Universidad de Alcalá
Escuela Politécnica Superior

GRADO EN INGENIERÍA ELECTRÓNICA DE COMUNICACIONES

Trabajo Fin de Grado

Diseño e implementación de un BMS de baterías Li-ion modular

Autor: Sergio Crespo Fernández-Hijicos

Tutor: Pedro Alfonso Revenga de Toro

2017

UNIVERSIDAD DE ALCALÁ
ESCUELA POLITÉCNICA SUPERIOR

Grado En Ingeniería Electrónica De Comunicaciones

Trabajo Fin de Grado

Diseño e implementación de un BMS de baterías Li-ion modular

Autor: Sergio Crespo Fernández-Hijicos

Tutor: Pedro Alfonso Revenga de Toro

TRIBUNAL:

Presidente: Francisco Javier Rodríguez Sánchez

Vocal 1º: Miguel Ángel García Garrido

Vocal 2º: Pedro Alfonso Revenga de Toro

Calificación: .....

Fecha: .....

A mis padres, porque sin ellos no habría sido posible y por entender que, aunque mis estudios han sido largos y difíciles, da sentido a mi vida terminar lo que empecé y tanto me gusta que es la ingeniería electrónica.

"Huyo de la vida regalada, de la ambición y la hipocresía, y busco para mi propia gloria la senda más angosta y difícil. ¿Es eso de tonto y mentecato?"

Don Quijote de La Mancha. Miguel de Cervantes

I. RESUMEN

El presente proyecto, "*Diseño e implementación de un BMS de baterías Li-ion modular*", consiste en diseñar un circuito de protección y ecualización de carga para baterías de litio, a esto llamamos BMS, pero el trabajo va más allá, puesto que en la PCB que se diseña, hay un componente que se encarga de hacer las medidas necesarias que posteriormente se enviarán por SPI a un microcontrolador, desde el cual además de enviar y recibir comandos, se podrán visualizar los datos adquiridos por un display RGB de 16 bits.

PALABRAS CLAVE

BMS, baterías, PCB, IsoSPI, Microcontrolador (uC).

II. ABSTRACT

The present project, *"Design and implementation of a modular Li-ion battery BMS"*, consists of designing a load protection and equalization circuit for lithium batteries, this is called BMS, but the work goes further, since in The PCB that is designed, there is a component that is responsible for making the necessary measurements that will be sent later on SPI to a microcontroller, from which in addition to sending and receiving commands, the data acquired will be displayed on a 16-bit RGB display.

KEYWORDS

BMS, batteries, PCB, IsoSPI, Microcontroller (uC).

III. RESUMEN EXTENDIDO

Dada la importancia del almacenamiento de energía en cualquier proyecto tecnológico, cada vez se tienen baterías recargables más ligeras con mayor capacidad, hoy en día las baterías comerciales más potentes en ese aspecto son las de ion de litio, en concreto las que se van a utilizar en este proyecto son diez celdas de "*GP Batteries"* modelo "*GP45EVLF"*, las cuales tienen unas características de tensión nominal de 3.2 V y capacidad 45 Ah.

Estas baterías necesitan un circuito que las proteja de sobretensión y que mida la temperatura en cada celda, que sea capaz de cortar la corriente por las mismas durante el proceso de carga, en caso de que se haya cargado o que haya aumentado excesivamente la temperatura por alguna razón. Este circuito es un *BMS* (*Battery Management System*), que se va hacer sobre una *PCB* (*Printed Circuit Board*), para ello, lo primero que hay que hacer es diseñar el esquemático del circuito pensando en la función que va a desempeñar, cómo se va a medir la tensión, la temperatura, cómo "desviar" la corriente por otro circuito cuando no queramos cargar una celda determinada en caso de que ya esté cargada o se haya calentado mucho.

Cuando se tenga el circuito funcional, se procederá a diseñar la PCB, para ello se necesita saber qué componentes individuales se van a utilizar, dado que se necesita conocer las dimensiones de los mismos para asignarle a cada uno la "huella" que le corresponda, la corriente que debe pasar por cada pista para darle mayor o menor grosor, la disposición sobre la placa porque lo que se pretende es atornillar la placa sobre el conjunto de baterías, además de un LED que indique cual celda está cargada y cual no.

El siguiente paso es, establecer una comunicación entre la placa o el componente *LTC6804* de "*Linear Tecnologies*" y un microcontrolador que se encargue de mostrar las medidas al usuario. El protocolo de comunicación que se va a utilizar es IsoSPI, una variante de SPI desarrollada por "*Linear Tecnologies*", consiste en el mismo protocolo SPI entre un maestro y un esclavo, pero se añade entre ellos un aislamiento galvánico extremo a extremo con transformadores entre un elemento y otro. Las funciones principales del uC (Micro Controlador) son comportarse como GUI (*Graphical User Interface*) del BMS, es decir, mostrar medidas, enviar y recibir comandos seleccionados por el usuario al BMS, etc.

IV. ÍNDICE GENERAL

V. ÍNDICE DE FIGURAS

VI. GLOSARIO DE ACRÓNIMOS Y ABREVIATURAS

BMS	Battery Management System	Sistema de gestión de batería
PCB	Printed Circuit Board	Tarjeta de circuito impreso
SPI	Serial Peripherical Interface	Interfaz serie periférico
IsoSPI	Isolated SPI	SPI con aislamiento
RGB [1]	Red Green Blue	Rojo Verde Azul
uC [2]	Micro Controller	Micro Controlador
ADC	Analogic-Digital Converter	Conversor Analógico-Digital
CRC	Cyclic Redundancy Check	Comprobación de redundancia cíclica
CAD	Computer-Aided Design	Diseño asistido por computadora
EDA	Electronics Design Automation	Automatización de diseño electrónico
IDE	Integrated Development Environment	Entorno de desarrollo integrado
NTC	Negative Temperature Coefficient	Coeficiente de temperatura negativo
TFG		Trabajo de Fin de Grado
EMI	ElectroMagnetic Interference	Interferencia electromagnética
IC	Integrated Circuit	Circuito integrado
MOSFET [3]	Metal-Oxide-Semiconductor / Field-Effect Transistor	Metal-óxido-semiconductor / transistor de efecto de campo
PMOS	P-channel MOS	MOS canal P

VTCMP	Voltage Threshold Comparator	Comparador de tensión limite
LED	Light-Emitting Diode	Diodo emisor de luz
GUI	Graphical User Interface	Interfaz gráfica de usuario
MOSI	Master Output–Slave Input	Sentido maestro-esclavo
MISO	Master Input–Slave Output	Sentido esclavo-maestro
PEC	Packet Error Code	Código de error de paquete

[1] RGB: Es una forma de modelar un color mediante la mezcla por adición de los tres colores primarios que le dan nombre (rojo, verde y azul).

[2] uC: La u hace referencia a la letra griega mu (μ), representa el prefijo micro.

[3] MOSFET: es un acrónimo compuesto a su vez de dos acrónimos en uno MOS y FET.

VII. MEMORIA

1. INTRODUCCIÓN

El trabajo debe plantearse diferenciando entre la parte *hardware* y *software*, que está muy ligado al planteamiento entre diseño e implementación.

Entendiendo por diseño, el uso de una herramienta *CAD* (*Computer-Aided Design*). En este trabajo se decidió utilizar KiCAD *EDA* (*Electronics Design Automation*), una herramienta *Open Source,* que permite el diseño de esquemáticos de circuitos electrónicos, circuitos impresos y visualización en 3D, huellas de componentes y generar archivos Gerber. Aunque permite diseñar huellas, la mayoría de componentes tienen formas y tamaños estandarizados, por tanto, KiCAD ya tiene esas huellas en librerías que no se necesitan instalar aparte, sino que ya vienen con la instalación base de la herramienta.

La otra herramienta *software* empleada es el IDE (*Integrated Development Environment*) para programar el uC, en este proyecto se decidió utilizar Keil uVision4 IDE de ARM ya que es el *software* más adecuado para programar un micro de ARM, puesto que contiene las librerías propias de dicho micro, además se ha empleado en asignaturas previas al TFG para realizar trabajos similares y por tanto, su manejo es conocido. esta herramienta combina gestión de proyectos, entorno de tiempo de ejecución, compilador, editor de código y modo de depuración (debug mode) dentro del cual se pueden usar los clásicos *breakpoints*, *watch windows* y control de ejecución, además, proporciona visibilidad completa a los periféricos del dispositivo (Timers, ADCs, etc.).

Por otro lado, tenemos el *hardware*, aparte de las baterías, están los componentes que hay que soldar a la placa que se ha diseñado como BMS, cuyo componente principal es el LTC6804 de *Linear*, el resto de componentes se detallarán en la parte de desarrollo del proyecto, también se utilizan sensores de temperatura NTC que, aunque no sean lineales, son pequeños y baratos. El uC, que se utilizará es el ARM Cortex-M3 LPC1768 de NXP, integrado en la tarjeta de desarrollo Mini-DK2, el motivo de usar esta tarjeta es el mismo que en el caso del Keil, es una tarjeta utilizada en asignaturas anteriores al TFG. Aunque podría valer cualquiera, por ejemplo, la Arduino UNO hubiera sido otra posibilidad, dado que es relativamente barata y muy universal en el sentido de que hay mucha documentación, código libre creado por usuarios, etc. aparte de un IDE propio de Arduino. La etapa intermedia entre el uC y el BMS, que traduce el protocolo serie SPI

a IsoSPI, compuesta fundamentalmente por el circuito integrado LTC6820, que es el encargado para dicha tarea, un conector RJ45 con transformadores internos que proporcionan aislamiento galvánico entre un terminal y otro, como ya se detallará mejor cuando se hable del protocolo IsoSPI, un cable Ethernet CAT 5 que comunica el BMS con esta etapa y una serie de resistencias y condensadores de polarización.

1.1 OBJETIVOS

Teniendo en cuenta que el trabajo parte de cero, es decir, que se ha diseñado exclusivamente para el propósito del proyecto, un circuito que encargado de hacer las funciones del BMS que son medidas, protección y ecualización de carga de baterías de litio, se ha diseñado una PCB para dicho propósito y se ha programado un microprocesador encargado de actuar como maestro en la comunicación con el BMS, dicha comunicación será de hasta 1Mbps y estará aislada mediante transformadores.

Sin duda es un trabajo ambicioso en el aspecto de los objetivos, puesto que trata de cómo aprender a realizar un proyecto de ingeniería muy completo cuyos objetivos principales forman parte del mismo objetivo que es diseñar un circuito BMS que funcione, es decir, que mida tensión y temperatura de las celdas y que sea capaz de dejar de cargar celdas individuales en caso de que su carga esté completa, pero para llegar a este objetivo global, se deben marcar otros objetivos previos o metas, éstos son:

1. Lo primero de todo, diseñar el propio circuito a modo de esquemático.

2. Elegir adecuadamente los componentes para implementar en el circuito, esto es importante porque los componentes irán soldados sobre una PCB y se deben conocer las dimensiones con exactitud.

3. Diseñar la PCB y encargar su fabricación, teniendo en cuenta lo anterior y prestando especial atención a las dimensiones de las baterías, puesto que la placa irá atornillada a ellas y atendiendo también al ancho de pista para asegurar que el flujo de corriente sea suficiente.

4. Programar un microcontrolador en modo maestro, para realizar la comunicación SPI con el BMS y diseñar un GUI (*Graphical User Interface*) por el que se controlen los comandos transferidos entre el BMS y el micro y mostrar al usuario los datos adquiridos.

5. Implementar un circuito secundario que realice la función de traducción entre protocolos SPI e IsoSPI.

2. BASE TEÓRICA

Lo primero que se debe tener en cuenta para este trabajo es que tenemos diez celdas de baterías de litio en serie y que se necesita un circuito, la parte de los detalles y la implementación es una cuestión que de momento no nos ocupa.

Entonces, lo primero que se debe pensar es que se necesita algo que funcione como interruptor, que permita cargar o no, cada celda individualmente según convenga, esto puede ser, por ejemplo, un transistor. Pero cómo decide un transistor cuándo está cargada cada celda, para esta tarea se necesita algún dispositivo con inteligencia suficiente para hacer tal función, esto podría ser un uC, puesto que se pueden programar y tienen un ADC para poder medir la tensión en una celda, uno de los problemas es que tenemos diez en serie y cada ADC está referenciado a masa, por tanto, no sirve para medir la tensión más alta, que puede estar a 37 voltios, pero como existen dispositivos dedicados a este tipo de tareas, lo que se debe hacer es pensar en uno que cumpla con las necesidades adecuadas, en el apartado siguiente se hablará del componente en cuestión.

Una vez solucionado el problema de adquisición de la tensión y del circuito que funcione como interruptor de carga de las baterías, se deben pensar el resto de problemas que plantean diseñar un circuito autónomo, como es la alimentación, en este caso, la alimentación se obtendrá de las propias baterías, ya que el componente que se ha decidido utilizar para la adquisición de tensión así lo permite, esto facilita que no se necesiten fuentes de alimentación externas.

Por otro lado, en cuanto a la parte de electrónica digital, se necesita un protocolo de comunicaciones serie, en este caso se ha optado por SPI e IsoSPI, que se detallarán más adelante.

2.1 LTC6804-2

Para este proyecto se utilizará el LTC6804-2, de *Linear Tecnologies* (LT), puesto que tiene una serie de características muy favorables al propósito del proyecto, por tanto, será el componente principal, en torno al cual se diseñará el circuito y la PCB, estas características son:

- Capacidad de medir hasta doce celdas en serie.

- Una arquitectura apilable que soporta cientos de celdas, puesto que se pueden utilizar múltiples dispositivos conectados en paralelo al procesador central, con cada dispositivo direccionado individualmente.
- Interfaz IsoSPI incorporada:
    - 1Mbps de comunicación serie aislada.
    - Se puede utilizar con un solo par trenzado, hasta 100 metros.
    - Baja susceptibilidad y emisiones de EMI (*ElectroMagnetic Interference*).
- Error de medida total máximo de 1,2 mV.
- Tiempo máximo de medida de todas las celdas de 290 µs en un sistema.
- Medida sincronizada de tensión y corriente.
- ADC Delta-Sigma de 16 bits con filtro de ruido de tercer orden de frecuencia programable.
- Diseñado para sistemas compatibles con ISO26262 (normativa de seguridad en un sistema eléctrico en un vehículo motor).
- Balanceo de celdas de forma pasiva con timer programable.
- 5 GPIO digitales o entradas analógicas para medir temperatura u otros sensores, además de la posibilidad de configurar una comunicación serie por I2C o como SPI Master.
- Corriente de alimentación de 4 µA en modo *Sleep*.
- Encapsulado en SSOP de 48 pines (0,5 mm de pitch).

Dicho esto, diseñaremos un circuito en base a este IC (*Integrated Circuit*). En las hojas de características del componente se describe detalladamente los componentes que debemos emplear para las distintas configuraciones. En este trabajo se configurará para funcionar en modo IsoSPI como ya se ha comentado.

2.1.1 LTC6804: DIAGRAMA DE ESTADOS

En relación a la alimentación y demás, es necesario saber cómo funciona este dispositivo, que se divide en dos partes, el funcionamiento del núcleo del circuito y el funcionamiento del circuito IsoSPI (encargado de la comunicación), que se describirá después. El funcionamiento del núcleo se describe de la siguiente manera:

El circuito puede estar en cuatro estados, "*SLEEP State*", "*STANDBY State*", "*REFUP State*" y "*MEASURE State*".

- SLEEP State: La tensión de referencia y los ADCs están apagados, los temporizadores "*Watchdog timer*" y "*Software discharge timer*", han finalizado su cuenta, la corriente de alimentación es reducida al mínimo, el puerto "IsoSPI" está en estado "*IDLE State*". Si se recibe una señal "*WAKEUP*", el circuito entra en el estado STANDBY State. Es decir, todo el circuito está en modo bajo consumo, a la espera de recibir una señal que lo active.

- STANDBY State: La tensión de referencia y los ADC están desactivados a la espera de que llegue un comando de adquisición de los ADCs o el bit REFON del registro de configuración esté a nivel alto, estos eventos harán que el circuito pase al estado MEASURE State o REFUP State respectivamente. En este estado, los timers están haciendo su cuenta, en caso de que termine sin ningún comando recibido, se vuelve al estado SLEEP. El pin DRIVE se pone a nivel alto (5,6 V), el cual se utiliza como señal de *enable* que active un convertidor reductor de tensión DC-DC de 5 V de salida, para proporcionar alimentación al circuito por el pin VREG. Se observa, que la diferencia de tensión entre los pines DRIVE y VREG es de 0,6 V, esto es porque una configuración de alimentación típica del dispositivo se hace con un transistor bipolar, poniendo el colector, a la tensión más alta de las baterías (V+), la base al pin DRIVE y el EMISOR al pin VREG.

- REFUP State: Para estar en este estado, el bit REFON del registro de configuración debe estar a nivel alto. Los ADCs están apagados, pero la tensión de referencia está encendida, preparada para recibir un comando de inicio de conversión de los ADCs. Este estado no es necesario, pero es útil porque desde este estado la adquisición de los datos será más rápida que desde el estado STANDBY, puesto que la referencia de tensión ya está encendida y solo se necesita recibir el comando de adquisición para empezar.

- MEASURE State: Tanto la referencia de tensión como los ADCs están encendidos, puesto que es el estado en el que se realiza la adquisición de datos de los ADCs. Cuando la conversión se ha completado el circuito regresa al estado STANDBY o REFUP, dependiendo del bit REFON.

En la *Figura 1.* Diagrama de estados LTC6804 se muestra el diagrama de estados de lo anteriormente descrito y del puerto IsoSPI, el cual se describe simplemente diciendo que mientras se esté enviando o recibiendo datos, se encuentra en el estado ACTIVE, si no, en el estado READY y si no se envía o recibe nada durante 5,5 ms, el puerto IsoSPI entra en el estado IDLE, a la espera de una señal WAKEUP.

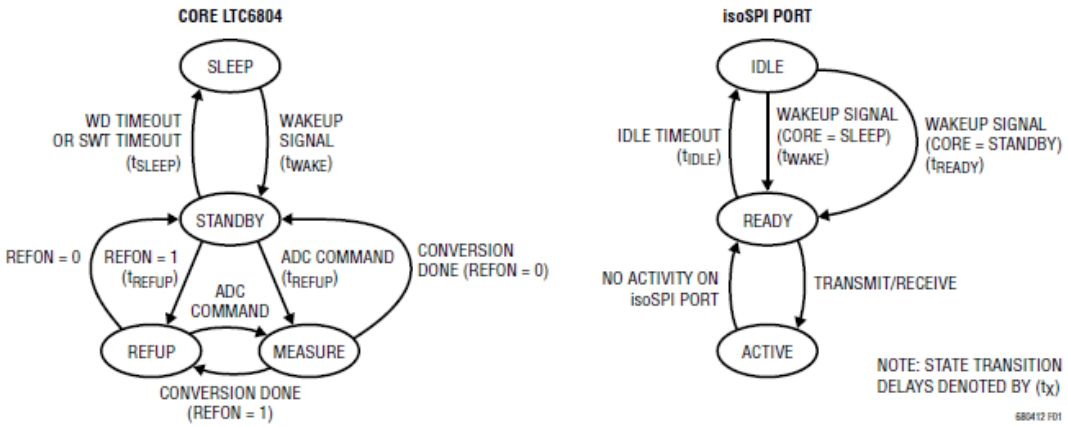


Figura 1. Diagrama de estados LTC6804

2.1.2 LTC6804: PUERTO ISOSPI

Continuando con el puerto IsoSPI y teniendo en cuenta el diagrama de estados, a continuación, se va a definir el funcionamiento del puerto y la forma general de los comandos para que se sepa de qué se habla durante el desarrollo del trabajo.

