Proyecto Fin de Carrera

Ingeniería de Telecomunicación

Prototipo de electrocardiógrafo portátil

Autor: Antonio Peralta Sánchez

Tutor: Fernando Muñoz Chavero

Departamento de Ingeniería Electronica
Escuela Técnica Superior de Ingeniería

Universidad de Sevilla
Sevilla, 2016

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Prototipo de electrocardiógrafo portátil

Autor:

Antonio Peralta Sánchez

Tutor:

Fernando Muñoz Chavero

Profesor titular

Departamento de Ingeniería Electronica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2016

Proyecto Fin de Carrera: Prototipo de electrocardiógrafo portátil

Autor: Antonio Peralta Sánchez
Tutor: Fernando Muñoz Chavero

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2016

El Secretario del Tribunal

Resumen

En este proyecto trata de crear un prototipo de electrocardiógrafo portátil, abarcando desde el diseño y fabricación de la PCB de adaptación y filtrado de la señal eléctrica proveniente del cuerpo humano hasta la preparación y programación de la placa de desarrollo Beaglebone Black Rev C. Esta placa contiene un procesador AM335x a 1GHz ARM Cortex-A8, al igual que dos PRUs o Programmable real-time unit, procesadores de alta velocidad (200MHz) y 32 bits con un ciclo único de acceso a un número elevado de pines de la placa y acceso total a la memoria interna y los periféricos. Este módulo será usado para leer el ADC de la placa y capturar la señal eléctrica del cuerpo humano, que luego será procesada para mostrar el resultado final.

Una vez capturada la señal se hará uso de Matlab para el filtrado final.

Índice

1 Introducción 9
1.1 Objetivo 9
2 Estado del arte 10
2.1 Sistemas embebidos 10
2.1.1 Arduino 10
2.1.2 Raspberry Pi 14
2.1.3 Beagleboard 17
2.1.4 BeagleBone 23
2.1.5 Conclusiones 31
2.2 Electrocardiograma y electrocardiógrafo 32
2.2.1 Electrocardiograma 32
2.2.2 Electrocardiógrafo 35
3 Diseño amplificador 41
3.1 Especificaciones 41
3.1.1 Señal de entrada 41
3.1.2 Señal de salida 42
3.1.3 Componentes 42
3.2 Diseño esquemático 44
3.3 Diseño PCB 50
3.4 Medidas experimentales 52
4 Placa de desarrollo 55
4.1 Preparación 55
4.1.1 Primeros pasos 55
4.1.2 Librerías necesarias 55
4.2 Programas 60
4.3 Experimentos 65
5 Conclusiones y líneas futuras 72
6 Referencias 73
7 Índice de Figuras 74

1 Introducción

1.1 Objetivo

La idea principal de este proyecto es el desarrollo de un prototipo de electrocardiógrafo portátil de bajo coste y fácil manejo.

Es necesaria la familiarización tanto con la parte médica como con la parte tecnológica para llegar al objetivo deseado. Por lo tanto, una de las tareas será adentrarse en el estudio de las características del electrocardiograma, y las señales eléctricas que lo componen.

Por otra parte, se realizará un estudio de la placa BeagleBone Black Rev C y sus distintos componentes para así conseguir una captura y procesado de la señal con la mayor eficacia posible y obtener todas las características necesarias para un prototipo exitoso. Esto conlleva adentrarse de manera muy específica en todos los módulos de los que está compuesta esta placa. Gracias a la arquitectura de ésta podremos separar la captura de la señal del hilo principal de ejecución de su sistema operativo de manera que obtengamos una precisión mucho mayor.

Se deberá diseñar una placa de adaptación de señal, y una aplicación software capaz de capturar la salida de dicha placa y su posterior procesado. Haremos uso de lenguajes tales como C y Python, de igual manera que se hará uso del software Matlab para el procesado final de la muestra.

2 Estado del arte

En este estado del arte nos adentraremos en dos temas diferentes, pero necesarios para el desarrollo del proyecto, las placas de desarrollo de hardware embebido y el electrocardiograma, tema en el que se incluirán los electrocardiógrafos, más concretamente aquellos de bajo coste, tanto diseño como procesamiento de las señales cardiacas.

2.1 Sistemas embebidos

Un sistema embebido(del inglés, embedded) también llamado sistema empotrado, hace referencia a que la electrónica o el sistema de control que lo compone se encuentra integrada en el sistema en sí, no pudiendo ser separada como podríamos hacer con un PC normal, algo que implica un compromiso con el tamaño y la eficiencia de este sistema.

Generalmente estos sistemas están diseñados para desempeñar funciones específicas de una manera óptima, al contrario de un ordenador personal(PC) que abarca un mayor número de funciones. Normalmente están integrados en un sistema heterogéneo mayor que pueden incluir partes mecánicas, eléctricas y/o electromecánicas.

En la actualidad, los sistemas embebidos forman parte de nuestra vida cotidiana, la mayoría de las veces de manera transparente al usuario final, como en elementos tan sencillos como microondas o sistemas más complejos como los sistemas de navegación de los automóviles o incluso en los sistemas de medición médicos.

En el presente documento nos centraremos en las plataformas de hardware embebidos, ya que una de ellas es la elegida para el desarrollo del proyecto. A continuación comentaremos algunas de las placas de desarrollo embebidas más utilizadas o más populares en la actualidad.

2.1.1 Arduino

Esta placa goza de una gran popularidad entre los usuarios de este tipo de hardware. Arduino es una plataforma de electrónica “open-source” o código abierto cuya filosofía es contar con hardware y software fáciles de usar, que proporcionan la posibilidad de realizar proyectos interactivos a cualquier persona de manera sencilla. Es digno de resaltar la palabra “cualquier” ya que podemos encontrar online cientos de proyectos desarrollados bajo esta plataforma, de manera que la información sobre esta es amplia y de fácil acceso.

