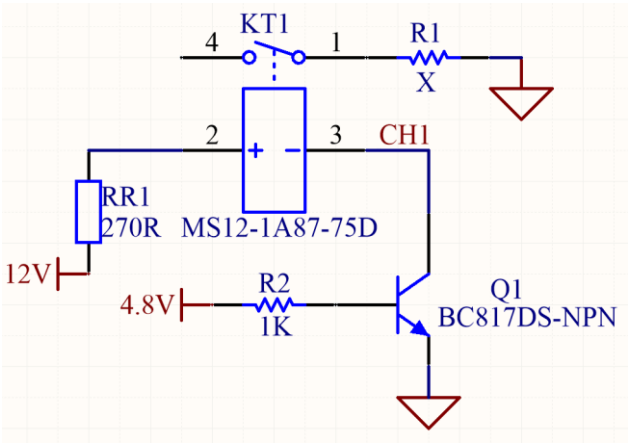


Figura 7 – Detalle de driver de relé y relé.

Teniendo en cuenta que el *port expander* se alimenta a 5 V las salidas activadas tendrán un voltaje de salida de 4.8 V, por la caída de 0.2 V que impone el propio

circuito de salida del *port expander*. Se calcula la corriente de base teniendo en cuenta que el voltaje en la base del transistor es 0,7 V.

.

$$I_b = \frac{4{,}8 - 0{,}7}{10^3} = 4{,}1\ mA$$

Se calcula cuál será la Ic en el peor caso:

$$I_{cmax} = \frac{12}{270} = 44\ mA$$

Para esas corrientes de colector y de base se mira entre la curva 8 y 9 de la Figura 8 Con esto se asegura que se hace funcionar el transistor en la zona de saturación y no en la zona activa.

En esta configuración el transistor se utiliza como interruptor, de tal manera que este siempre trabaja en saturación o corte.

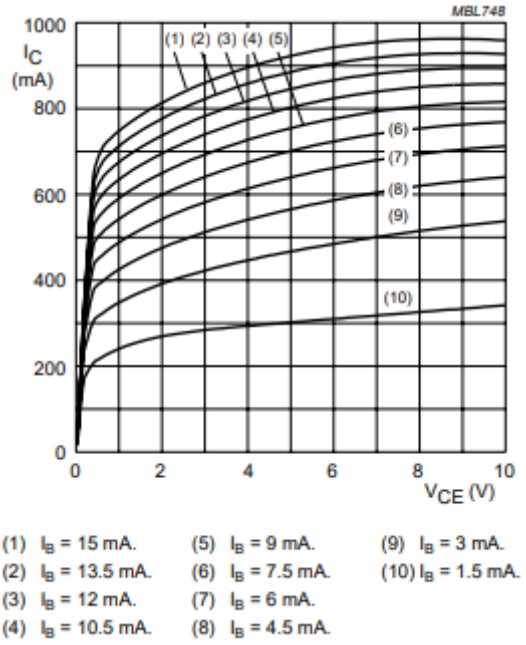


Figura 8 – Corriente de colector para distintas corrientes de base en función de Vce

La Vce para esas corrientes será como mucho de 0.1 V por lo que la tensión de activación será de:

$$V_{ACT} = 12 - V_{CE} = 11.9\ V$$

El relé tiene una tensión de *Pull-In* mínima de 8.4 V por lo que al superar esté umbral se activará actuando de este modo las resistencias de entrada en baja impedancia.

2.1.3 Filtro antialiasing

El filtro *antialiasing* añadido para eliminar las réplicas producidas en todo proceso de muestreo es de un polo, suficientemente selectivo para la aplicación y menos penalizado en cuanto a ganancia y fase que no si se utilizase uno de más polos. La frecuencia de corte se ha calculado para 2.5 MHz, ya que es la frecuencia

máxima que se podrá muestrear con un ADC de 5 MSPS, según el criterio de Nyquist, $f_s \geq 2B_w$ La frecuencia de corte para un filtro paso bajo de un polo viene dada por:

$$f_c = \frac{1}{2\pi RC}$$

$$R = \frac{1}{2\pi f_c C} = \frac{1}{2\pi f * 2.5 * 10^6 * 100 * 10^{-12}} = 636.61\ \Omega$$

Fijando el valor de capacidad del condensador por uno conocido y existente, en el caso práctico se ha escogido 100 pF, se despeja la R de la ecuación resultando un valor de 636.61 Ω. Se seleccionan al final resistencias de 634 Ω con 1% de tolerancia, pues son las más próximas que se han podido encontrar cerca del valor teórico y la variación de la frecuencia de corte es insignificante respecto del valor teórico al ser tan solo 2.61 Ω de diferencia entre las calculadas y las utilizadas finalmente.

2.1.4 Reguladores y referencia de voltaje

En los dos reguladores de voltaje se han colocado diodos BAS40-05 entre la entrada y la salida para aliviar la corriente típica que se suele dar en muchos reguladores. Dicha corriente circula de la salida a la entrada en el momento de desconexión de la alimentación, al imponerse 0 V en la entrada y quedar la salida con 1.8 V o 5 V momentáneamente debido al efecto de retención de voltaje que generan los condensadores de salida, por ello se genera una corriente hacia atrás que atraviesa el regulador degradándolo con el tiempo. También se ha hecho uso de condensadores de distintos valores dentro de un conjunto de ellos establecidos por el fabricante para asegurar la estabilidad de los mismos tanto en la salida como en otros pines. Se ha procurado posicionarlos lo más alejadamente posible de las líneas que llevan la señal y chips críticos, al ser los reguladores componentes de naturaleza ruidosa.

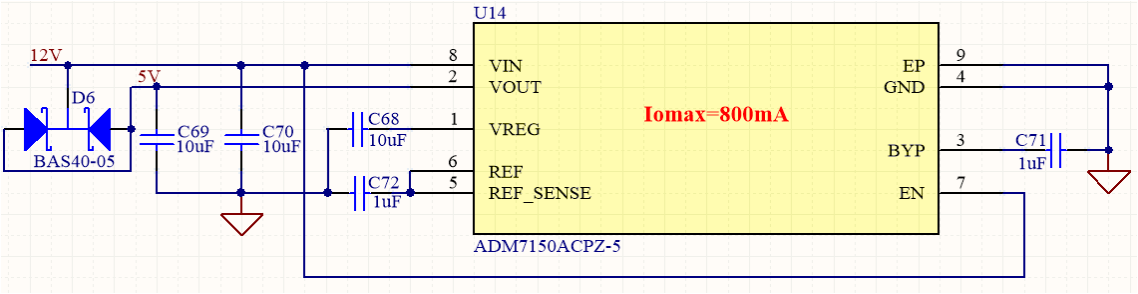


Figura 9 – Regulador de 5V

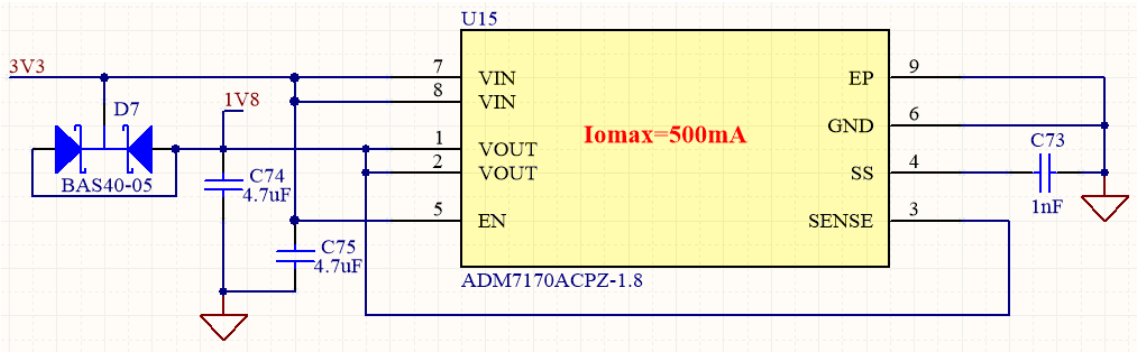


Figura 10 – Regulador de 1.8V

En la salida de la referencia de tensión se decidió hacer uso de seguidores de tensión, ya que la corriente de salida del componente es de tan solo 10 mA, y se desconoce cuál podría ser el pico de demanda de corriente requerido por los 4 ADCs, de esta manera se evita el posible efecto de carga.

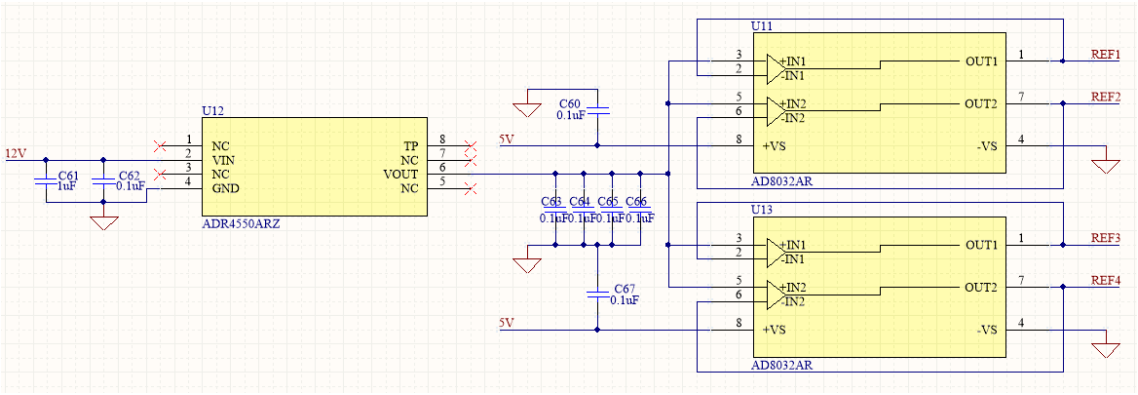


Figura 11 – Referencia de tensión de 5V con seguidor de tensión para cada ADC.

2.1.5 Pinout

La conexión con la FPGA se ha realizado de acuerdo al estándar VITA-57. En total se hace uso de los pares diferenciales CLK, CNV, DCO y D, para la comunicación entre cada uno de los 4 ADCs y la FPGA. Por tanto, esto consumiría 16 pares diferenciales libres en total. Por otro lado, la señal de *trigger* de entrada ocupa otro par diferencial libre adicional.

Se hace uso de dos pines ya incluidos en la especificación VITA-57 para la comunicación I2C entre FPGA y *port expander*.

Del mismo modo las líneas de alimentación de 12 V y 3.3 V van a través de pines especificados en el estándar para esto, por lo que en total se hace uso de 17 pares diferenciales libres adicionales.

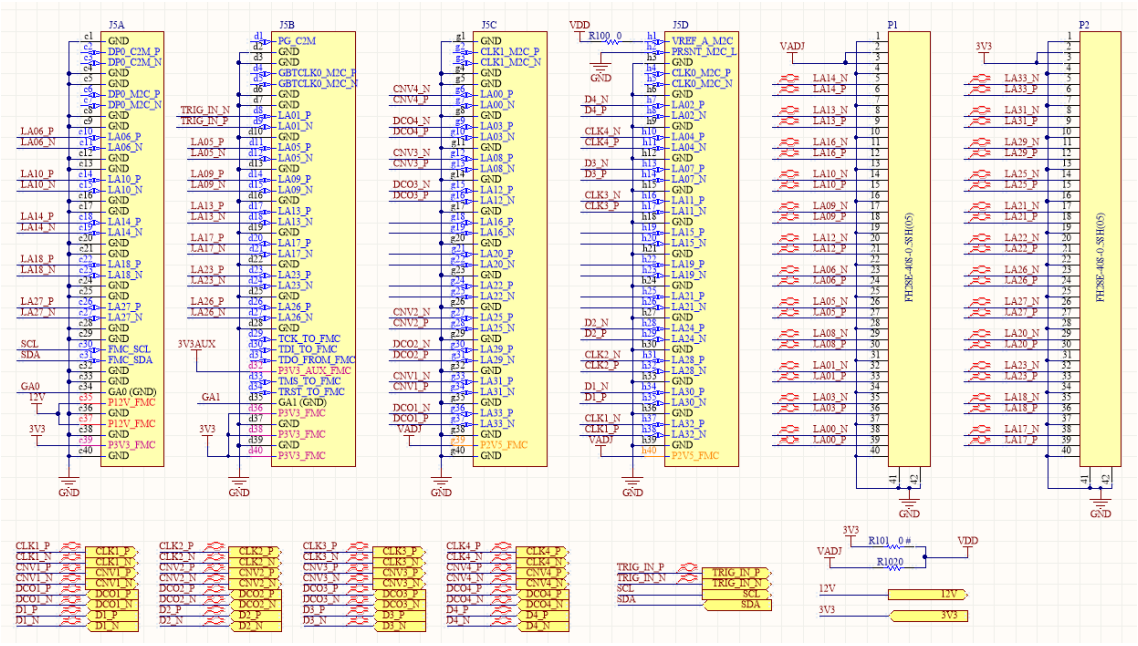
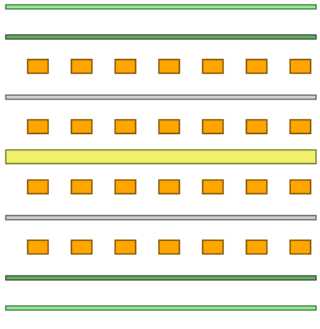


Figura 12- Asignación de señales a conector Samtec y conectores FCC.

2.2 Diseño de la PCB

2.2.1 Selección del sustrato

Una de las primeras cosas que se ha de tener en cuenta es el sustrato real sobre el que se va a fabricar la placa. Saber de antemano las características eléctricas de éste, es un paso necesario para intentar tener la máxima precisión posible en el diseño de las pistas, en concreto referido a tener controlada con precisión la impedancia de éstas y de este modo tener las mínimas pérdidas por pistas desadaptadas. En este caso el soporte para las pistas sobre el que se realizará el prototipo será FR-4 de 4 capas, y las características de este pueden ser vistas en la documentación que proporciona la empresa que se encargará de la fabricación de la PCB. En la Figura 13, se puede ver ya la configuración del *stack* una vez metidos estos datos del *datasheet* del sustrato en la ventana de Board Layer Stack del menú de Altium.



Layer Name	Type	Material	Thickness (mm)	Dielectric Material	Dielectric Constant
Top Overlay	Overlay				
Top Solder	Solder Mask/Co...	Surface Material	0.0762	Solder Resist	3.5
Top Layer	Signal	Copper	0.035		
Dielectric1	Dielectric	Prepreg	0.175	FR-4	4.2
L2_GND	Signal	Copper	0.035		
Dielectric2	Dielectric	Core	1.13	FR-4	4.35
L3_VCC	Signal	Copper	0.035		
Dielectric3	Dielectric	Prepreg	0.175	FR-4	4.2
Bottom Layer	Signal	Copper	0.035		
Bottom Solder	Solder Mask/Co...	Surface Material	0.0762	Solder Resist	3.5
Bottom Overlay	Overlay				

Figura 13 – Pila de capas en tabla de configuración de capas de Altium.

El sustrato formado por 4 capas conductoras tiene una constante dieléctrica de 4.2 para capas superior e inferior. El uso de un sustrato con mejor constante dieléctrica repercute en el menor ancho de las pistas y en las menores pérdidas, si se escogiese un Rogers en lugar de un FR-4 se tendría también el beneficio

de tener dicha constante dieléctrica muy controlada al no ser un dieléctrico de naturaleza fibrosa como el FR-4, pero la diferencia de precio en la fabricación de la PCB puede ser de hasta 4 veces mayor y el beneficio no es tan grande al no trabajar en frecuencias muy altas.

2.2.2 Disposición de componentes

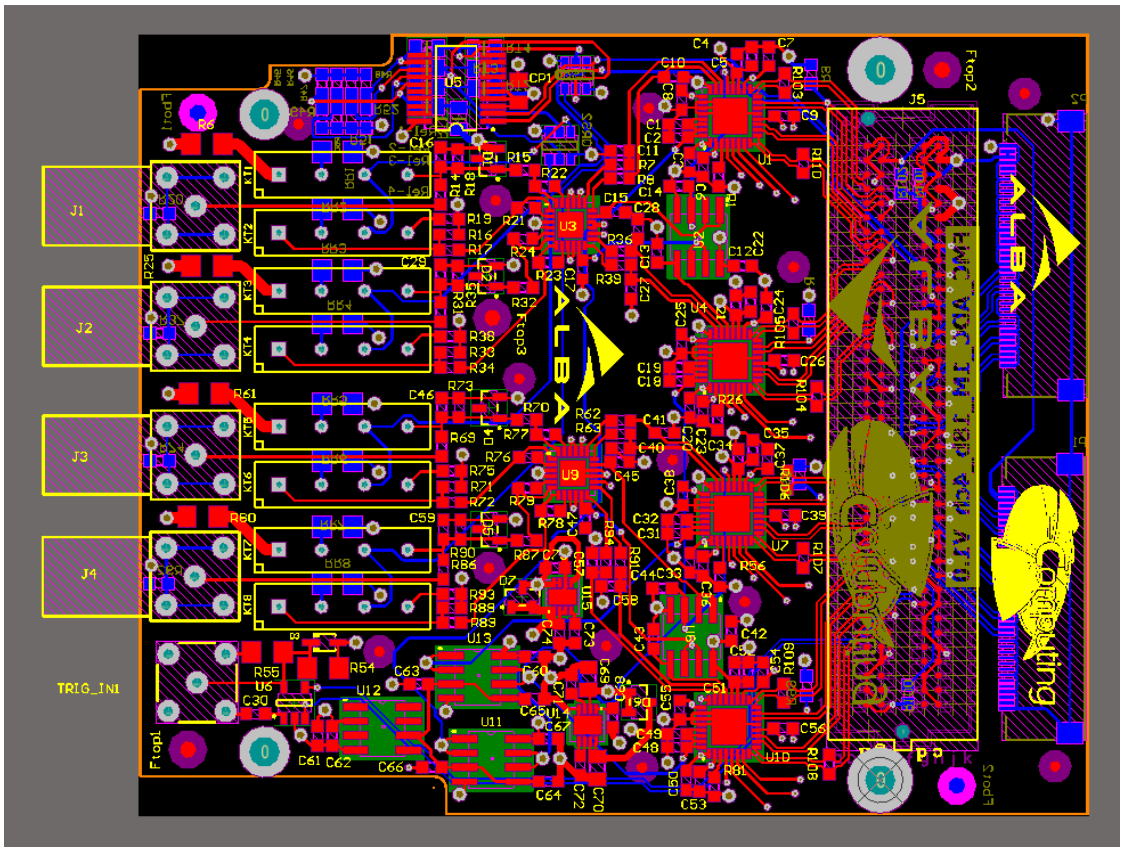


Figura 14 – Vista superior una vez finalizado el diseño de la PCB, en azul capa inferior y en rojo superior. Se han omitido las capas intermedias GND y alimentaciones.

En todo el espacio que ocupa la PCB se han dispuesto los componentes atendiendo a ciertos criterios para intentar mantener la integridad de la señal lo máximo posible. Dado que el espacio está bastante limitado se ha priorizado en una disposición tal que ninguna de las líneas por las que va la señal de entrada atraviese una vía, para evitar de esta manera los efectos inductivos que ésta pudiera ocasionarnos.

Para no salirse de la especificación sobre la que se ha hecho el prototipo, los componentes más altos como son los relés, se han puesto en una zona donde VITA-57 permite una altura máxima mayor.

De esta manera en la capa superior se pueden encontrar los conectores, relés, drivers de ADCs, ADCs, referencia de tensión, reguladores, seguidores de tensión, *port-expander*, filtro *antialiasing*, resistencias, así como todos los filtros y condensadores de desacoplo y estabilidad que se han puesto.

En la capa inferior se han instalado las resistencias de bobina de los relés, así como los transistores que controlan el estado del relé. Además, se ha dejado preventivamente una serie de resistencias de 0 Ω que luego no serán montadas,

para controlar vía hardware la acción de los relés, en caso de haber algún problema con el *port expander* o los transistores, o como medida de urgencia.

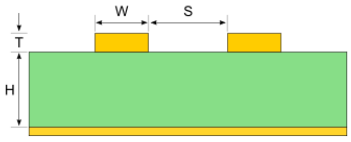
También es en esta capa donde se instalan los dos conectores FCC de 40 pines. En ambas capas se han puesto diferentes resistencias de 0 Ω que como se ha dicho son útiles para configuraciones estáticas o para tener libertad para hacer test o rediseños.

2.2.3 Pistas

Es importante calcular adecuadamente los anchos de las pistas a fin de establecer la impedancia característica óptima para tener las mínimas pérdidas por desadaptación. En el diseño se han utilizado pistas de 50 Ω por ser una impedancia característica habitual. Un hecho que podría llamar un poco la atención es el estilo de trazado de pistas utilizado, el cual impone algún que otro ángulo, pero cuyos efectos nocivos en la señal se han considerado despreciables al ir ésta a una frecuencia máxima de 2.5 MHz.

2.2.3.1 Pistas diferenciales de 100Ω

Para los 4 pares que conectan el ADC con la FPGA, así como las pistas que llevan la señal a digitalizar del driver al ADC, se ha utilizado el modelo de la Figura 15, estableciendo una separación de 0.2 mm y un ancho de pistas de 0.31 mm. Se han tenido en cuenta las tolerancias máximas de fabricación que ofrece el fabricante de la PCB para escoger un valor de ancho y separación que tenga sentido. Para hacer el trazado de este tipo de pistas en Altium, se ha hecho uso de la herramienta de trazado de pistas diferenciales, así como de diversas reglas que ayudan a optimizar dicho despliegue, manteniendo por ejemplo un ancho constante entre las pistas para tener controlada la adaptación o ayudas para establecer pistas diferenciales con la misma longitud por pista.
Se ha de añadir además un resistor de 100 Ω entre las pistas para establecer la impedancia diferencial de 100 Ω resultante. Esto se hace en las señales de salida, pero no en las de entrada del ADC, ni en las que conectan el ADC con el driver porque para estos casos la resistencia la pone la FPGA o el chip respectivamente.



INPUTS			
Trace Thickness	T	0.035	mm
Substrate Height	H1	0.175	mm
Trace Width	W	0.31	mm
Trace Spacing	S	0.2	mm
Substrate Dielectric	Er	4.2	

OUTPUTS	
Odd (Z): 50.0 ohms	Even (Z): 51.9 ohms
Common (Z): 25.9 ohms	Differential (Z): 100 ohms

Figura 15 – Cálculo para pistas diferenciales de impedancia característica 100 Ω

Para el cálculo, la separación entre pistas ha sido fijada a 0.2 mm ya que 0.15 mm es el límite de fabricación de la tecnología con la que será fabricada la PCB, por eso se ha puesto algo un poco mayor a fin de asegurar. Con esto y una anchura de 0.31 mm se consiguen pistas de impedancia característica 50 Ω e impedancia diferencial 100 Ω.

2.2.3.2 Pistas single-ended de 50Ω

Para el caso de pistas *single-ended* la anchura que permite que la línea tenga una impedancia característica de 50 Ω es 0.346 mm. En este caso el modelo es el utilizado en la Figura 16 – Cálculo para pistas single-ended de impedancia característica 50 Ω.

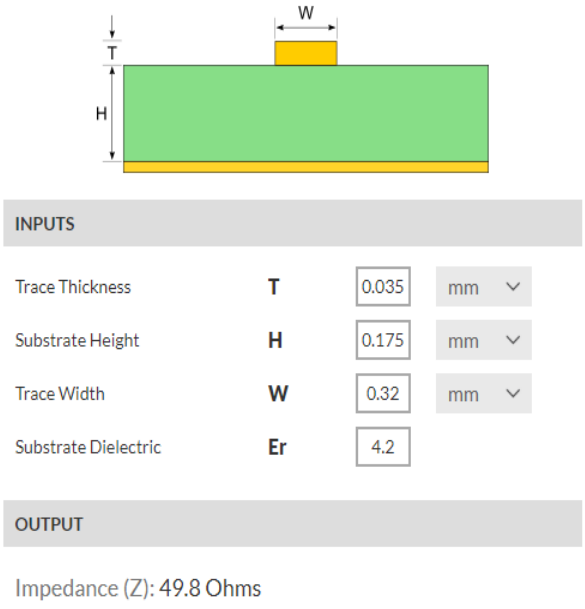


Figura 16 – Cálculo para pistas single-ended de impedancia característica 50 Ω

2.2.4 Capas

Para este desarrollo se ha optado por una PCB de 4 capas. Se realizará una descripción general de cada capa conductora.