Cuando se recibe una señal WAKEUP, que no es más que una trama vacía de contenido, pero suficiente para estimular el puerto IsoSPI, el pin IBIAS se encenderá a 2 V, utilizándose como alimentación de un divisor resistivo externo para definir la intensidad de corriente que tendrán los pulsos diferenciales del protocolo IsoSPI, el punto intermedio del divisor resistivo se conecta al pin de entrada ICMP, esto es la tensión del comparador interno del dispositivo, encargado de definir un nivel bajo y un nivel alto.

En la *Figura 2*, se observa el esquema interno del puerto IsoSPI y las conexiones externas, de las de polarización (IBIAS) que hemos hablado y también los pines IPA e IMA que al hacer la diferencia o resta IPA – IMA proporcionan la propia señal isoSPI de la que se hablará más adelante.

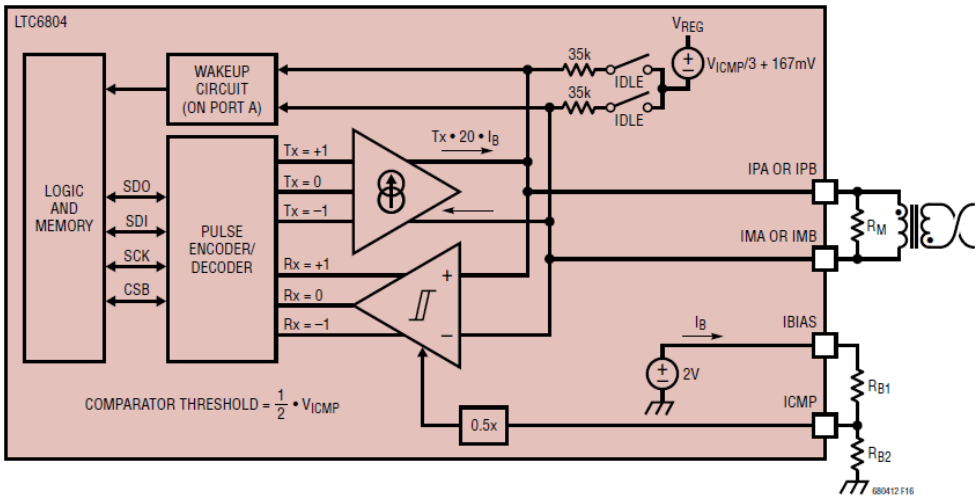


Figura 2. Esquema interno del puerto isoSPI

Donde R_{B1} y R_{B2} son el mencionado divisor resistivo y R_M es la resistencia de adaptación de impedancias del bus por el que fluirán las tramas del protocolo IsoSPI. Para calcular el valor de estas resistencias debe saberse que, la corriente que circulará por el bus es veinte veces la calculada por dividir los dos voltios del pin IBIAS entre la suma de resistencias, es decir:

$$I_{DRIVE} = 20 * I_{BIAS} \quad ; \quad I_{BIAS} = \frac{2V}{R_{B1}+R_{B2}}$$

Por otro lado, la tensión límite del comparador (TCMP) es la mitad de la tensión en el pin ICMP, es decir:

$$V_{ICMP} = I_{BIAS} * R_{B2} \quad ; \quad V_{TCMP} = 0{,}5 * V_{ICMP}$$

Por tanto, la amplitud en tensión de los pulsos diferenciales debe ser mayor que la tensión límite calculada (TCMP). Para definir esta amplitud de pulsos se debe tener en cuenta la resistencia R_M, de manera que la amplitud total es el resultado de multiplicar dicha resistencia por la corriente de circulación (DRIVE).

$$V_A = I_{DRIVE} * (R_M * 0{,}5)$$

Donde V_A es la amplitud positiva de la señal, resultando en tres niveles de tensión para definir el protocolo IsoSPI que son $+V_A$, 0 y $-V_A$, donde un nivel positivo resulta del suministro de corriente en IPA y de la ausencia de corriente en IMA a través de la resistencia de carga R_M, y

viceversa en un voltaje negativo, cuando ambas salidas están apagadas, la resistencia de carga fuerza la salida diferencial a 0V.

2.1.3 LTC6804: PROTOCOLO ISOSPI

Para entender el protocolo IsoSPI es necesario saber un par de cosas sobre SPI. El bus SPI es un protocolo estándar de comunicaciones serie con flujo de bits regulado de forma síncrona entre un maestro y un número indefinido de esclavos, la comunicación básica es entre un maestro y un esclavo, en la que el maestro mediante una señal de habilitación llamada *chip select* inicia comunicación con el esclavo e intercambian información con un tamaño de bits fijado por el maestro, el cual también habilita la señal de reloj y dos canales unidireccionales, uno en sentido maestro-esclavo (MOSI) y otro en sentido esclavo-maestro (MISO)

El protocolo IsoSPI es un protocolo propio de *Linear Technology* que lo único que pretende es poder utilizar SPI con aislamiento galvánico entre terminales, dado que, al existir dicho aislamiento, la componente continua de la señal que se envíe va a ser eliminada, por tanto, las señales deben ser diferenciales en las que el estado neutro sean los 0 voltios y pulsos de datos de amplitud bipolar (entre $+V_A$ Y $-V_A$), mientras que en SPI, la amplitud de los pulsos es unipolar (entre 0 y +VA). Esto hace que exista una mayor seguridad por estar aislados eléctricamente un terminal y otro, e inmunidad al ruido.

Por tanto, lo que se debe hacer es generar la comunicación SPI desde el uC y enviárselo a un dispositivo capaz de traducir SPI a IsoSPI, este dispositivo es el LTC6820 que convierte las cuatro señales SPI estándar (CS, SCK, MOSI y MISO) en pulsos diferenciales IsoSPI para ser enviados y recibidos sobre un par de hilos (típicamente un par trenzado), en este trabajo se ha utilizado un cable de conexión Ethernet CAT 6. En la *Figura 3* se ve la forma de estas señales, atendiendo a la señal IP-IM se observa que hay pulsos un poco más anchos que otros y que unos cambian de alto a bajo y otros de bajo a alto.

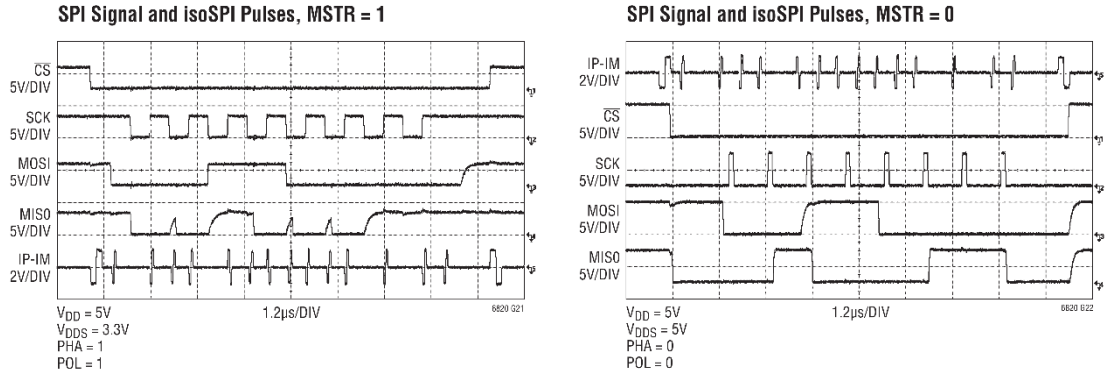


Figura 3. Equivalencia SPI e isoSPI

A continuación, se va a detallar la forma de onda de los pulsos IsoSPI y cómo entienden estos dispositivos el protocolo, pues lo hacen gracias a unos *timers* de pulsos o filtros que discriminan entre pulsos cortos y largos, puesto que hay dos duraciones características, pero también si la transición es de bajo a alto o viceversa, en la *Figura 4* se observan estas dimensiones caracterizadas por los tiempos y diferenciando entre positivos y negativos.

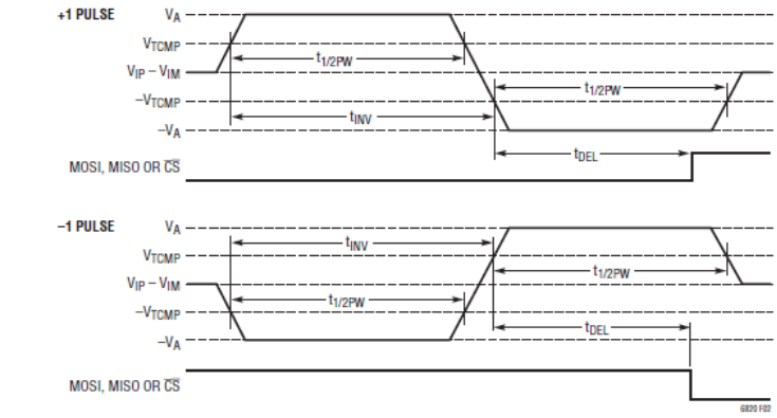


Figura 4. Caracterización de pulsos isoSPI

Ahora que se sabe cómo diferenciar unos pulsos de otros, falta saber qué significa cada uno de ellos. Todos los pulsos empiezan y acaban valiendo 0 V, que es el valor neutro entre cada pulso. Atendiendo al tipo de transición se ha visto que hay dos tipos +1 si la transición es de alto a bajo y -1 si la transición es de bajo a alto. Por otro lado, atendiendo a la duración también hay dos

tipos *Long*, que se usa para determinar si hay un evento en el pin CS y *Short*, que se usa para determinar los datos de los canales MOSI y MISO.

- Long +1**:** Significa que el CS cambia de bajo a alto, es decir, que se desactiva. Un nivel alto seguido de un nivel bajo, cada uno de ellos dura 150 ns (300 ns en total). Solo se transmite desde el lado maestro.

- Long -1: Significa que el CS cambia de alto a bajo, es decir, que se activa**.** Es un nivel bajo seguido de nivel alto (300 ns en total). Solo se transmite desde el lado maestro.

- Short +1**:** Significa que hay un evento en SCK y el MOSI = 1. Es un nivel alto seguido de bajo (100 ns en total). Solo se transmite desde el lado maestro.

- Short -1: Significa que hay un evento en SCK y el MOSI/MISO = 0. Es un nivel bajo seguido de alto (100 ns en total). Se transmite desde el lado maestro y se devuelve desde el lado esclavo.

Desde el lado *master* existen los cuatro tipos de eventos anteriormente mencionados, ya que, es el que se encarga de habilitar al esclavo y en datos deja claro cuando hay nivel alto y cuando bajo, pero desde el lado esclavo solo se pueden devolver pulsos *short -1*, de forma que, cuando el esclavo quiere enviar un nivel alto (MISO = 1), en el par IP-IM no se produce ningún evento en ese flanco de reloj, volviendo a la *Figura 3.* Equivalencia SPI e isoSPI, en la gráfica de la izquierda (MSTR=1) la forma de la señal MISO se ha generado en función de lo que se recibe por IP-IM, se observa que cada ciclo de reloj, en el evento de alto a bajo que no se reciba nada por IP-IM, MISO pasa a nivel alto de forma automática, pero si hay un *short -1*, MISO pasa a nivel bajo, se observa que en estas situaciones el MISO pasa a nivel alto durante un instante y baja rápidamente a nivel bajo, que son las pequeñas aletas de tiburón que se muestran.

2.1.4 LTC6804: Formato de los comandos y el PEC

En primer lugar, se va a definir la señal WAKEUP que tanto se ha mencionado anteriormente, que consiste habilitar y deshabilitar el CS (en IsoSPI, es un *long -1* seguido de un *long +1*) esto es suficiente para estimular el puerto IsoSPI y que pase al estado READY o ACTIVE, si se continúan enviando comandos, como ya se ha visto.

Tanto para escritura como para lectura hay un formato único que es el siguiente, se envían dos bytes del propio comando, el PEC (*Packet Error Code*) que son otros dos bytes utilizados para asegurarse de que la información se ha enviado sin errores de bits y a continuación los bytes de datos, en la *Figura 5* se muestra este formato de manera gráfica.

8	8	8	8	8		8	8	8
CMD0	CMD1	PEC0	PEC1	Data Byte Low	…	Data Byte High	PEC0	PEC1

Figura 5. Formato de los comandos

En cuanto al campo CMDx, hay dos tipos de formato, dado que se pueden utilizar varios stacks en un sistema en el que todos estén comunicados cada uno con su LTC6804, los comandos pueden ser tipo *Broadcast* (para todos) o *Address* (para uno concreto). El tipo *Broadcast* tiene los cinco bits de mayor peso con valor cero y los siguientes once con el valor de la tabla de comandos que da el fabricante y el tipo *Address* tiene el bit de mayor peso de valor uno, y los cuatro siguientes con el valor de la dirección que se le haya asignado por *hardware*. En las *Figura 6* y *Figura 7* se muestran ambos tipos de formatos.

NAME	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CMD0	WR	0	0	0	0	0	CC[10]	CC[9]	CC[8]
CMD1	WR	CC[7]	CC[6]	CC[5]	CC[4]	CC[3]	CC[2]	CC[1]	CC[0]

Figura 7. Formato CMD tipo Broadcast

NAME	RD/WR	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CMD0	WR	1	a3*	a2*	a1*	a0*	CC[10]	CC[9]	CC[8]
CMD1	WR	CC[7]	CC[6]	CC[5]	CC[4]	CC[3]	CC[2]	CC[1]	CC[0]

Figura 6. Formato CMD tipo Address

A continuación, se va a definir el código de comprobación de errores llamado PEC anteriormente mencionado. Es un CRC (*Cyclic Redundancy Check*) de 15 bits, cuyo valor se calcula para todos los bits en un grupo de registros en el orden en que vayan pasando, utilizando un valor inicial de PEC fijo y con el polinomio característico: x15 +x14 + x10 + x8 + x7 + x4 + x3 + 1. Dicho esto, el proceso de cálculo del PEC, es el siguiente:

1) Inicializar el PEC al valor 000 0000 0001 0000 (15 bits).

2) Para cada bit DIN (dato de entrada al cálculo del PEC) hay que establecer:

   - IN0 = **DIN** XOR PEC[14]
   - IN3 = IN0 XOR PEC[2]
   - IN4 = IN0 XOR PEC[3]
   - IN7 = IN0 XOR PEC[6]
   - IN8 = IN0 XOR PEC[7]
   - IN10 = IN0 XOR PEC[9]
   - IN14 = IN0 XOR PEC[13]

3) Actualizar los 15 bits del PEC de la siguiente manera:

   - PEC[14] = IN14; PEC[13] = PEC[12]; PEC[12] = PEC[11]; PEC[11] = PEC[10]; PEC[10] = IN10; PEC[9] = PEC[8]; PEC[8] = IN8; PEC[7] = IN7; PEC[6] = PEC[5]; PEC[5] = PEC[4]; PEC[4] = IN4; PEC[3] = IN3; PEC[2] = PEC[1]; PEC[1] = PEC[0]; PEC[0] = IN0.

4) Vuelta al paso 2, hasta que todo el dato sea recorrido. El PEC final debe ser de 16 bits, de los cuales, los 15 bits de mayor peso son los calculados y añadiendo un 0 en el bit de menor peso.

La *Figura 8* ilustra el algoritmo descrito anteriormente.

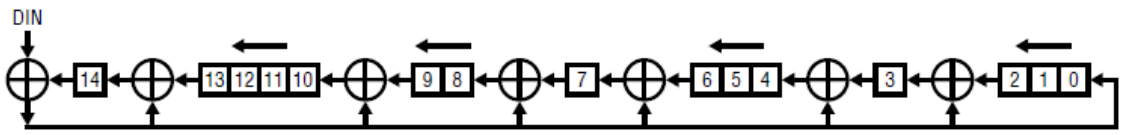
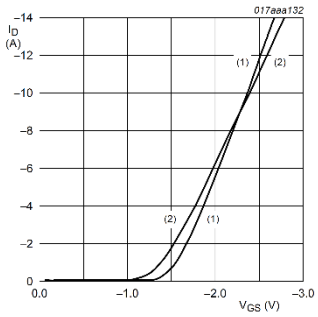


Figura 8. Algoritmo de cálculo del PEC

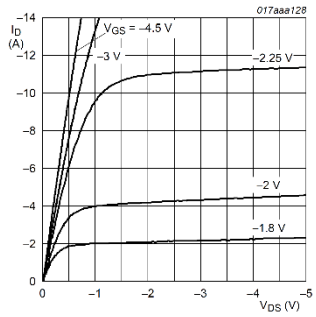
2.2 MOSFET CANAL P

Otro componente que requiere especial interés es el que usamos de interruptor para decidir qué celda debe continuar cargándose y cual no, éste es el transistor MOSFET canal P (se ha elegido canal P y no N, por conveniencia de uso con el LTC6804), a este transistor se le hará funcionar en dos regiones de funcionamiento que son corte y óhmica o lineal. De las curvas características del PMOS sacadas de las hojas de características del componente (PMV48XP), que vemos en la ***¡Error! No se encuentra el origen de la referencia.***, podemos sacar las siguientes conclu-



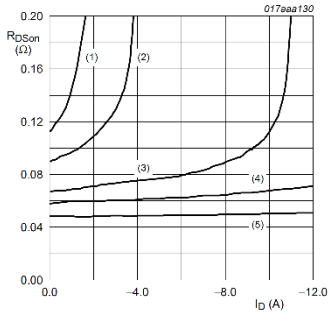
$V_{DS} > I_D \times R_{DSon}$
(1) T_j = 25 °C
(2) T_j = 150 °C

Fig 10. Transfer characteristics: drain current as a function of gate-source voltage; typical values



T_j = 25 °C

Fig 6. Output characteristics: drain current as a function of drain-source voltage; typical values



T_j = 25 °C
(1) V_{GS} = -1.8 V
(2) V_{GS} = -2.0 V
(3) V_{GS} = -2.25 V
(4) V_{GS} = -3.0 V
(5) V_{GS} = -4.5 V

Fig 8. Drain-source on-state resistance as a function of drain current; typical values

Figura 9. Curvas del PMOS

siones:

Cuando está en corte, el transistor se comporta como un circuito abierto, para que esto suceda V_{th} debe ser menor que V_{GS}.

Es decir, que si V_{th} = -1V, V_{GS} debe ser superior a -1 V, por tanto, el BMS le enviará a la puerta un nivel alto de tensión igual a la tensión en la celda, esto se sabe por el circuito interno mostrado en la *Figura 10*. Por ejemplo y teniendo en cuenta que la batería debe tener no menos de unos 2 V y no más de 4,2 V que es su carga máxima, la batería se cargará normalmente.

$$V_G = 3{,}5\ V \quad ; \quad 2\ V \leq V_S \leq 4{,}2\ V \quad ; \quad 1.5\ V \leq V_{GS} \leq -0{,}7\ V$$

Pero cuando la celda esté completamente cargada, el BMS le enviará a la puerta del PMOS un nivel bajo, si miramos la **¡Error! No se encuentra el origen de la referencia.** observamos que

para una V_{GS} = -4,5 V y una I_D = -1 A, que es la corriente de carga, el PMOS está en zona óhmica, lo que significa que se comporta como una resistencia de aproximadamente R_{DS} = 0,05 Ω y la corriente circulará por el PMOS y no por la celda en cuestión.