Esta plataforma es hardware y software, algo que la diferencia de la mayoría de las placas y microcontroladores, ya que los entornos de desarrollo así como el lenguaje de programación de arduino y las placas en las que se ejecutan han sido desarrollados de la mano, lo que nos garantiza la compatibilidad y la sencillez de los desarrollos sobre esta.

Algo que diferencia esta placa sobre las demás es que ésta no es una computadora, sino un microcrontrolador, por lo que sobre ella no podrá ejecutarse un sistema operativo. Para hacerla funcionar deberemos utilizar su entorno de desarrollo, disponible tanto para Windows, Linux o Mac OS, y una vez desarrollado el código, cargarlo mediante conexión usb en la placa.

A continuación vamos a describir la placa y sus componentes un poco más en detalle. Nótese que describiremos la Arduino Uno.

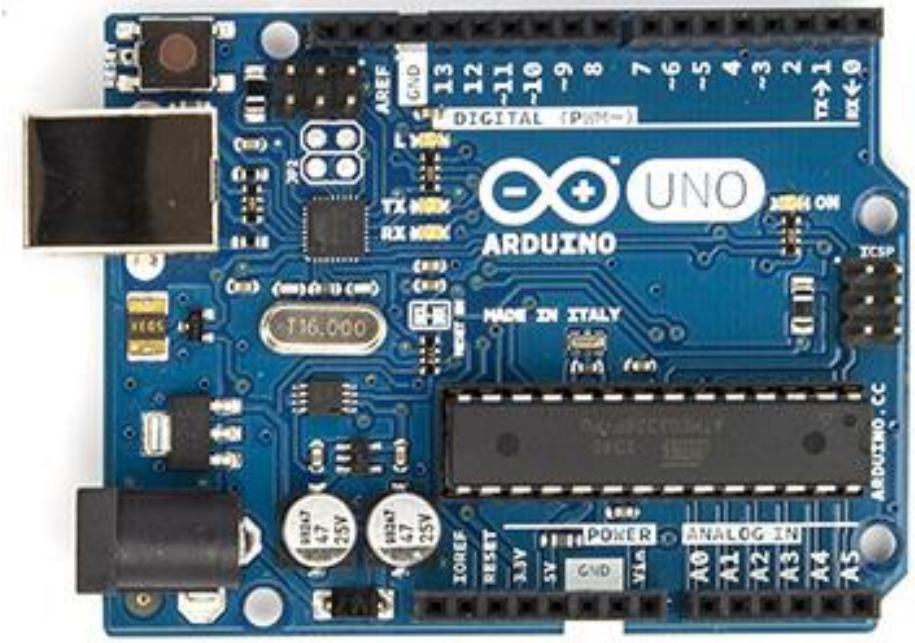

Fig. 1 Arduino Uno

2.1.1.1 Microcontrolador

Esta placa utiliza el ATmega328, un microcontrolador Atmel de 8 bits programable en flash, de alto rendimiento y bajo consumo perteneciente a la familia Atmel AVR 8-bit Microcontroller.

- Arquitectura:

Tiene una avanzada arquitectura RISC(*Reduced Instruction Set Computer*), lo que se traduce en instrucciones de tamaño fijo y presentadas en un reducido número de formatos, y en que solo las instrucciones de carga y almacenamiento acceden a la memoria de datos.

Dispone de 131 instrucciones la mayoría de un ciclo de reloj de duración, 32x8 registros de propósito general, y es capaz de ejecutar hasta 20 MIPS (*Millions of Instructions per Second*).

- Memoria:

Contiene segmentos de memoria de alta resistencia no volátiles. Más concretamente, dispone de 32 KBytes de memoria Flash de programa, 1 KByte de memoria EEPROM(Electrically Erasable Programmable Read-Only Memory) y 2KBytes de memoria SRAM(Static Random Access Memory).

- Características de los periféricos:
    - Dos timers/contadores de 8 bits con preescalador y comparador.
    - Un timer/contador de 16 bits con preescalador, comparador y modo captura.
    - Contador en tiempo real con oscilador separado.
    - Seis canales PWM.
    - 14 canales ADC de 10 bits.
    - USART serie programable.
    - Interfaz serie SPI maestro/esclavo.
    - Watchdog timer programable con oscilador.
    - Interrupciones y despertar según cambios en pines.
- Características especiales del microcontrolador:
    - Power-on Reset y detección de Brownout programable.
    - Oscilador calibrado interno.
    - Fuentes de interrupción tanto internas como externas.
- 23 entradas/salidas programables
- Voltaje soportado de 1.8 a 5.5V

2.1.1.2 Alimentación

La placa Arduino Uno puede ser alimentada via USB o con una fuente de alimentación externa y en caso de tener las dos conectadas la fuente es seleccionada automáticamente pudiendo ser la fuente externa tanto un alimentador AD-to-DC como una batería.

Los pines de salidas dan un voltaje de 5V, configurados por un regulador interno a la placa.

2.1.1.3 Memoria

No contiene módulos de memoria a parte de la memoria del microcontrolador.

2.1.1.4 Entradas y salidas

La placa Arduino Uno dispone de 14 pines digitales, los cuales pueden ser configurados tanto de entrada como de salida, operando a 5V. Cada pin puede proporcionar o recibir 20 mA como recomendación, disponiendo todos ellos de una resistencia de pull-up interna, desconectada por defecto, de 20-50K ohmios. El valor máximo recomendado para no dañar el microcontrolador es de 40mA en cualquiera de los pines de entrada/salida.