- Top layer, L1_TOP

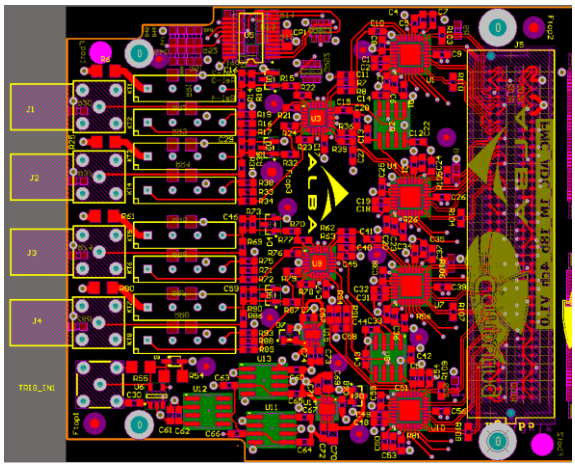


Figura 17 – Vista Altium L1_TOP layer

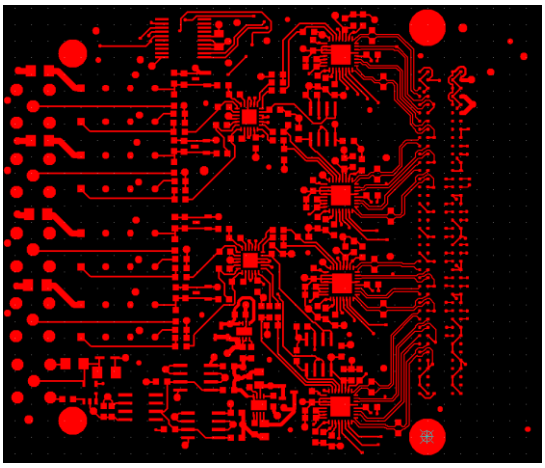


Figura 18 – Gerber L1_TOP layer

Contiene la mayor parte de redes del prototipo y a ella están conectados casi todos los componentes del diseño, se ha intentado colocar el máximo número de componentes en esta capa, cuidando lo máximo posible el camino que recorre la señal de entrada hasta el ADC.

- Medium layer, L2_GND

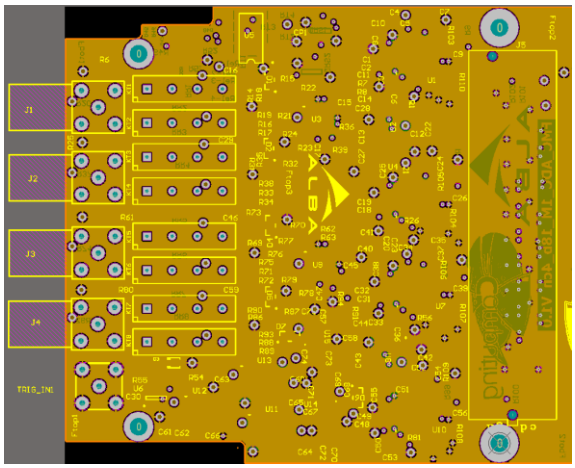


Figura 19 – Vista Altium L2_GND layer

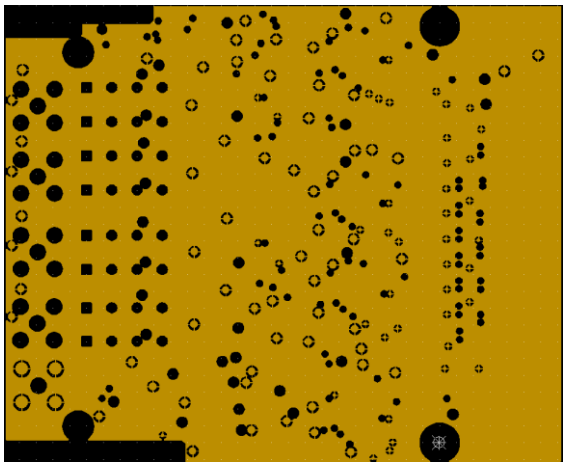


Figura 20 – Gerber L2_GND layer

En esta capa se decidió meter un plano de masa entero justo por ser la siguiente capa más inmediata de las señales principales, de igual manera que en los modelos utilizados para el cálculo de las pistas.

- Medium layer, L3_VCC

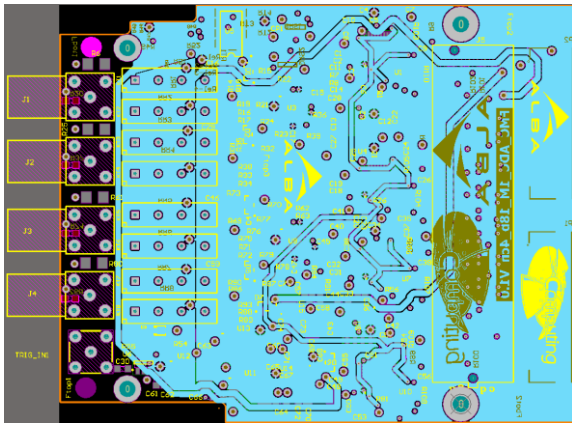


Figura 21 – Vista Altium L3_VCC layer

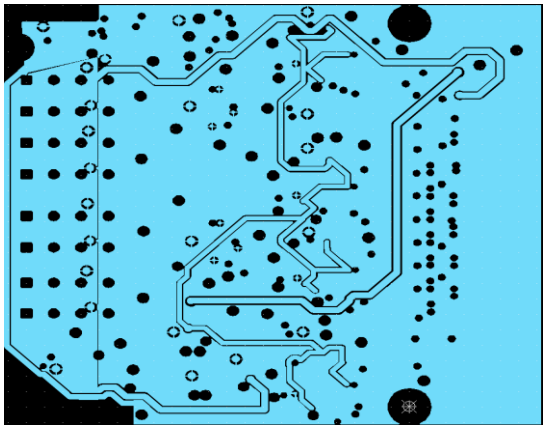


Figura 22 – Gerber L3_VCC layer

En esta capa se pusieron planos de alimentación para los 4 distintos voltajes que son utilizados por todos los componentes que conforman la placa. Es decir: 1.8 V, 3.3 V, 5 V y 12 V. Por lo que todo el plano conductor de esta capa se divide en 4, aunque la mayor parte de la extensión es ocupada por la zona de 5 V, ya que la mayoría de componentes llevan alimentación de 5 V.

-Bottom layer, L4_BOTT

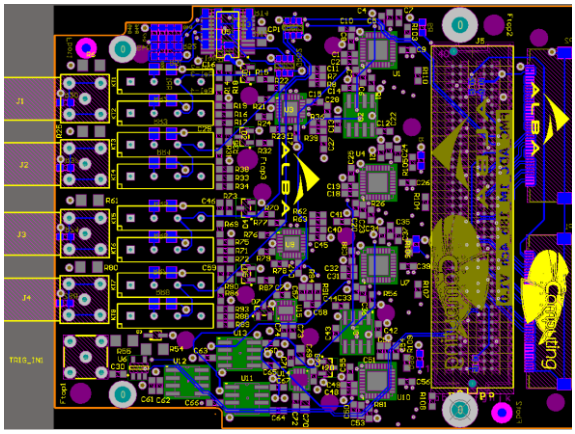


Figura 23 – Vista Altium L4_BOTT layer

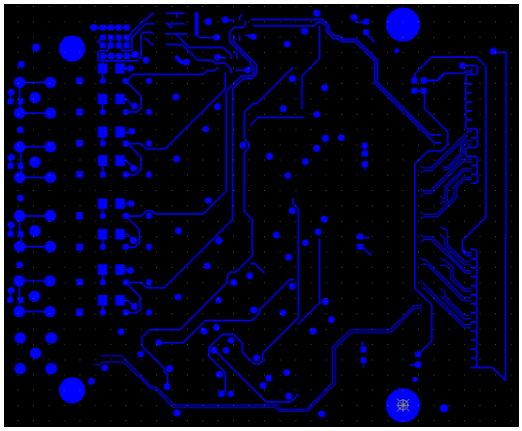


Figura 24 – Gerber L4_BOTT layer

En la última capa solo se han situado algunas pistas auxiliares que no convenía, principalmente por cruces y espacio, que fuesen situadas en la primera capa y cuyo impacto por atravesar vías no repercute en nada crítico. En este caso los drivers de los relés, así como sus resistencias de cola han sido colocados en esta última capa. También los conectores FCC 40, y las resistencias de 0 Ω que nos permiten la activación manual por hardware de los relés, mediante un sencillo puente.

Además de estas 4 capas conductoras, existen otras 7 capas adicionales que han de ser definidas. Algunas son definidas semiautomáticamente durante el proceso de posicionamiento de componentes (si se dispone de *footprints* adecuados) o ruteado de pistas. Otras han de ser definidas directamente:

- Capa de serigrafía superior e inferior:

Estas capas conforman la serigrafia que será utilizado para los designadores de los componentes si existe espacio para ello o para el logotipo de la empresa y división que realiza el diseño.

- Capa de *pads* superior e inferior:

La capa de *pads* contiene la información de donde van situados los *pads* sobre los que se situarán las patillas o *pads* de los componentes.

- Capa de máscara de soldadura superior e inferior:

Sirve para definir la capa de lacado para zonas de cobre expuesto donde no se desea aplicar pasta de soldadura y que están expuestas al aire para proteger la zona de la corrosión y cortocircuitos.

- Capa con el borde que conforma los límites físicos de la PCB:

Esta capa simplemente define los límites de la PCB. En este caso la forma de la PCB se ha sacado del estándar VITA-57.

2.2.5 Fabricación de la PCB

Una vez está terminado el diseño de la PCB, se generan los ficheros Gerber que se mandan al fabricante.

Además de las 7 capas adicionales es necesario también proveer al fabricante del fichero que contiene la información para taladrar las vías. Éste no contiene más que las coordenadas, los diámetros y el tipo de vía de cada uno de ellos, es decir metalizada o no.

También se han tenido que tener en cuenta las tolerancias de fabricación que da el fabricante para valorar si las posibles restricciones en cuanto a precisión, por ejemplo, a la hora de fabricar el ancho de las pistas, compromete significativamente en el buen funcionamiento del prototipo. Las tolerancias fueron comprobadas y tenidas en cuenta durante todo el diseño de la PCB.

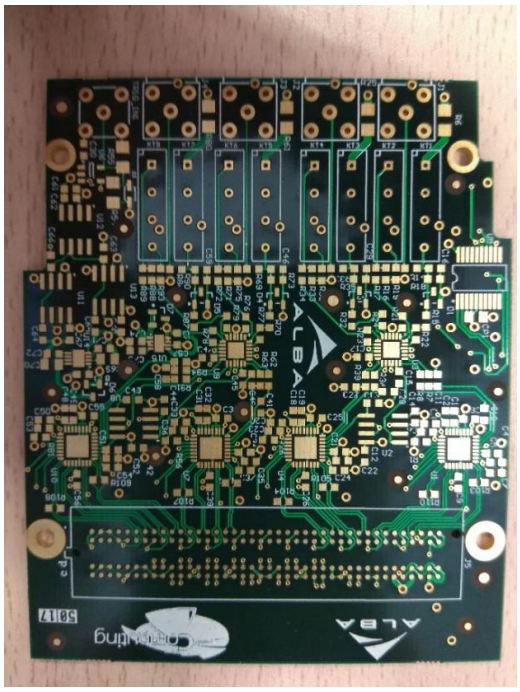


Figura 25 – Parte superior PCB

Figura 26 – Parte inferior PCB

2.2.6 Montaje

En el diseño de esta FMC se han escogido encapsulados pequeños debido a la falta general de espacio, en concreto tanto para drivers, como para ADCs y reguladores se ha seleccionado el encapsulado LFCSP. Este tipo de encapsulado es complicado de soldar, y por ello ha sido necesario enviar la placa a un montador especializado, el cual mediante máquina de posicionado y horno ha soldado los componentes de este tipo junto con el conector Samtec, puesto que sus pines se encuentran en la parte inferior y no son accesibles. El resto de componentes al ser SOIC, SSOP, SOT-23 y familia SMD 0603 ha sido posible soldarlos manualmente en el propio laboratorio del Sincrotón ALBA.

Para ayudar al proceso de posicionado tanto de placa como de componentes en el proceso de montaje, se han añadido fiduciales en la PCB, y por lo tanto en los Gerber. Estos han sido colocados en diagonal respecto de los chips más críticos y precisos, de tal manera que la máquina tenga una referencia igual de X y de Y cerca de estos puntos.

En concreto hay dos tamaños de fiduciales según se quieran usar para referenciar elementos grandes o pequeños. La regla es sencilla: fiduciales grandes para elementos grandes y fiduciales pequeños para elementos pequeños.

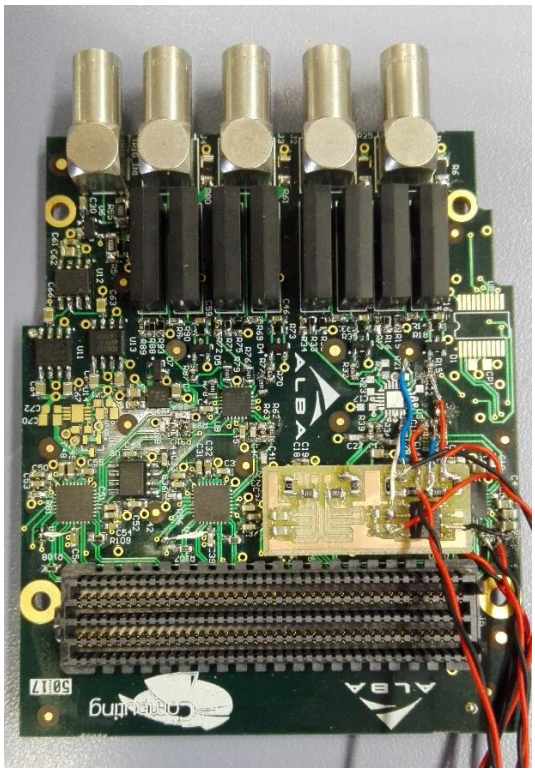


Figura 27 – Vista superior montada

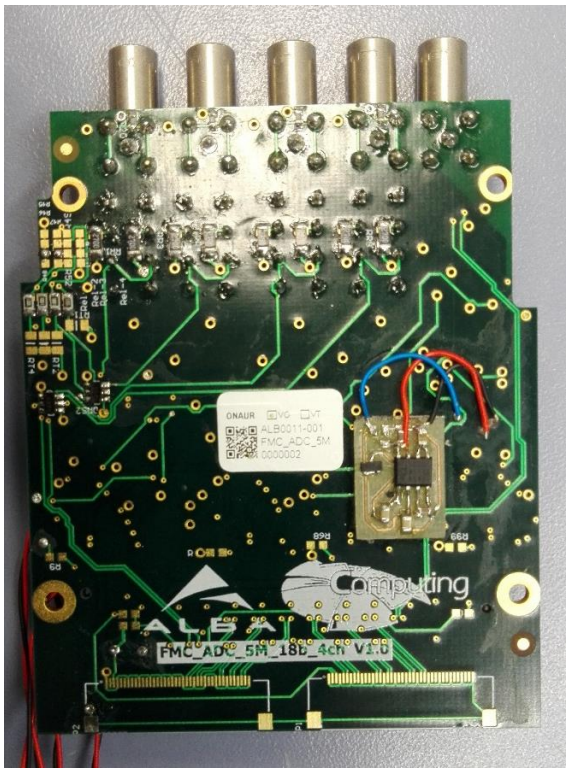


Figura 28 – Vista inferior montada

2.2.7 Reworks

La realización de los *reworks* para los que ha sido necesario fabricar una placa externa, se ha realizado utilizando la máquina fresadora de prototipado de PCB del laboratorio de electrónica del Sincrotón ALBA.

2.2.7.1 Drivers

Debido a un error por no realizar una revisión lo suficientemente exhaustiva del *datasheet* del AD4932 en la fase inicial de diseño, se pudo comprobar a posteriori que el AD4932 no era adecuado por tener un *Output Voltage Swing* de −Vs + 1.15 V a +Vs – 1.15 V. Teniendo en cuenta que es alimentado a 5 V esto hacía que los límites de señal de salida fuesen de ±3.85V los cuales quedan mal ajustados al rango dinámico del ADC (+-5 V) perdiendo un total 2.3 V de margen dinámico. Esto equivale a tener un número de bits efectivos de 16.79, es decir 1.21 bit menos de resolución tal y como se ha calculado a continuación:

$$bits_{eff} = log_2\left(\frac{2^{18}}{2.6}\right) = 16.97bits$$

Considerado esto dentro de las especificaciones mínimas, pero a la vez mal aprovechado el ADC, se prefirió la opción de buscar otro driver y realizar un *rework*. Dado que la placa tiene el propósito de prototipo y el tiempo es ajustado para un proyecto tan largo, sólo se hizo el montaje y testeo completo de un canal.

Para ello hubo que buscar otro driver, se priorizó en encontrar uno que tuviese características eléctricas principales parecidas a las del AD4932, que fuese pin-compatible y que estuviese disponible con el mismo encapsulado para evitar realizar un *rework*, pero no fue posible y por ello se decidió por el AD4940, que tiene las mismas características que el AD4932 pero un *Output Voltage Swim* mínimo de −Vs + 0.1 V a +Vs – 0.1 V por lo que la señal de salida será de ±4.9 V ajustándose mucho mejor al rango dinámico de entrada del ADC.

Todo esto hace que aunque se haya puesto énfasis en evitar transmitir la señal principal a través de vías y que es algo conseguido en el diseño inicial, transcurriendo tan solo por la capa superior, el hecho de haberse realizado este *rework* supone la violación de este objetivo, ya que se obliga a conectar una pequeña placa auxiliar en la parte superior, para la cual ha sido necesario realizar una conexión con cables de *rework* y hacer pasar la señal a través de ellos, lo que supone no mantener la impedancia deseada en este tramo de señal.

Aunque esto sea así, es justificable por el hecho de que en el caso de no querer renunciar al uso del AD4932 y evitarse el *rework*, perder tanto margen dinámico es más perjudicial que no asumir la pequeña distorsión producida por el cable de *rework*.

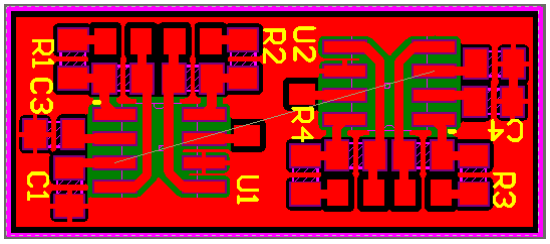


Figura 29 – Detalle rework driver

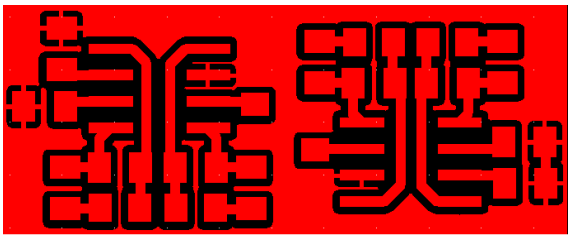


Figura 30 – Gerber top layer rework driver

Se puede observar en la Figura 29 y 30 que la placa tiene posibilidad de que se le añadan 2 drivers, por lo que serían necesarias 2 para hacer el rework de los 4 canales. Aunque como se ha dicho anteriormente solo se ha montado completamente el canal 1, tal y como puede verse en la Figura 27.

Una cosa que se pensó fue en utilizar el AD4932 inicial en el canal 2 a modo de prueba. Ya que su *voltage swing* permite salidas de ±3.85 V, una opción hubiese sido configurar el ADC para utilizarlo con margen dinámico de ±4.096 V en lugar de ±5 V. De esta manera se perderían solo 0.492 V de margen dinámico. Se intuía que la mejor opción era utilizar el driver que permitía tener un output voltage swing mayor, ya que utilizando el rango dinámico en el ADC de ±5 V se obtiene un incremento en la SNR de 1.7 dB respecto de utilizar ±4.096 V.

A pesar de ello la idea de hacerlo era comparar entre el canal 1 y el 2 cada uno con un driver distinto. Por falta de tiempo al final esta intención se descartó y se centraron los esfuerzos solo en finalizar el canal.

2.2.7.2 Regulador de tensión

Debido a un problema de última hora en el stock de los proveedores de electrónica no se podía disponer del regulador ADM7150ACPZ-5 hasta 3 meses más tarde, con lo que hubo que optar también por la inclusión de un *rework* en la parte inferior para ajustar el *footprint* de un nuevo regulador, en este caso un ADM7150ARDZ-5.0-R7 que servía perfectamente para la función ya que no es más que el mismo modelo con otro encapsulado, por lo que tenía el inconveniente de tener otro *footprint*. No pudo encontrarse tampoco ninguno pin-compatible con el anterior, por lo que fue necesario recurrir a otro *rework* por la inexistente posibilidad de esperar 3 meses por el regulador escogido en su momento para continuar con el desarrollo.

Respecto al *rework* del regulador, es una simple adaptación para soportar un encapsulado SOIC-8 con el chip y sus condensadores correspondientes para mantenerlo estable y conectarlo a las vías mediante cable de *rework* en algún posible punto.

Para este tipo de *reworks* es importante emplear sustrato con una sola capa de cobre, ya que se debe minimizar el riesgo de hacer cortocircuito con algún componente más de la FMC una vez esté situada sobre ésta. Por ello es ideal utilizar sutrato con la parte inferior sin recubrimiento de cobre.

En este caso a nivel de perjuicio debido al *rework* las prestaciones no se ven tan resentidas al no tratarse de un elemento por donde pasa la señal. El montaje de este *rework* puede verse en la Figura 2728.

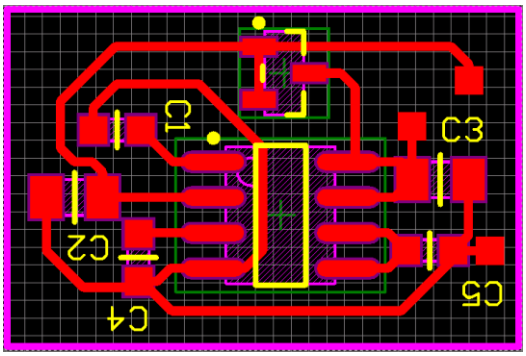


Figura 31 – Detalle rework regulador

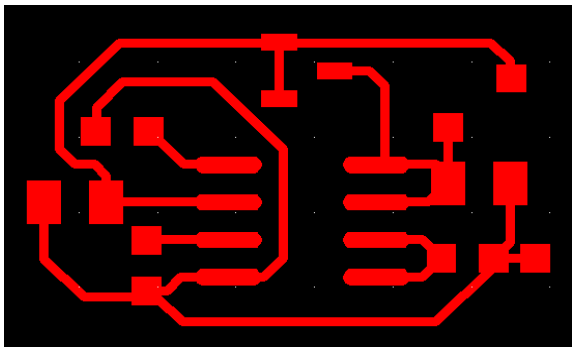


Figura 32 – Gerber top layer rework regulador

Para el regulador de 5V hubo que situar la placa en la parte inferior de la FMC.

2.2.7.3 Rework para cambiar PIN EN3 de 0 a 1

Tal y como se comenta en la sección 1.3.1. hubo que hacer un *rework* para cambiar el pin EN3 de GND a 1.8 V y poder activar la salida en modo común del ADC. Para hacerlo se tuvo que cortar primero la pista y después eliminar con el *cutter* la máscara de soldadura para poder soldar un cable de EN3 a 1.8 V. En la Figura 33 se puede ver más en detalle este rework para los canales 3 y 4.