El circuito interruptor PMOS del cual estamos hablando es el de la *Figura 10*.

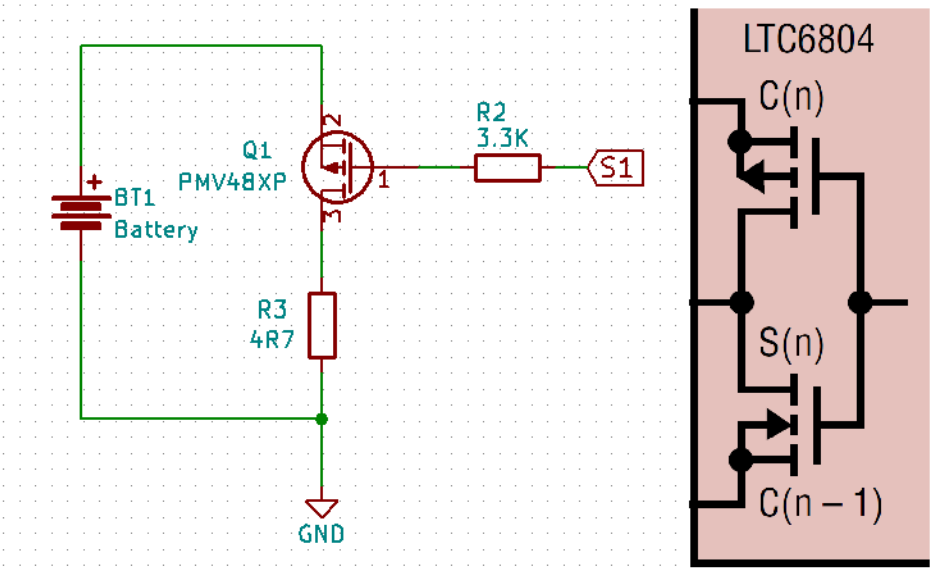


Figura 10. Circuito interruptor

Donde S1 representa la señal de control de la tensión en la puerta que decide si el PMOS debe estar en corte o no. Del transistor, el pin 1 es la puerta (G), el 2 es la fuente (S) y el 3 es el drenador (D) y además se ha representado el circuito interno de la salida S del LTC6804.

3. DESARROLLO DEL TRABAJO

Para la descripción detallada del trabajo, se debe seguir el mismo orden y diferenciación que ya se ha hecho en apartados anteriores, es decir, primero se va a hablar del circuito pensado y diseñado.

A continuación, se desarrolla cómo se ha diseñado la tarjeta PCB en que se implementará el circuito, el *software* utilizado, los componentes elegidos (tanto los que se soldarán en la PCB como los que se utilizarán en circuitos externos a ella).

Por último, se desarrollará con tanto detalle como sea posible, cómo se ha implementado la comunicación entre el BMS y el uC, esto es, el circuito de traducción SPI a IsoSPI y del código de programación escrito en C que se ejecutará en el uC.

3.1 CIRCUITO DISEÑADO

Atendiendo a las necesidades de las baterías y teniendo en cuenta el IC LTC6804, se ha diseñado el circuito que se implementará en la PCB, el cual consta de varias partes, que se irán describiendo una a una. En esta sección se va a describir a la vez, el paso previo al diseño en la PCB, que es hacer el circuito esquemático.

3.1.1 CIRCUITO DE PROTECCIÓN Y ECUALIZACIÓN

Empezando por el circuito de interruptores con los MOSFETS canal P explicado anteriormente, al cual se han añadido algunos elementos. A este circuito a partir de ahora le llamaremos “circuito de protección y ecualización” de carga en las baterías, puesto que esa es su función, ya que es capaz de ecualizar la caga de las celdas haciendo que la corriente deje de circular por ellas una vez cargadas, a la vez que permite que el resto se sigan cargando, por tanto, se consigue que se carguen todas por igual.

En cuanto a la protección se ha dispuesto de un fusible de 2 A de corriente máxima, para prevenir de cortocircuitos u otros hechos no deseados.

Como se puede observar en la *Figura 11*, se han añadido elementos con respecto a la Figura 10, aunque para hacer más visible el circuito se ha recortado la imagen, se aprecia la reproducibilidad del mismo y como ya se ha dicho, el BMS es de diez celdas en serie. En primer lugar, en la fuente del transistor se ha colocado una etiqueta "CELLn" que indica donde debe ir el borne positivo de cada celda y n es el número de celda, aunque además se ha añadido un fusible limitador de corriente a 2 A, entre la fuente del transistor y el propio borne de cada celda como medida de protección como se indica a la derecha de la *Figura 11*.

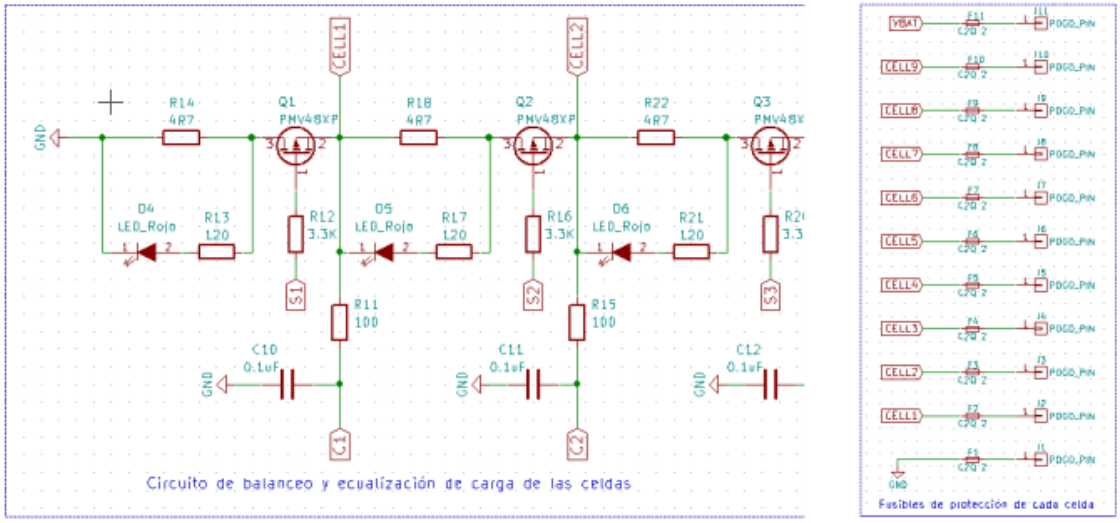


Figura 11. Circuito de ecualización

Se ha añadido el filtro paso bajo RC necesario para la medida de la tensión en cada celda, con una R = 100 Ω y una C = 100 nF, como indica el *datasheet* del LTC6804, por otro lado, se ha añadido también un LED indicador de que ya se ha cargado la batería con su correspondiente resistencia de polarización.

3.1.2 CIRCUITO DE ALIMENTACIÓN

La siguiente parte del circuito que se va a explicar es la correspondiente a la alimentación del circuito. Se ha optado por alimentar el circuito con la propia tensión de las baterías sin necesidad de una fuente externa, porque a pesar de que consumirá energía de las baterías aun cuando no se esté usando, interesa tener un circuito que funcione y sea capaz de medir la tensión en las mismas también cuando no se esté cargando, por tanto, si necesitara una fuente de alimentación externa, se necesitaría o bien, dos fuentes, una para el circuito y otra para cargar las baterías, o bien una sola con la que se alimenten ambas, pero en este caso solo se podría utilizar mientras se está cargando.

Por tanto, la placa está alimentada directamente por las baterías que monitoriza. Como etapa de potencia se utiliza un conversor reductor DC-DC *Buck*, el cual reduce entre los 20 a los 37 voltios de tensión mínima y máxima respectivamente del acumulador de baterías (tensión de entrada) a una tensión de salida regulada de 5 voltios para el LTC6804. El conversor utilizado es el MAX17551 de *Maxim Integrated*, el cual, con los componentes de polarización utilizados es capaz de convertir tensiones de entrada entre 6 y 60 voltios, suficientes para esta aplicación.

Entre la tensión de las baterías y la tensión de entrada al conversor DC-DC anteriormente mencionado se pondrá un diodo Schottky para protección de polaridad inversa y un diodo TVS para proteger contra transitorios de gran tensión.

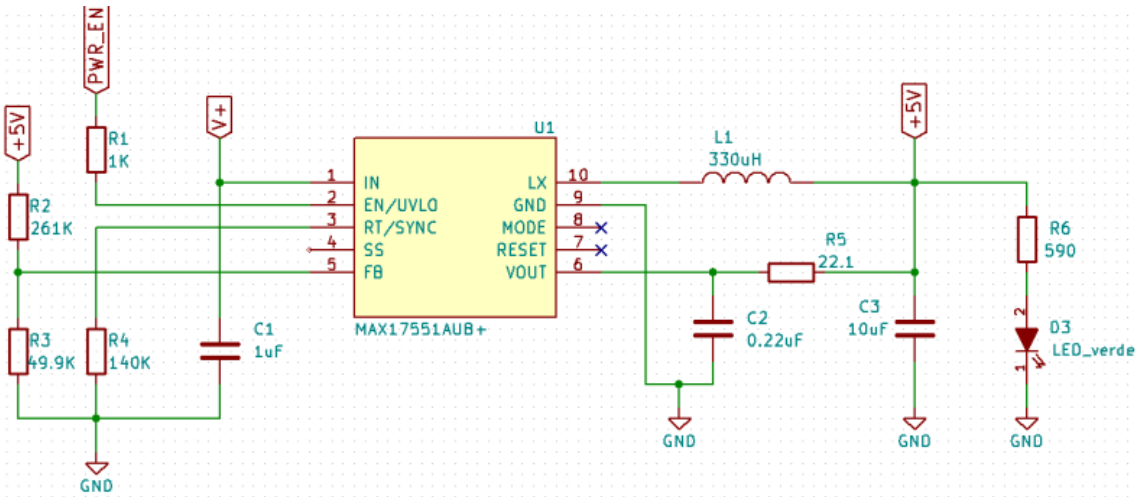


Figura 12. Conversor DC-DC Buck

Como se puede observar en la *Figura 12*, hay tres etiquetas que son señales q van a otras partes del circuito que son, +5V, PWR_EN y V+.

+5V es la salida del conversor DC-DC de 5 voltios necesaria para fijar la tensión de V_{REG} del LTC680, que a su vez sirve para distintas funciones del circuito, tales como indicar si se usa el puerto de comunicación por IsoSPI (entrada ISOMD), SWTEN que es otra entrada digital que debe estar conectada a V_{REG} para habilitar el timer por software.

PWR_EN va directamente conectado a la salida de un regulador interno del IC que se llama DRIVE, sirve para regular la tensión de V_{REG}.

V+ es la tensión más alta de las baterías, conectada por diodo Schottky descrito anteriormente, este circuito se presenta en la *Figura 13*.

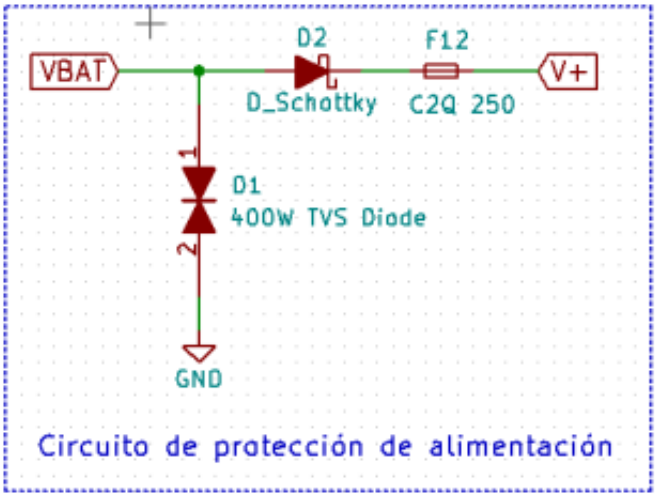


Figura 13. Tensión máxima de baterías

Una cosa a tener en cuenta sobre la V+ es que al conectarlo a la entrada V+ del LTC6804 hay que conectarlo a través de un filtro paso bajo RC, aunque en teoría esa entrada debe ir conectada a la tensión más alta de las baterías directamente, el filtro mencionado y los diodos Schottky y TVS, de los que se hablaba antes, hacen que se produzca una leve caída de potencial en la celda más alta, esto se debe tener en cuenta a la hora de medir, puesto que la celda más alta tendrá menos precisión que el resto.

3.1.3 LTC6804. CIRCUITO DE MONITORIZACIÓN

Aunque ya se habló en la base teórica sobre el circuito integrado LTC6804, es el componente más importante para el propósito del trabajo, puesto que es el responsable de toda la monitorización y en gran medida de la ecualización ya que es el que activa los MOSFET, gracias a que es el componente que contiene los ADCs en serie para medir la tensión en cada celda. La conexión de cada pin se muestra en la *Figura 14*.

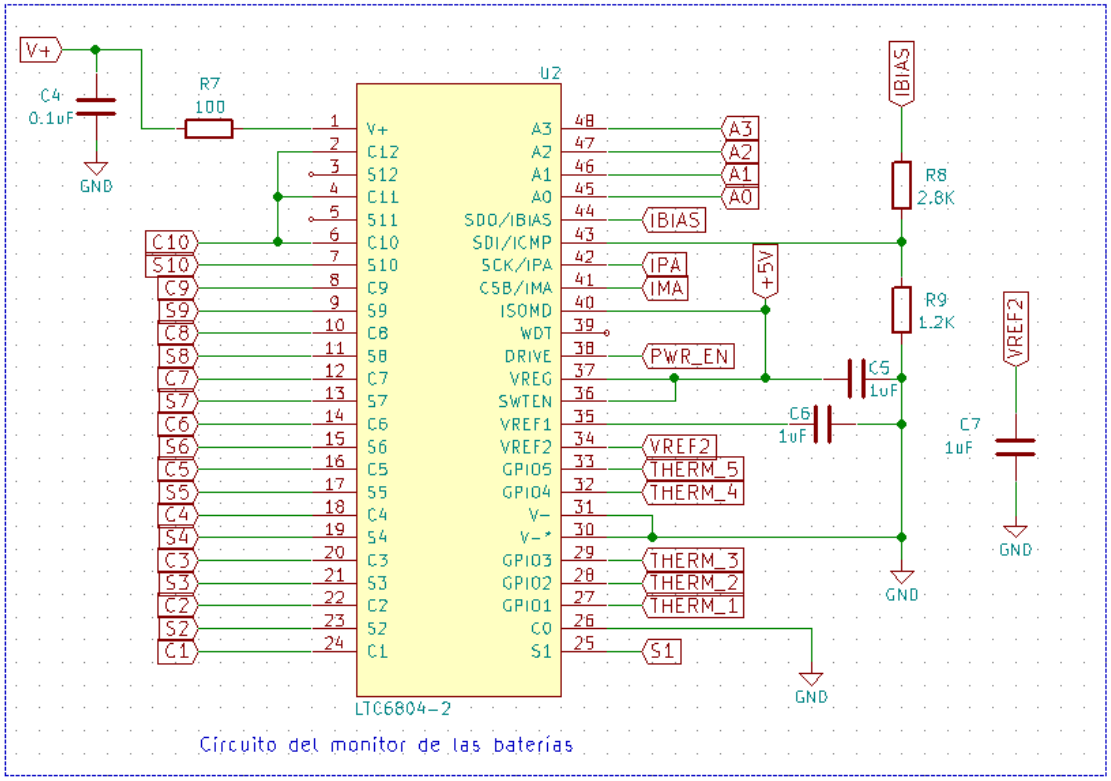


Figura 14. Conexiones LTC6804

Como se puede observar, en el pin 1, está V+ de la q hablamos en la sección del circuito de alimentación.

Aparte de eso se va a explicar la conexión de cada pin. En primer lugar, en el lado izquierdo del integrado se tiene que cada conexión Cn es la entrada de tensión analógica de cada celda como vimos en la *Figura 11*, Sn son las salidas que controlan la puerta de cada MOSFET.

Atendiendo al lado derecho se tiene, de arriba a abajo:

Las entradas A3:A0, indican la dirección del dispositivo con que se está comunicando el maestro del sistema (hay que recordar que se está utilizando SPI), ya que, como dice el título del presente trabajo, es un BMS modular, es decir, que se pueden conectar más circuitos como éste en caso de que se quisiera hacer una ampliación en tensión, para la aplicación en que se vaya a utilizar. Más adelante se hablará del circuito con el switch con que van conectadas estas entradas, pero de momento solo decir que en este trabajo se utiliza la dirección más baja (0000), ya que el acumulador consta de un solo segmento de diez celdas.

IBIAS (Isolated Interface Current Bias): se conecta un divisor resistivo entre IBIAS y V^- (GND) para fijar el nivel de corriente de salida de las señal diferencial IPA-IMA, en este trabajo, como está activado el interfaz IsoSPI, la tensión de salida de este pin son 2 V y la corriente de salida de los dos puertos de comunicación IsoSPI (IPA, IMA) será fijada a veinte veces el nivel de corriente del pin IBIAS, el valor de las resistencias R_{B1} y R_{B2} se va a definir siguiendo los pasos que se describieron en el apartado de la base teórica, en la sección del pin IBIAS, por tanto:

$$I_{DRIVE} = 20 * I_{BIAS} = 10\ mA \quad ; \qquad I_{BIAS} = \frac{2V}{2.8K+1.2K} = 0.5\ mA$$

Por otro lado, la tensión límite del comparador (TCMP) es la mitad de la tensión en el pin ICMP, es decir:

$$V_{ICMP} = I_{BIAS} * R_{B2} = 0.5 * 1.2 = 0.6\ V \qquad ; \qquad V_{TCMP} = 0.5 * V_{ICMP} = 0.3\ V$$

Por tanto, la amplitud en tensión de los pulsos diferenciales será:

$$V_A = I_{DRIVE} * (R_M * 0.5) = 10 * 10^{-3} * (120 * 0.5) = 0.6\ V$$

IPA/IMA (Isolated 2-Wire Serial Interface Port A): IPA (Plus) e IMA (Minus) son el par diferencial de entrada salida utilizado como bus de comunicación IsoSPI.

ISOMD (Serial Interface Mode): Si se conecta este pin a V_{REG} Se activa el modo de comunicación IsoSPI, en caso de que se conectara a masa, nuestro dispositivo funcionaría como SPI a cuatro hilos estándar.

WDT (WatchDog Timer Output Pin): Es una salida en drenador abierto y se ha optado por dejarlo desconectado, puesto que es opcional.

DRIVE: Es la salida de un regulador interno, sirve para gobernar la tensión regulada en V_{REG}, como ya se comentó en el circuito de alimentación, es la señal de *enable* del convertidor reductor DC-DC.

VREG: Regulador de entrada a 5 voltios.

SWTEN (Software Timer Enable): Este timer se habilita conectándolo directamente a V_{REG}. Es un timer que se encarga de los tiempos de descarga.

VREF1: Tensión de referencia de los ADCs, debe conectarse con un condensador de bypass a masa.

VREF2: Se conecta igual que VREF1 y sirve de tensión de referencia para cuando se mida la tensión de los termistores por los GPIOs utilizados como entrada.