Algunos de los pines de la placa tiene una función específica que pasaremos a describir a continuación:

- Comunicación serie: los pines 0(Rx) y 1(Tx) se utilizan para comunicaciones serie, y están debidamente conectados con los correspondientes pines del chip USB-to-TTL serie del ATmega
- Interrupciones externas: pines 2 y 3. Pueden ser configurados para activar una interrupción por umbral o cambio de valor.
- PWM: los pines 3, 5, 6, 9, 10 y 11 proporcionan salida PWM de 8 bits.
- SPI: los pines 10, 11, 12 y 13 soportan comunicaciones mediante el protocolo SPI(Serial Periferical Interface).
- LED: existe un led en el pin 13, estando encendido a nivel alto y apagado a nivel bajo.
- TWI: el pin A4 o SDA y el A5 o SCL soportan comunicaciones TWI(Two Wire Interface) o I2C(Inter integrated Circuit).

Arduino Uno también dispone de 6 entradas analógicas(A0 a A5) las cuales proporcionan 10 bits de resolución cada una. Por defecto miden de 0 a 5 voltios pero es posible cambiar dicho valor utilizando el pin AREF, el cual proporciona el voltaje de referencia para las entradas analógicas.

2.1.1.5 Comunicaciones

La placa Arduino Uno ofrece diversas facilidades para la comunicación entre ésta y un ordenador, otra placa, u otro microcontrolador. Tal como se ha comentado en el apartado anterior, el microcontrolador que contiene soporta comunicaciones serie pero, además de eso, podemos acceder a él por USB apareciendo en nuestro ordenador como un puerto COM más.

Está disponible para el usuario numerosas librerías tanto para comunicaciones serie, para usar el bus I2C o para la comunicación por SPI.

2.1.2 Raspberry Pi

Raspberry Pi es un ordenador de placa reducida, ordenador de placa única u ordenador de placa simple (SBC, Simple Board Computer) de bajo coste desarrollado en Reino Unido por la Fundacion Raspberry Pi, con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas. De hecho, el gran acierto de su creador, Eben Upton, fue la creación de un dispositivo de bajo coste que daba la posibilidad de mejorar las habilidades de programación y el entendimiento del hardware a un nivel básico y, gracias a su reducido tamaño y bajo precio dieron alas a entusiastas de la electrónica que desarrollaron proyectos para los que un simple microcontrolador, como puede ser Arduino, no bastaban.

Esta placa ha ido evolucionando mucho desde su creación, y han ido apareciendo distintos modelos con distintas características. En este documento nos centraremos en el modelo más actualizado y potente que actualmente ofrece la fundación, la Raspberry Pi 3 Modelo B.

Fig. 2 Raspberry Pi 3 Model B

La Raspberry es un dispositivo open-source, a excepción del chipset Broadcomm SoC (System on a Chip), el cual abarca gran cantidad de elementos principales de la placa

Este dispositivo si puede ser considerado una computadora, ya que dispone de microprocesador, GPU, es capaz de correr sistema operativo, desde un sistema Linux básico hasta una versión del actual Windows 10. Pasaremos a continuación a describir la placa más en detalle.

2.1.2.1 Chipset

El chipset de este modelo de Raspberry Pi es el Broadcom BCM2387, el cual cuenta con un procesador de cuatro núcleos de arquitectura ARM Cortex-A53 de 64 bits, a 1.2GHz y con 32KB de memoria cache de nivel 1 y 512KB de nivel 2, acompañado de nada menos que 1 GHz de memoria RAM LPDDR2. Dicho procesador soporta conectividad Wireless LAN 802.11 b/g/n y Bluetooth 4.1 . Es digno de mención, que el procesador de este modelo, tiene un rendimiento 10 veces superior al que traía el primer modelo de la placa.

La integración de conectividad inalámbrica WiFi y Bluetooth(de bajo consumo) en la propia placa es un avance que los usuarios demandaban desde hacía mucho tiempo, ya que antes debían disponer de dispositivos a parte para poder disfrutar de estas características o incluso hacer uso del puerto Ethernet, motivo por el cual esta placa perdía cierto atractivo.

La GPU(Graphics Processor Unit o Unidad de Procesamiento Gráfico) es de doble núcleo con co-procesador multimedia. Proporciona Open GL 2.0, para la creación y procesado de gráficos 2D y 3D, aceleración hardware OpenVG, para los gráficos 2D, y es capaz de reproducir vídeo a 1080p y 30 FPS(Frames per Seconds) con el códec H.264 .

El sistema operativo se inicia desde una tarjeta micro SD, pudiendo ser desde una versión Linux a una versión de Windows 10, tal como se comentó anteriormente.

Este chipset se alimenta mediante un puerto micro USB contenido en la placa a 5V1 y 2.5A.

Toda esta potencia, unida a su reducida dimensión(tan solo 85 x 56 x 17 mm) hacen de este dispositivo una gran herramienta para desarrollar proyectos cuya complejidad dependerá del interés del usuario.

2.1.2.2 Conectores

La placa dispone de un número elevado de conectores que pasaremos a enumerar a continuación:

- Puerto Ethernet: 10/100 BaseT.
- Salida de vídeo: Dispone de puerto HDMI(High-Definition Multimedia Interface) y conector RCA.
- Salida de audio: Dispone de una salida Jack de 3.5mm, una HDMI, y cuatro salidas USB 2.0 .
- GPIO(General Purpose Input Output o Entradas/salidas de propósito general): La placa dispone de cabezal de expansión de 40 pines.
- Cámara: Para conexión de cámara dispone de 15 pines MIPI(Mobile Industry Processor Interface) CSI-2 (Camera Serial Interface)

- Display: para la conexión de un display dispone de un puerto DSI(Display Serial Interface).