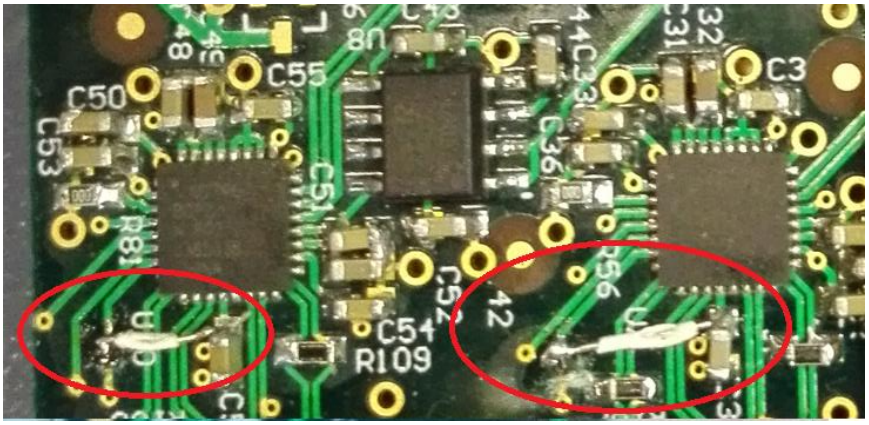


Figura 33 – Rework para habilitar a 1 el pin EN3.

En el momento de colocar los componentes y pistas, se intuyó que los pines de configuración podrían ser problemáticos y podría ser útil poder acceder a ellos, por lo que se decidió dejar las 4 pistas correspondientes más largas de lo normal por si había que intervenir sobre dichos pines.

3. PROGRAMACIÓN DE LA FPGA

En este capítulo se hace un recorrido por todo el proceso de desarrollo e integración del firmware hecho en VHDL. La Spec tiene una FPGA Xilinx Spartan 6 100T.

Para el desarrollo VHDL se hizo uso del entorno ISE 14.3 del propio Xilinx. Aunque actualmente se utiliza Vivado en sustitución de ISE, como mejor y más completo entorno de simulación y programación de FPGAs. Se ha optado por la utilización de IDE 14.3 al ser un entorno conocido y ya muy testeado en otros proyectos realizados anteriormente en el sincrotrón ALBA.

3.1 VHDL Propio

En este apartado se describe el VHDL desarrollado para implementar una funcionalidad básica. En la Figura 34 se muestra la jerarquía de módulos, así como las señales están conectadas, antes de haberse realizado la integración final con otros módulos.

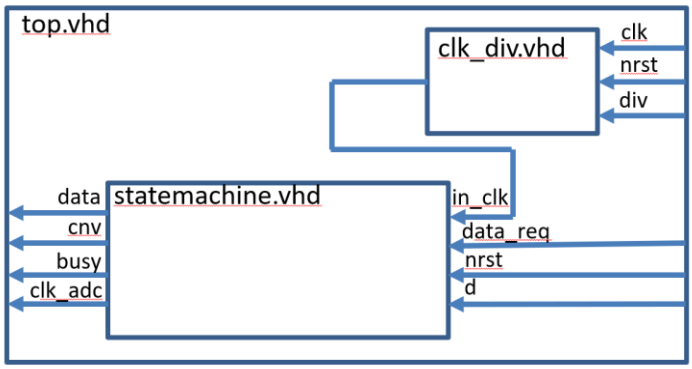


Figura 34 – Módulos custom VHDL

3.1.1 Máquina de estados

Lo primero de todo será pensar en una máquina de estados con todos los estados posibles que podrá tener el ADC. La definición de estados es sencilla, 4 estados son suficientes para realizar dicha síntesis en VHDL.

-IDLE: El primer estado de todos se da cuando la máquina de estados está libre esperando una activación a través de la señal DATA_REQ proveniente de otros módulos de la FPGA (o del sistema de control) y que despierta del IDLE la máquina de estados para enviar la señal de conversión al ADC y dar comienzo una adquisición.

-CONV: Este estado ocurre mientras se está enviando la señal de conversión hacia el ADC.

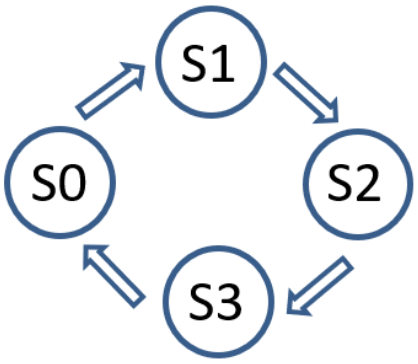
-ACQ: Este modo está presente durante la fase de adquisición, antes de empezar a enviar los datos, pero ya mientras adquiere éstos. Las especificaciones dicen que el tiempo entre conversiones mínimo es de 200 ns,

por lo que en el código VHDL se ha de expresar eso en función de números de ciclo. Para el caso de reloj rápido a 125MHz se calcula cómo:

$$\frac{t_{MSB} - t_{CNVH}}{t_{CLK}} = \frac{200\text{ns} - 10ns}{8ns} = 23.75$$

Es decir, se han de esperar 24 ciclos de reloj.

-DATA: En este estado la máquina está enviando ya los 18bits de la muestra en serie, junto con el clock para evaluarlos.



S0 – IDLE – Hasta recibir DATA_REQ
S1 – CNV – Enviando señal de conversión
S2 – ACQ – Mientras adquiere
S3 – DATA – Enviando los datos

Figura 35 – Diagrama de estados

Para entender mejor cómo funciona la máquina de estados se repasarán las entradas y salidas de esta, y qué señales lleva cada una. De la declaración del código de la Figura 36 se puede ver:

```
entity statemachine is
    port( clk, DATA_REQ, D, nrst : in std_logic;
          BUSY, CNV, CLK_ADC : out std_logic;
          DATA : OUT STD_LOGIC_VECTOR(17 DOWNTO 0)
          );
end statemachine;
```

Figura 36 – Puertos de la máquina de estados

Es decir, la máquina de estados tiene una entrada para reloj (clk), otra para iniciar el proceso de digitalización en el ADC (DATA_REQ), otra por donde entran los datos que vienen del ADC (D) y por último una entrada de *negative reset* (nrst) que sirve para llevar la máquina a un estado inicial.

Como salidas se tiene una señal de estado de ocupación (BUSY), que indica si la máquina está ocupada, es decir, en estado de envío de señal de conversión, de espera para enviar el dato o de envío de dato. La señal BUSY solo está en 0 cuando la máquina de estados se encuentra en Idle. La otra de las salidas es la línea que envía los datos (DATA) hacia otro módulo a través del módulo top, en este caso en forma de vector de 18 bits, por corresponder a los 18 bits que conforman cada uno de los samples digitalizados por el ADC. Los datos son enviados en serie mediante señal diferencial de bajo voltaje, LVDS.

Para comprobar el correcto funcionamiento de la máquina de estados, se realizará una simulación con el simulador Isim incluido dentro del *framework* ISE que permitirá examinar el correcto comportamiento de las señales en lo que a lógica se refiere.

Ha servido como referencia y ayuda en la comprensión del ADC la Figura 37, obtenida del *datasheet* del fabricante y en la que se muestra el funcionamiento del chip ADC en modo *echoed-clocked*.

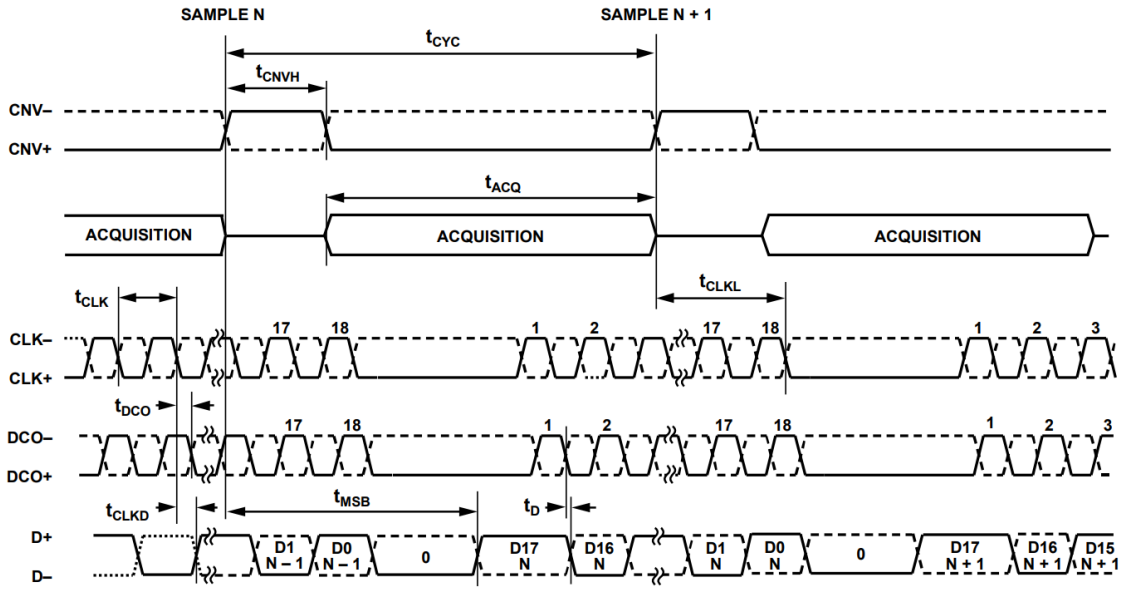


Figura 37 – Echoed-clocked interface mode timing diagram

Es necesario para realizar cualquier simulación establecer las señales de entrada que se quieran forzar, para ello se tiene que agregar al proyecto un archivo .vhd en el que se generan entre otras cosas el reloj y la señal de inicio de adquisición (DATA_REQ). Cabe resaltar la visualización de la señal automat, la cual es interna y contiene el estado actual de la máquina. Esta señal es utilizada en el código del módulo para activar y desactivar las señales en función de dicho estado.

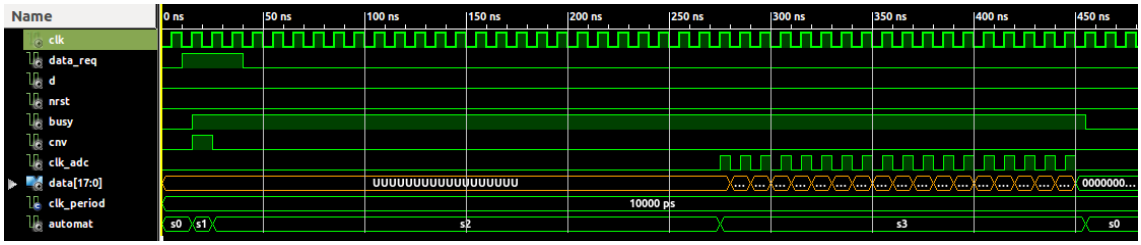


Figura 38 – Simulación máquina de estados mediante Isim

En la captura anterior se puede observar como la señal CNV es activada en el momento que la señal DATA_REQ es detectada en alta durante un flanco de subida de reloj. Se ve también como la señal CNV se ha hecho de tal manera que vuelve automáticamente a 0 tras un ciclo de reloj de haber sido activada. Para mandar el impulso por DATA_REQ que dé comienzo a una adquisición se

ha tenido que programar dicha señal en el archivo .vhd correspondiente al *test bench* para realizar esa simulación.

Es importante también fijarse en cómo se cumplen los ciclos de reloj necesarios para realizar otra conversión, es decir 24, y como la señal automat va cambiando sus valores según el estado actual de la máquina.

El árbol de módulos y organización del proyecto antes de realizar la integración final quedaría como se muestra en la Figura 39, una vez ya han sido añadidos los ficheros necesarios para ejecutar el *test bench* del módulo o módulos que interesen.

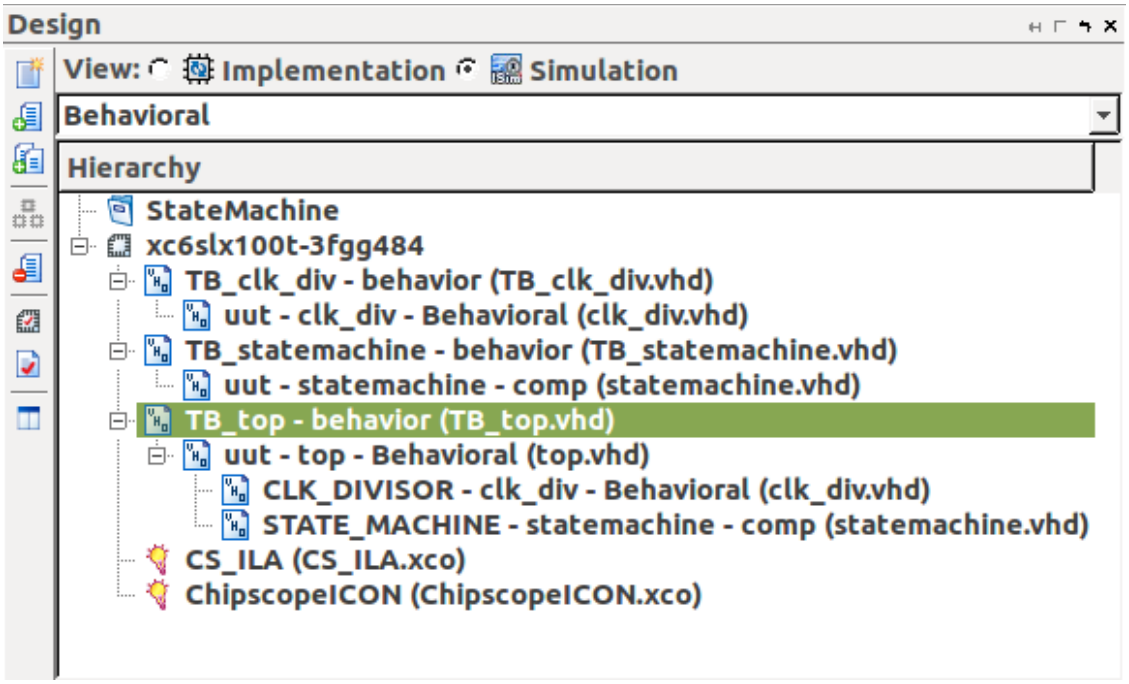


Figura 39 – Árbol de módulos de proyecto

3.1.2 Clock divisor

El módulo DIV se creó para *debug*, ya que en operación o test la máquina de estados siempre irá a 125 MHz. DIV permite dividir el *clock* mediante la siguiente ecuación:

$$f_{clk_{div}} = \frac{f_{clk}}{4 * DIV}$$

Esta no es la manera más optima de realizar esto, aunque es una forma muy rápida que permite controlar fácilmente la frecuencia de funcionamiento simplemente cambiando el registro DIV. Hay que recordar que la FPGA proporciona un reloj de 125 MHz. Una manera más complicada pero mejor de hacer esto es utilizando un PLL.

Un problema que se tuvo como consecuencia de hacerlo así fue que el *trigger* que llega del sistema de control tiene una temporización de *clock* rápido, 125

MHz, y la máquina de estados se está gestionando con un *clock* salida del divisor de clock, por lo que se tuvo que poner en el módulo top un latch del trigger rápido proveniente del sistema de control hasta que la máquina de estados capturase este trigger. De esta manera ninguno de los triggers que antes se perdían ahora se pierden.

En la Figura 40 se muestra el código correspondiente al latch implementado en el módulo top. Básicamente es un proceso para registrar DATA_REQ por flanco de subida de clk y mantener data_reg_r hasta un flanco de subida de clk_d asegurando de esta manera el disparo.

```
process (clk) begin
    if (rising_edge(clk)) then
        if DATA_REQ = '1' then
            data_req_r <= '1';
            clk_d_zerocross <= '0';
        else
            if (clk_d = '1') and (clk_d_zerocross = '1') then
                data_req_r <= '0';
            end if;
            if (clk_d = '0') then
                clk_d_zerocross <= '1';
            end if;
        end if;
    end if;
end process;
```

Figura 40 – Proceso para hacer un latch del trigger rápido

3.1.3 Top

En la parte más alta de la jerarquía se encuentra el módulo top, que engloba a los 2 módulos anteriores. Es dentro de este módulo donde se conectan entre ellos la salida de reloj del módulo DIV, con la entrada de reloj del módulo máquina de estados, también es este módulo el que se integra dentro de los otros módulos sobre los que será integrado este *custom* VHDL, aquí también es donde se puso el *latch* necesario por el problema del DIV del apartado anterior En el apartado 3.2 se hablará de la parte de integración

3.2 Integración con código VHDL

Para interaccionar con la FMC se han integrado todos los módulos en el firmware de un proyecto ya existente en el sincrotrón ALBA, el cual es un electrómetro[6] de alta precisión basado en la misma arquitectura FMC-SPEC-SBC sobre la que se ha integrado este prototipo. Los módulos que no eran necesarios así cómo todo lo prescindible se ha eliminado a fin de hacer más rápida la compilación y simplificar la solución. En la Figura 41, se puede ver el VHDL propio ya integrado en el resto de los módulos.

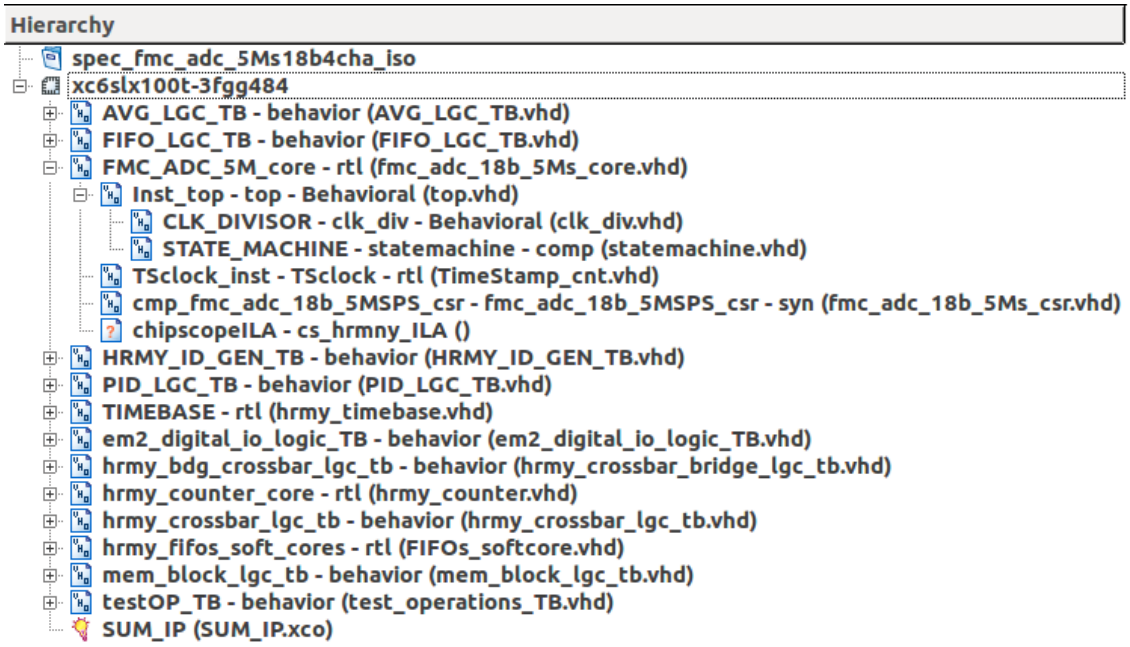


Figura 41 – Árbol de módulos una vez ya integrado el custom VHDL en el proyecto EM#

Aunque los propósitos son distintos ahorrará bastante tiempo reutilizar una parte que permite interaccionar con la FMC de forma casi inmediata, tan solo haciendo algunos ajustes en el código y mapeando adecuadamente los puertos de los módulos.

En este prototipo se hizo uso de 2 buses distintos para control y datos tal y como se muestra en la Figura 42. Estos buses, aunque tienen objetivos distintos se relacionan uno con otro.

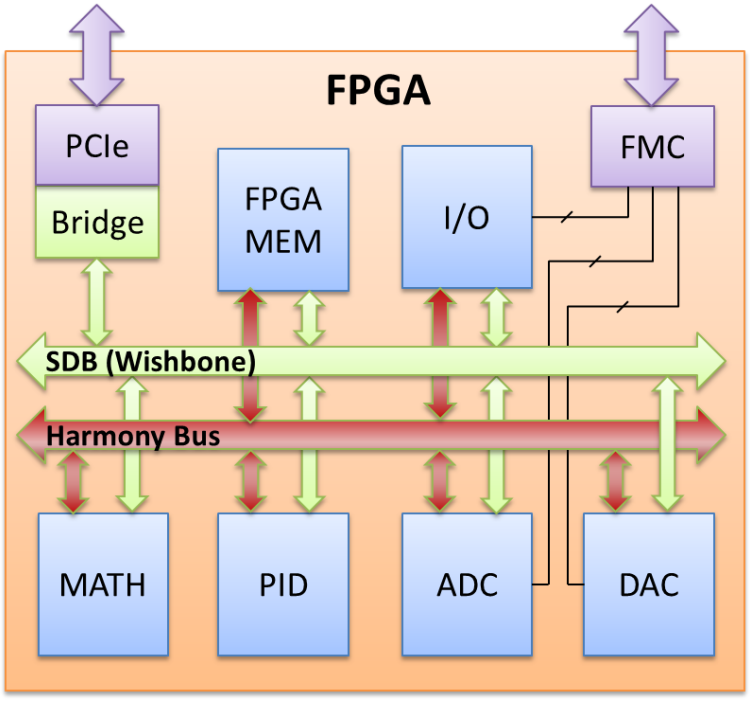


Figura 42 – Topología interna FPGA con representación de los buses utilizados

3.2.1 Self-Describing Bus

Nacido en el *CERN* en base a la necesidad de detectar el contenido de un dispositivo lógico específico una vez programado. Si el bus FPGA interno pudiera enumerar su propio contenido, se obtendrían muchas ventajas.

Se diseñó para que fuera lo más simple posible, pero abierto a futuras extensiones sin introducir problemas de compatibilidad. Su bus interno es Wishbone[8], aunque las estructuras han sido diseñadas genéricas para que otras implementaciones de bus puedan usarlas.

El Self-Describing Bus[10] permite enumerar los núcleos que están activos en el binario FPGA actual, ya sea desde la computadora host o desde la CPU interna del *softcore* de la propia FPGA. También se usa como sistema de archivos simple en dispositivos EEPROM, de modo que tanto el host como el *softcore* que están en la FPGA pueden acceder fácilmente a los datos.

Para la generación del core Wishbone de esta FMC se ha hecho uso del Wishbone generator[9]. Esta herramienta genera automáticamente el código VHDL de un *core* Wishbone esclavo, así como un informe en donde se detallan los registros creados. Es dentro de este *core* Wishbone dónde se incluye el VHDL propio para controlar el hardware de este proyecto (FMC_ADC_5M_core en la Figura 41). En la Figura 43 se puede ver una captura con los registros Wishbone creados que se usarán para leer valores, enviar *triggers* y ajustar el divisor de *clock* como convenga sacado de dicho informe.

H/W Address	Type	Name	VHDL/Verilog prefix	C prefix
0x0	REG	Control register	fmc_adc_core_ctl	CTL
0x1	REG	CLK DIVISOR	fmc_adc_core_div	DIV
0x2	REG	ADC Chanel 1	fmc_adc_core_adc_ch1	ADC_CH1
0x3	REG	ADC Chanel 2	fmc_adc_core_adc_ch2	ADC_CH2
0x4	REG	ADC Chanel 3	fmc_adc_core_adc_ch3	ADC_CH3
0x5	REG	ADC Chanel 4	fmc_adc_core_adc_ch4	ADC_CH4
0x6	REG	Samples counter	fmc_adc_core_samples_cnt	SAMPLES_CNT
0x7	REG	ID assignation	fmc_adc_core_id	ID

Figura 43 – Memory *map summary*

3.2.2 Harmony

Debido a que el SDB ocupa un ancho de banda en el bus PCIe para la comunicación entre SBC y SPEC (FPGA), y también debido a que el software de la SBC no puede garantizar una latencia de tiempos del orden de microsegundos, no se pueden hacer adquisiciones largas utilizando como bus directo PCIe.
Para solucionar este problema se desarrolló el bus Harmony [3] para el proyecto Em# del sincrotrón *ALBA* [6]. Este es un bus interno de la FPGA, que permite la

transferencia rápida de información entre módulos. De esta forma, se puede hacer una adquisición de muchas muestras con un *sampling rate* alto con el módulo VHDL que gestiona el ADC y enviarlas de forma rápida a través del bus Harmony a un módulo VHDL que gestione una FIFO o una EEPROM. Esta información se sacará de una forma más lenta y sin asegurar latencias a posteriori con el SDB y a través de PCIe. Harmony, además, incluye *timestamp* en todos los datos.