GPIO (5:1): Se utilizan como entradas de tensión analógica y comparten los ADCs internos del LTC6804, pero en este caso se utilizan para medir la temperatura de las celdas a través de termistores NTC.

V^-: son dos pines de alimentación negativa, es decir, la masa del circuito, deberán cortocircuitarse externamente.

En la *Figura 16* se muestra el diagrama de bloques del circuito interno del LTC6804. En la *Figura 15*, se ve la conexión anteriormente mencionada del direccionamiento del dispositivo con un *switch* o interruptor.

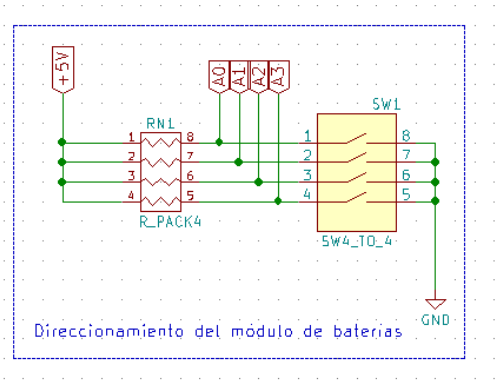


Figura 15. Direccionamiento del módulo BMS

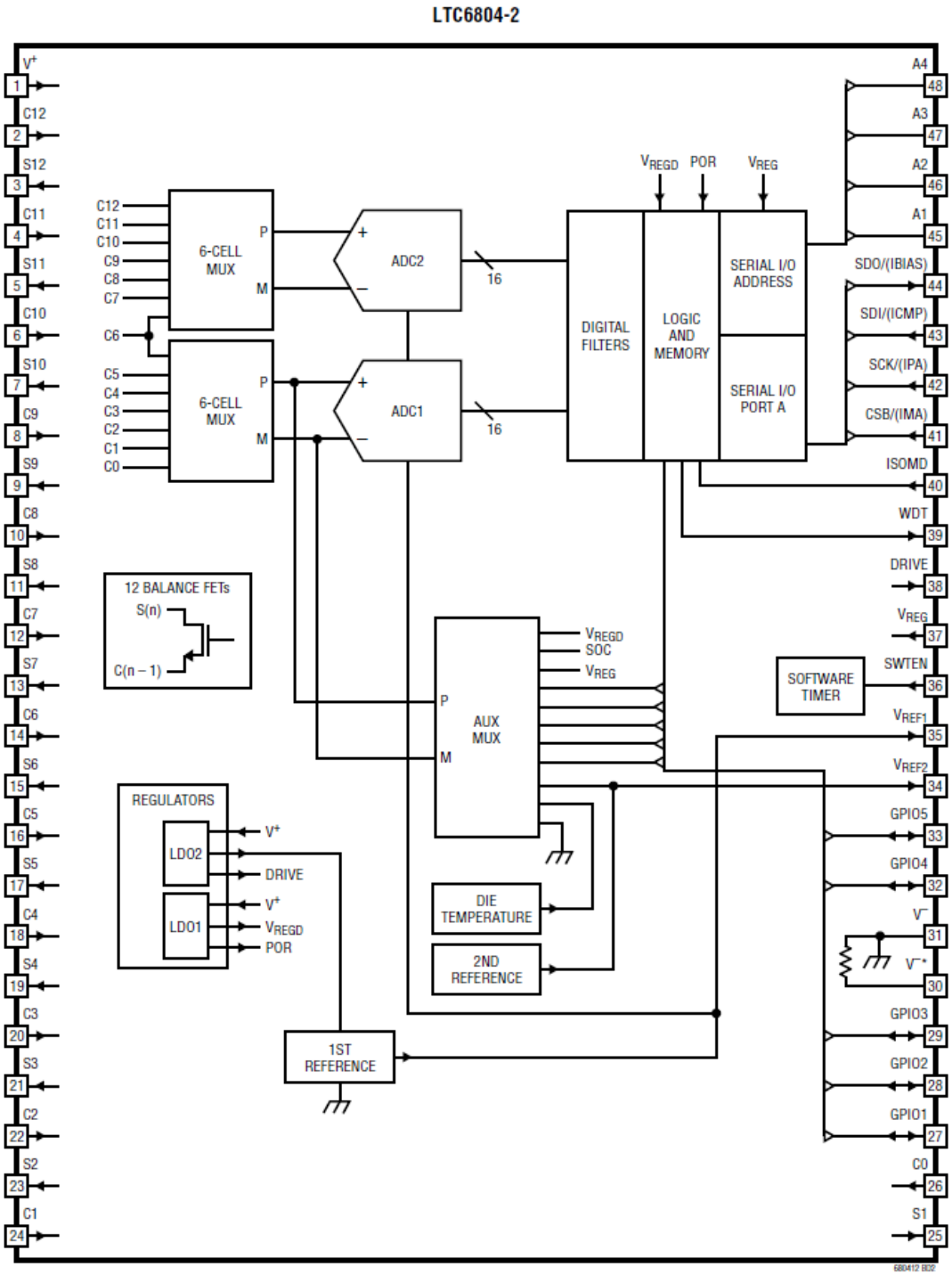


Figura 16. Diagrama de bloques LTC6804

3.1.4 CIRCUITO PARA MEDIR TEMPERATURA

En cuanto a la medida de temperatura, se van a utilizar los pines GPIO (1:5), situando los sensores en la unión de dos celdas, de forma que se mida la temperatura cada dos. El circuito que se ha diseñado para implementar en la PCB, son las resistencias de 10 KΩ conectadas a VREF2 y a unos conectores que sirvan de interfaz con el sensor para poderlo llevar fuera de la PCB. Se puede ver claramente en la *Figura 17*.

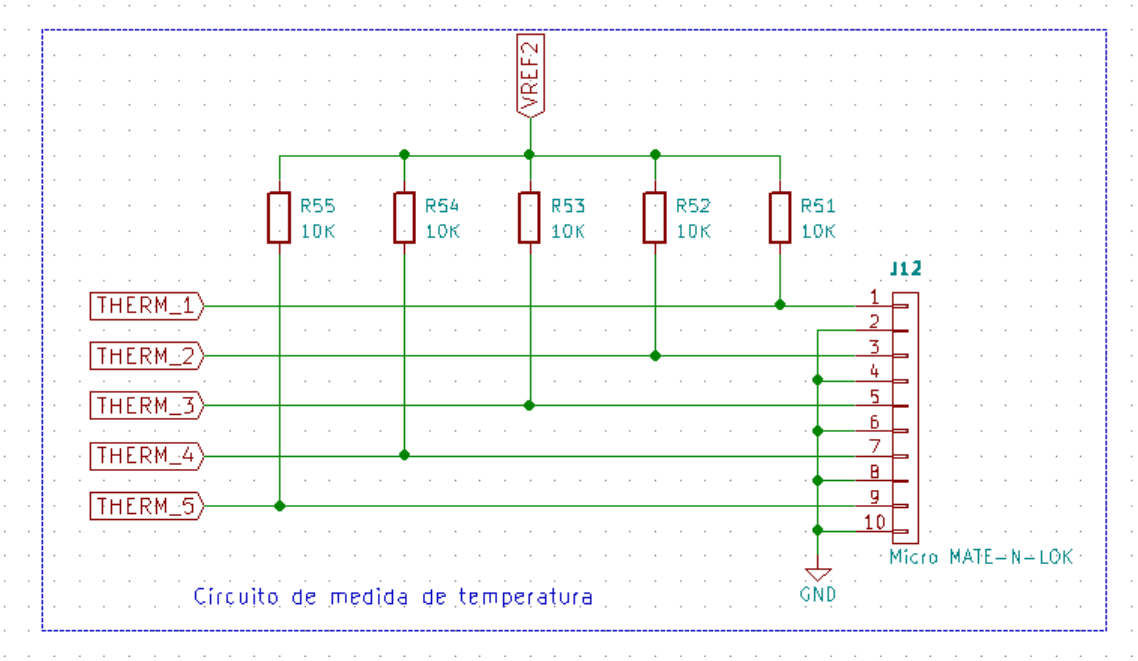


Figura 17. Circuito de polarización para los termistores

La idea es hacer la medida de temperatura mediante un divisor resistivo, donde a temperatura ambiente, el termistor se comporta como una resistencia de 10 KΩ y por tanto, la tensión es (VREF2)/2. A medida que aumenta la temperatura, el valor de la resistencia es menor, entonces, la tensión que se mide en el puerto GPIO es menor y viceversa si disminuye la temperatura. En el programa principal, se hará una tabla de valores para cada rango de temperaturas (cada 5ºC), dado que, uno de los inconvenientes de este sensor es que no es lineal.

3.1.5 CONECTORES RJ45

Por último, para realizar la conexión física entre distintos módulos BMS o entre un BMS y el uC, se ha pensado utilizar conectores RJ45, ya que en su interior tienen transformadores para el aislamiento galvánico necesario, además de darle un acabado más profesional, cómodo y universal que si se dejan los cables como par trenzado al aire. Esto también hay que añadirlo al esquemático del circuito para posteriormente hacer el conexionado en el diseño de la PCB.

En la *Figura 18* se ven las conexiones con los conectores mencionados.

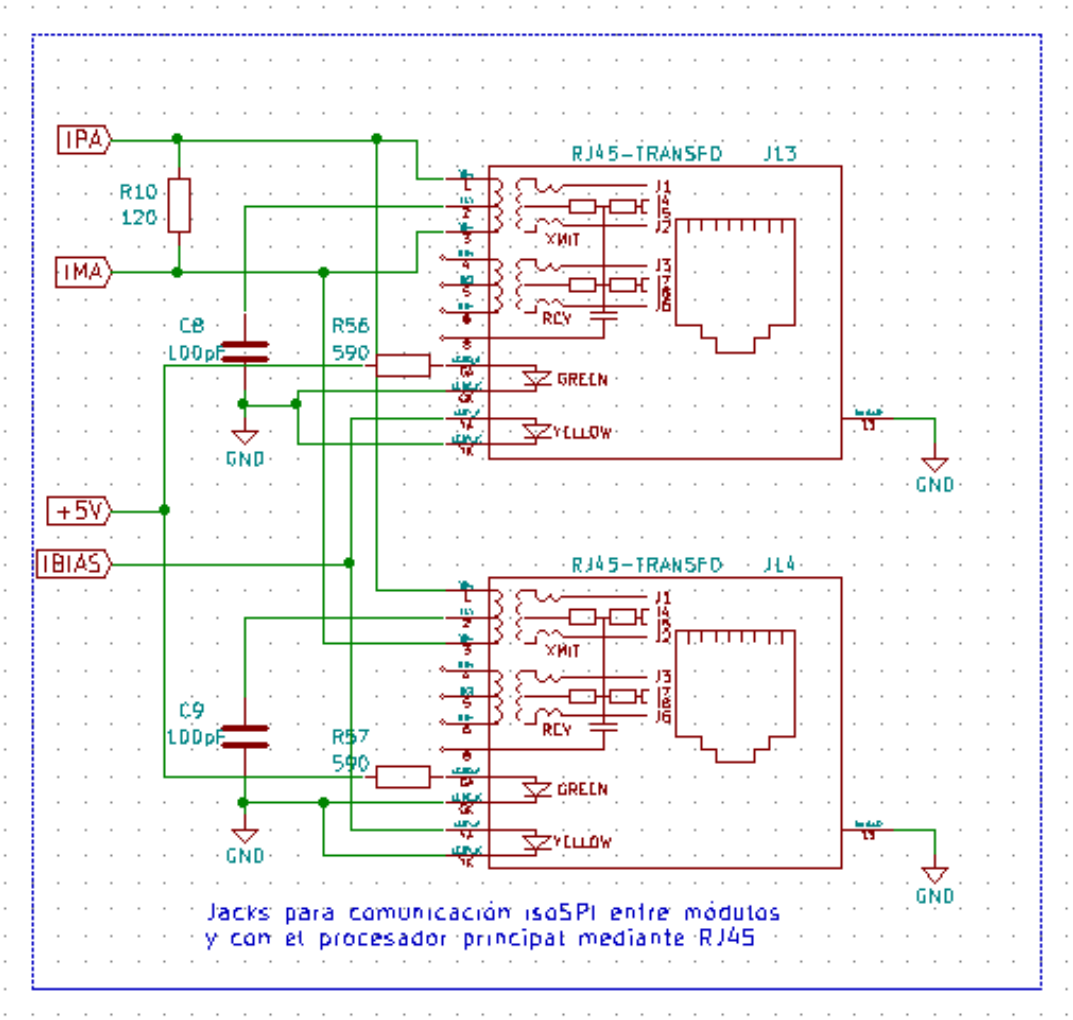


Figura 18. Conexionado con RJ45

3.2 CIRCUITO TRADUCTOR SPI A ISOSPI

Una vez descrito todo el esquemático del circuito que se va a implementar en la PCB, se va a proceder a explicar el circuito necesario para interpretar los comandos SPI a IsoSPI y viceversa.

El circuito es el LTC6820, al que se le han añadido componentes pasivos necesarios para su correcto funcionamiento y la aplicación del trabajo, en la *Figura 19* se muestra el diagrama de bloques con sus conexiones externas, que se explicarán a continuación.

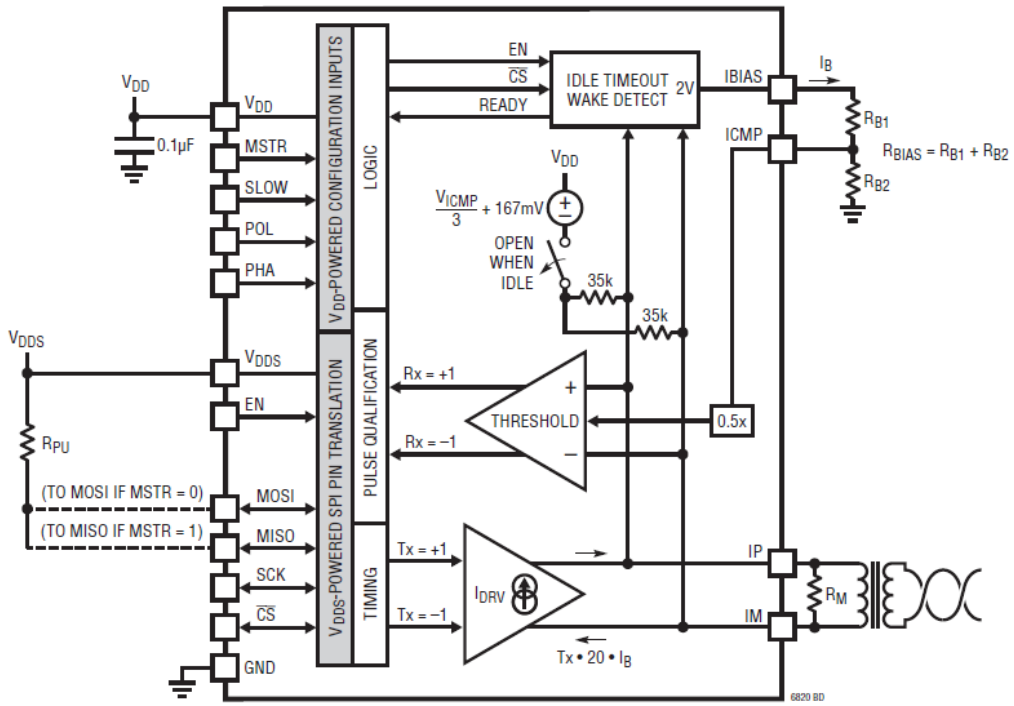


Figura 19. Diagrama de bloques LTC6820

En primer lugar, la alimentación V_{DD} se ha fijado a 5 V y V_{DDS} a 3.3 V, puesto que esta es la tensión que entenderá el dispositivo como nivel alto en la comunicación con el uC.

Dado que este dispositivo está situado en el lado master de la comunicación (es el intérprete del uC), R_{PU} se conecta al pin MISO. Por otro lado, R_{B1}, R_{B2} y R_M deben tener el mismo valor que en el BMS, estos valores son R_{B1} = 2.8 KΩ, R_{B2} = 1.2 KΩ y R_M = 120 Ω.

A continuación, se va a mostrar una trama isoSPI capturada con el osciloscopio. Esta captura se corresponde al comando WRCFG (Write Configuration Register Group) de tipo *address* que recordemos que debe ser:

Tipo address (1 bit)	Address (4 bits)	CMD (11 bits)
1	0000	00000000001

O dicho en hexadecimal 0x8001, cuyo PEC es 0x4D7A.

Figura 20. ISOSPI, comando WRCFG

Aunque en la *Figura 20*, no se aprecien las formas de onda, se va a explicar lo mejor posible la interpretación de la imagen. El canal 1 (señal amarilla) es IPA, el canal 2 (señal verde) es IMA, y la señal Math (Morada) es IPA-IMA, con una amplitud:

$$+450\ \text{Mv} > V_A > -450\ \text{mV}\ \left({}^{200\text{mV}}/_{\text{DIV}}\right)$$

Se ha especificado los datos sobre la imagen para ver que son correctos, pero lo importante es que se puede apreciar que por cada división horizontal hay 5 pulsos y teniendo en cuenta que hay 5 µs/DIV, significa que la velocidad de transmisión es:

$$bitrate = {}^{5\ bits/DIV}/_{5\mu s/DIV} = 1 * 10^6 bits/s$$

Es decir, 1Mbps. En la siguiente imagen, (*Figura 21*) se ha hecho un zoom al comienzo de la trama para que se aprecien las transiciones.

Figura 21. IsoSPI, Inicio comando WRCFG

Ahora sí se aprecia perfectamente las transiciones de alto a bajo y viceversa en la señal morada, que es la importante, también se ve perfectamente que el *long -1* correspondiente a la activación del CS, es el doble de ancho que los siguientes pulsos e incluso tiene más amplitud.

3.3 DISEÑO DE LA PCB

Una vez que se ha definido todo el circuito que compondrá el BMS, sobre un esquemático que se ha ido definiendo en el apartado anterior paso a paso, lo siguiente que se debe hacer es directamente el diseño de la PCB. Este diseño se ha realizado con la herramienta de software libre y gratuito KiCAD EDA.

Antes de pasar al diseño de la PCB, se debe de crear la Netlist, que es una lista con todas las conexiones, el nombre de una conexión es una red, por ejemplo, GND es el nombre de la red de masa y todos los componentes conectados a masa, tendrán una conexión con la red GND. Hecho esto, hay que asignar una huella a cada componente y entonces comenzar con el diseño.

Lo primero que hay que tener en cuenta a la hora de diseñar una PCB son las dimensiones de los diferentes elementos. En este caso, las dimensiones más restrictivas son las que tienen que ver con las celdas, puesto que la PCB se atornillará sobre ellas y por tanto, se necesita que los taladros en los que se coloque el tornillo coincidan con el borne correspondiente. En total son diez celdas en serie, por lo que se necesitan once taladros, uno de masa o referencia y los otros diez las tensiones en cada celda. Estas dimensiones se pueden apreciar en la vista desde la planta del segmento de baterías que se muestra en la *Figura 22*.