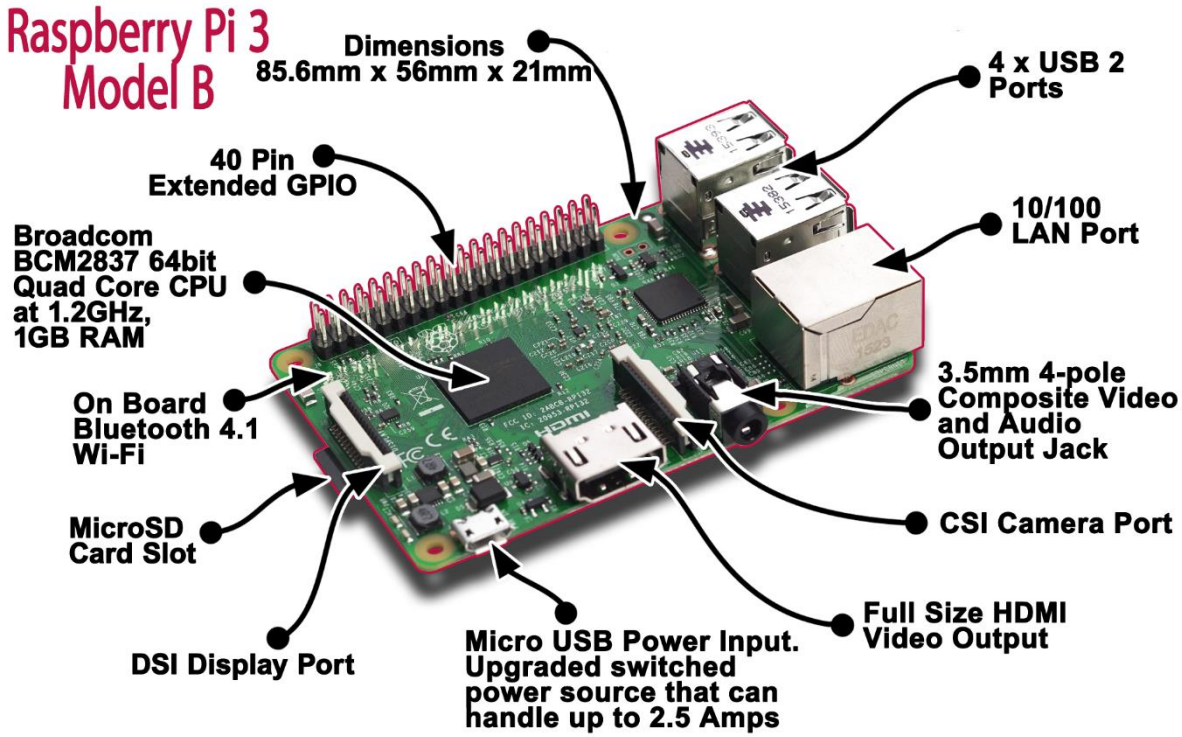

Fig. 3 Raspberry Pi 3 Model B

Desde la Fundación Raspberry Pi ofrecen pruebas de rendimiento al usuario para que pueda contrastar resultados según los modelos disponibles, en los que muestran la rapidez de esta última a nivel de procesador, procesos multihilo, acceso a los GPIO, utilización de los gráficos, etc.

2.1.3 Beagleboard

La fundación BeagleBoard.org es una organización sin ánimo de lucro con sede en Estados Unidos quenació con la idea de proporcionar un sistema para la enseñanza de software y hardware libre.

Pensada para usos en ámbitos universitarios, un grupo de personas entre las que se encontraban empleados de Texas Instrumentes, crearon el sistema embebido de código abierto, BeagleBoard. La fundación Beagelboard.org no recibe ningún beneficio económico por la venta de cada tarjeta, siendo esta una de las razones por las que resultan tan económicas. La ayuda para la producción original la proporcionó Digi–Key junto con Texas Instruments, pero actualmente la distribución de las tarjetas BeagleBoard se ha abierto a decenas de distribuidores en todo el mundo.

La BeagleBoard básicamente es una placa base de bajo costo que tiene un procesador de baja potencia ARM de la serie Cortex-A, producido por Texas Instruments. Estos procesadores son de la categoría “System on Chips” (SoCs), es decir, circuitos integrados que contienen todos los componentes de una computadora o sistema electrónico, especialmente diseñados para sistemas portátiles y tecnología móvil. Pensados inicialmente con la idea de permitir a las distribuciones de Linux mejorar la compatibilidad con los dispositivos ARM, actualmente y gracias a su éxito, BeagleBoard tiene el apoyo de numerosas distribuciones de Linux y el soporte por parte de casi todos los desarrolladores, como Ubuntu, QNX, Windows Embedded y Android.

La BeagleBoard está equipada con un conjunto mínimo de funciones para evitar elevar el coste de la placa innecesariamente. No obstante mediante la utilización de interfaces estándar, la BeagleBoard es altamente extensible pudiéndose añadir muchas características e interfaces. Pese a que los creadores de la BeagleBoard no tuvieron en mente hacer uso de las placas en productos finales, toda la información de diseño está disponible de forma libre y se puede utilizar como la base para la realización de otros productos. Como se comentó en la introducción, esta es una de las razones principales por las que se ha usado la BeagleBoard en este proyecto, la idea principal sería aprender a desarrollar aplicaciones en este hardware para luego poder crear un dispositivo que utilice como base el diseño hardware empleado en la BeagleBoard.

La fundación BeagleBoard.org tiene a disposición del usuario distintas placas de desarrollo dependiendo de las características y precio deseados. A continuación se muestran en una tabla los diferentes modelos disponibles.