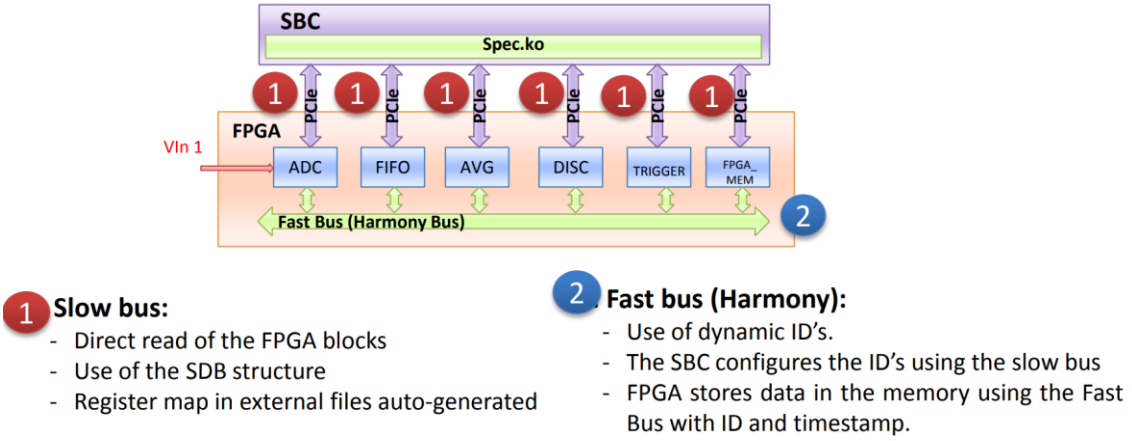


Figura 44 – Harmony, descripción

3.3 Escritura de firmware

Para escribir el firmware existen dos formas principales de hacerlo.

1. Utilizar la herramienta IMPACT incluida en el Framework ISE para escribir en la FPGA a través de la interfaz JTAG y por medio de un programador de Xilinx, simplemente indicándole el archivo binario generado que contiene el firmware.

   Esta opción es fácil y sencilla, pero como la memoria de la FPGA es volátil, en cuanto se interrumpa la alimentación, el firmware será eliminado teniendo que repetir la operación de escritura cada vez que se reinicie o interrumpa la alimentación. Esta es la forma más rápida y flexible en las primeras fases de integración del firmware. Una vez se tiene un firmware estable con el que hacer pruebas, es conveniente utilizar el segundo método para ahorrar tiempo.

2. Copiando el archivo con el binario en la SBC en una carpeta concreta, y de esta forma se carga el firmware en la FPGA en cada arranque gracias un driver específico desarrollado en el *CERN*

4. TEST Y CARACTERIZACIÓN

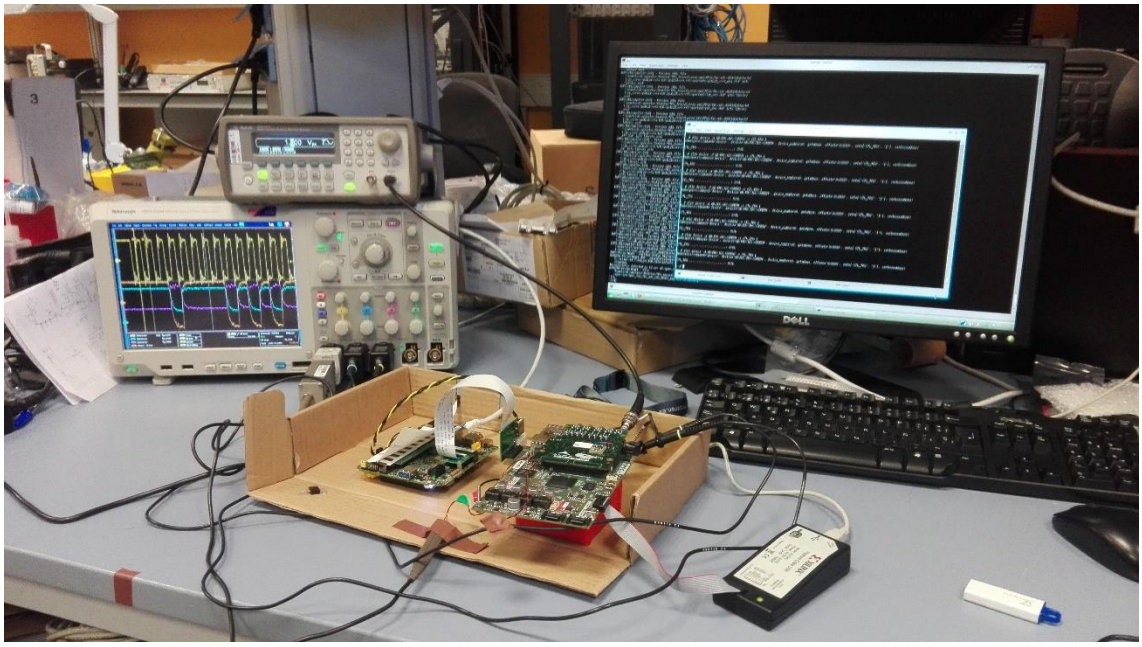


Figura 45 – Setup de medida de señales, adquisiciones y programación de FPGA

4.1 Propuesta de setup

Cómo ya se nombró en la introducción, la puesta en marcha y control sobre la FMC se realizará mediante una arquitectura compuesta por una SBC, la FPGA Carrier (SPEC) y la FMC que se ha fabricado.

Para visualizar las señales de interés entre FPGA y ADC, se ha utilizado un osciloscopio de 4 canales conectando una sonda diferencial en el puerto 1, y dos sondas normales en los puertos 2 y 3, de esta manera y con ayuda del canal matemático según el caso se pueden ver simultáneamente dos señales, o tres señales. La conexión de las sondas a la PCB se ha realizado directamente con la misma sonda, mediante cables de *rework* soldados a las señales de interés o con pinzas de miniatura especiales para sonda diferencial.

La SPEC y su FMC se manejan a través del driver de la SPEC, que se ejecuta en la SBC, y utilizando PCI-Express como interfaz de conexión. Este driver desarrollado por el *CERN* hace uso del SDB para acceder a registros internos de la FPGA y así controlar su funcionamiento y extraer datos

El manejo de la SBC se realiza mediante una sesión SSH desde una máquina de escritorio a través de la red Ethernet del sincrotrón, a la que ambas están conectadas.

Para las primeras versiones de firmware, en la que es más común tener que realizar varias programaciones en poco tiempo, se hizo uso del método de programación de firmware a través de JTAG con el programador de Xilinx. Una vez conseguido un firmware libre de errores que permitió realizar adquisiciones

correctamente y trabajar sobre él se optó por cargar el firmware en la SBC, siendo autocargado en la FPGA cada vez que se reiniciase el sistema.

Además de osciloscopio se han utilizado generadores de señal tanto para generar rampas como valores estáticos de voltaje en todo el margen de entrada de la FMC.

Debido a la limitación de tiempo se decidió realizar las pruebas solo en alta impedancia de entrada, es decir dejando desactivados los relés. Con esto se dejaría para el final el montaje y puesta en marcha del *port expander* y se dispondría de más tiempo para centrarse en otras fases que exigían más urgencia.

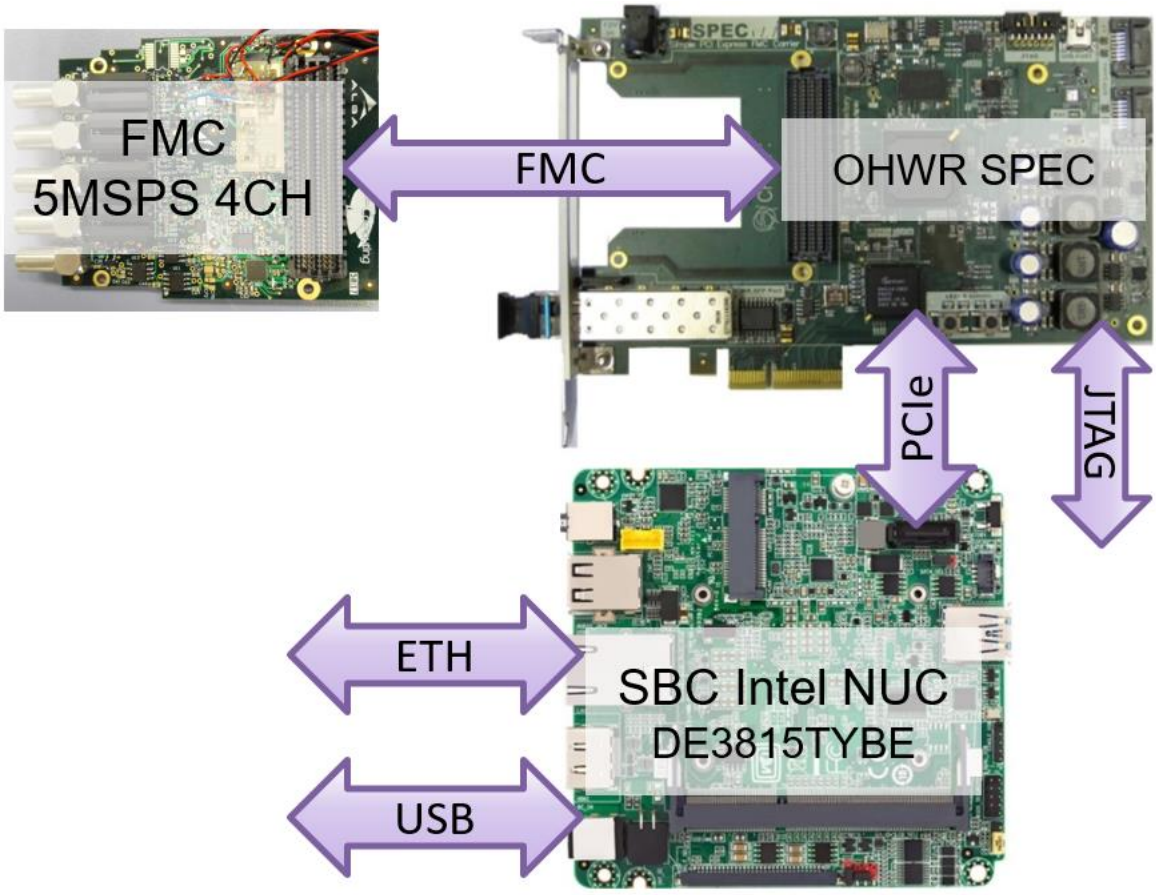


Figura 46 – Arquitectura SPEC-SBC-FMC

4.2 Escritura y lectura de los registros

Gracias a SDB es posible realizar lecturas y escrituras inmediatas de los registros desde consola. Un comando junto con su salida para ver los registros es el de la Figura 47.

```
~ # alin device
Namespace(command='device', device='list', device_number=0, get=None, offset='0x0000', set=None, verbose=None)
List of devices:

-  WB-HRMY-AVERAGE
-  WB-FMC-FV-CONTROL
-  WB-HRMY-PID
-  WB-FMC-ADC-CORE
-  WB-HRMY-MEMORY
-  WB-HRMY-ID-GEN
-  EM2_DAC
-  WB-HRMY-TIMEBASE
-  SPI
-  WB-EM2-DIGITAL_IO
-  WB-HRMY-FIFO
-  WB-FMC-ADC-CORE5M
-  WB-HRMY-COUNTER
-  WB-HRMY-CROSSBAR
```

Figura 47 – Registros de Wishbone de todo el sistema

Para este caso interesa más ver los registros que corresponden solo al *core* de la FMC. Para ello se puede utilizar el comando de la Figura 48 qué referencia a dicho módulo.

```
~ # alin device -d WB-FMC-ADC-CORE5M
Namespace(command='device', device='WB-FMC-ADC-CORE5M', device_number=0, get=None, offset='0x0000', set=None,
verbose=None)

VendorId:       0x000000000000alba
Device:         WB-FMC-ADC-CORE5M
Product:        0x00000013
Version:        0x00000002
First Address:  0x3200L
Last Address:   0x323fL
Description:    Wishbone slave for FMC ADC 18bits 5MSPs core

ADC_CH1 ....................: 0xefdbL
ADC_CH2 ....................: 0x0L
ADC_CH3 ....................: 0x0L
ADC_CH4 ....................: 0x0L
CTL_ADC_FSM ................: 0x1L
CTL_ADQ_M ..................: 0x0L
CTL_RST ....................: 0x0L
CTL_TRG ....................: 0x0L
CTL_TRIGID .................: 0x0L
DIV ........................: 0x0L
ID_CH1_ID ..................: 0x0L
ID_CH2_ID ..................: 0x0L
ID_CH3_ID ..................: 0x0L
ID_CH4_ID ..................: 0x0L
SAMPLES_CNT ................: 0x0L
```

Figura 48 – Registros de Wishbone referidos a la FMC

Cómo se puede observar en la Figura 48, ya es posible ver el registro con el valor en hexadecimal digitalizado por el ADC del canal 1 tras haber lanzado un *trigger.*

Algo que fue útil posteriormente para analizar las señales digitales, fue enviar manualmente *triggers* y capturar en el osciloscopio el intercambio de señales entre FPGA y FMC, para ello la forma es:

alin device -d WB-FMC-ADC-CORE5M -s CTL_TRG 1

El CTL_TRG es además un tipo de registro que ha sido definido en SDB para que automáticamente vuelva a 0 tras haber estado en 1 al ser escrito.

También es posible desde aquí cambiar el valor de DIV para modificar la frecuencia del *clock* de entrada, esto se haría mediante el siguiente comando:

alin device -d WB-FMC-ADC-CORE5M -s DIV 64

4.3 Script para registro de muestras

Teniendo una forma de acceder a los registros y escribir en ellos a través de consola, se ha elaborado un script en Python para automatizar el envío de *triggers* y las adquisiciones y escritura en fichero de las muestras a estudiar.

En el código del script, incluido en el Anexo, se puede ver como se envían *triggers* en intervalos programables mediante la modificación de la variable de tiempo de la función time.sleep() y se escriben las muestras en un fichero. El software de alto nivel no asegura una latencia exacta entre muestras, pero si aproximada.

4.4 Caracterización

Por supuesto la caracterización es una fase de las más importantes en todo nuevo prototipo. La caracterización que se tenía pensada para este prototipo estaba planeada para ser más exhaustiva, pero por problemas de tiempo se decidió realizar una más sencilla en la que se medirían ciertos parámetros como pruebas representativas, ya que el gasto de tiempo en adquirir la cantidad de muestras necesaria para realizar una buena caracterización es muy elevado.

Se incluye en la parte de caracterización una serie de pruebas que se hicieron para visualizar en el osciloscopio el correcto comportamiento del conjunto. Esto podría haberse hecho con CHIP Scope, una extensión que permite visualizar las señales internas de la FPGA. Visualizar señales externas también sería posible poniendo un registro en cada una de ellas y visualizándolo. Al final se decidió que la mejor manera era utilizar una sonda diferencial y 2 normales para visualizar simultáneamente las señales de interés accediendo directamente a ellas con el osciloscopio.

En los casos de la Figura las señales representadas corresponden al reloj (CLK±) en el canal 1, lazo positivo del par diferencial de salida de datos (D+) en el 2 y lazo negativo de par diferencial de salida de datos (D-) en el 3. El 4 corresponde al dato en si, que es el canal matemático que resta el canal 2 del canal 3, esto ha sido necesario de hacer para medir la señal de dato por ser diferencial. En el caso del reloj ha sido posible medir la señal directamente utilizando una sonda diferencial.

La Figura 49, fue tomada con DIV=64, es decir una frecuencia de reloj de aproximadamente 500 kHz. El valor devuelto a través del registro

correspondiente en el SDB en hexadecimal es 3cc0c, el cual corresponde con el dato que se lee en la señal de salida del ADC.

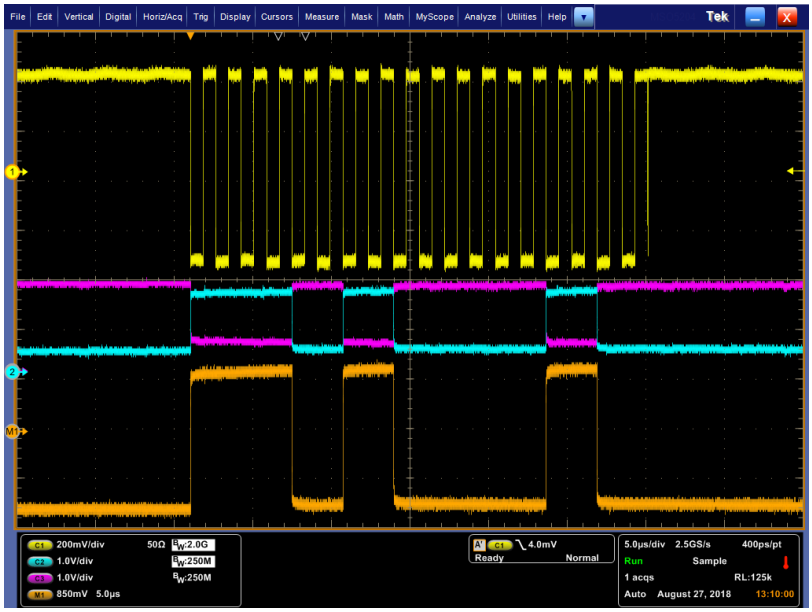


Figura 49 – CLK y DATA para DIV=64, 0x3cc0c

Para el caso de la Figura 50, con DIV=10 y frecuencia de 3.125 MHz, el valor leído en el registro es 3cb42 que nuevamente corresponde con el dato que se ve en la pantalla del osciloscopio.

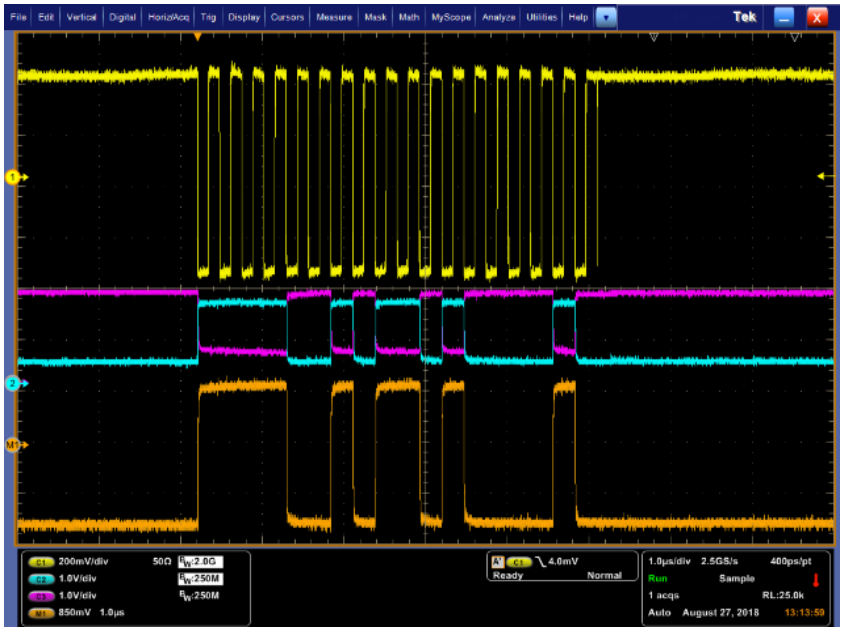


Figura 50 – CLK y DATA para DIV=10, 0x3cb42

En el caso de la Figura 51, y ya con el DIV a 2 y una frecuencia de unos 15.625 MHz, la distorsión empieza a ser más apreciable, el dato leído en el registro es 3cc2c, que también coincide con el dato que sale del ADC.

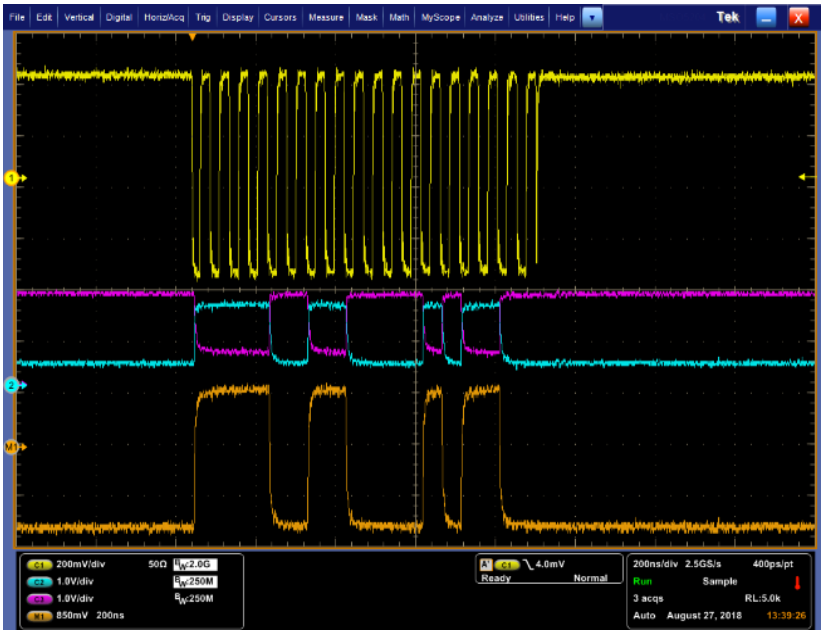


Figura 51 – CLK y DATA para DIV=2, 0x3cc2c

Fijando un valor de DIV a 0, es decir 125 MHz de reloj, el valor capturado en la FPGA es bf mientras que en el osciloscopio observamos que debería ser 1ff. Esto puede ser visto en la Figura 52. Esta diferencia como puede observarse corresponde al corrimiento de un bit de izquierda a derecha.

Figura 52 – CLK y DATA para DIV=0, 0xbf

Esto puede ser fácilmente visto simplemente comparando el valor leído en el osciloscopio con el valor contenido en la muestra.

	HEXADECIMAL	BINARIO
Valor muestra FPGA	0xbf	000000000010111111
Valor ADC osciloscopio	0x1ff	000000000101111111

Esta desincronización sucede porque el ADC pone el siguiente bit en la señal D después de un tiempo (t_clkd) del flanco de bajada del reloj que le llega, y la FPGA capta el dato en el próximo flanco de subida del reloj que ella misma genera. Asumiendo que el retardo de la pista de *clock* que va de la FPGA a este ADC es t_d_clk, que el retardo del ADC en poner el dato después de un flanco de bajada de reloj es t_clkd, y que el retardo de la pista D que va del ADC a la FPGA es t_d_D, cuando la suma de estos tres retardos es mayor a medio ciclo de reloj, el bit actual se captará en el siguiente flanco de subida de reloj en la FPGA. De esta forma, la palabra del ADC captada por la FPGA tiene un corrimiento de bit a la derecha, siendo de esta forma de un valor aproximadamente de la mitad de lo que debería ser.

Para solucionar esto habría que ajustar el retraso introducido por la longitud de las pistas en la salida de la misma en la FPGA, algo que por falta de tiempo no se hizo. Otra forma es utilizar el ADC en modo *echoed-clocked*, como ya se hace, pero adquiriendo los datos de entrada con la copia del reloj que el mismo ADC proporciona. De esta forma, tanto los datos como el reloj tienen un retardo similar (al ser DCO y D de distancias similares), suprimiéndose así los problemas de sincronización. Esta estrategia, que complica enormemente la máquina de estados, se ha considerado fuera del objetivo de este proyecto.

La siguiente parte de la caracterización consistió precisamente en visualizar el reloj (CLK±) con la sonda diferencial y el canal 1, y la rama positiva de la señal de conversión (CNV+) y el dato (D+) por los canales 2 y 3 respectivamente. En la Figura 53 se puede apreciar como a 125 MHz (DIV=0) el t_msb=200 ns de la Figura 37 se cumple.