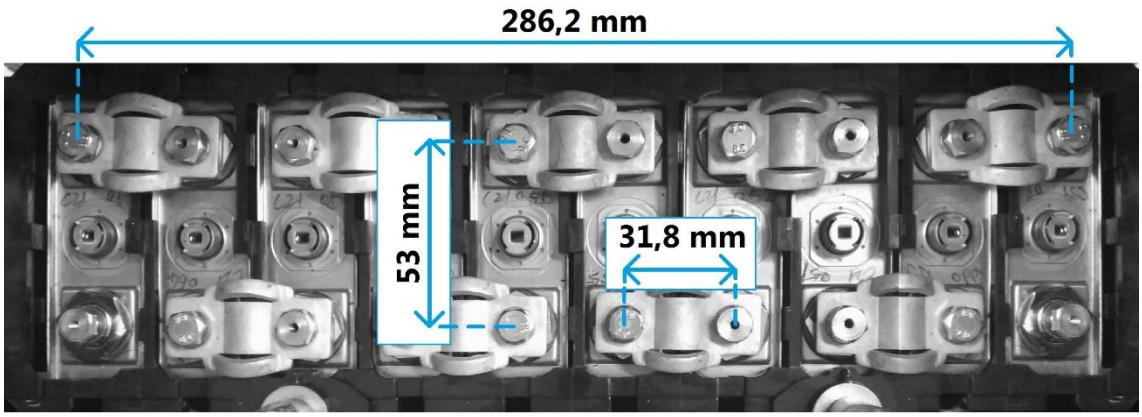


Figura 22. Planta de baterías con cotas

Dadas las dimensiones de la *Figura 22*, el margen de dimensiones de la placa, para que no sobresalga de las baterías ni sea demasiado pequeña es, a lo largo entre 290 y 320 mm y a lo ancho entre 60 y 95 mm, entonces se decide hacer una PCB de 300x70 mm, en la *Figura 23*, se ilustra la forma previa de la PCB con la colocación de los taladros de los bornes de las baterías.

Una vez delimitada la forma de la PCB, se carga la Netlist y aparecen todas las huellas asignadas a cada componente, lo único que se debe hacer es colocarlos donde nos parezca mejor y pasar al ruteado.

Llegados a este punto, cabe decir que, hay componentes que importa donde estén colocados y componentes que no tanto, por ejemplo, el LTC6804 basta con situarlo más o menos en el centro pero ningún sitio en concreto, sin embargo, los conectores RJ45 deben estar colocados en el borde, los LEDs indicadores de cada celda, conviene que estén situados encima de cada celda en cuestión y el caso más preciso es el de los taladros para los tornillos obviamente deben coincidir con el borne de la celda determinada.

Para tener estos puntos localizados, lo primero que se ha hecho es, delimitar los bordes de la placa con las cotas significativas necesarias y dibujos para señalizar donde van los taladros como se muestra en la *Figura 23*.

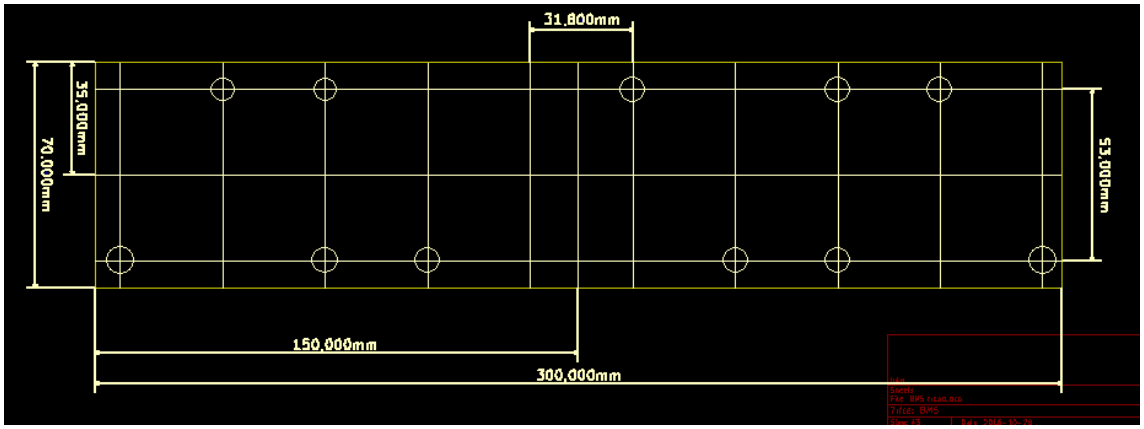


Figura 23. Bordes de la placa y medidas de interés

Una vez hecho esto, se procede a situar cada huella en su sitio para poder pasar al proceso de ruteado. En la *Figura 24* se muestra la placa con los componentes sin rutear.

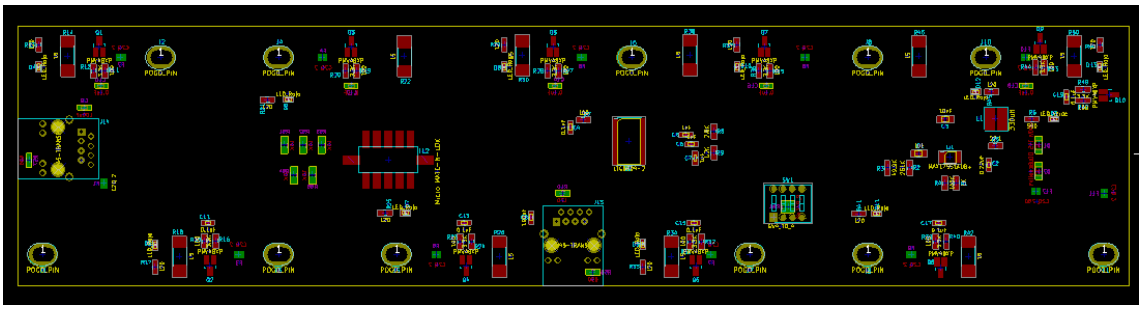


Figura 24. Placa con componentes sin rutear

Para rutear la placa, se han elegido dos anchos de pista distintos, dependiendo de si son pistas por donde va a pasar más corriente o menos. La corriente más alta que va a atravesar este circuito es la de carga de las baterías, que por conveniencia de seguridad va a ser típicamente de un amperio, pero nunca mayor de dos. Para calcular el ancho de pista necesario necesitamos saber la sección del conductor y su conductividad. KiCAD tiene una herramienta muy útil que es una calculadora electrónica que, entre otras cosas, sirve para calcular el ancho de pista necesario para una determinada corriente, para acceder a ella hay que pulsar en "Herramientas" y después "Abrir Pcb Calculator".

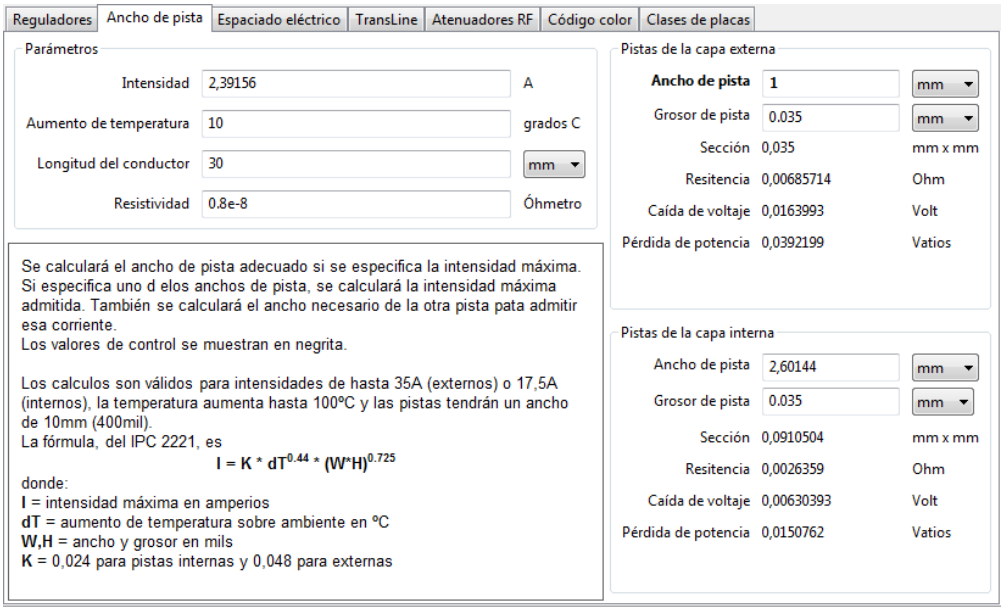


Figura 25. Calculadora de placas KiCAD

En la *Figura 25* se muestra que con un ancho de pista de 1 mm puede circular una corriente de casi 2,4 A, por tanto, se ha elegido este tamaño.

Para el resto de conexiones que no necesitan esta corriente, se ha elegido un ancho de pista de 0,2 mm por donde puede circular una corriente de hasta 0,745 A, suficiente para la aplicación.

La forma de unir estas pistas a sus redes correspondientes se ha hecho de forma completamente manual, con el propósito de dejar la red GND sin conectar, esto se ha hecho para después crear una zona rellena en la capa de abajo para la red GND, de esta manera, todas las conexiones que vayan a masa, bastará con pasar una vía de la capa de arriba a la de abajo, aunque la capa de abajo no es completa GND puesto que se ha necesitado pasar pistas por esta capa de otras redes y dejar un espacio de seguridad más que suficiente para el espacio que ocupen las tuercas y arandelas que se utilicen al atornillar los bornes.

El resultado se muestra en la *Figura 27* donde se pueden apreciar las conexiones sin la zona rellena y con ella. En la *Figura 26* se puede observar la placa con el visor en 3D de KiCAD.

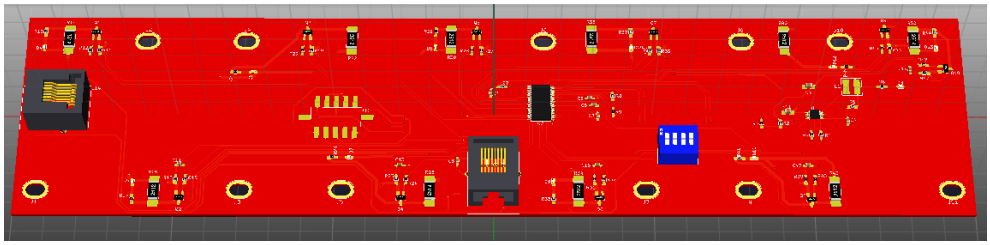


Figura 26. Vista 3D de la PCB

Por último añadir que, una vez diseñada la PCB, hay que exportar los archivos Gerber que contienen toda la información necesaria para la fabricación de la PCB, convierten el diseño en codigo de texto plano. Al fabricante se le deben enviar al menos los archivos Gerber de los taladros, los cortes de los bordes, las capas donde se haya escrito (silk screen), las de máscara y las de cobre.

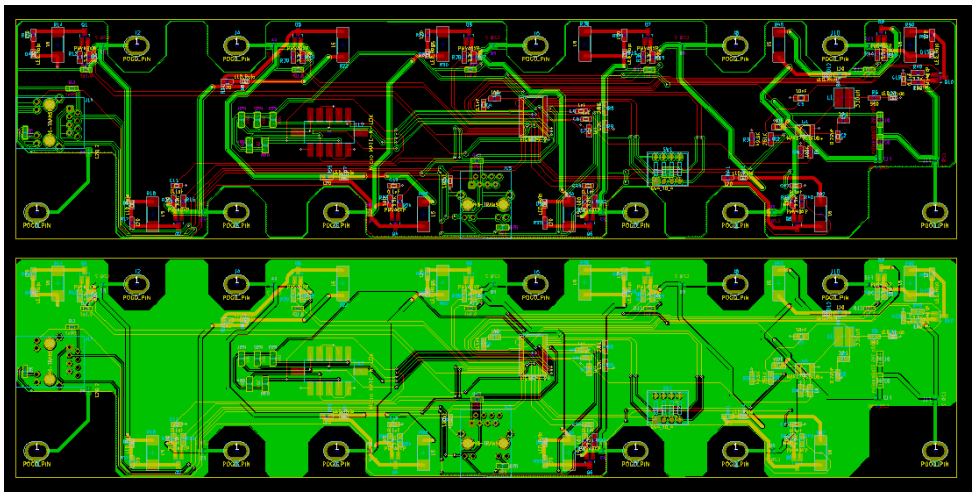


Figura 27. Vistas del diseño de la PCB

3.4 PROGRAMA DISEÑADO

La programación del uC que se ha utilizado de *master* en el trabajo se ha escrito en código c, la tarjeta de desarrollo es la Mini-DK2 cuyo procesador o *core* pertenece a la familia LPC17xx, es un ARM Cortex M-3, en este caso el LPC1768 y el entorno de desarrollo es Keil uVision4 IDE de ARM. Esta tarjeta ha sido utilizada en la asignatura de Sistemas Electrónicos Digitales Avanzados del Grado en Ingeniería Electrónica de Comunicaciones (GIEC) de la Universidad de Alcalá.

La idea de utilizar esta tarjeta de desarrollo es que, además de que no es necesario comprar una nueva para el trabajo porque ya se ha utilizado anteriormente, se puede utilizar la pantalla táctil como GUI (*Graphical User Interface*) para seleccionar los comandos que se quieran enviar y visualizar los datos que nos devuelva el BMS, el problema es que el LPC1768 tiene dos puertos SPI, y la pantalla utiliza los dos, uno para visualizar y otro para recoger los datos al pulsar, por tanto, las opciones son generar el protocolo SPI con cuatro pines en modo GPIO o bien, usar un módulo SSP configurado en modo SPI, que aparentemente es mejor opción y es la que se ha decidido emplear.

Por tanto, hay que programar un menú en el cual, podamos visualizar datos y enviar comandos, además de programar el módulo SSP en modo SPI que sirva para comunicar el uC con el BMS.

3.4.1 MÓDULO SSP CON INTERFAZ SPI

Lo primero que se ha programado es el interfaz o modo SPI del módulo SSP, para la aplicación del trabajo se ha utilizado el SSP0 con una velocidad de transmisión de bits o tasa de bits (*Bitrate*) de hasta 1 Mbps, que es el máximo bitrate que admite el BMS.

Por otro lado, se ha decidido poner los registros de los parámetros de configuración CPOL=1 y CPHA=1. El primero es la polaridad del reloj y el segundo es la fase de MOSI y MISO, al estar los dos a nivel alto, el estado del reloj permanece a nivel alto cuando se esté en estado ocioso (CS desactivado) y la información se envía en cada transición de alto a bajo y se lee en cada transición de bajo a alto de reloj, es decir, activo a nivel bajo.

Debajo se puede ver la función de inicialización del SSP0 con comentarios aclarativos de lo que se ha explicado en el párrafo anterior.

```
/*!-----    Función de inicialización    ------*/

void SSP0_Init() {
      // CS using GPIO P1.21
      LPC_GPIO1->FIODIR |= 1 << 21;
      // Power SSP
      LPC_SC->PCONP     |= 1 << 21;
      //PLCK_SSP0 = 100MHz/2 = 50MHz
      LPC_SC->PCLKSEL1 &=~ 3<<10;
      LPC_SC->PCLKSEL1 |= 2<<10;
      // P1.20 -> SCK0
      LPC_PINCON->PINSEL3 &= ~ (3 << 8);
      LPC_PINCON->PINSEL3 |= 3 << 8;
      // P1.23, P1.24 -> MISO, MOSI
      LPC_PINCON->PINSEL3 &= ~ (3 << 14 | 3 << 16);
      LPC_PINCON->PINSEL3 |= 3 << 14 | 3 << 16;
      // Normal operation, SSP controller enabled as MASTER
      LPC_SSP0-> CR1 = 2;
      LPC_SSP0-> CR0 = (7 << 0) |          // 8 bit
                       (1 << 6) |          // CPOL = 1
                       (1 << 7) |          // CPHA = 1
                       (0 << 8);           // SCR = 0
      // bitrate = PCLK / [CPSR x (SCR + 1)]
      // bitrate = 50MHz / [50 x (0 + 1)] = 1 Mbps
      LPC_SSP0-> CPSR  |= 50;  // tiene que ser numero par máx = 254
      LPC_GPIO1->FIOSET |= 1 << 21;        // CS=1
}
```

En cuanto al SSP0, solo queda crear funciones para enviar datos, recibirlos o ambas acciones en la misma función. En este caso se ha optado por hacer las tres funciones, pero se va a mostrar solo la que contiene las dos acciones en la misma función, para no redundar en lo mismo.

```
/*!-----    Funcion de envio y recepcion       ------*/

// *tx_Data: Array de datos para escribir en el MOSI del SPI port
// tx_len: Tamaño del array tx_Data
// *rx_data: array que almacenará los datos leidos por el MISO
// rx_len: Tamaño del array rx_Data

void envio_recep(uint8_t *tx_Data,
                 uint8_t tx_len,
                 uint8_t *rx_data,
                 uint8_t rx_len ) {
      uint8_t i;
      for(i = 0; i < tx_len; i++){
            LPC_SSP0->DR = tx_Data[i];
            /* Wait if Tx FIFO is full. */
            while (!(LPC_SSP0->SR & TNF));
            LPC_SSP0->DR;
      }

      /* Wait until Tx finished, drain Rx FIFO. */
      while (LPC_SSP0->SR & (BSY | RNE)) {
            LPC_SSP0->DR;
      }

      for(i = 0; i < rx_len; i++) {
            LPC_SSP0->DR = 0xFF;
            /* Wait while Rx FIFO is empty. */
            while (!(LPC_SSP0->SR & RNE));
            rx_data[i] = LPC_SSP0->DR;
      }
}
```

Obsérvese que no se ha activado ni desactivado el CS, puesto que, se hará cada vez que se use la función, para que, en el caso de que se quieran enviar dos o más comandos seguidos, se activará antes del primer comando y desactivará después del último, sin necesidad de hacerlo con cada uno de ellos.

A continuación, se va a ilustrar mediante capturas hechas con un analizador lógico, una serie de tramas SPI correspondientes a comandos utilizados en el trabajo. Esto servirá para describir gráficamente el formato de los comandos definidos en la sección de la base teórica.

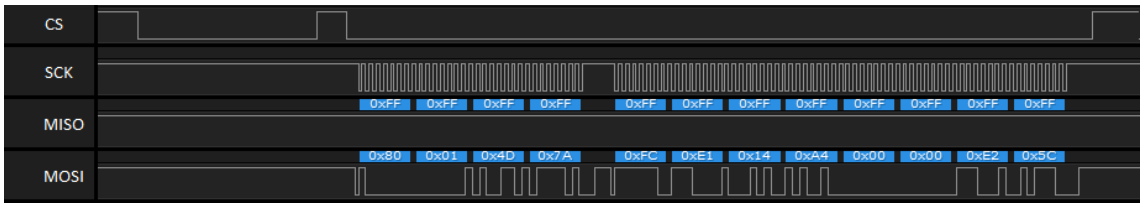


Figura 28. SPI, Comando WRCFG

En la *Figura 28* se ve que activamos el CS sin enviar ni recibir ningún tipo de información antes de empezar con el envío de comandos, esto es lo que se llama *dummy byte* utilizado como señal WAKEUP, esta señal, como ya se explicó en el apartado del diagrama de estados del LTC6804, sirve para que el circuito salga del estado SLEEP y a la vez, el puerto IsoSPI entre en el estado READY. Acto seguido se envía por el MOSI el comando WRCFG en modo *address* con dirección 0000 y los once bits del campo CMD son en hexadecimal 0x001, los dos siguientes bytes son el PEC, que en este caso tiene de valor 0x4D7A y a continuación se envían los seis bytes de datos del registro de configuración, que corresponden con la propia configuración que se le quiere establecer al BMS, seguidos de los dos últimos bytes que son el PEC de estos datos.