	Seeed Studio BeagleBone Green	BeagleBoard.org BeagleBone Black	BeagleBoard.org BeagleBoard-xM	BeagleBoard.org BeagleBoard-X15
Processor	AM3358 ARM Cortex-A8	AM3358 ARM Cortex-A8	DM3730 ARM Cortex-A8	AM5728 ARM Cortex-A15
Maximum Processor Speed	1GHz	1GHz	1GHz	1.5GHz
Analog Pins	7	7	0	TBD
Digital Pins	65 (3.3V)	65 (3.3V)	53 (1.8V)	TBD
Memory	512MB DDR3 (800MHz x 16), 4GB on-board storage using eMMC, microSD card slot	512MB DDR3 (800MHz x 16), 2GB (4GB on Rev C) on-board storage using eMMC, microSD card slot	512MB LPDDR (333MHz x 32), microSD card slot	2GB DDR, 4GB on-board storage using eMMC, microSD card slot
USB	HS USB 2.0 Client Port, LS/FS/HS USB 2.0 Host Port	HS USB 2.0 Client Port, LS/FS/HS USB 2.0 Host Port	4 Port LS/FS/HS USB Hub, HS USB 2.0 OTG Port	SS USB 3.0 Host, HS USB 2.0 OTG Port (TBD)
Video	cape add-ons	microHDMI, cape add-ons	DVI-D (via HDMI connectors), S-Video	HDMI, TBD
Audio	cape add-ons	microHDMI, cape add-ons	3.5mm stereo jack	3.5mm stereo jack
Supported Interfaces	4x UART, 8x PWM, LCD, GPMC, MMC1, 2x SPI, 2x I2C, A/D Converter, 2xCAN Bus, 4 Timers	4x UART, 8x PWM, LCD, GPMC, MMC1, 2x SPI, 2x I2C, A/D Converter, 2xCAN Bus, 4 Timers	McBSP, DSS, I2C, UART, LCD, McSPI, PWM, JTAG, Camera Interface	TBD
MSRP	$39	$49	$149	$249

Fig. 4 Modelos BeagleBoard.org

La fundación ha desarrollado diferentes placas con las cuales ha ido mejorando su diseño. En este documento pasaremos a describir el modelo más potente disponible para el usuario, la BeagleBoard-X15 .

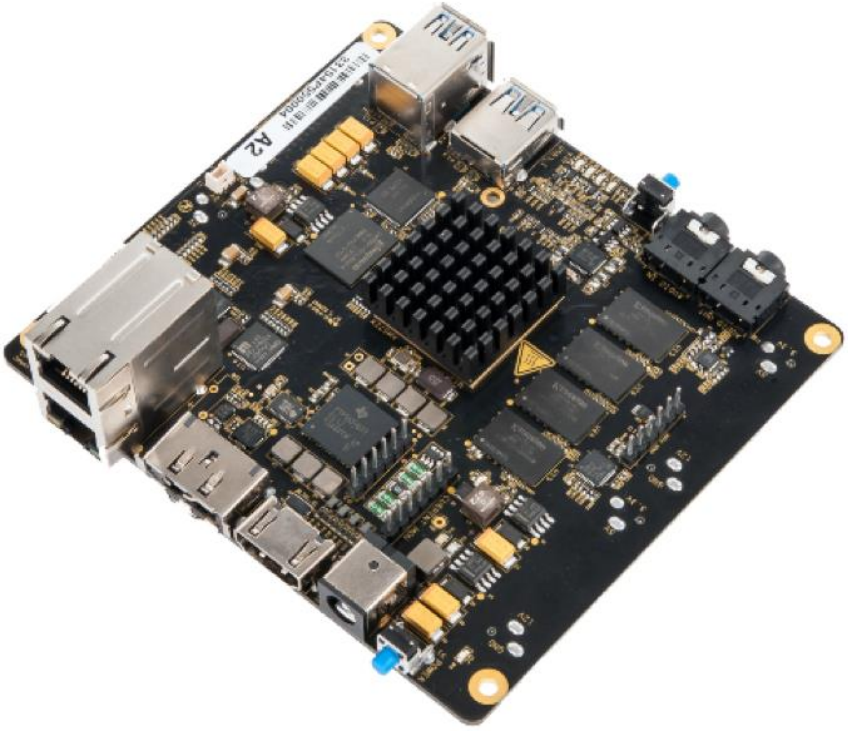

Fig. 5 BeagleBoard-X15

En las siguientes imágenes podemos observar la parte superior y la parte inferior de la placa con todos sus componentes diferenciados.