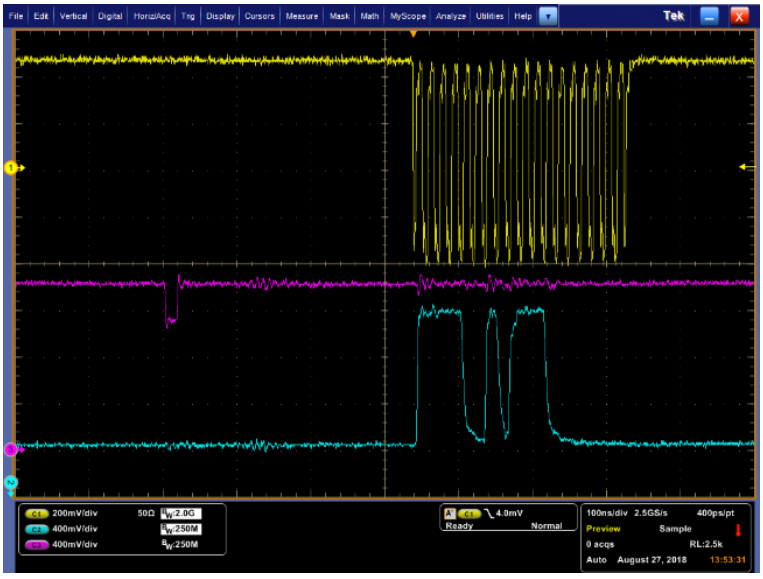


Figura 53 – Señales CLK±, CNV+ y D+ para DIV=0

Algo curioso que también se puede ver en la Figura 51 y Figura 53, que sucede en numerosas ocasiones, es el *glitch* que se puede ver al final del último ciclo de reloj. Este *glitch* se debe a la utilización del módulo DIV como divisor de reloj principal, aunque no afecta negativamente al comportamiento en ocasiones

puede descoordinar la adquisición en el ADC, pero no se han observado problemas. Al no ir a velocidades excesivamente altas esto no debería ser un problema

En la última parte de la caracterización se midió la señal de reloj (CLK±) y la señal de conversión (CNV±). Como se puede observar en la Figura 54, en el momento que se envía el pulso de CNV se esperan los 24 ciclos de reloj antes calculados hasta tener disponible el dato.

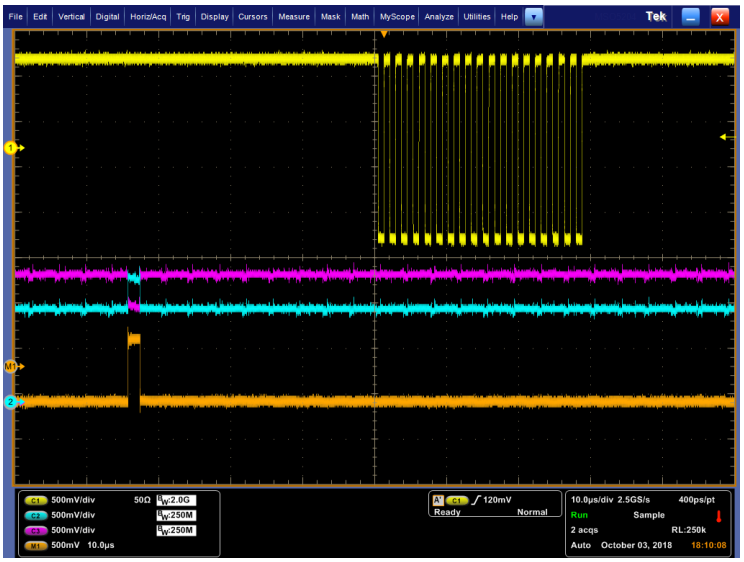


Figura 54 – Señales CLK±, CNV± para DIV=64

4.4.1 Caracterización con pila. Desviación estándar y ruido presente en la medida

En esta caracterización se utilizará una pila de 1.2 V recargable como entrada, la cual ha sido medida con un multímetro y marca un voltaje de 1.224 V. Se ha decidido el uso de una pila debido a la estabilidad que esta ofrece. Esto es algo crítico en esta prueba ya que se pretende medir la variación en la que oscilan los valores de cada una de las muestras y con ello deducir la desviación estándar a partir de un voltaje estable conocido. Se ha creído conveniente por tanto en esta, como en la siguiente caracterización pasar las muestras hexadecimales a decimal, para realizar las representaciones. Es importante tener en mente el hecho de que se tienen 262144 posibles valores diferentes, es decir; si el rango dinámico del ADC se divide entre la cantidad de saltos (valores posibles) se obtiene el salto de bit. De esta manera también se va teniendo en mente el salto de bit comparándolo con el ruido, ya que, a mayor resolución en bits más afectará el ruido comparativamente al salto, al estar los niveles cada vez más cerca entre sí. Este salto para un ADC de 18 bits y 20 Vpp de margen dinámico es de 76,29 uV.

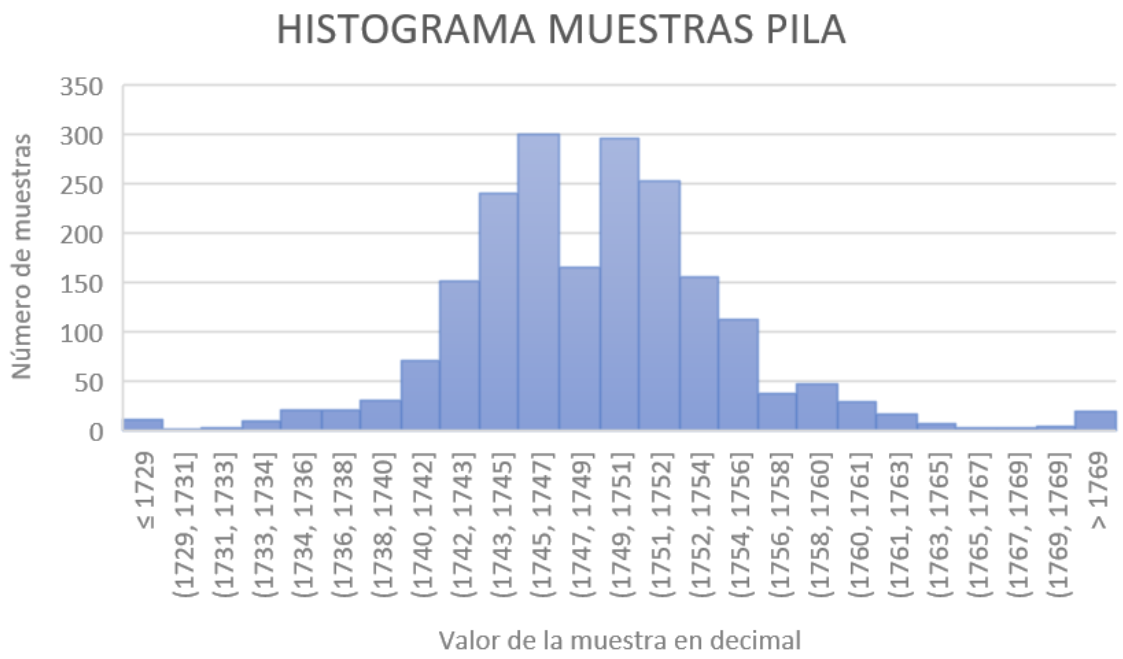


Figura 55 – Histograma de muestras pila recargable 1.224V

La desviación estándar para este conjunto de muestras es de 6.59. En la Figura 55 se puede apreciar que el margen de valores entre los que se distribuye es relativamente bajo, la mayoría de muestras están repartidas entre 10 valores distintos. Es decir, el error oscila en un margen de 762,9 uV lo que comparado con los 2^{18} valores posibles y teniendo en cuenta que es la primera iteración de diseño de esta tarjeta adquisidora, se consideran resultados más que aceptables.

4.4.2 Caracterización con rampa. Linealidad del ADC

En este caso se usará instrumentación externa, el generador de funciones BK 4009 será el encargado de generar una rampa de subida y bajada entre -10 V y 10V durante un intervalo de 200 segundos.

Durante ese tiempo se realizarán alrededor de 2000 adquisiciones y se mostrarán en una gráfica para ver con ello la linealidad de una manera bastante generalista, pero siendo una buena referencia como primera y rápida caracterización.

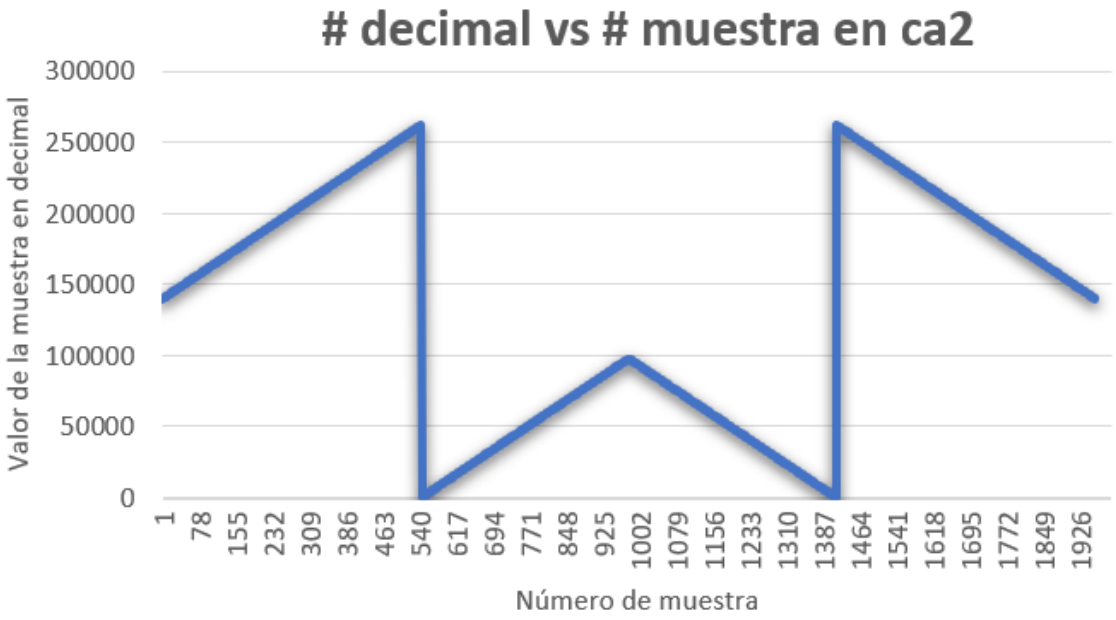


Figura 56 – Rampa obtenida de las muestras

Tal como se aprecia en la Figura 56, la linealidad en términos cualitativos es muy buena. Es importante el hecho de cómo se recorren esos valores, se ha de tener en cuenta que el ADC saca los datos representados en complemento a 2, por lo que el barrido de la Figura 56 en realidad es entre -10 V y 10 V. En la Figura 57 se muestra la gráfica una vez ya aplicado el complemento a dos.

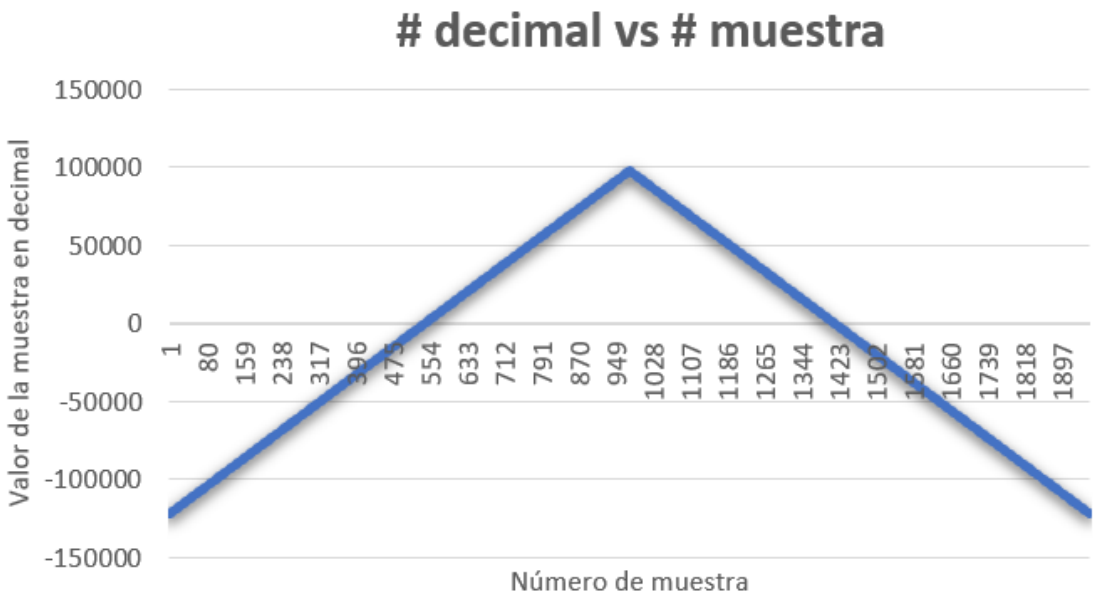


Figura 57 – Rampa obtenida de las muestras una vez aplicado el complemento a dos

4.4.3 Caracterización del driver

Debido a la descompensación existente en cada una de las ramas del driver, se detectó durante las pruebas que esto generaba un offset de 0.46 V. Para entender mejor el funcionamiento de éste y estudiar de forma más clara el posible impacto en el mal funcionamiento del diseño, se decidió hacer una pequeña caracterización del driver en distintas circunstancias de entrada de señal con las que se ha elaborado la Tabla 2.

	In+	In-	In*	Vout+	Vout-	Vout*
cc	1,874V	1,869V	0,000V	2,257V	2,720V	0,463V
ca	2,263V	2,241V	0,043V	2,721V	2,257V	0,464V
-10V	0,152V	0,145V	-0,002V	0,182V	4,819V	-4,623V
-5V	1,010V	1,006V	-0,002V	1,216V	3,760V	-2,552V
0V	1,850V	1,846V	0,001V	2,234V	2,744V	-0,519V
5V	2,730V	2,721V	0,001V	3,324V	1,674V	1,612V
10V	3,594V	3,587V	0,001V	4,341V	0,638V	3,698V

Tabla 2 – Relaciones entrada-salida

* *Las medidas In y Vout son una medida diferencial directa entre ambos pines, no la diferencia entre cada voltaje respecto de GND.*

Lo primero que se puede deducir de la Tabla 2, es que existe un offset de 0.46 V que desplaza la señal. Se puede apreciar a la salida con las entradas en cc o ca. Se ha de asegurar que, para los casos más extremos de señal de entrada, las tensiones a la salida del driver no caen fuera de los límites del margen dinámico del ADC. Para los 2 casos más extremos, que corresponden a entradas de 10 V y -10 V las salidas son respectivamente, 3.698 V y -4.623 V, por lo que no se tendría problema en cuanto a tener la señal fuera de rango por ser ±10 V los límites de la entrada diferencial del ADC. No obstante, de esta forma se estarían perdiendo un total de 1.7 V de margen dinámico. Lo que equivale a que se tenga una resolución efectiva de 17.23 bits.

$$bits_{eff} = log_2\left(\frac{2^{18}}{1.7}\right) = 17.23\ bits$$

Esto necesitaría un reestudio de los valores de las resistencias de adaptación, así como un reajuste de la realimentación del driver, ya que la ganancia se podría aumentar más.

Por otra parte, cabe señalar que este offset debido a la descompensación de impedancias entre las 2 ramas de entrada es porque al ser sensiblemente diferentes generan una diferencia de tensión en la entrada que se traduce como un desplazamiento en tensión en la salida.

5. PRESUPUESTO

Para realizar el presupuesto con el coste total se han realizado dos tablas diferentes, una solo para los componentes de la placa y otra para otro tipo de gastos generados por la fabricación y montaje de la misma.

En la Tabla 3, se pueden ver los designadores de todos los componentes que conforman el prototipo, precio por unidad, cantidad, breve descripción, así como código de distribuidor. Para hacerlo se ha modificado un BOM, generado directamente desde Altium.

La referencia del campo Supplier Reference ha sido puesta utilizando una notación para facilitar la labor a la hora de realizar el pedido. Las referencias están precedidas de un prefijo M_, F_, D_ y R_ que corresponden respectivamente a los distribuidores Mouser, Farnell, Digikey y RS. Lo que sigue al guion es la referencia interna de cada uno de los componentes en el proveedor escogido para cada uno. Al tratarse de un prototipo, la cantidad de unidades que se van a producir es baja, por tanto, en general se ha pagado un alto precio en cada componente al no recibir las bonificaciones por cantidad que los distribuidores ofrecen.

El campo Mount se ha creado en este caso para cada componente para tener información tanto para el montador como para el desarrollador de la placa, respecto de que componentes, por distintos motivos, no deben ser montados. Es decir, su *footprint* existirá al existir el componente como tal, pero luego no serán montados ya que su utilidad puede ser futura o con otros propósitos por los que no interesa que el componente se monte por defecto. Este es el caso de las resistencias que se han instalado para activar manualmente los relés que modifican la impedancia de las entradas de la FMC.

En la Tabla 3, se han desglosado los costes generados por la fabricación de la PCB, y por el montaje de los componentes más críticos. En un principio la idea era que el montador montase todos los componentes de la placa, pero por problemas de plazos se decidió montar solo los componentes con encapsulado LFCSP y el conector Samtec, montando el resto de componentes manualmente. De la misma forma que al comprar componentes se consiguen grandes beneficios por volumen, en la fabricación de la PCB y montaje de componentes de la misma sucede lo mismo, al ser un número de unidades bajo se paga un alto precio por cada una de ellas.

Se puede justificar el desarrollo de una solución basada en FMC+SPEC en sustitución de las DAQ Adlink tanto a nivel económico como a nivel de prestaciones. Las tarjetas adquisidoras que serán sustituidas serán las ADLINK modelo DAQ-2005 y DAQ-2010, con 16bits-500ksps y con 14bits-2MSPS respectivamente. Ambas con 4 canales y mismo precio, 1541€.

La solución que se implantará tiene un coste aproximado de 600€ respecto de la SPEC, y de otros 600€ respecto de la FMC. Se cuentan 600€ y no 826€ de la FMC porque una vez que se haga producción el coste de fabricación descenderá

mucho. Esto significa que se ahorrarán unos 341€ por unidad, que se tiene mejor resolución y que se tendrá mejor velocidad una vez se haya programado el firmware para ello.

Supplier Reference	Value	Description	Designator	Mount	Qty	Price/unit	Subtotal
F_2447060	0 R	Resistor, 0R, 0.063W, 0603	R9, R40, R45, R46, R47, R48, R49, R50, R51, R52, R68, R99, R101	NO	13	0.01 €	0.07 €
F_2447587	1k	Resistor 0805 0.1W 1%	RT1, RT2, RT3, RT4	NO	4	0.01 €	0.04 €
R_685-1315	-	Hirose FFC Connector, 0.5mm Pitch, Bottom Contact, 40way	P1, P2	NO	2	1.41 €	2.82 €
D_541-2659-1-ND	50R	Resistor, 50R, 2.4W, 1206	R6, R25, R61, R80	YES	4	1.62 €	6.48 €
D_AD8032AR-ND	-	Double buffer	U2, U8, U11, U13	YES	4	4.57 €	18.28 €
D_ADA4932-2YCPZ-R2CT-ND	-	ADC Driver	U3, U9	YES	2	12.08 €	24.16 €
D_ADM7150ACPZ-5.0-R7CT-ND	-	5V regulator	U14	YES	1	7.61 €	7.61 €
D_MCP23008T-E/SSCT-ND	-	SPI Port Expander 8bits I/O, SSOP	U5	YES	1	0.96 €	0.96 €
F_1301713	0.1uF	Capacitor, MLCC, X7R, 0.1uF, 10%, 50V, 0603	C1, C3, C4, C5, C6, C7, C8, C9, C12, C13, C14, C15, C17, C18, C20, C21, C22, C23, C24, C25, C26, C30, C31, C33, C34, C35, C36, C37, C38, C39, C42, C43, C44, C45, C47, C48, C50, C51, C52, C53, C54, C55, C56, C60, C62, C63, C64, C65, C66, C67	YES	50	0.21 €	10.30 €
F_1757895	-	Dual NPN BJT Transistor, SOT-457	QRS1, QRS2	YES	2	0.24 €	0.48 €
F_1829192	-	MMBZ12VDL - DIODE, DUAL TVS, 40W, 8.5V, SOT23-3	D1, D2, D4, D5	YES	4	0.18 €	0.74 €
F_2280655	1nF	Capacitor, MLCC, 1nF, 50V, X7R, 0603	C16, C29, C46, C59, C73	YES	5	0.18 €	0.88 €
F_2310668	100nF	Capacitor 0.1uF 10% 25V XR7	CP1	YES	1	0.09 €	0.09 €
F_2433507	-	SAMTEC_ASP-134604-01. Connector VITA 57, 160 Male contacts in 4 row	J5	YES	1	11.24 €	11.24 €
F_2447060	0R	Resistor, 0R, 0.063W, 0603	R1, R14, R20, R26, R31, R37, R56, R69, R74, R81, R86, R92, R100, R102	YES	14	0.01 €	0.07 €
F_2447587	1k	Resistor 0805 0.1W 1%	Rel-1, Rel-2, Rel-3, Rel-4	YES	4	0.01 €	0.04 €
F_2561909	25R	Resistor, 25R, 0.250W, 0603	R19, R38, R75, R93	YES	4	0.13 €	0.52 €
F_2727421	-	1V8 regulator	U15	YES	1	2.31 €	2.31 €
F_3817866	-	LEMO EPL.00.250.NTN - SOCKET, 00, RIGHT ANGLE, PCB, 50OHM	J1, J2, J3, J4, TRIG_IN1	YES	5	16.38 €	81.90 €
F_GRM21BR61E106KA73L	10uF	Capacitor, MLCC, X5R, 10uF, 10%, 25V, 0805	C68, C69, C70	YES	3	0.62 €	1.87 €
M_584-AD7960BCPZ	-	ADC 5MSPS, 18bit	U1, U4, U7, U10	YES	4	42.77 €	171.08 €
M_584-ADR4550ARZ	-	5V Reference	U12	YES	1	5.25 €	5.25 €
M_660-RN73H1JT9881F100	9.88k	Resistor, 9.88kOhm, 0603, 1%, 100mW, 75V	R16, R33, R71, R88	YES	4	0.22 €	0.88 €
M_810-C2012JB1C106K08C	10uF	Capacitor, MLCC, X5R,10uF, 10%, 50V, 0603	C2, C19, C32, C49	YES	4	0.23 €	0.92 €
M_810-CGA3E1X7R1C105AC	1uF	Capacitor, MLCC, X7R, 1uF, 10%, 50V, 0603	C61, C71, C72	YES	3	0.11 €	0.33 €
M_876-MS12-1A87-75D	-	12V RELE, MS12-1A87-75D	KT1, KT2, KT3, KT4, KT5, KT6, KT7, KT8	YES	8	3.66 €	29.28 €
R_679-0465	4.22k	4.22kΩ, ±1%, 0,1W, Película Gruesa	R21, R22, R23, R24, R76, R77, R78, R79	YES	8	1.30 €	10.40 €
R_815-1386	4.7uF	Capacitor, MLCC, 4.7µF, ±10%, 25V	C74, C75	YES	2	0.25 €	0.50 €
S_ 509-677	-	¡Ultra low capacitance, ESD protection diode, SOD323	D3	YES	1	0.14 €	0.14 €
R_ 660-8262	-	SN65LVDS1DBV. High Speed Differential Line Driver	U6	YES	1	7.91 €	7.91 €
R_122-9582	634R	0603, 637R, 1%	R7, R8, R36, R39, R62, R63, R91, R94	YES	8	4.08 €	32.64 €
R_122-9621	733k	Resistor, 732kΩ, ±0.1%, 0.063W	R18, R35, R73, R90	YES	4	3.84 €	15.36 €
R_122-9663	8.45k	Resistor, 8.45kΩ, ±0.1%, 0.063W	R15, R17, R32, R34, R70, R72, R87, R89	YES	8	0.20 €	1.63 €
R_223-2120	100R	Resistor 100Ohm, ±1%, 0,25W	R54, R55	YES	2	0.02 €	0.03 €
R_264-4523	100pF	0603, MLCC, 5%, 50V	C10, C11, C27, C28, C40, C41, C57, C58	YES	8	0.03 €	0.26 €
R_679-1951	270R	Resistor 0805 0.1W 1%	RR1, RR2, RR3, RR4, RR5, RR6, RR7, RR8	YES	8	3.05 €	24.40 €
R_810-2097	100R	Resistor, Thick Film, 100R, 0.1W, 0.5%, 0603	R103, R104, R105, R106, R107, R108, R109, R110	YES	8	0.03 €	0.25 €
R_826-9967	-	Schottky Barrier Double Diode, common cathode	D6, D7	YES	2	0.08 €	0.15 €
						TOTAL	*472.26 €*

Tabla 3 – Lista total de componentes

Fabricación PCB	3	530 €	177 €
Montaje chips/conector	3	530 €	177 €
Componentes	-	472 €	472 €
		TOTAL	826 €

Tabla 4 – Precio total incluyendo fabricación y montaje

6. CONCLUSIONES Y FUTURAS MEJORAS

Se hablará punto por punto acerca de los requerimientos mínimos de la aplicación y su consecución.