De esta adquisición solo añadir que como se puede observar, en el MISO solo hay niveles altos porque el comando WRCFG, es solo de escritura.

Ahora se va a mostrar en la *Figura 29*, el comando RDCV utilizado para leer los datos adquiridos por los ADCs del BMS, para que se vea que, efectivamente, el BMS envía los datos al MISO si el comando es de lectura.

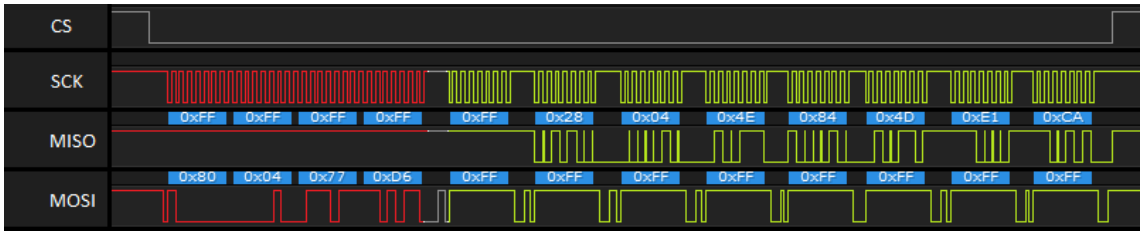


Figura 29. SPI, comando RDCV

En este caso, el campo CMD es 0x004, correspondiente al comando de lectura del registro "*Read Cell Voltage Register group A*", se llama así porque el tamaño de estos registros es de seis bytes y la lectura de cada celda son dos bytes, por tanto, se necesitan cuatro registros como estos

llamados "*group A, group B, group C y group D*", dado que el LTC6804 puede leer de hasta 12 celdas. Se ha definido en color rojo la parte correspondiente al comando más su PEC y en color verde la parte de los datos de lectura más su PEC.

3.4.2 MÁQUINA DE ESTADOS DEL PROGRAMA

Para describir la máquina de estados, se va a tratar de explicar comparando lo que se muestra por el GUI creado en la pantalla de la Mini DK2 con el funcionamiento interno del programa. Lo primero que hay que tener en cuenta, es si nuestro BMS está cargando baterías o no, esta diferenciación se hace, porque si no está cargando, es posible volver a los estados del BMS de IDLE por parte del puerto IsoSPI y SLEEP por parte del circuito, en caso de que no se hayan enviado comandos durante sus respectivos, fin de cuenta o *timeouts* y así reducir al mínimo un consumo innecesario. Pero si está cargando, se necesita saber constantemente el estado de cada celda para que, en caso de que alguna haya finalizado su carga, poder dejar de cargarse y así comenzar con la ecualización de carga, de modo que se estarán enviando constantemente comandos de conversión y lectura de los ADCs. Todo esto, se estará haciendo en paralelo a la máquina de estados que se definirá a continuación:

El estado inicial es el menú principal (MAIN_MENU), que es el primero que se ve representado en la pantalla, éste a su vez tiene tres submenús, uno que sirve para mostrar los registros de estado del BMS (STATUS_MENU), otro con las medidas de tensión y temperatura de cada celda (MEASURE_MENU) y un menú de selección de número de stacks (ADDRESS_MENU) que sirve para el direccionamiento de dispositivos LTC6804-2 que se tienen en total monitorizándose desde el micro, aunque en este trabajo siempre será un, pero se ha implementado para casos futuros en los que se quiera ampliar el número de stacks. Desde el MAIN_MENU, se puede acceder al resto de menús y de ellos a éste, pero no se puede acceder al resto entre ellos, es decir, no se puede acceder desde el STATUS_MENU al MEASURE_MENU, por ejemplo.

El menú de información de estado del BMS, (STATUS_MENU), consta de dos botones, uno para actualizar la información y otro para volver al MAIN_MENU, además, muestra dos indicadores de colores en la parte superior izquierda para saber si se ha recibido el PEC de los comandos RDCFG y ADSTAT correctamente, si estos indicadores son verdes significa que los PEC son correctos y si son rojos, incorrectos. Además, en el STATUS_MENU se visualizan los datos de los comandos mencionados anteriormente, estos datos son:

- Nivel de descarga máximo permitido (undervoltage) dado en voltios.
- Sobretensión (overvoltage) dado en voltios, cuando una celda alcance este valor, se activará el MOSFET que hay en paralelo a dicha celda, para que la corriente circule por él y así, se deje de cargar la celda mientras que las otras continúan sin necesidad de desconectar nada.

- SOC (Sum of cells) es la suma de tensiones de todas las celdas, dada en voltios y en porcentaje. No confundir con el SOC (State of charge) que es el mismo concepto, pero entendido para una celda en particular y no para la suma de todas las que hay en el Stack.

- Bit Refon, es un bit que hace que las referencias de tensión se mantengan activas, esto se utiliza para lo siguiente: Cuando el BMS quiere capturar una medida de tensión, pasa del estado Standby al estado Measure esto tarda un tiempo t_{refup} que es el tiempo que tarda en activarse las tensiones de referencia de los ADCs, para evitar este tiempo, el bit Refon, se coloca a nivel alto, entonces el BMS en lugar de estar en el estado Standby, se sitúa en el estado Refup y las conversiones se realizan inmediatamente.

- Tensión de alimentación analógica Vreg, se corresponde a la tensión de referencia de 5 voltios.

- Tensión de alimentación digital Vregd.

- Internal Die Temperature (ITMP).

El menú de medida de tensión y temperatura en cada celda (MEASURE_MENU), al igual que el STATUS_MENU, consta de los dos botones (actualizar datos y volver al MAIN_MENU) y los dos indicadores de colores de comprobación del PEC, en este caso de los comandos RDCV y RDAUX.

El menú de direccionamiento de Stacks de baterías (ADDRESS_MENU), es el menú en el que seleccionamos el número de ICs que hay en el sistema, se muestra un numero de dos dígitos con uno de ellos resaltado para saber cuál se está cambiando, un teclado numérico de 0 a 9, un botón para elegir el dígito de la izquierda y otro para elegir el de la derecha y otro botón para volver al MAIN_MENU. En caso de que el dígito de la izquierda sea mayor que uno, valdrá uno y si el número global es mayor que dieciséis, el valor será dieciséis, esto es, porque no se puede direccionar un número mayor, puesto que las entradas de selección de dirección son cuatro.

En la *Figura 30* se muestra un Statechart de la máquina de estados con pseudocódigo explicativo.

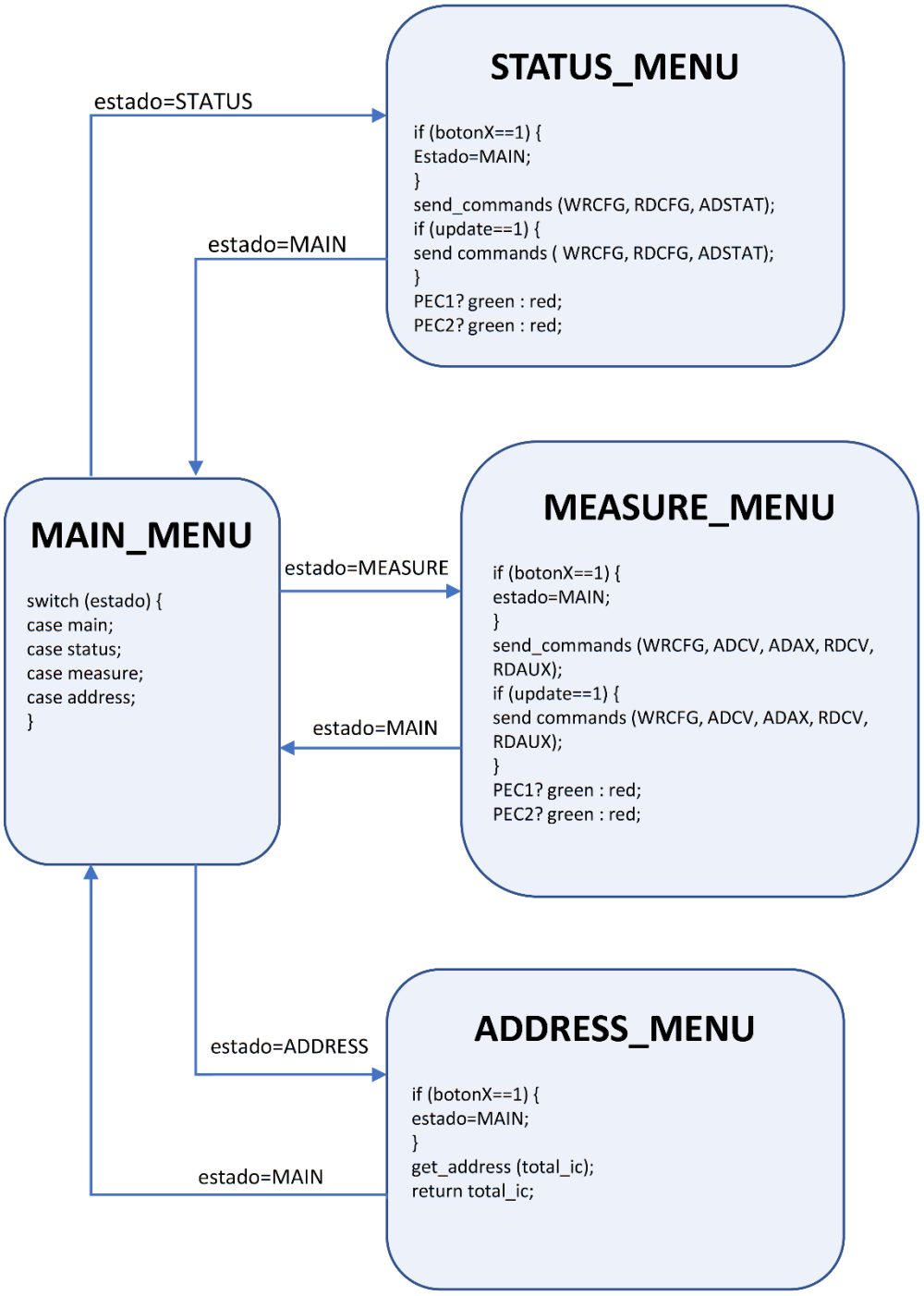


Figura 30. Statechart

VIII. CONCLUSIONES E IDEAS FUTURAS

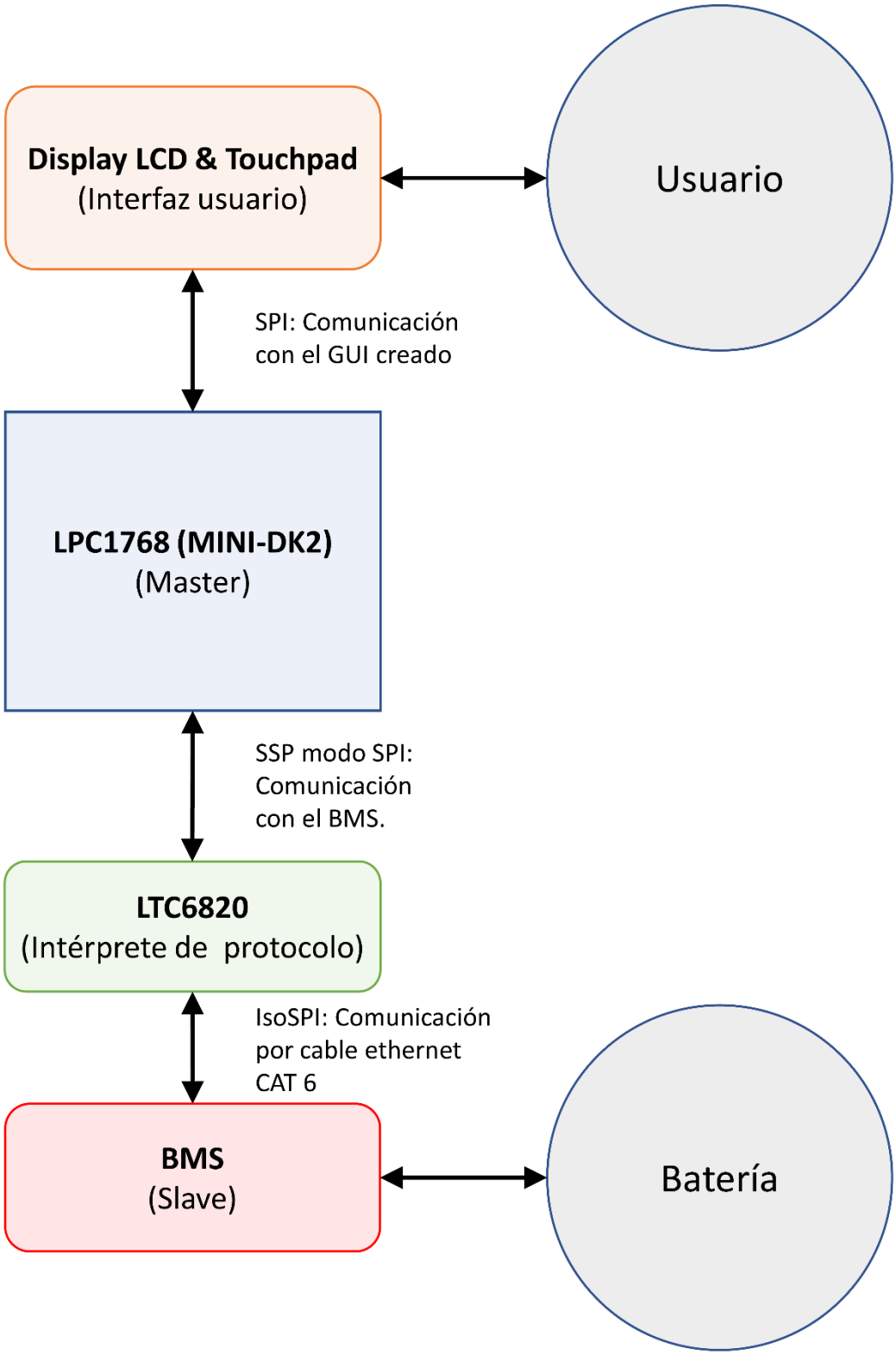
Se ha conseguido desarrollar la propuesta inicial de un circuito BMS de baterías Li-ion modular con los objetivos iniciales y funciona correctamente.

El sistema es estable, además de la robustez debida al aislamiento galvánico entre maestro y esclavo, la velocidad de conversión de los ADC del LTC6804 es muy alta y de mucha precisión, el tiempo de visualización es lento debido exclusivamente a la velocidad del uC utilizado y al tiempo de adquisición y conversión de los ADCs del BMS.

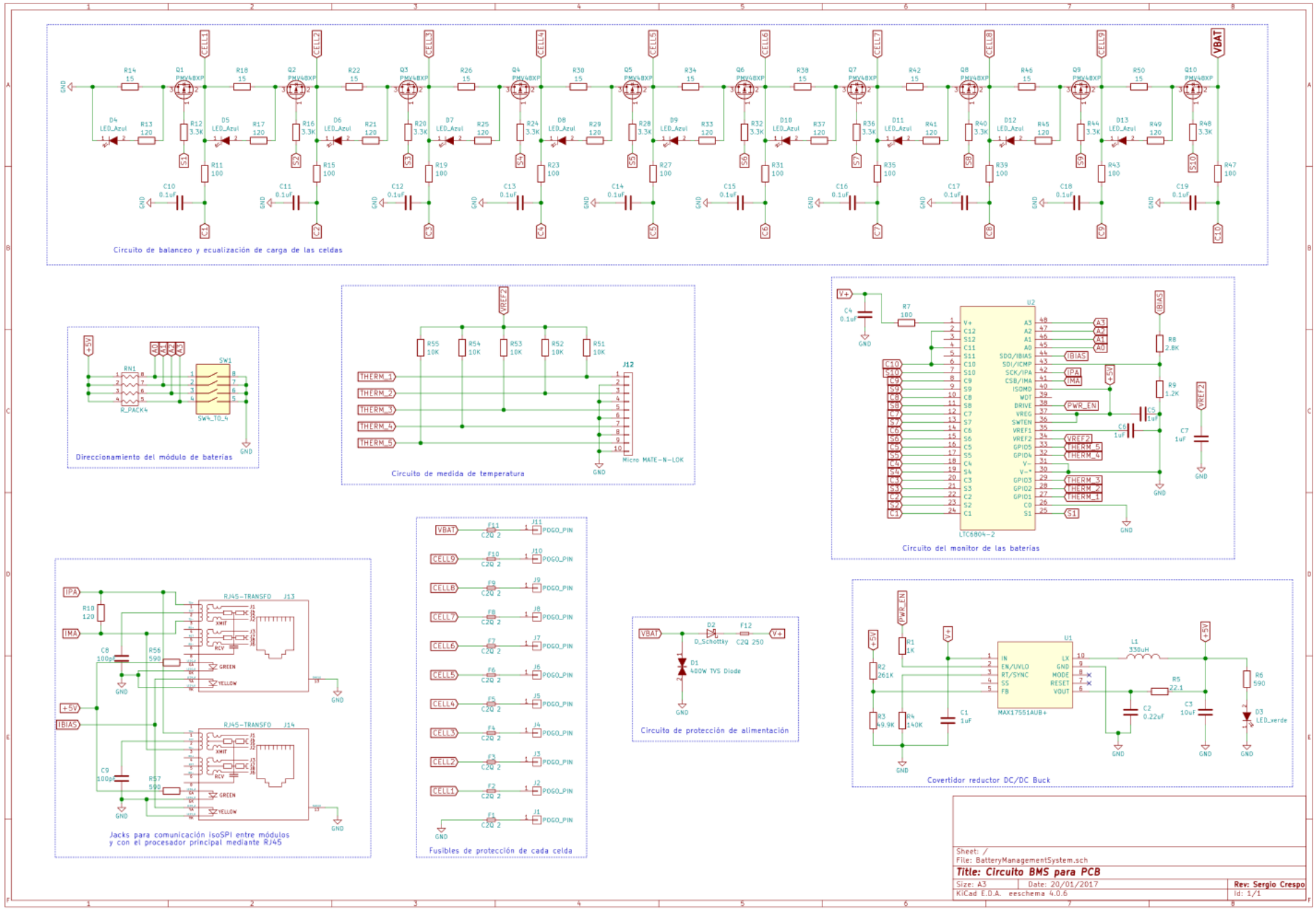
De cara al futuro, la idea es poder usar este mismo circuito, pero multiplicado por un número finito de *stacks* en serie, de modo que se consiga la tensión suficiente para alimentar el motor eléctrico de un automóvil, con los debidos cambios que haya que realizar en el código del microcontrolador, sistemas de comunicación añadidos como podría ser un módulo bluetooth por el cual se puedan comunicar el controlador y un ordenador, dispositivo móvil, etc. Además de otros problemas que se puedan presentar durante el desarrollo. O bien, utilizando el circuito como base, rediseñar la PCB de forma modular y universal para poder usarlo con distintos modelos de baterías con distintas dimensiones.