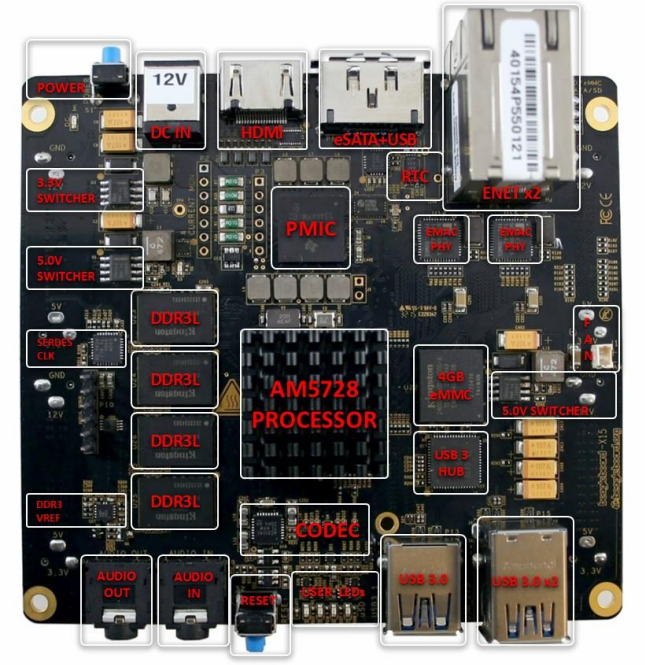

Fig. 6 BeagleBoard-X15 TOP

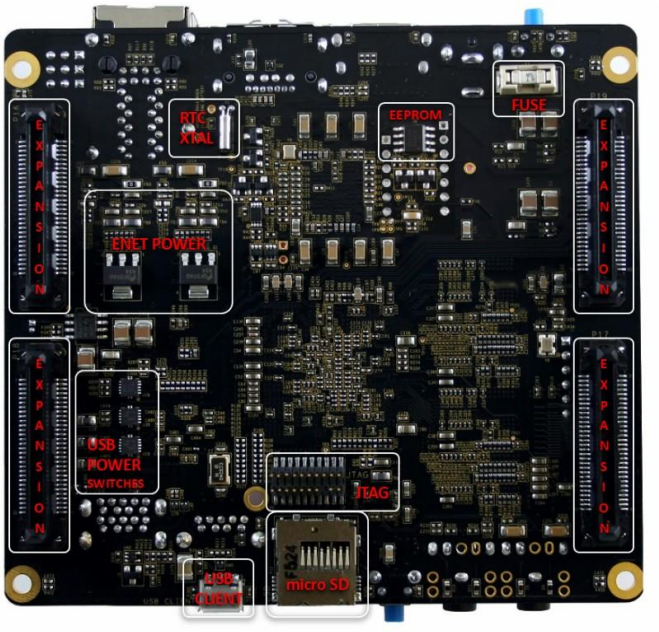

Fig. 7 BeagleBoard-X15 BOTTOM

2.1.3.1 Procesador

La BeagleBoard-X15 consta de un procesador firmado por Texas Instruments(TI) Sitara AM5728 que integra a su vez un microprocesador de propósito general ARM Cortex-A15 de doble núcleo a 1.5GHz a la misma vez que 2 DSPs(Digital Signal Processor) C66x de punto flotante a 700MHz.

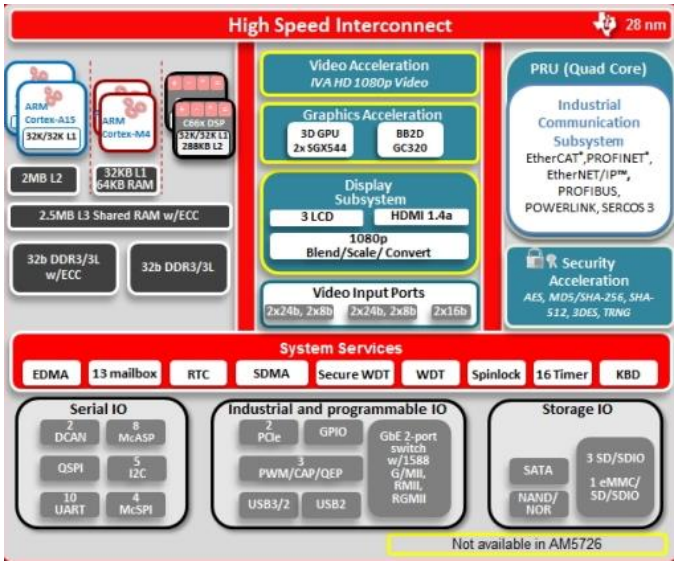

Fig. 8 Diagrama de Bloques AM5728

El procesador AM5728 es el encargado de gobernar cada parte de la placa. Es un sistema SoC(System on Chip) igual que el utilizado en la placa original pero mucho más avanzado.

Los sistemas SOC integran en un único chip toda la funcionalidad de un sistema digital completo basado en microprocesador. Esto incluye al menos uno o varios núcleos o unidades de proceso (CPUs, *Central Processing Units*), memoria y periféricos. También se pueden encontrar bloques de lógica reconfigurable y bloques analógicos integrados en el mismo chip.

El hecho de integrar todo el sistema, proporciona una gran reducción de tamaño y aumento de la fiabilidad, ya que todos los componentes están sometidos al mismo y único proceso de fabricación, lo que permite reducir costes y tiempo de diseño de los sistemas embebidos finales. Además, la propagación de señales es mejor a nivel de silicio que a nivel de PCB (*Printed Circuit Board*), mejorando así la capacidad de comunicación de los subsistemas internos. El encapsulado que permite esa integración mencionada es de tipo POP (*Package on Package*), donde la memoria está soldada sobre el propio chip, de modo que no es posible visualizarlo directamente. Con este tipo de encapsulado se consigue reducir el número de pistas necesarias para la conexión de las memorias optimizando el espacio en la PCB.

Los procesadores de aplicaciones Sitara ARM AM572x están fabricados para alcanzar las necesidades de procesamiento de los productos embebidos modernos. Estos dispositivos proporcionan altos niveles de procesamiento a través de la máxima flexibilidad y total integración de una solución de procesador mixto.

La programabilidad es proporcionada por su microprocesador de doble núcleo ARM Cortex-A15 de estructura RISC con la extensión Neon, y sus dos DSPs de punto flotante. Permite a los desarrolladores mantener el control de las funciones a parte de otros algoritmos programados en los DSPs y los coprocesadores, lo que reduce la complejidad del software.

Adicionalmente a esto, TI proporciona un completo conjunto de herramientas de desarrollo para el ARM y los DSPs, incluyendo compiladores para código C, un optimizador para el DSP y una interfaz de depuración donde examinar la ejecución del código.