- Frecuencia de muestreo 1 MSPS: En un principio se tenía pensado hacer pruebas de velocidad para saber hasta donde se podía llegar teniendo en cuenta que el ADC es de 5 MSPS. Luego se comprobó que el tiempo disponible no era suficiente para realizar pruebas de este tipo. Además, para llegar a esos 5 MSPS teóricos sería necesario hacer uso de 2 máquinas de estado simultáneamente, con lo que el código VHDL se complicaba enormemente y por lo que se decidió hacer una implementación con una sola máquina de estados. No obstante, se da por hecho que implementando la segunda máquina de estados no debería existir motivo por el que no se puede superar 1 MSPS e incluso llegar a 5MSPS. De hecho, aunque sea una sola muestra y corriéndose un bit ya se hicieron adquisiciones a 3.63 MSPS. Esto puede ser visto en la Figura 51 y se calcula como:

$$f_{muestreo} = \frac{1}{(5.5 * 50 * 10^{-9})} = 3.63MSPS$$

- Resolución de 16 bits: Este es un parámetro que se cumple y además se mejora, al ser capaz la FMC de adquirir muestras de 18 bits pero que en realidad son 17.23 bits efectivos hasta que se termine de reajustar.

- Entrada de señal *single-ended*: Fue necesario hacer uso de un driver para poder asegurar una señal de entrada desbalanceada. La entrada del ADC tiene entradas diferenciales. Se comprobó el mercado y las opciones que existían para ADCs con entrada directamente desbalanceada, pero ninguno era adecuado por sus características y era preferible la opción al final escogida, incluir un driver para pasar la señal de *single-ended* a diferencial y utilizar un ADC de entradas diferenciales. Por tanto, este objetivo se da por cumplido.

- Rango de entrada ±10 V: El rango de entrada permite señales de ±10 V, aunque no queden perfectamente ajustadas al rango de entrada del ADC por lo que el objetivo también se da por cumplido.

- INL < 2 LSB: Aunque el ADC tiene como especificación un INL típico de 0.85 LSB y máximo de 2 LSB, esta medida de linealidad no ha podido ser comprobada en la caracterización. Aunque se espera que el INL siga por debajo de 2 LSB, no se puede asegurar que este objetivo haya sido cumplido al no disponer de medida alguna.

- VITA-57 compatible: Desde un principio se hicieron todas las medidas de PCB y se pusieron los componentes con una disposición tal que sus medidas fuesen compatibles con el estándar. Así mismo también se cuidó el ruteado de las líneas en el conector Samtec para no salirse de la

especificación, por lo que este es un objetivo cumplido. Para ser estrictos se ha de decir que los conectores FCC de 40 pines no están permitidos por la especificación, por lo que la placa será VITA-57 compatible siempre y cuando dichos conectores no se monten.

- Máximo número de líneas de control FMC-FPGA libres: Este no es un objetivo como tal, sino una directriz que en todo momento se tuvo presente, aunque casi siempre se consideraron prioritarias unas especificaciones mejores antes que no mantener libres un mayor número de líneas de control.

Teniendo en cuenta que es un prototipo se da por satisfecho el desarrollo y puesta en marcha del mismo. El limitado tiempo del proyecto ha impedido realizar una caracterización más exhaustiva para valorar con más precisión las verdaderas prestaciones de la tarjeta adquisidora. También por este motivo no ha sido posible trabajar más en buscar unas resistencias óptimas en la red de entrada que permitiesen ajustar de forma más precisa la señal al margen del ADC ganando con ello resolución. Tampoco el tiempo fue el suficiente para soldar el *port expander* e implementar el código VHDL necesario para activar los relés, algo que se consideró menos interesante y prioritario. A pesar de esto la placa está diseñada con esta característica y es posible hacerla funcionar en el momento en que se monte el *port expander* y se añada en el firmware el código necesario para cambiar un registro a través de SDB y mediante señalización I2C.

Se ha establecido la base de lo que será un dispositivo que suplirá y mejorará las funciones de lo que hasta el momento estaban haciendo equipos comerciales que quedarán fuera de producción.

Ya que el propósito de este prototipo es llegar a un desarrollo final que sustituya a los actuales DAQ Adlink. Para hacer efectiva esta sustitución existen cosas a corregir y mejorar en este prototipo para una futura versión que se lleve a producción. Para ello se deberá hacer una caracterización más completa del prototipo ya fabricado que la aquí realizada, a fin de identificar otros posibles problemas no identificados en esta memoria. El objetivo es tomar las medidas oportunas para subsanarlos en la fase de diseño de la segunda versión. A continuación, se hace un repaso a la lista de correcciones y mejoras necesarias para la próxima versión.

1. Cambiar el *footprint* para eliminar el *rework* y utilizar definitivamente el AD4940 en lugar del AD4932. Respecto al *rework* del regulador no será necesario cambiar nada, al ser un problema de falta stock temporal se puede usar el mismo que en un principio.
2. Modificar la PCB para conectar el pin EN3 a 1.8V y no a GND.
3. Ajustar los valores de las resistencias de la red de entrada para eliminar la descompensación en la entrada del driver y eliminar el offset generado por ello para hacer coincidir con más precisión la señal y el margen dinámico del ADC.
4. Implementar segunda máquina de estados para poder hacer funcionar la FMC a 5MSPS por canal.

7. BIBLIOGRAFÍA Y RECURSOS

7.1 Bibliografía y documentación

[1] https://www.cells.es/en/accelerators/front-ends

[2] Texas Instruments. *"Choose the right A/D converter for your application"*

[3] M. Broseta†, X. Serra-Gallifa, J. Avila-Abellán, S. Blanch, G. Cuní, D. Fernández-Carreiras, O. Matilla, M. Rodriguez, J. Salabert *"Present and future if harmony bus, a real-time high speed bus for data transfer between FPGA cores"*. [Barcelona], ALBA Syncrothron, ICALEPCS 2017

[4] J. Ávila *"Generador de campo magnetico pulsado"* [Master Tesis]. [Barcelona], Universitat Politecnica de Catalunya, 2015.

[5] M. Broseta†, X. Serra-Gallifa, J. Avila-Abellán, S. Blanch, G. Cuní, D. Fernández-Carreiras, O. Matilla, M. Rodriguez, J. Salabert *"Present and future if harmony bus, a real-time high speed bus for data transfer between FPGA cores"*. [Barcelona], ALBA Syncrothron, ICALEPCS 2017

[6] J. Ávila, "Electrometer comes to life", ALBA Syncrothron, 2017, http://accelconf.web.cern.ch/AccelConf/icalepcs2017/talks/tuapl04_talk.pdf

[7] M. Broseta, X. Serra *"Gateware software framework at ALBA: Current application in Electrometer at ALBA"* [Barcelona], ALBA Syncrothron, 2016

[8] Wishbone Computer Bus, http://opencores.org/ howto/wishbone

[9] Wishbone Generator, https://www.ohwr.org/projects/wishbone-gen

[10] Self-describing Bus, https://www.ohwr.org/projects/sdb

[11] ANSI-VITA-57.1-Rev 2010, http://cds.cern.ch/record/1172409/?ln=es

[12] Volnei A. Pedroni, *"Circuit Design with"* VHDL, 2004

[13] D. Amos, A. Lesea, R. Richter *"FPGA-Based Prototyping Methodology Manual: Best Practices in Design-For-Prototyping",* 2011

7.2 Figuras

Figura 1. Texas Instruments. 2009. Choose the right A/D converter for your application [online]. Available at: https://www.ti.com/europe/downloads/Choose%20the%20right%20data%20converter%20for%20your%20application.pdf

Figura 2. Wikipedia. 2018 Successive approximation ADC [online]. Available at: https://en.wikipedia.org/wiki/Successive_approximation_ADC

Figura 5. Analog Devices 2016. AD4932 Datasheet [online]. Available at: http://www.analog.com/media/en/technical-documentation/data-sheets/ADA4932-1_4932-2.pdf

Figura 6. Analog Devices 2016. AD4932 Datasheet [online]. Available at: http://www.analog.com/media/en/technical-documentation/data-sheets/ADA4932-1_4932-2.pdf

Figura 8. Nexperia. 2002. NPN general purpose double transistor datasheet [online]. Available at: https://assets.nexperia.com/documents/data-sheet/BC817DS.pdf

Figura 15. EEWEB. 2010. Microstrip Impedance Calculator [online]. Available at: https://www.eeweb.com/tools/edge-coupled-stripline-impedance

Figura 16. EEWEB. 2010. Microstrip Impedance Calculator [online]. Available at: https://www.eeweb.com/tools/edge-coupled-stripline-impedance

Figura 37. Analog Devices. 2014. 18-Bit, 5 MSPS PulSAR Differential ADC datasheet [online]. Available at: http://www.analog.com/media/en/technical-documentation/data-sheets/AD7960.pdf

Figura 42. Openhardware. 2014. Gateware software framework at ALBA – Current Application in electrometer [online]. Available at: https://ohwr-production.s3-eu-west-1.amazonaws.com/uploads/attachment/file/4798/CERN_talk.pdf?X-Amz-Expires=600&X-Amz-Date=20181010T144612Z&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAJDWHW5JNHWMBZXXQ/20181010/eu-west-1/s3/aws4_request&X-Amz-SignedHeaders=host&X-Amz-Signature=bbfc58f044c86645c9af73030926197d7fc688601f8ea7d5cfd53382eb9fb0a3

Figura 44. Openhardware. 2014. Gateware software framework at ALBA – Current Application in electrometer [online]. Available at: https://ohwr-production.s3-eu-west-1.amazonaws.com/uploads/attachment/file/4798/CERN_talk.pdf?X-Amz-Expires=600&X-Amz-Date=20181010T144612Z&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAJDWHW5JNHWMBZXXQ/20181010/eu-west-1/s3/aws4_request&X-Amz-SignedHeaders=host&X-Amz-Signature=bbfc58f044c86645c9af73030926197d7fc688601f8ea7d5cfd53382eb9fb0a3

Reliz

Figura 46. Openhardware. 2014. Gateware software framework at ALBA – Current Application in electrometer [online]. Available at: http://accelconf.web.cern.ch/AccelConf/icalepcs2017/talks/tuapl04_talk.pdf