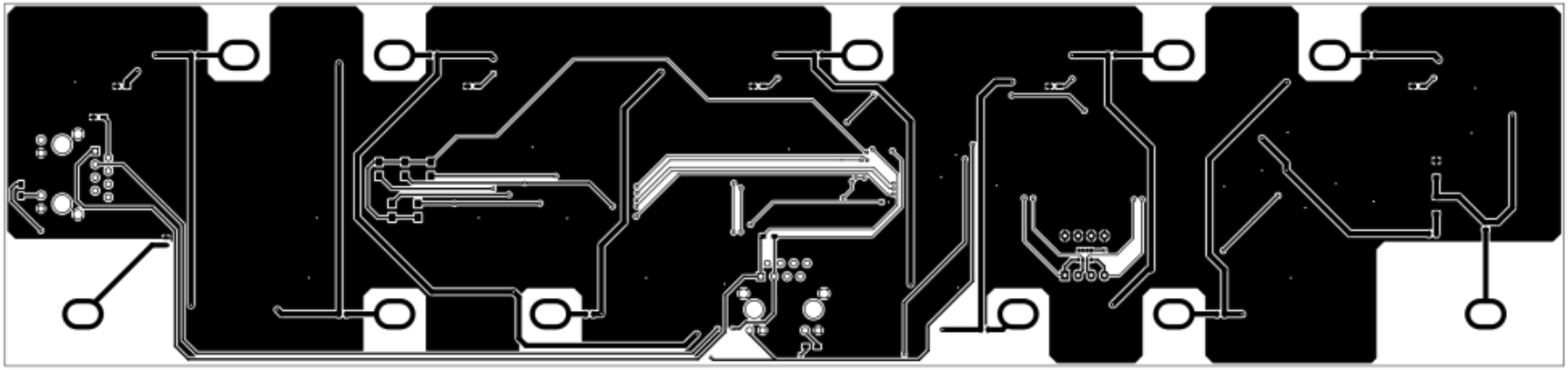
IX. PLANOS Y DIAGRAMAS

ÍNDICE DE PLANOS



Plano 1. Diagrama de flujo general

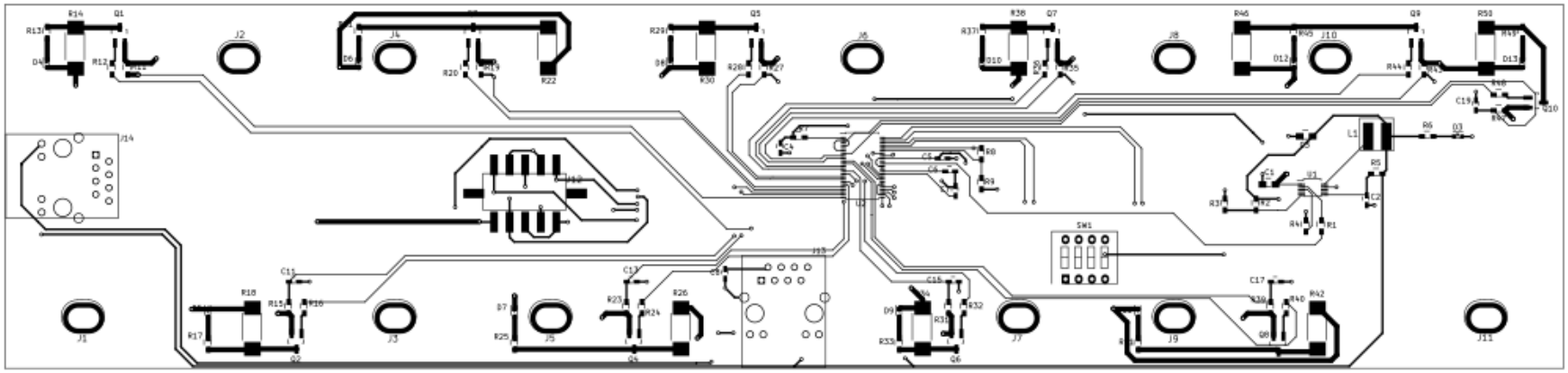

Circuito de balanceo y ecualización de carga de las celdas
Direccionamiento del módulo de baterías
Circuito de medida de temperatura
Circuito del monitor de las baterías
Jacks para comunicación isoSPI entre módulos
y con el procesador principal mediante RJ45
Fusibles de protección de cada celda
Circuito de protección de alimentación
Covertidor reductor DC/DC Buck
Sheet: /
File: BatteryManagementSystem.sch
Title: Circuito BMS para PCB
Size: A3
Date: 20/01/2017
KiCad E.D.A. eeschema 4.0.6
Rev: Sergio Crespo
Id: 1/1



Sheet: /
File: esp_bottom_Cu.kicad_pcb

Title: Gerber bottom view

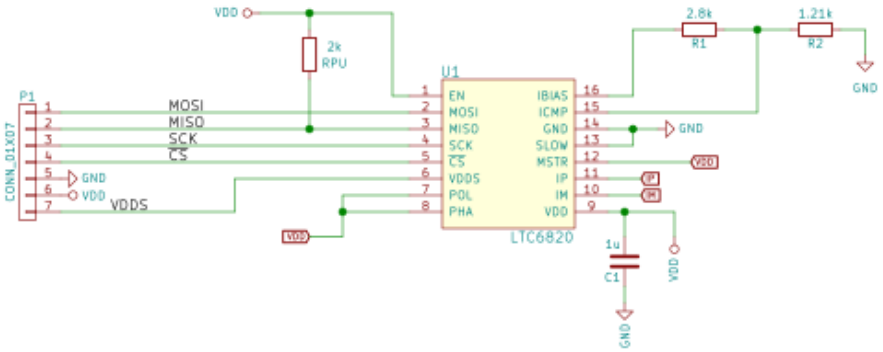
Size: A3	Date: 20/01/2017	Rev: Sergio Crespo
KiCad E.D.A. eeschema 4.0.6		Id: 1/1

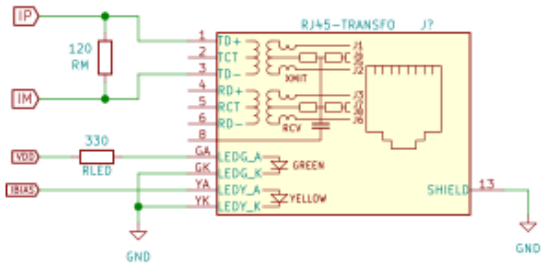


Sheet: /
File: [illegible]

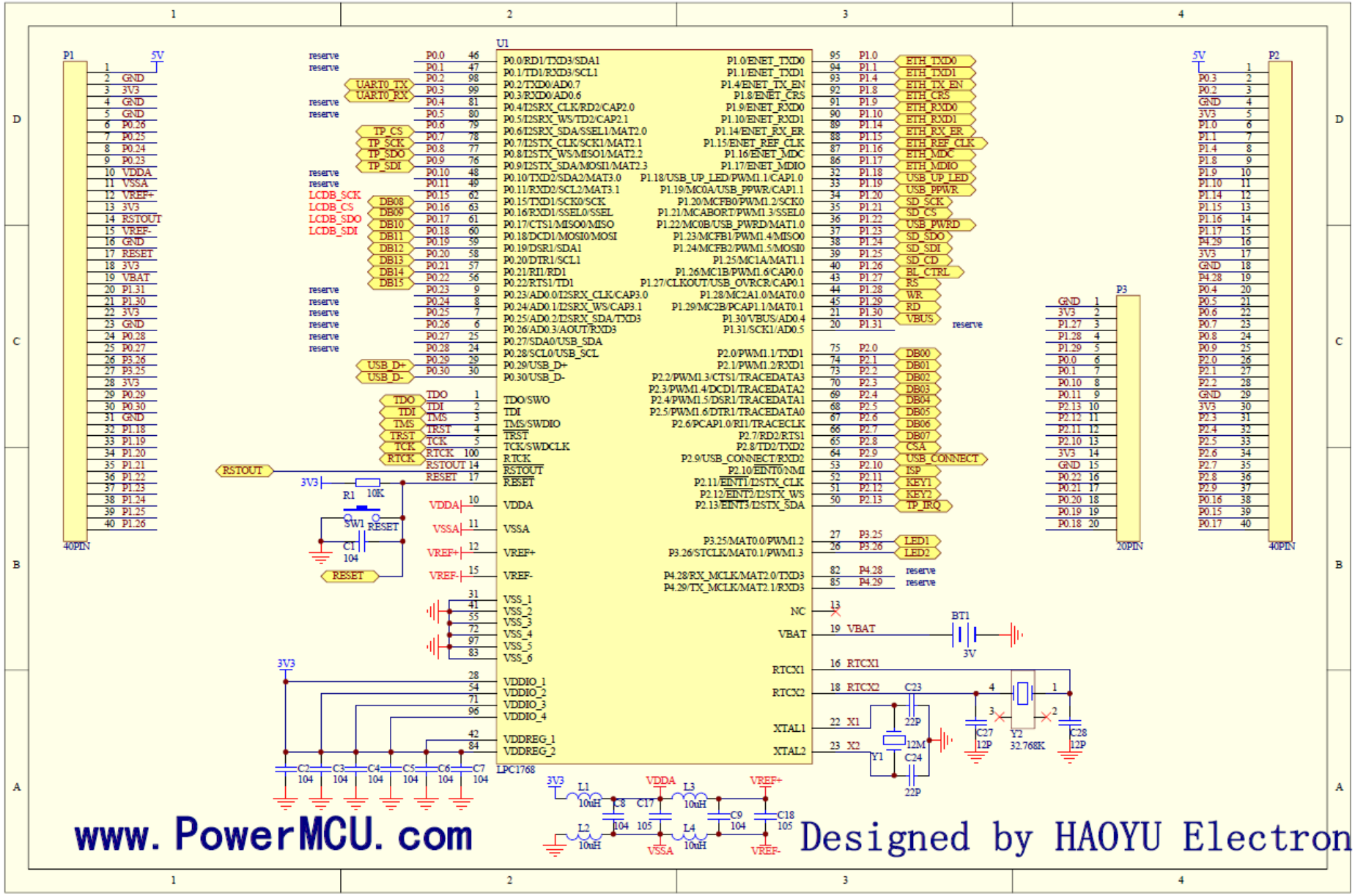
Title: Gerver top view

Size: A3	Date: 20/01/2017	Rev: Sergio Crespo
KiCad E.D.A. eeschema 4.0.6		Id: 1/1

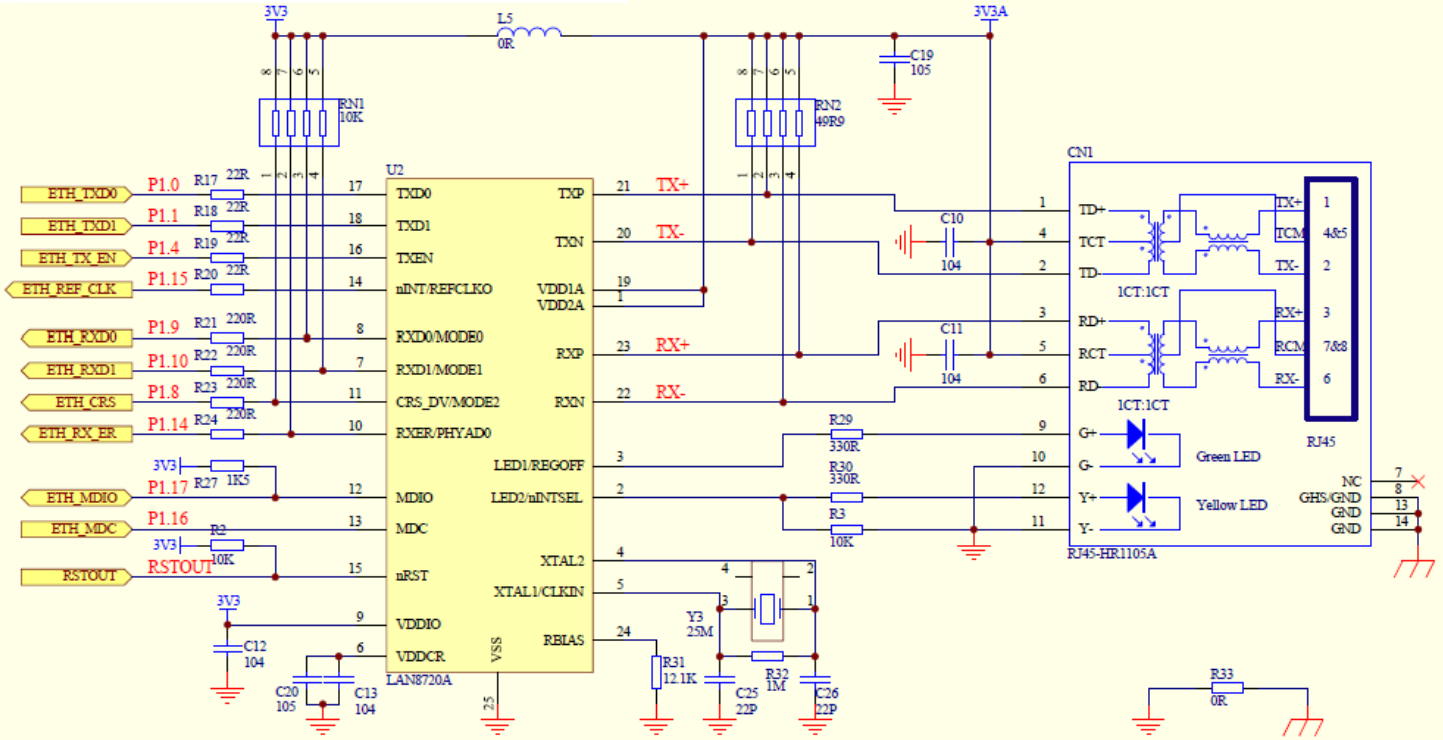




Sheet: /		
File: isoSPI.sch		
Title: isoSPI Module		
Size: A4	Date: 20/01/2017	**Rev: Sergio Crespo**
KiCad E.D.A. kicad 4.0.6		Id: 1/1

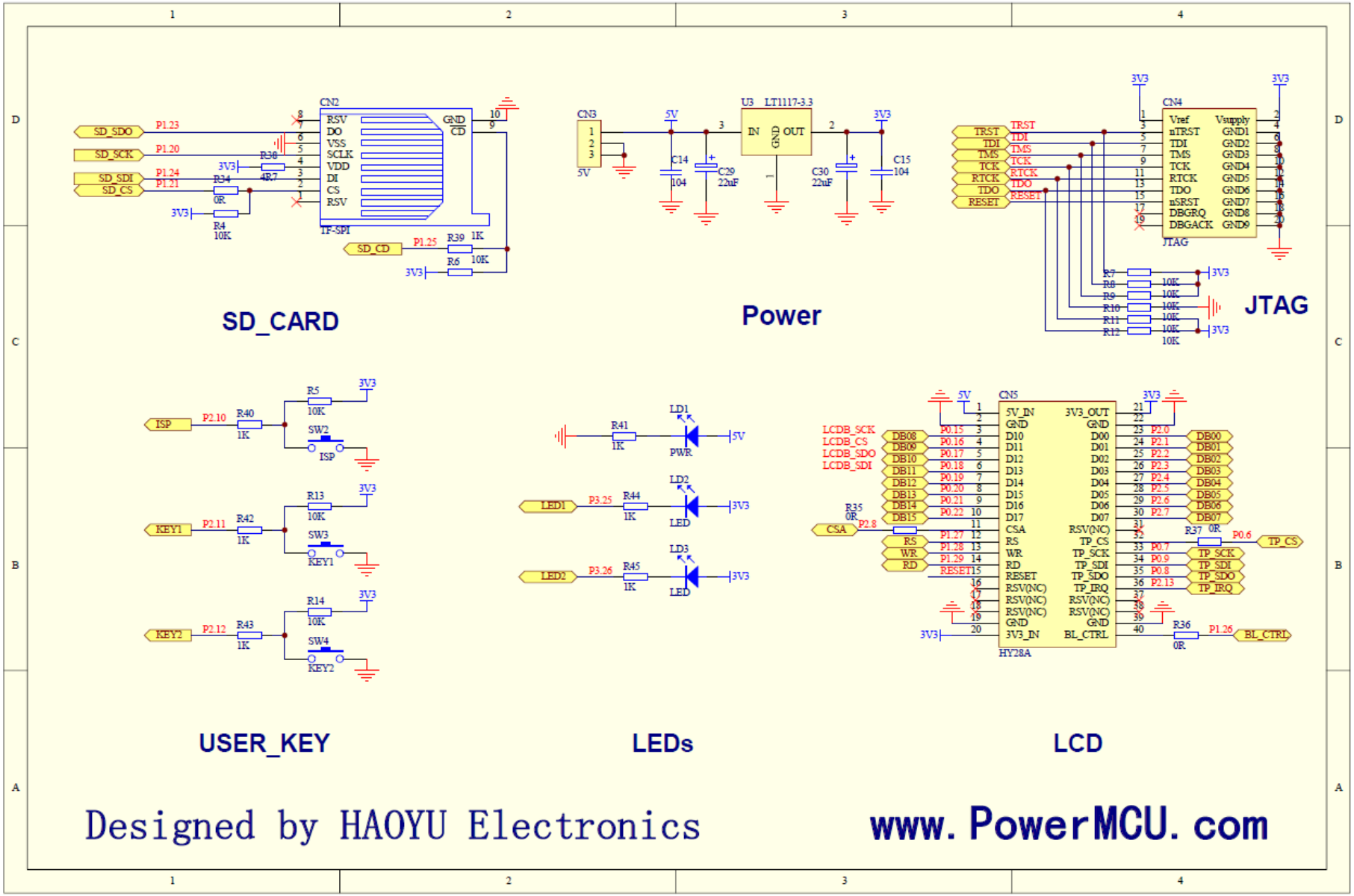

P1
40PIN
U1
LPC1768
P2
40PIN
P3
20PIN
www.PowerMCU.com
Designed by HAOYU Electron

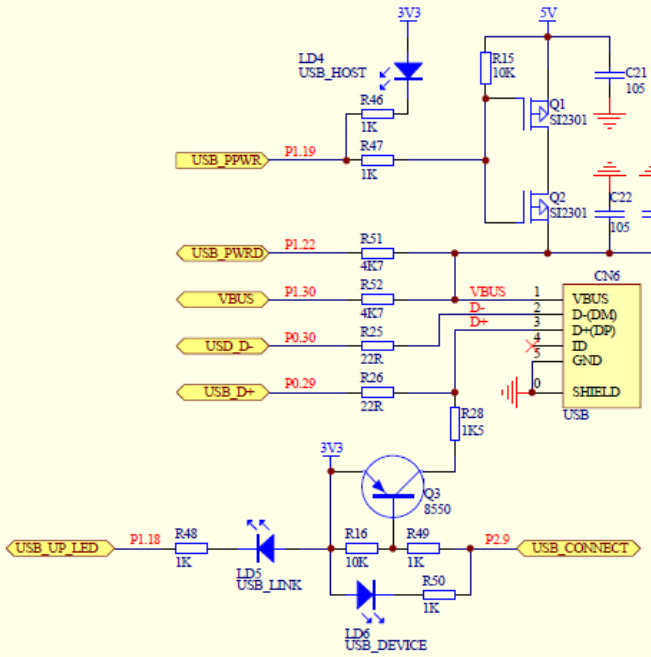
HAOYU ELECTRONICS



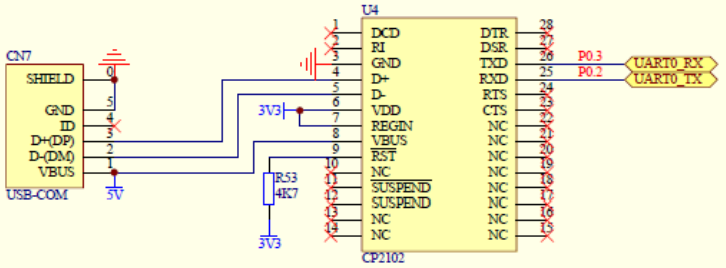
Designed by HAOYU Electronics

www.PowerMCU.com


SD_CARD
Power
JTAG
USER_KEY
LEDs
LCD
Designed by HAOYU Electronics
www.PowerMCU.com



USB Host/Device



USB to UART

Designed by HAOYU Electronics

www.PowerMCU.com

X. PLIEGO DE CONDICIONES

REQUISITOS DE HARDWARE

- Ordenador Intel Core i3-6xxx con las siguientes características mínimas:
    - CPU cuatro núcleos a 3,7 GHz.
    - Memoria RAM 4 GB.
    - Tarjeta gráfica, basta con la integrada en la CPU de Intel.
    - Disco duro de 500 GB.
- Disco duro externo de 1 TB para copias de seguridad.
- Tarjeta de desarrollo Mini-DK2 con pantalla táctil incorporada.
- Latiguillo RJ45 CAT 6, mínimo 10 cm y máximo 100 m de cable.
- Baterías de "*GP Batteries*" modelo "*GP45EVLF*", con tensión nominal 3.2 V y capacidad 45 Ah.
- Fuente de alimentación variable de tensión mínima 45 V y limitador de corriente de al menos 2 A.