Pasamos a nombrar los componentes del SoC más detalladamente:

- Subsistema microprocesador ARM de doble núcleo Cortex-A15.
- Dos DSPs C66x VLIW(Very Long Instruction Word) de punto flotante
    - Código compatible con C67x y C64x+
    - Capaz de realizar lo equivalente a 32 multiplicaciones 16 x 16 bits de punto fijo por ciclo
- 2.5MB de RAM nivel 2 On-Chip(No un módulo externo).
- Dos módulos EMIF(External Memory InterFace) DDR3/DDR3L.
    - Soporta hasta DDR3-1066.

    - Hasta 2GB por módulo EMIF.
- Coprocesador ARM de doble núcleo Cortex-M4
- IVA-HD Subsystem (Codec)
- Display Subsystem
    - Soporta video Full-HD 1920 x 1080p a 60 fps.
    - Multiples entradas y una salida de video.
    - Gráficos 2D y 3D.
    - Controlador de display con DMA(Direct Memory Access)
    - Codificador HDMI: HDMI 1.4ª y DVI 1.0
- 2 PRU-ICSS (Programmable Real-Time Unit and Industrial Communication SubSystem) de doble núcleo.
- Subsistema de aceleración de gráficos 2D.
    - Vivante GC320 Core
- VPE (Video Processing Engine).
- GPU(Graphics Processor Unit) de 3D de doble núcleo PowerVR SGX544 .
- Hardware de aceleración de encriptados
    - AES, SHA, RNG, DES y 3DES
- Tres modulos VIP(Video Input Port)
- GPMC(General-Purpose Memory Controller)
- Controlador EDMA(Enhanced Direct Memory Access).
- Dos puertos Gigabit-Ethernet(GMAC)
- 16 timers de 32 bits de propósito general.
- 5 puertos I2C.
- 1 Timer Watchdog de 32 bits
- 10 módulos UART/IrDA/CIR
- Interfaz Quad SPI (QSPI)
- Interfaz SATA gen2
- MCASP (Multichannel Audio Serial Port)
- USB 3.0
- USB 2.0
- PCI-Express
- Módulos de DCAN(Dual Controller Area Network)
    - Protocolo CAN 2.0B
- Hasta 247 Entradas/Salidas de propósito general(GPIO)
- Gestión de Encendido, reset y del reloj
- Tecnología CTools de depuración On-Chip
- Tecnología CMOS de 28 nm

Como se puede comprobar en las especificaciones, el procesador de la BeagleBoard-X15 es de una alta potencia y ofrece numerosas posibilidades a un precio muy competitivo, lo que hace que la funcionalidad de la placa abarque un amplio abanico de opciones.

2.1.3.2 Memoria

La dispone, a parte de la memoria indicada en el procesador, de 2GB de memoria RAM DDR3, al igual que 4GB de memoria flash on-board eMMC de 8 bits.

2.1.3.3 Conectividad

La BeagleBoard-X15 dispone de diferentes conectores que la dotan de una amplia conectividad.

- 2 Puertos Gigabit Ethernet
- 3 Puertos USB 3.0
- 2 Puertos USB 2.0
- eSATA(external Serial Advanced Technology Attachment)
- Salida de video HDMI
- Ranura para tarjeta microSD
- Entrada/salida de audio estéreo
- 4x60 pines con PCI, LCD, mSATA, UARTs, SPI/I2C/CAN

2.1.4 BeagleBone

La Fundación BeagleBoard.org pone a disposición del usuario, como se ha comentado anteriormente, distintas placas según las prestaciones deseadas, a las cuales por supuesto va unido su precio.

Bajo el nombre de BeagleBone, existen dos modelos distintos, BeagleBone Green y BeagleBone Black. En este documento describiremos la BeagleBone Black, y más concretamente su modelo BeagleBone Black rev(revision) C.

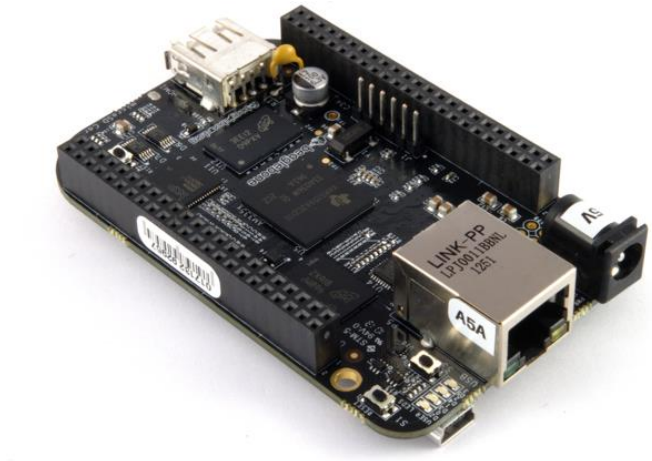

Fig. 9 BeagleBone Black Rev C

En la página web de la Fundación BeagleBoard.org describen la placa como una plataforma de desarrollo con gran soporte y de bajo coste para desarrolladores y gente afín a estas tecnologías. Señalan la rapidez de esta para lanzar el sistema operativo Linux(lo inicia en menos de 10 segundos) y fácil de acceder a ella, gracias a un simple cable USB.

Antes de comenzar con la descripción detallada se muestra un diagrama de bloques de los componentes clave de la placa.