ANNEXOS

TÍTOL DEL TFG: Design of a FPGA Mezzanine Card compatible with VITA 57 standard

TITULACIÓ: Grau en Enginyeria de Sistemes de Telecomunicació

AUTOR: Jesús Álvarez Alonso

DIRECTOR: Pere Lluís Gilabert

SUPERVISOR: José Ávila Abellán

DATA: 16 de septiembre del 2018

Contenido

8. ESQUEMÁTICOS

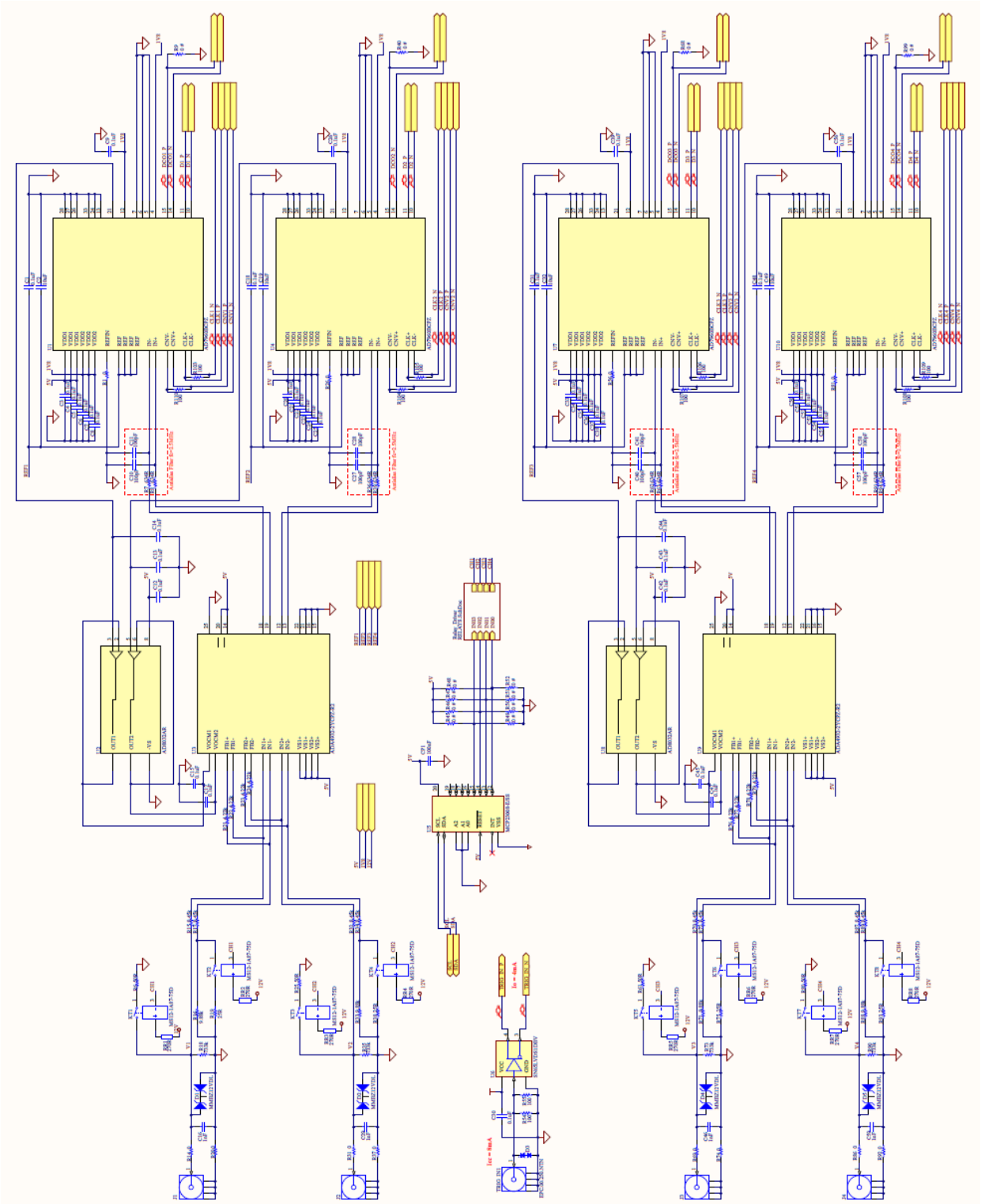


Figura 58 – Cadena completa de canales y port expander

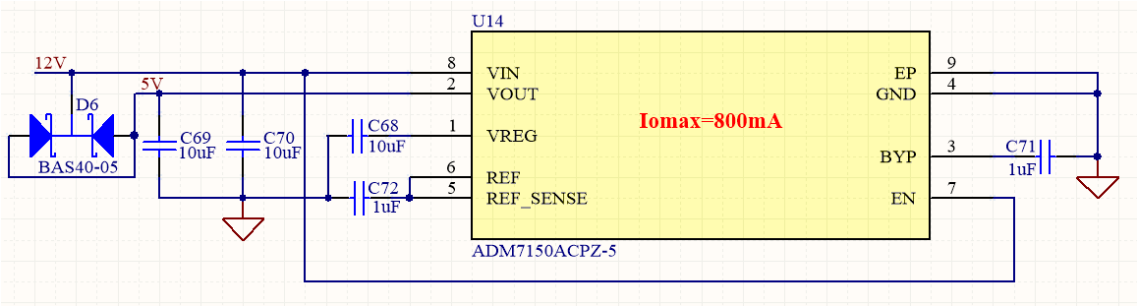


Figura 59 – Regulador de 5V

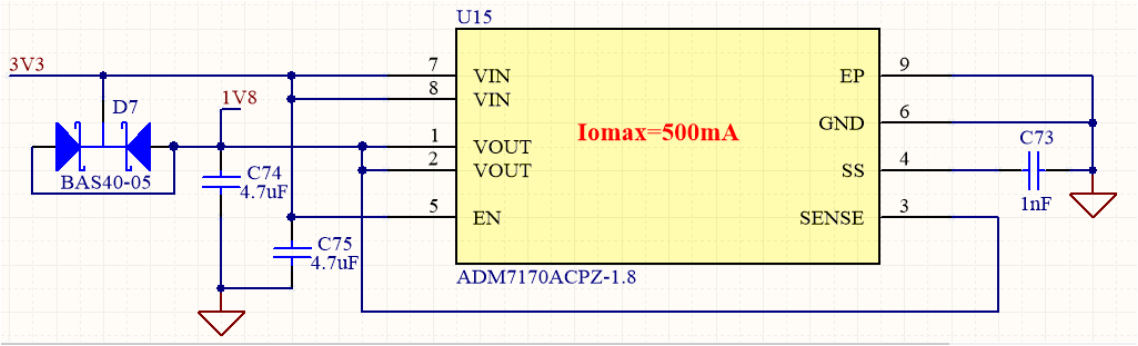


Figura 60 – Regulador de 1.8V

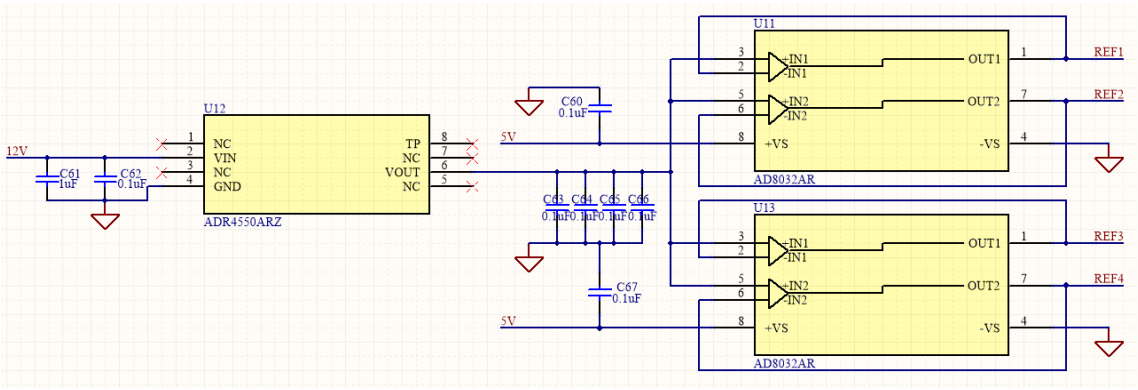


Figura 61 – Referencia de tensión de 5V y seguidores de tensión

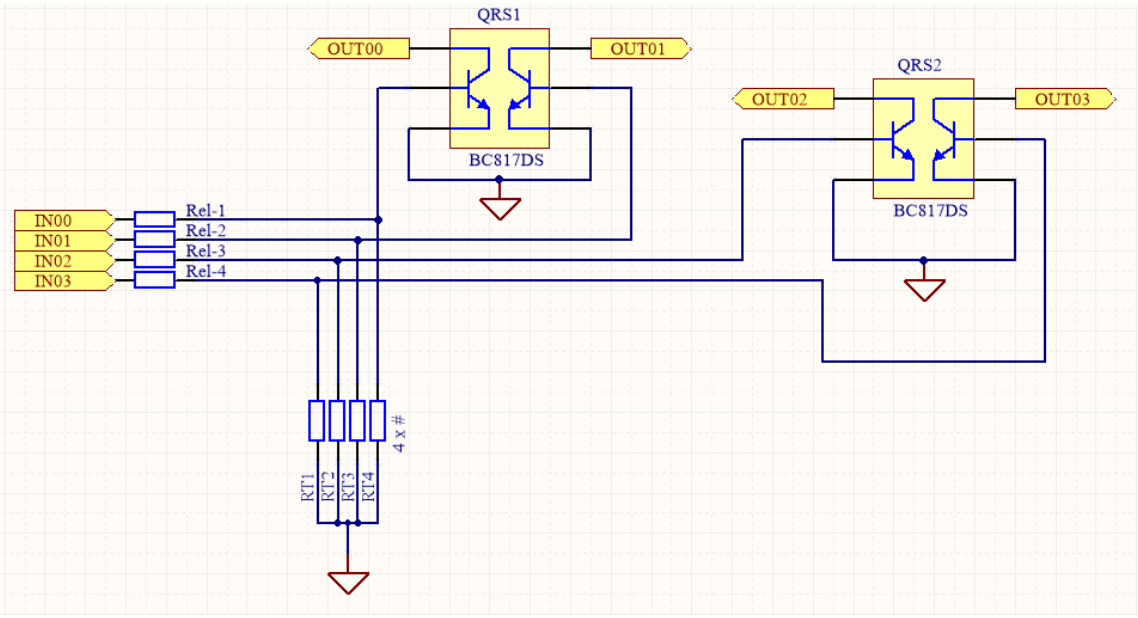


Figura 62 – Transistores para activación de relés

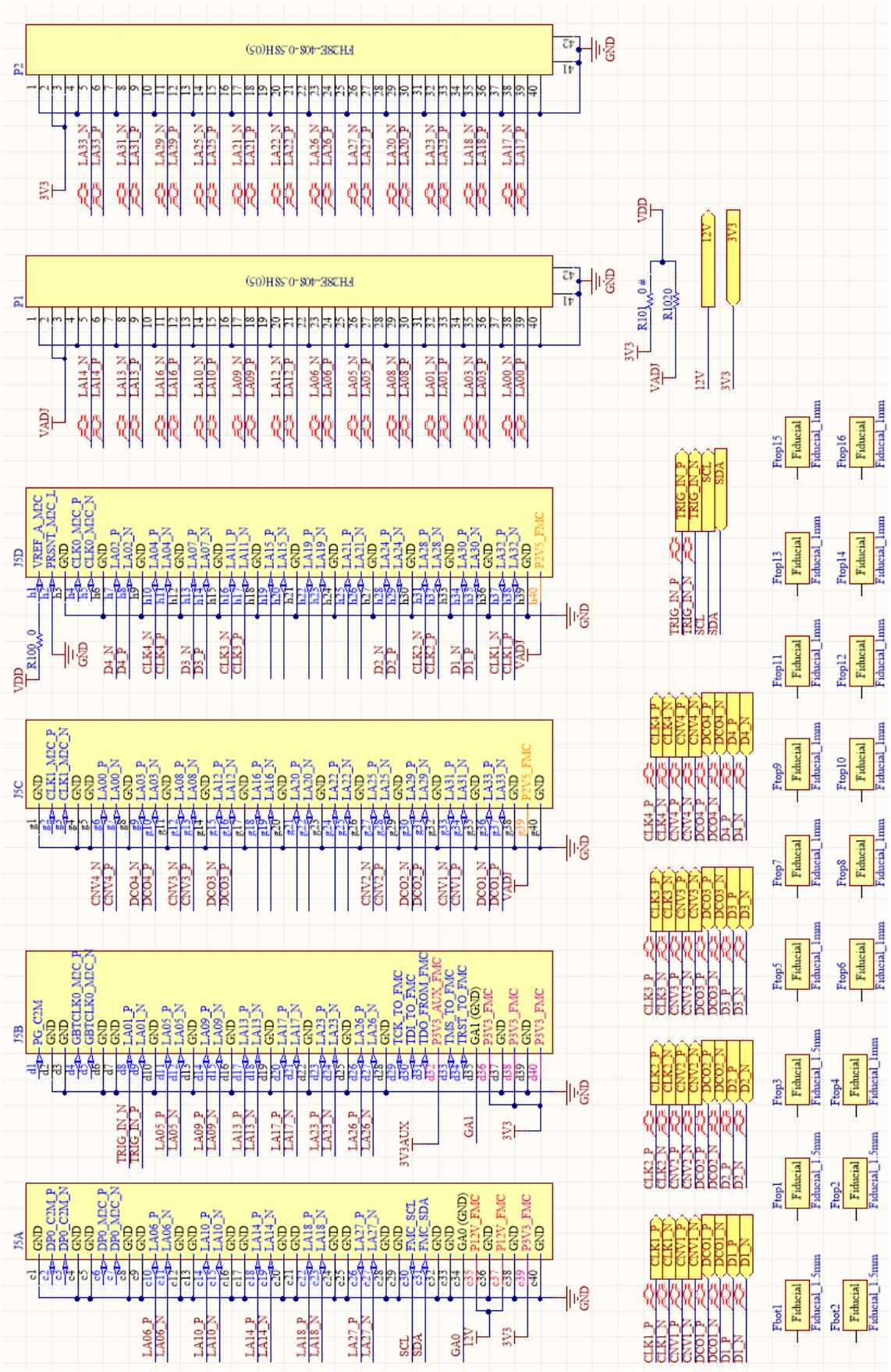


Figura 63 – Pinout conector Samtec y FCC 40 pines

9. GERBERS

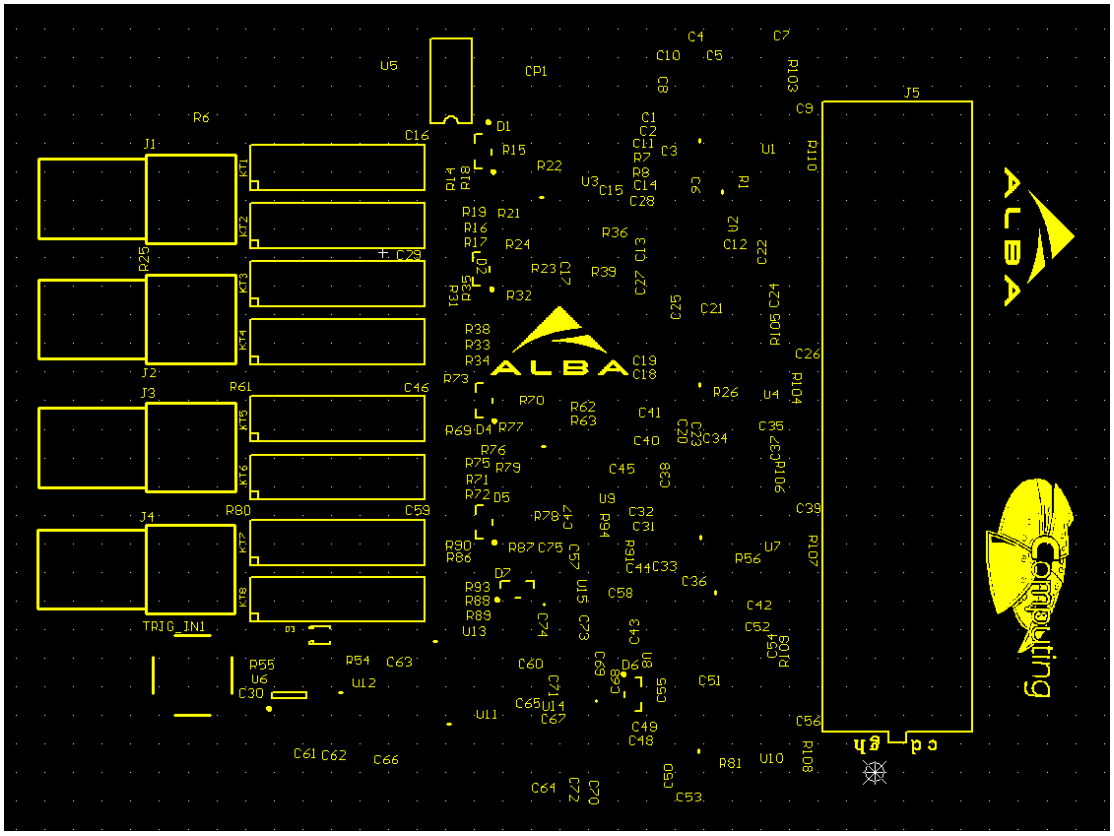


Figura 64 – Capa de serigrafia superior

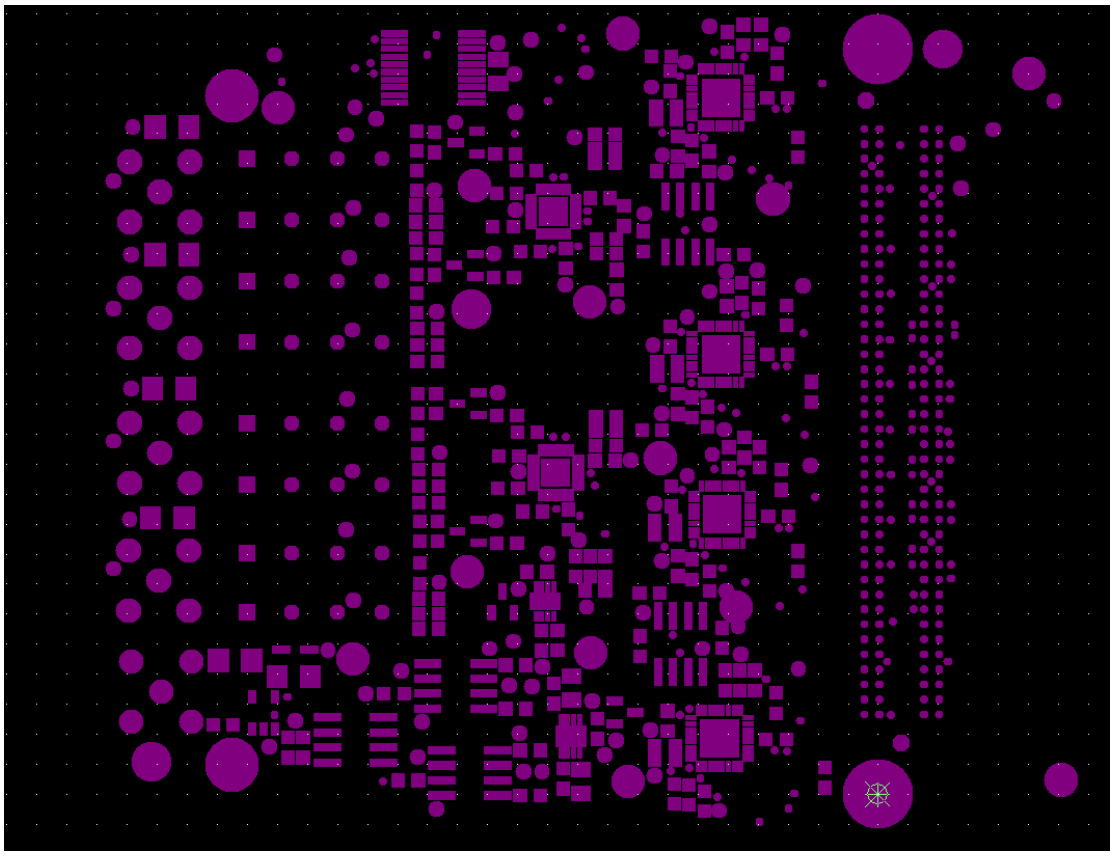


Figura 65 – Capa de máscara de soldadura superior

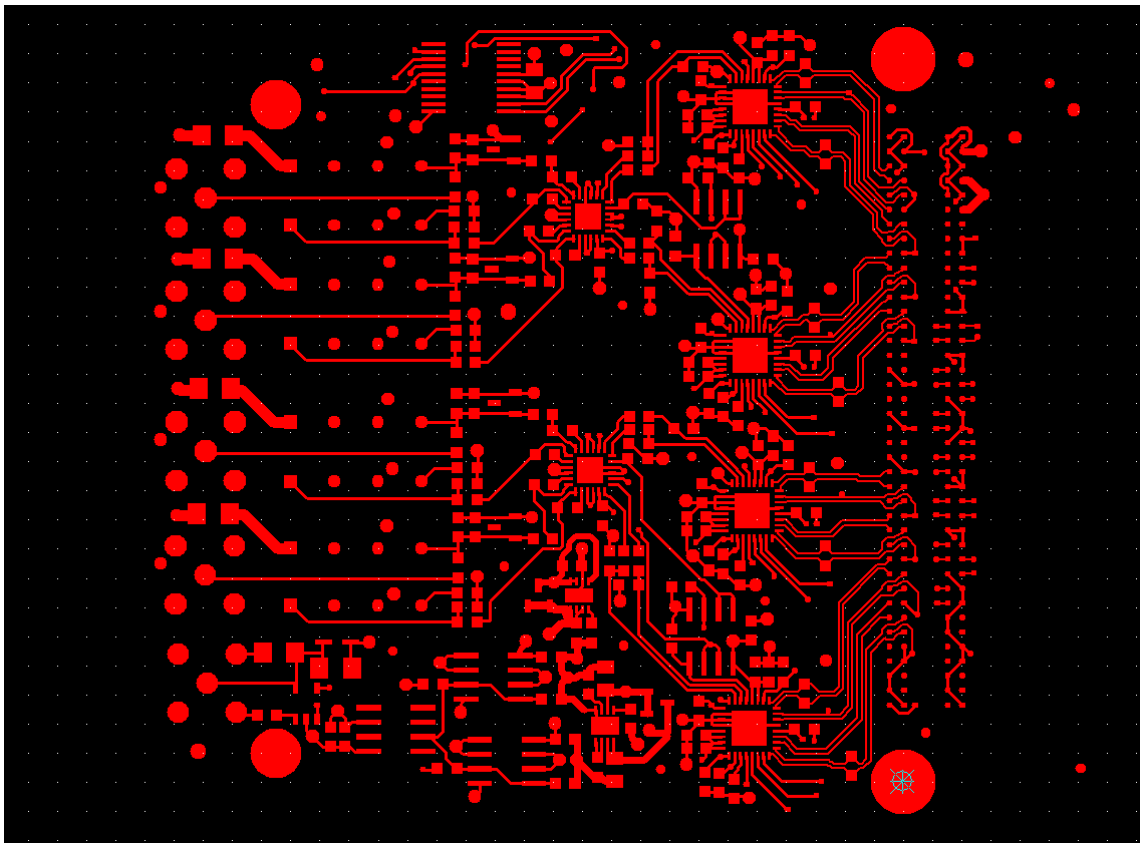


Figura 66 – Capa de pistas superior

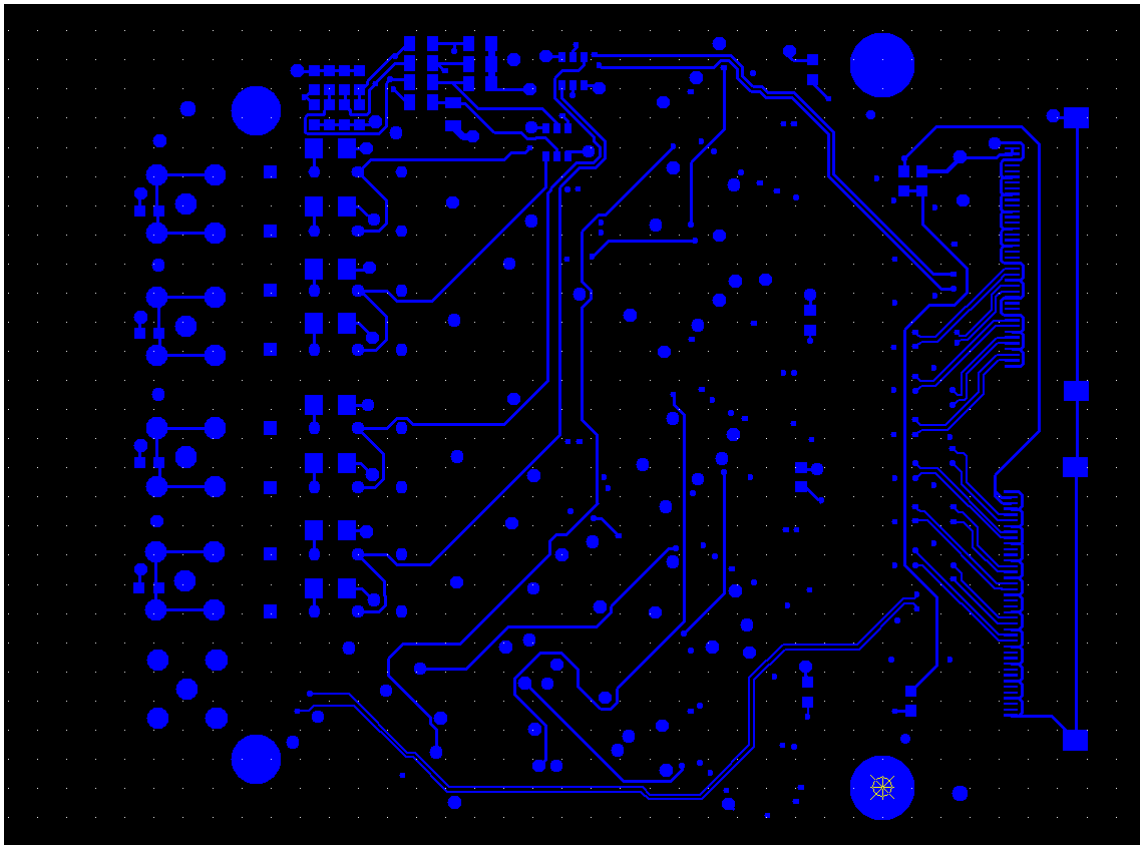


Figura 67 – Capa de pistas inferior

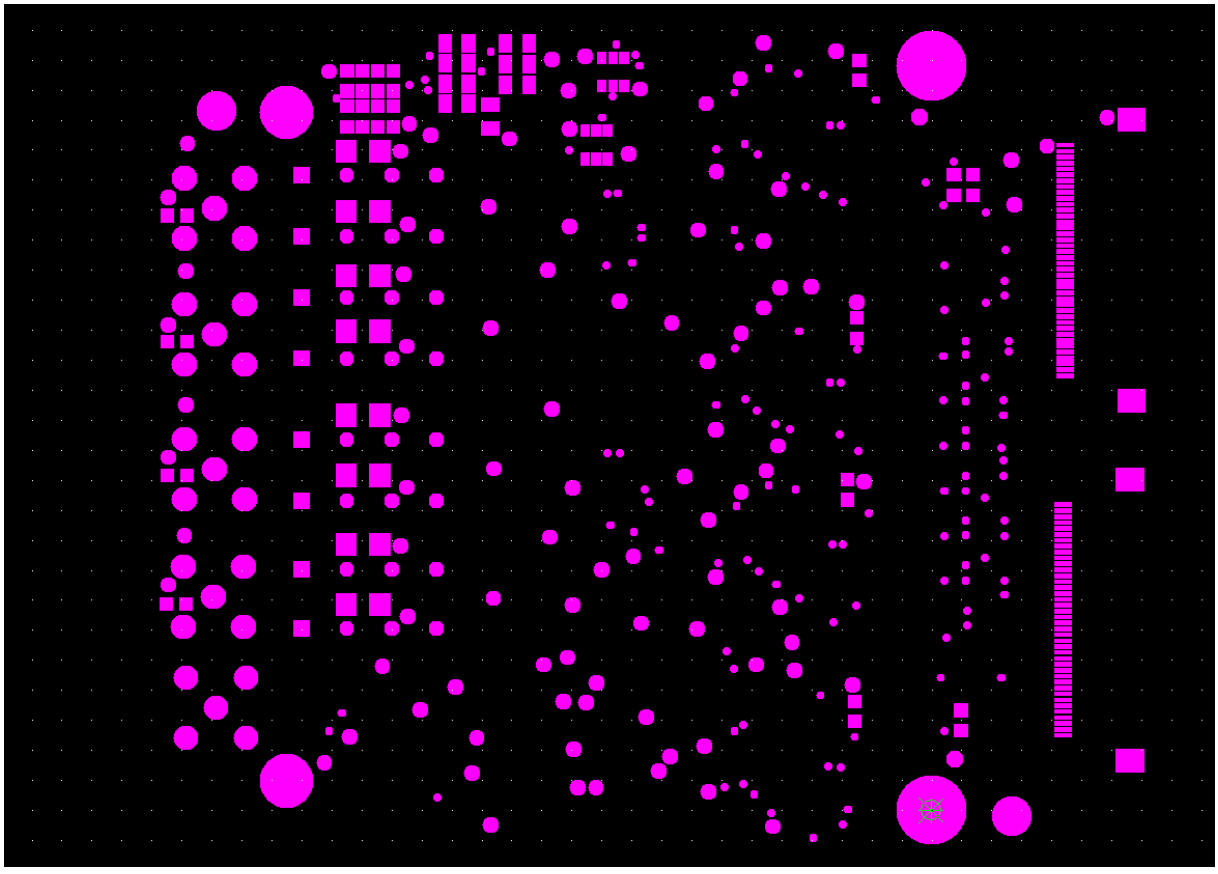


Figura 68 – Capa de máscara de soldadura inferior

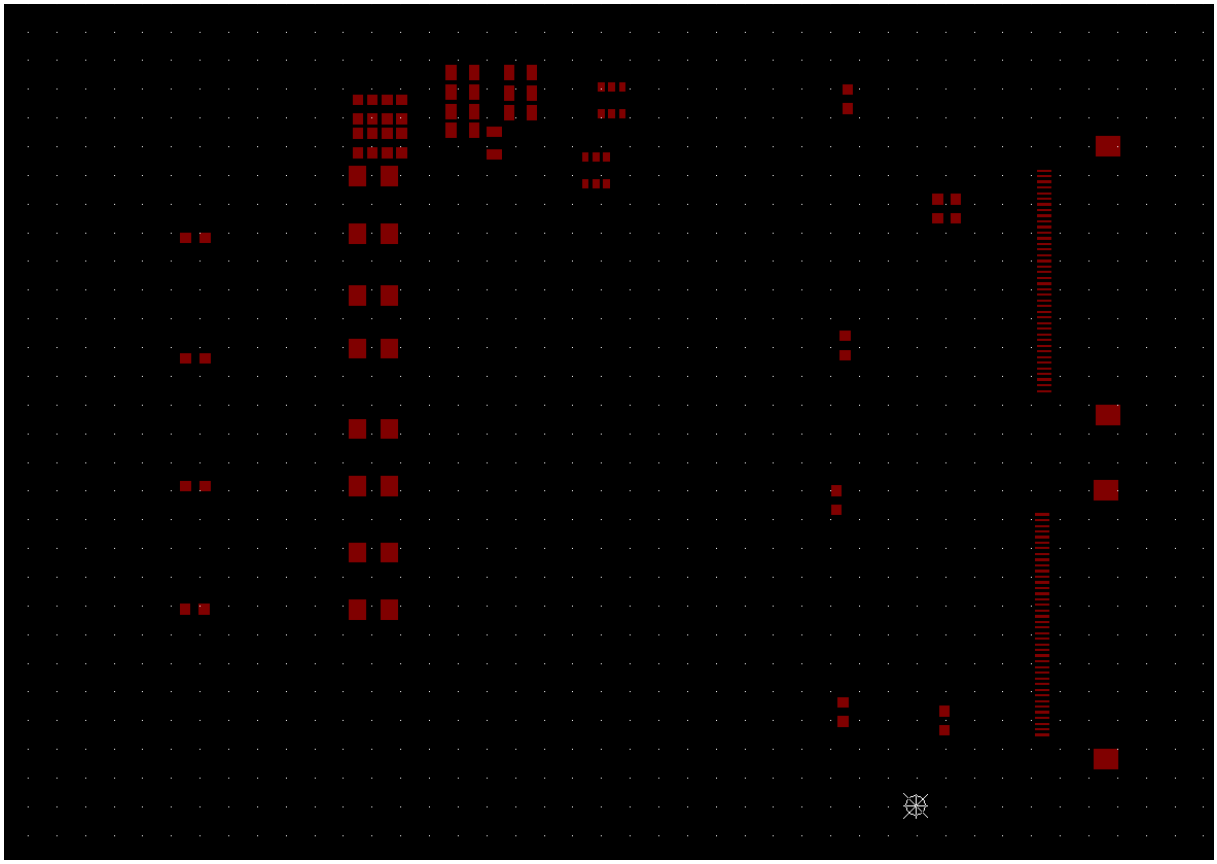


Figura 69 – Capa de pads inferior

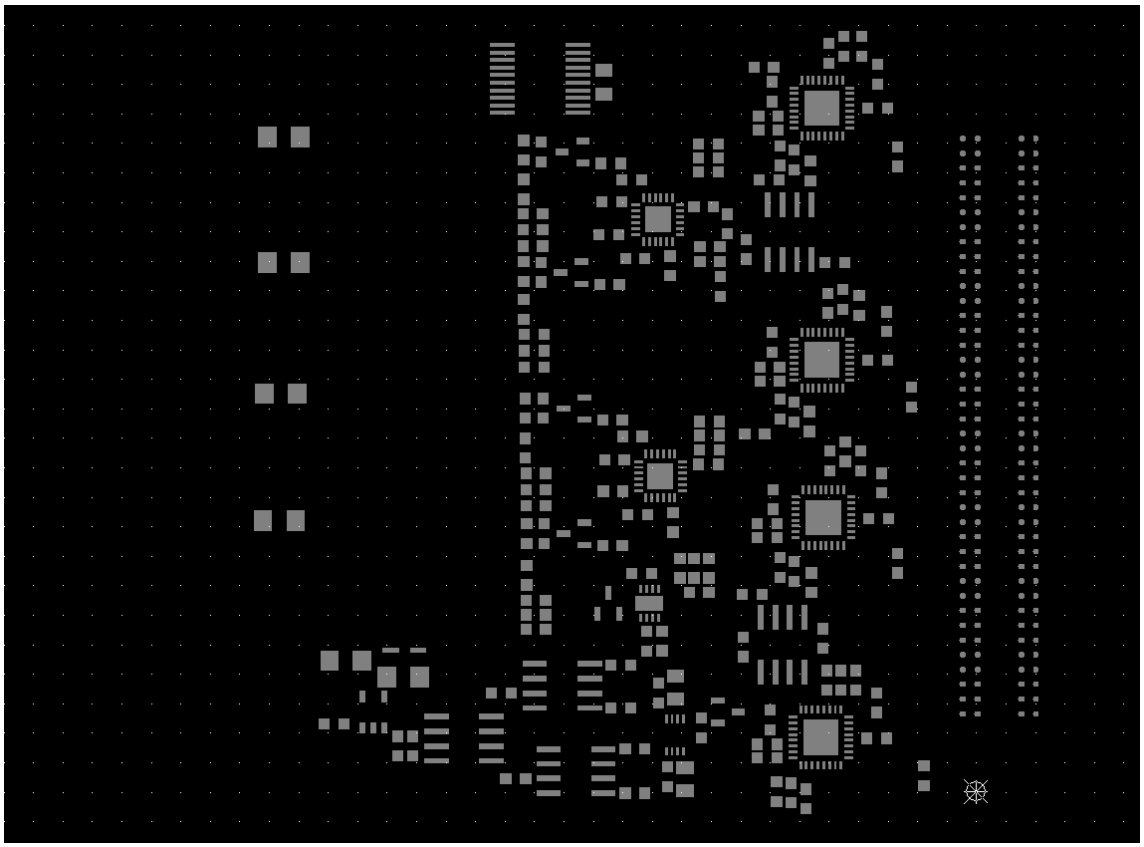


Figura 70 – Capa de pads superior

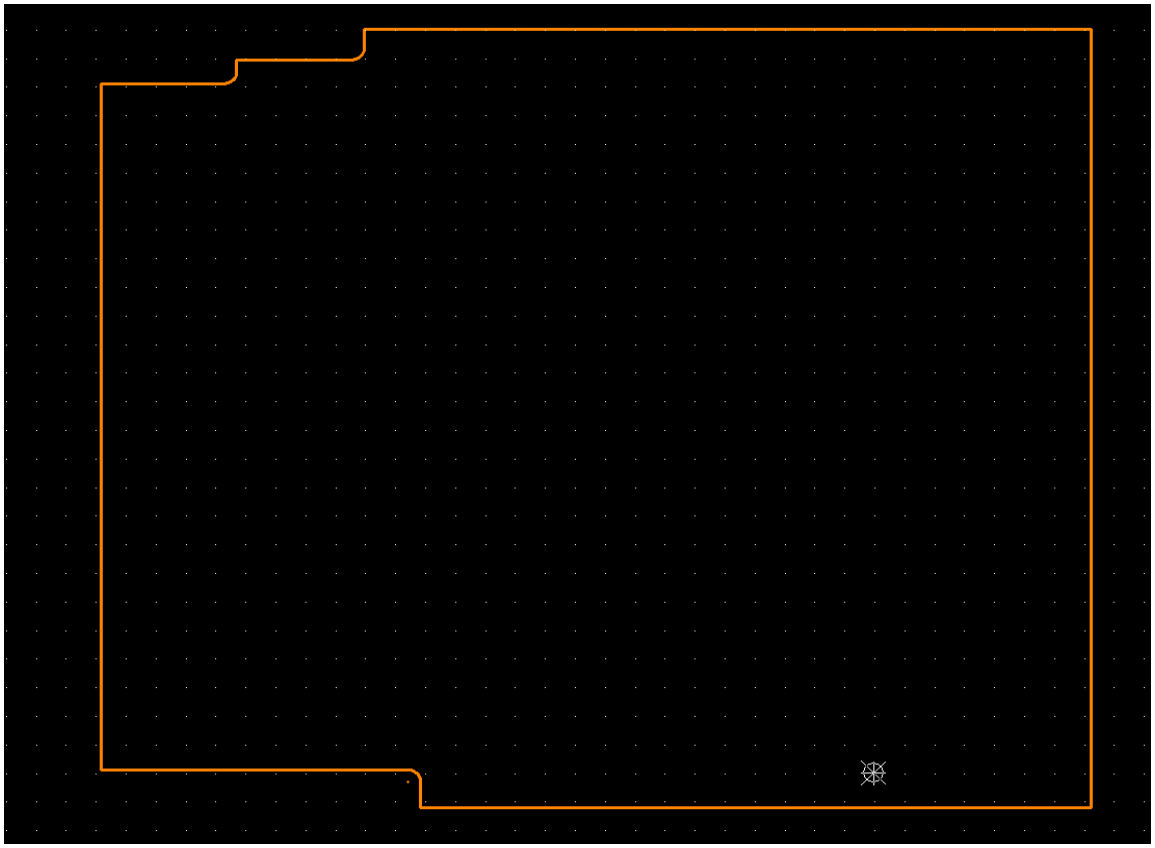


Figura 71 – Capa de contorno

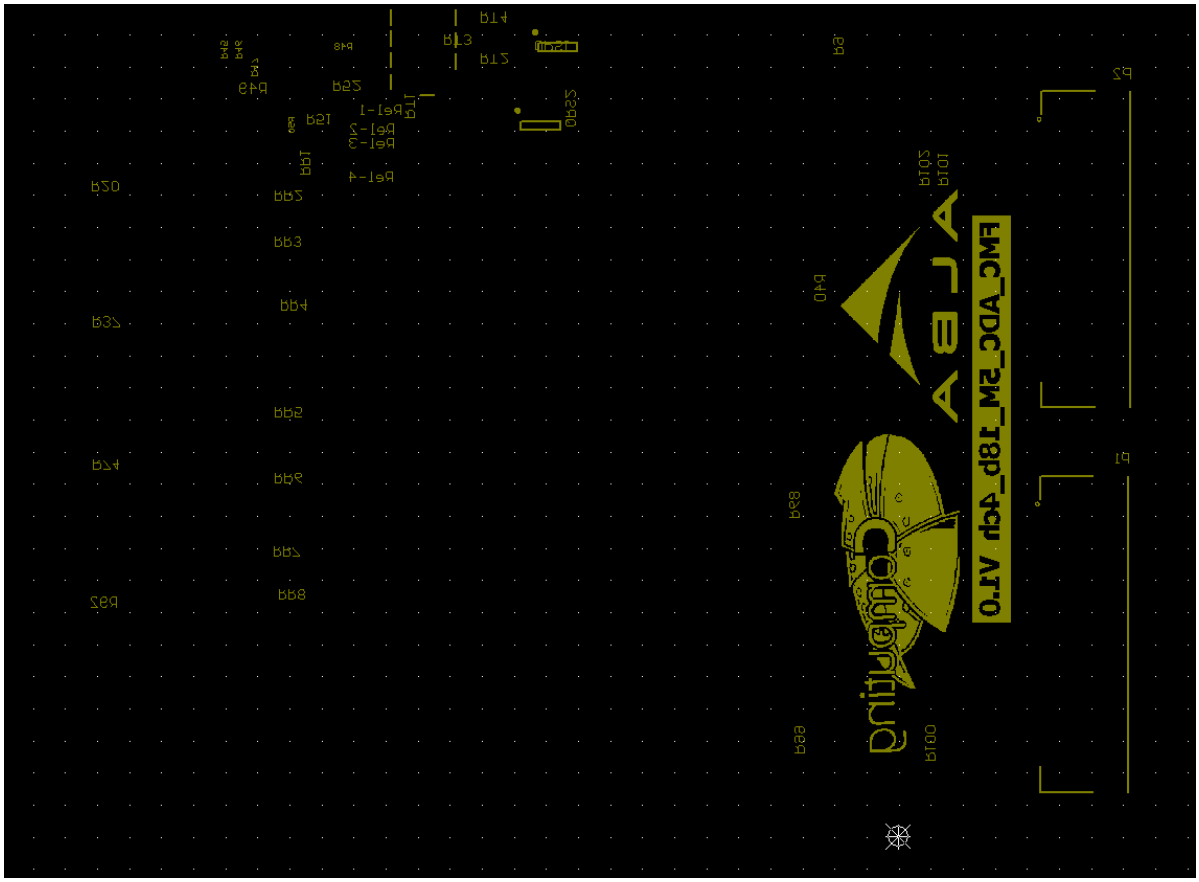


Figura 72 – Capa de serigrafía inferior

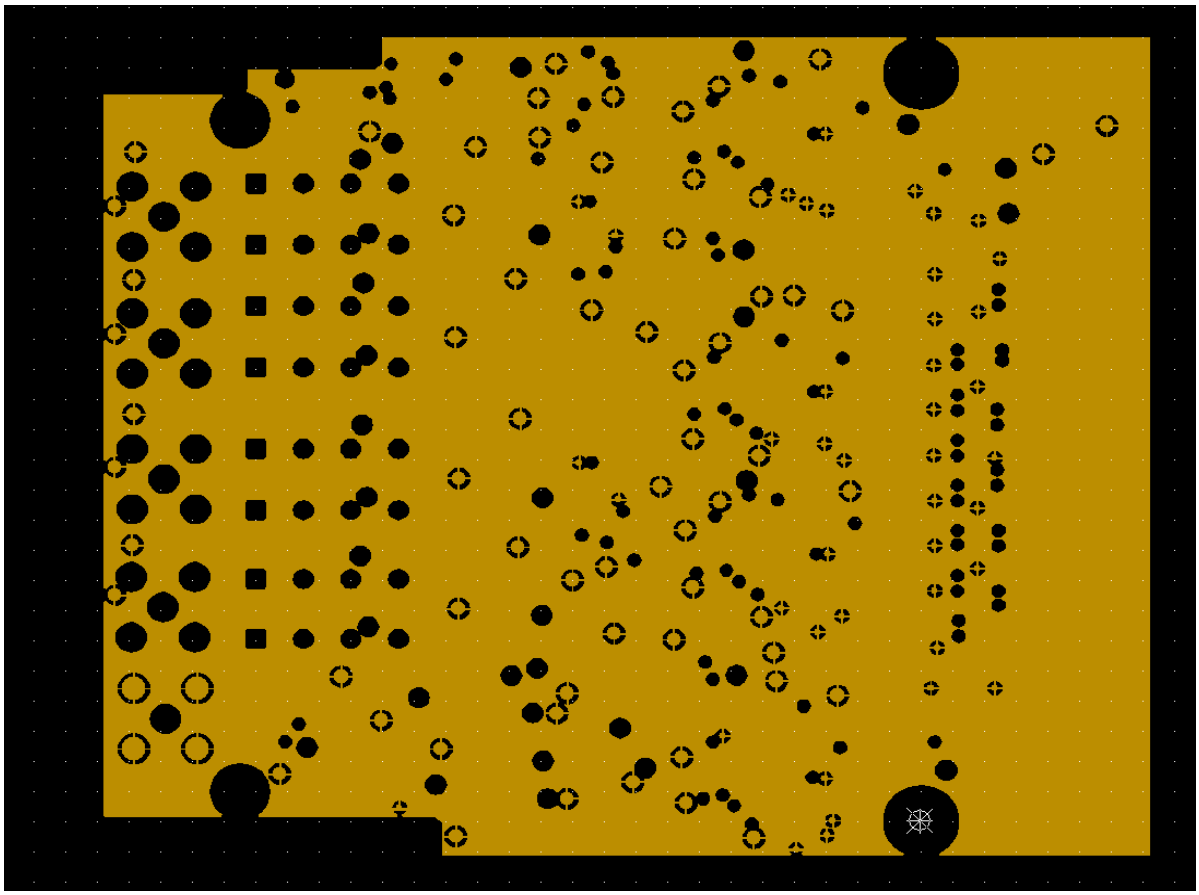


Figura 73 – Capa con plano de tierra

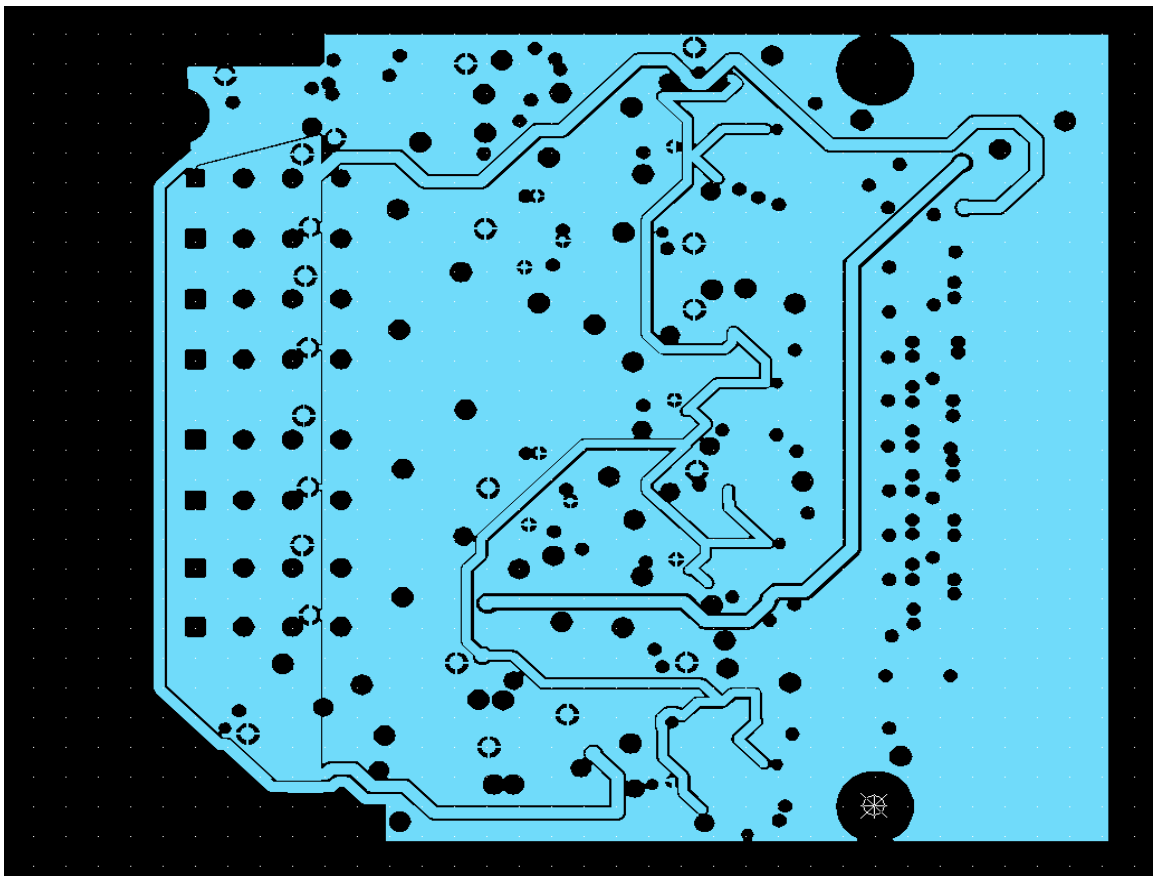


Figura 74 – Capa con planos de alimentación

10. VHDL

10.1 STATEMACHINE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity statemachine is
    port( clk, DATA_REQ, D, nrst : in std_logic;
            BUSY, CNV, CLK_ADC : out std_logic;
            fsm_state   : out std_logic_vector(1 downto 0);
            DATA : OUT STD_LOGIC_VECTOR(17 DOWNTO 0)
            );
end statemachine;

--s0 IDLE
--s1 CNV
--s2 ACQ
--s3 DATA

architecture comp of statemachine is

    type states is (s0 ,s1 ,s2 ,s3);
    signal automat: states;

    signal Daux : std_logic_vector (17 downto 0);
    signal counter  : std_logic_vector(4 downto 0); --Contador de 5
bits
    signal counter2 : std_logic_vector(4 downto 0); --Contador de 5
bits

begin
    process (clk)
    begin
      if clk'event and clk = '1' then
            case automat is
                when s0 =>
                    fsm_state <= "01";
                    if DATA_REQ = '1' then automat <= s1;
                    else automat <= s0;
                    end if;
                when s1 =>
                    fsm_state <= "00";
                    automat <= s2; --DATA_REQ no es condición
necesaria para que la máquina siga funcionando, sólo es el "disparo"
para salir de IDLE --Ya que se pasa directamente al siguiente estado,
en este estado (s1-CNV) sólo se está un ciclo de clk (T_CLK=10ns)
                    counter <= "00000"; --Se prepara el contador
                when s2 =>
```

```
                    fsm_state <= "10";
                    if counter = "11000" then
                        automat <= s3; --"XXXXX" debe ser el número de
ciclos de reloj necesarios de espera entre que se baja el CNV y el D17
está disponible. Es decir "XXXXX"= (t_MSB - t_CNVH)/(T_CLK) - 1; El -1
es porque se inicializa el contador a 0 desde el anterior estado
                        counter2 <= "00000"; --Se prepara el segundo
contador
                    else
                        automat <= s2;
                        counter <= counter + "00001";
                    end if;
                when s3 =>
                    fsm_state <= "11";
                    if counter2 = "10001" then automat <= s0; --Si el
contador llega a 17, es que se han captado 18 datos.
                    else automat <= s3; counter2 <= counter2 +
"00001";
                    end if;
            end case;
        end if;
    end process;

    process (clk,automat)
    begin
      if clk'event and clk = '1' and automat = s3 then
            Daux <= Daux(16 downto 0) & D;
      end if;
    end process;

    DATA <= Daux;
    CNV <= '1' when automat = s1 else '0';
    CLK_ADC <= clk when automat = s3 else '1';
    BUSY <= '1' when automat /= s0 else '0';
end comp;
```

10.1.1 TEST BENCH STATEMACHINE

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY TB_statemachine IS
END TB_statemachine;

ARCHITECTURE behavior OF TB_statemachine IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT statemachine
    PORT(
          clk, DATA_REQ, D, nrst : in std_logic;
            BUSY, CNV, CLK_ADC : out std_logic;
            DATA : OUT STD_LOGIC_VECTOR(17 DOWNTO 0)
            );
    END COMPONENT;
```

```
      --Inputs
   signal clk : std_logic := '0';
    signal data_req : std_logic := '0';
    signal d : std_logic := '0';
    signal nrst : std_logic := '0';

    --Outputs
   signal busy : std_logic := '0';
    signal cnv : std_logic := '0';
    signal clk_adc : std_logic := '0';
    signal data : std_logic_vector(17 downto 0);

   -- Clock period definitions
   constant clk_period : time := 10 ns;

BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: statemachine PORT MAP (
          clk => clk,
            DATA_REQ => DATA_REQ,
            D => D,
            nrst => nrst,
            BUSY => BUSY,
            CNV => CNV,
            CLK_ADC => CLK_ADC,
            DATA => DATA
            );


   -- Clock process definitions
   clk_process : process
   begin
        clk <= '0';
        wait for clk_period/2;
        clk <= '1';
        wait for clk_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
        wait for 20 ns;
        DATA_REQ <= '1';
        wait for 200 ns;
        wait for clk_period*3;
        DATA_REQ <= '0';
      wait for 100 ns;
      wait for clk_period*10;
        -- insert stimulus here

      wait;
   end process;

END;
```

10.2 DIV

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity clk_div is
port( in_clk, nrst : in std_logic;
                div :  in std_logic_vector(31 downto 0);
            out_clk : out std_logic
        );
end clk_div;

architecture Behavioral of clk_div is
signal aux: STD_LOGIC;
    signal counter: STD_LOGIC_vector(31 downto 0);
begin
     process (nrst, in_clk) begin
        if (nrst = '0') then
            aux <= '1';
            counter <= "00000000000000000000000000000000";
          elsif in_clk'event and in_clk = '0' then
            if (counter >= div) then
                aux <= NOT(aux);
                counter <= "00000000000000000000000000000000";
            else
                counter <= counter +
"00000000000000000000000000000001";
            end if;
        end if;
    end process;
     out_clk <= aux;
end Behavioral;
```

10.3 TOP

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
-- use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
-- library UNISIM;
-- use UNISIM.VComponents.all;

entity top is
```

```
    PORT(
        clk : IN std_logic;
        DATA_REQ : IN std_logic;
        D : IN std_logic;
        nrst : IN std_logic;
        BUSY : OUT std_logic;
        CNV : OUT std_logic;
        DIV : IN std_logic_vector(31 downto 0);