REQUISITOS DE SOFTWARE

- Sistema operativo Windows 10 Home.
- IDE de ARM, Keil uVision V4 con compiladores de código C y C++.
- Librerías de uso del display GLCD y Touchpanel.
- Librerías de Firmware de ARM, que son LPC17xx CMSIS y CM3 (Cortex-M3).

- Librería Linduino LTC68042.cpp adaptada a Arduino, posteriormente editada.
- Editor de texto inteligente en C (o cualquier lenguaje de programación), Notepad++.
- Programa de diseño electrónico asistido, KiCAD EDA.

XI. PRESUPUESTO

En este apartado se describe el coste total del proyecto.

Se ha dividido en tres secciones, una dedicada al coste del *hardware*, otra al *software* y otra a la mano de obra.

COSTE DEL HARDWARE

En la *Tabla 1* se muestra el precio de cada elemento de la placa en la que se ha implementado el BMS y el precio total.

Concepto	Valor	Cantidad	Precio unidad	Precio
Resistencia	1K	1	0,09 €	0,09 €
Resistencia	261K	1	0,09 €	0,09 €
Resistencia	49.9K	1	0,09 €	0,09 €
Resistencia	140K	1	0,09 €	0,09 €
Resistencia	22.1	1	0,09 €	0,09 €
Resistencia	590	3	0,09 €	0,27 €
Resistencia	100	11	0,01 €	0,11 €
Resistencia	2.8K	1	0,09 €	0,09 €
Resistencia	1.2K	1	0,09 €	0,09 €
Resistencia	10K	5	0,01 €	0,05 €
Resistencia	120	11	0,01 €	0,11 €
Resistencia	3.3K	10	0,01 €	0,10 €
Resistencia	4R7	10	0,44 €	4,40 €
Array de riesistencias	4X10K	1	0,09 €	0,09 €
Switch	4 to 1	1	0,51 €	0,51 €
Condensador	1uF	1	0,23 €	0,23 €
Condensador	0.22uF	1	0,09 €	0,09 €
Condensador	10uF	1	0,17 €	0,17 €
Condensador	0.1uF	11	0,04 €	0,47 €
Condensador	1uF	3	0,17 €	0,51 €
Condensador	100pF	2	0,09 €	0,18 €

Diodo	TVS 400W DIODE	1	0,41 €	0,41 €
Diodo	Schottky	1	0,36 €	0,36 €
Diodo LED	LED azul	10	0,31 €	3,10 €
Diodo LED	LED verde	1	0,37 €	0,37 €
PMOS	PMV48XP	10	0,26 €	2,55 €
Fusible	C2Q 2	11	0,28 €	3,05 €
Bobina	330uH	1	0,61 €	0,61 €
Convertidor DC-DC	MAX17551AUB+	1	2,36 €	2,36 €
BMS	LTC6804-2	1	16,82 €	16,82 €
Conector	Jack RJ45	2	3,92 €	7,84 €
Conector	Female Connector	1	2,50 €	2,50 €
Conector	Male Connector	1	0,58 €	0,58 €
Crimp	contact rcpt	10	0,14 €	1,36 €
PCB	5pcs (min)	1	14,00 €	14,00 €
Total				65,84 €

Tabla 1. Coste placa BMS

En la

Concepto	Valor	Cantidad	Precio unidad	Precio
Traductor protocolos	LTC6820	1	4,36	4,36
Conentor	Jack RJ45	1	3,92	3,92
Condensador	100 nF	2	0,21	0,42
Resistencia	2K8	1	0,09	0,09
Resistencia	1K2	1	0,09	0,09
Resistencia	2K	1	0,09	0,09
Resistencia	120	1	0,09	0,09
PCB (prototipado)	5pcs (5x7 cm)	1	3,07	3,07
Total				12,13

Tabla 2 se muestra el precio de la placa que sirve de traducción de protocolos basada en el LTC6820.

Concepto	Valor	Cantidad	Precio unidad	Precio
Traductor protocolos	LTC6820	1	4,36	4,36
Conentor	Jack RJ45	1	3,92	3,92

Condensador	100 nF	2	0,21	0,42
Resistencia	2K8	1	0,09	0,09
Resistencia	1K2	1	0,09	0,09
Resistencia	2K	1	0,09	0,09
Resistencia	120	1	0,09	0,09
PCB (prototipado)	5pcs (5x7 cm)	1	3,07	3,07
Total				12,13

Tabla 2. Coste Placa isoSPI

En la *Tabla 3* se muestra el coste del resto del *hardware* necesario, suponiendo que no se disponga de él, como son un ordenador desde el que trabajar, el cual se ha estimado un precio medio incluyendo monitor, teclado y ratón, etc. Estos elementos harware, son bienes amortizables, el precio de desamortización se ha calculado en base a que el tiempo de amortización para equipos informáticos es de tres años, para equipos electrónicos de cinco años y para herramientas de diez años. Además, hay otros conceptos consumibles que, en cada caso, el porcentaje de amortización será el porcentaje usado sobre el total.

Concepto	**Precio**	**Cantidad amortizada**	**Porcentaje amortización**	**Precio desamortización**
PC Intel Core i3-6xxx	700,00 €	1 año	33%	231,00 €
Mini-DK2	32,30 €	1 año	20%	6,46 €
USB-MiniJTAG	11,66 €	1 año	20%	2,33 €
Cable (200 m.)	10,85 €	2 m.	1%	0,11 €
Juego de alicates de electrónica	11,90 €	1 año	10%	1,19 €
Jumpers macho-hembra (40)	1,25 €	1 año	10%	0,13 €
Soldador	42,90 €	1 año	20%	8,58 €
Rollo de estaño (100 g.)	16,19 €	10 g.	10%	1,62 €
Pinzas para soldar	1,42 €	1 año	10%	0,14 €
Flux (15 ml.)	8,21 €	5 ml.	33%	2,71 €
Total	836,68 €			254,27 €

Tabla 3. Coste hardware general

COSTE DE SOFTWARE

Con el software sucede algo similar al apartado anterior, también son bienes amortizables y al tratarse de sistemas informáticos, el tiempo de amortización será de tres años. En la *Tabla 4* se muestra el precio del *software*. Hay que decir que para usar Keil, hace falta una licencia de pago, porque la versión de prueba es insuficiente para la carga del programa que se va a utilizar. Pero para saber lo que cuesta se necesita hacer una solicitud y en este caso, la licencia que se ha utilizado es gracias a la Universidad de Alcalá, que mientras se esté conectado a la red *eduroam*, se puede utilizar el Keil en su versión profesional.

Concepto	Precio	Cantidad amortizada	Porcentaje amortización	Precio desamortización
SO Microsoft windos 10 Home	135,00 €	1 año	33%	44,55 €
KiCAD EDA	0,00 €	1 año	33%	0,00 €
Keil uVision V4	0,00 €	1 año	33%	0,00 €
Microsoft Office 2016	149,00 €	1 año	33%	49,17 €
Total	284,00 €			93,72 €

Tabla 4. Coste del software

COSTE TOTAL

Visto todo lo anterior y considerando un coste de ingeniero junior de 30 €/h incluyendo gastos por seguridad social y otros impuestos de contratación. Siendo la jornada completa de 8 horas, 5 días a la semana y que un año tiene aproximadamente 44 semanas, el total de horas en un año asciende a 1760 horas. Por tanto, el coste total del proyecto ascenderá, según la *Tabla 5*, a los 64.468,75 €.

Concepto	**Coste**
Placa BMS	119,84 €
Placa isoSPI	12,13 €
Otro hardware	254,27 €
Software	93,72 €
Coste de ingeniero	52.800,00 €
Subtotal sin IVA	53.279,96 €
IVA (21%)	11.188,79 €
Total	**64.468,75 €**

Tabla 5. Coste total del proyecto

XII. MANUAL DE USUARIO

La tarjeta Mini-DK2 debe alimentarse con una fuente externa como podría ser una pila de 9 V, o con el cable USB a un convertidor AC-DC, (el clásico cargador de móvil, por ejemplo). Además, deberá conectarse a la placa basada en el LTC6820, con los pines de alimentación de 5 V a V_{DD}, el de 3,3 V a V_{DDS}, GND y los pines de comunicación SPI (SCK, CS, MOSI y MISO).

La placa basada en el LTC6820 debe conectarse a la placa BMS, a través de un cable Ethernet CAT 6, que proporciona aislamiento entre el maestro y el esclavo de la comunicación gracias a los transformadores del RJ45.

Y la placa BMS, debe atornillarse a las celdas de la batería que va a monitorizar, atendiendo a cuál es el conector de masa (J1 en la placa) y cuál es el conector de tensión máxima o V+ (J11 en la placa).

FUNCIONAMIENTO Y ENVÍO DE COMANDOS

Una vez realizadas las conexiones, hay que saber el funcionamiento de los menús del GUI creado.

El menú inicial que se encuentra en la pantalla al encender es el menú principal, representado en la *Figura 31*. Se ha representado este menú con el botón de cargando en *on* y en *off*. En el cual se tienen cuatro botones (Status Menu, Measure Menu, Address Menu y el botón indicador de si se está cargando o no).

El botón indicador de carga sirve para adquirir cada segundo de manera periódica la tensión en cada celda, esto existe para automatizar el proceso de carga sin necesidad de estar pendiente de la misma. Por otro lado, si cargando está en *off*, el sistema volverá al modo de bajo consumo si no se le envía ningún comando.

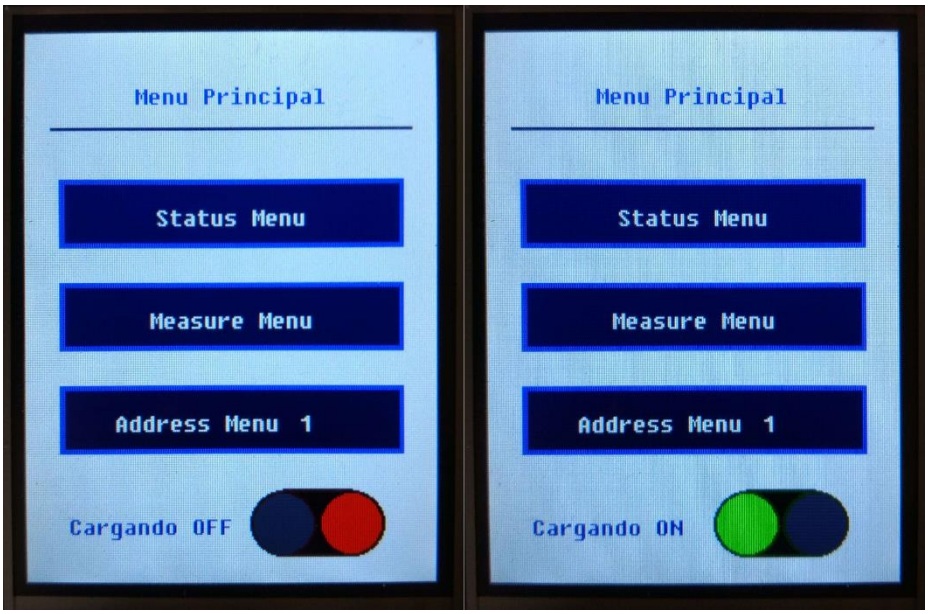


Figura 31. Menu principal

Si se pulsa en el botón *Status Menu*, el uC enviará al BMS los comandos de configuración, tanto de lectura, para poder mostrar en la pantalla el estado de las, como escritura, para establecer valores como el de *overvoltage* o el bit *REFON*, explicados en la memoria de este trabajo. El resultado es el que se muestra en la *Figura 32*.

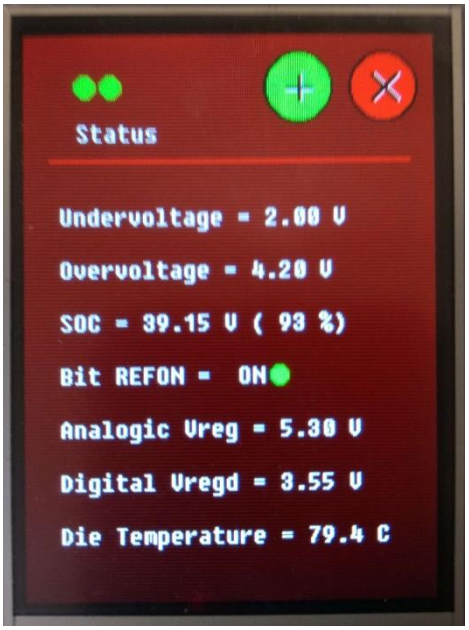


Figura 32. Status menu

Los indicadores verdes de la parte superior izquierda de la figura significan que se ha recibido el PEC (*Packet Error Code*) de los comandos RDCFG y RDSTAT correctamente, de lo contrario aparecerían en rojo. Por otro lado, en la parte superior derecha tenemos dos botones, uno verde con una cruz, que es el botón refrescar, para volver a enviar los comandos, para comprobar si ha cambiado algo o no y otro botón rojo con un aspa, que es el botón para regresar al menú principal.

Si se pulsa el botón Measure Menu, el uC enviará al BMS los comandos de adquisición de medidas de tensión y temperatura. En el display se mostrarán dichos valores de cada celda, numeradas del uno al diez. El resultado es el de la *Figura 33*.

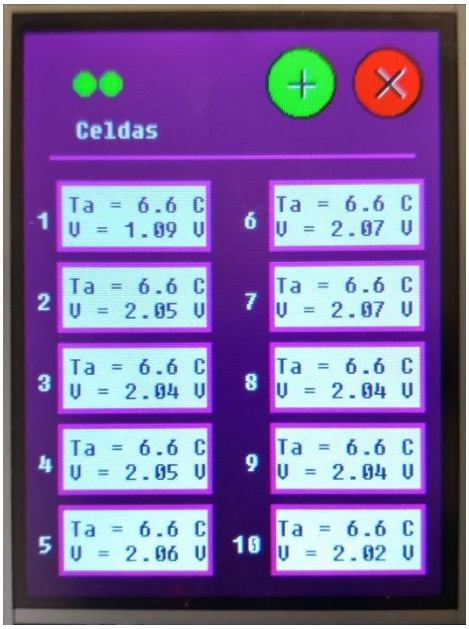


Figura 33. Measure menu

En este caso los indicadores de PEC, son de los comandos RDCV y RDAUX.

Por último, está el botón Address Menu, que al pulsarlo no envía ningún comando al BMS, sino que, sirve para establecer el número de *stacks* de baterías que se tiene. Esto hace que internamente le asigne una dirección a cada *stack*, si se tiene uno, la dirección será 0000, si se tienen dos, las direcciones serán 0000 y 0001. Estas direcciones están representadas en binario y deben coincidir con las determinadas por *hardware* mediante el *switch* que hay en la placa del BMS. En la *Figura 34* se representa este menú y sus diferencias con los anteriormente descritos se detallarán a continuación.

Figura 34. Address menu

Como se puede observar, hay un teclado numérico y en la parte superior izquierda hay dos dígitos que representan el número de *stacks* que estamos escribiendo, uno de ellos está resaltado con fondo negro, que es el dígito que se está editando, se puede cambiar al dígito de la izquierda pulsando el botón rojo oscuro que aparece en la parte inferior izquierda y volver a seleccionar el dígito de la derecha pulsando el botón verde que hay en la parte inferior derecha. Independientemente del número que se escriba, la variable total_ic nunca será mayor que dieciséis, puesto que, con cuatro bits la dirección más alta que se puede obtener es quince (total_ic – 1).

XIII. BIBLIOGRAFIA

[1] Documentation | KiCad EDA, 2017. *Kicad-pcb.org* [online].

[2] DAOWD, MOHAMED, ANTOINE, MAILIER, OMAR, NOSHIN, LATAIRE, PHILIPPE, VAN DEN BOSSCHE, PETER and VAN MIERLO, JOERI, 2014, Battery Management System—Balancing Modularization Based on a Single Switched Capacitor and Bi-Directional DC/DC Converter with the Auxiliary Battery. *Energies*. 2014. Vol. 7, no. 5, p. 2897-2937. DOI 10.3390/en7052897. MDPI AG

[3] *LTC6804-1/LTC6804-2 Multicell Battery Monitors*. (2014) (2nd ed.). McCarthy Blvd., Milpitas. Retrieved from http://www.linear.com/

[4] *Convertir SVG a DOC (WORD) (Online y Gratis) — Convertio*. (2017). *Convertio.co*. Retrieved 23 June 2017, from https://convertio.co/es/svg-doc

[5] *isoSPI Isolated Communications Interface*. (2013) (1st ed.). McCarthy Blvd., Milpitas. Retrieved from http://www.linear.com/

[6] *LPC17xx User manual*. (2017) (2nd ed.). Retrieved from http://www.nxp.com/

[7] Navarro, K. (2017). *¿Cómo funciona el protocolo SPI? | Panama Hitek*. *Panama Hitek*. Retrieved 23 June 2017, from http://panamahitek.com/como-funciona-el-protocolo-spi

[8] *Testing BMS | Scienlab*. (2017). *Scienlab.com*. Retrieved 23 June 2017, from http://www.scienlab.com/test-solutions/testing-bms

[9] *Transistor de efecto de campo metal-óxido-semiconductor*. (2017). *Es.wikipedia.org*. Retrieved 23 June 2017, from https://es.wikipedia.org/wiki/Transistor_de_efecto_de_campo_metal-%C3%B3xido-semiconductor

[10] BARSUKOV, YEVGEN and QIAN, JINRONG, 2013, *Battery power management for portable devices*. Boston: Artech House.

[11] ANDREA, DAVIDE, 2010, *Battery management systems for large lithium-ion battery packs*. Boston: Artech House.

[12] HO, WILLIAM and JI, PING, 2007, *Optimal production planning for PCB assembly*. London : Springer.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA SUPERIOR

Universidad de Alcalá