        CLK_ADC : OUT std_logic;
        fsm_state    : out std_logic_vector(1 downto 0);
        DATA : OUT std_logic_vector(17 downto 0)
        );
end top;

architecture Behavioral of top is
signal clk_d : std_logic;
signal clk_d_zerocross : std_logic;
signal data_req_r : std_logic;

component statemachine is
      port( clk, DATA_REQ, D, nrst : in std_logic;
              BUSY, CNV, CLK_ADC : out std_logic;
              fsm_state : out std_logic_vector(1 downto 0);
              DATA : OUT STD_LOGIC_VECTOR(17 DOWNTO 0)
            );
    end component;

component clk_div is
        port( in_clk, nrst : in std_logic;
                div :  in std_logic_vector(31 downto 0);
                out_clk : out std_logic
           );
end component;

begin

CLK_DIVISOR : clk_div
                port map( in_clk=>clk,
                          nrst=>nrst,
                          div => div,
                          out_clk=>clk_d
                           );

STATE_MACHINE : statemachine
                port map( clk=>clk_d,
                          DATA_REQ=>data_req_r, --DATA_REQ, --
                          D=>D,
                          nrst=>nrst,
                          BUSY=>BUSY,
                          CNV=> CNV,
                          CLK_ADC=>CLK_ADC,
                          fsm_state => fsm_state,
                          DATA=>DATA
                          );


--PROCESO para registrar DATA_REQ por flanco de subida de clk y
mantener data_reg_r hasta un flanco de subida de clk_d

process (clk) begin
    if (rising_edge(clk)) then
```

```
        if DATA_REQ = '1' then
            data_req_r <= '1';
            clk_d_zerocross <= '0';
        else
            if (clk_d = '1') and (clk_d_zerocross = '1') then
                data_req_r <= '0';
            end if;
            if (clk_d = '0') then
                clk_d_zerocross <= '1';
            end if;
        end if;
    end if;
end process;


end Behavioral;
```

11. WISHBONE GENERATOR

```
peripheral {
  name = "WB-FMC-ADC-Core5M  ";
  vendor_id = "000000000000a1ba";
  device_id = "00000013";
  version   = "00000002";
  date      = "20180525";
  description = "Wishbone slave for FMC ADC 18bits 5MSPs core";
  hdl_entity = "fmc_adc_18b_5MSPS_csr";

  prefix = "fmc_adc_core";

  reg {
    name = "Control register";
    prefix = "ctl";

    field {
      name = "Trigger ID";
      description = "Is the trigger ID that starts the adquisition.";
      prefix = "TRIGID";
      type = SLV;
      size = 8;
      align = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "Reset ADC";
      description = "When this bit is set to 1 is done a Reset ADC and
flags";
      prefix = "RST";
      type = MONOSTABLE;
      size = 1;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "TRIG";
      description = "Manual trigger";
      prefix = "TRG";
      type = MONOSTABLE;
      size = 1;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "Busy";
      description = "When this bit is set to 0 the ADC is done";
      prefix = "adq_m";
      type = BIT;
      size = 1;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };
```

```
    field {
      name = "State machine status";
      description = "States:\n0: IDLE\n1CNV\n2ACQ\n3DATA\n";
      prefix = "adc_fsm";
      type = SLV;
      size = 2;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

    reg {
    name = "CLK DIVISOR";
    prefix = "div";

    field {
      name = "CLK_DIVISOR";
      description = "Clock divisor to controll the speed";
      type = SLV;
      size = 32;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
      };
     };

    reg {
    name = "ADC Chanel 1";
    prefix = "adc_ch1";

    field {
      name = "ADC Chanel1";
      description = "Reading of channel 1 of 18bit ADC AD7960";
      type = SLV;
      size = 32;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

    reg {
    name = "ADC Chanel 2";
    prefix = "adc_ch2";

    field {
      name = "ADC Chanel2";
      description = "Reading of channel 2 of 18bit ADC AD7960";
      type = SLV;
      size = 32;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

    reg {
    name = "ADC Chanel 3";
    prefix = "adc_ch3";

    field {
      name = "ADC Chanel3";
      description = "Reading of channel 3 of 18bit ADC AD7960";
      type = SLV;
```

```
      size = 32;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

    reg {
    name = "ADC Chanel 4";
    prefix = "adc_ch4";

    field {
      name = "ADC Chanel4";
      description = "Reading of channel 4 of 18bit ADC AD7960";
      type = SLV;
      size = 32;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };


  reg {
    name = "Samples counter";
    prefix = "samples_cnt";

    field {
      name = "Samples counter";
      description = "Counts the number of samples.\n It is reset on
START and then counts the number of pre-trigger + post-trigger
samples";
      type = SLV;
      size = 32;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

  reg {
    name = "ID assignation";
    prefix = "ID";

    field {
      name = "Cha1 ID";
      description = "ID used to send the data adquiered on Channel 1.
If it is 0 no data is generated.";
      prefix = "CH1_ID";
      type = SLV;
      size = 8;
      align = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "Cha2 ID";
      description = "ID used to send the data adquiered on Channel 2.
If it is 0 no data is generated.";
      prefix = "CH2_ID";
      type = SLV;
      size = 8;
      align = 8;
```

```
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "Cha3 ID";
      description = "ID used to send the data adquiered on Channel 3.
If it is 0 no data is generated.";
      prefix = "CH3_ID";
      type = SLV;
      size = 8;
      align = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };

    field {
      name = "Cha4 ID";
      description = "ID used to send the data adquiered on Channel 4.
If it is 0 no data is generated.";
      prefix = "CH4_ID";
      type = SLV;
      size = 8;
      align = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };
  };

};
```

12. SCRIPT PYTHON

```
from alin.base import AlinDevice
import time


# To Upload a device use ALinDevice with the following parameters:
#          device= device's name
#          number= device's number in case there is more than 1. By
default 0.
ADC_modul = AlinDevice(device='WB-FMC-ADC-CORE5M', number=0)



#stop any other proccess that uses DIGITAL_IO module
#Setup here, using web or SCPI commands, DIODIFF1 and DIODIFF2 as
input and DIO1 as output.

acq = [[ADC_modul, 'CTL_TRG', 1,   'Acquiring', 'ADC_CH1'],


         ]



def ejecutarTabla(table):
    for item in table:
        func = item[0]
```

```
        reg = item[1]
        value = item[2]
        comment = item[3]
        reg2 = item[4]

        f=open("medidas_pila.txt","a")
        print comment
        i=0
        while i<2000: # Controla cuanto tiempo dura la adquisición
          func.writeAttribute(reg,value)
          f.write(hex(func.readAttribute(reg2))+"\n")
          print reg,'\t\t', hex(func.readAttribute(reg2))
          time.sleep(0.1) # Controla cada cuanto se hace un trigger
          i=i+1
          print i
        f.close()

print "Linear Test"

opciones = {

    1: acq,

}

while True:

    print "choose option (1) ",
    o = int(input())
    ejecutarTabla(opciones[o])
```