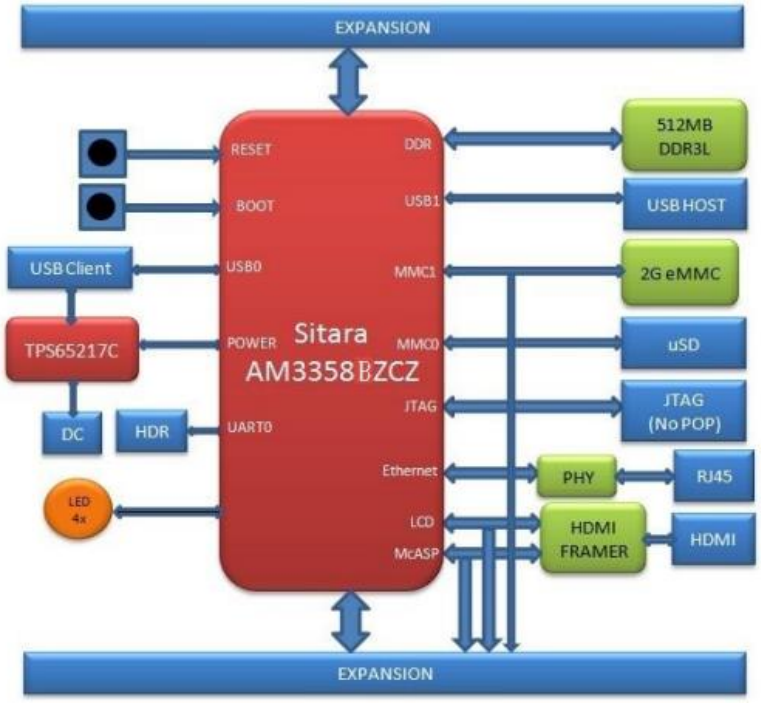

Fig. 10 BeagleBoard Black key components Block Diagram

2.1.4.1 Procesador

Esta placa está diseñada para el uso del procesador Sitara AM3358BZCZ100, del cual se muestra a continuación el diagrama de bloques.

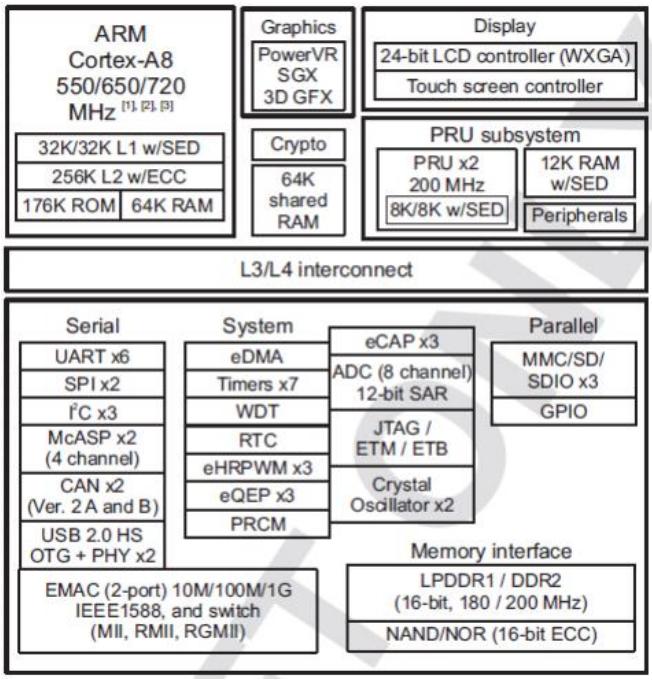

Fig. 11 AM3358BZCZ Diagrama de Bloques

Pasemos a describir de manera más específica los componentes de procesador:

- Microprocesador Sitara ARM Cortex-A8 de 32 bits y estructura RISC
  - Coprocesador NEON SIMD(Single Instruction , Multiple Data).
  - 32KB de memoria para instrucciones de nivel 1 y 32KB de cache de datos.
  - 256KB de cache de nivel 2 con código de corrección de error(ECC o Error Correction Code).
  - 176KB de memoria ROM On-Chip para el arranque .
  - 64KB de RAM dedicada.
  - Soporte para simulación y depuración(JTAG).
  - Controlador de interrupciones(hasta 128 peticiones de interrupción).
- Memoria On-Chip
  - 64KB para controlador de memoria RAM de propósito general(OCMC o On-Chip Memory Controller).
  - Accesible
  - Soporta retención para un rápido despertar
- Interfaces de memoria externa(EMIF)
  - Controlador de mDDR(LPDDR), DDR2, DDR3, DDR3L
    - Bus de datos de 16 bits, 1GB de espacio direccionable.
  - GPMC(General-Purpose Memory Controller)

    - Módulo de localización de error(ELM o Error Locator Module)
- Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem(PRU-ICSS).
    - Soporte para protocolos tales como EtherCAT, PROFIBUS, etc.
    - Dos unidades PRU(Programmable Real-Time Units)
        - Procesador de 32 bits de estructura RISC capaz de operar hasta 200MHz
        - 8KB de RAM para instrucciones y otros 8KB para RAM de datos ambos con paridad
        - Multiplicador de 32 bits de ciclo único y acumulador de 64 bits
        - Módulo mejorado de GPIO que proporciona soporte para las entradas/salidas y captura paralela de señal externa.
    - 12KB de memoria RAM con paridad compartida.
    - Tres bancos de registros de 120 Bytes accesible por cada PRU.
    - Controlador de interrupciones
    - Bus de interconexión interno para conectar maestros externos e internos a los recursos contenidos en el PRU-ICSS
    - Periféricos internos
        - Puerto UART. Hasta 12 Mbps.
        - Módulo mejorado de captura(eCAP).
        - Puerto MDIO(Management Data Input/Output).
- Módulo para la gestión de del reloj, reset y alimentación(PRCM).
- Reloj en tiempo real (Real Time Clock o RTC).
- Periféricos
    - Hasta 2 puertos USB 2.0 OTG.
    - Hasta 2 Industrial Gigabit Ethernet MACs(10, 100, 1000 Mbps).
    - Hasta 2 Puertos CAN(Controller-Area Network).
    - Puertos serie de audio multicanal(McASPs).
    - Hasta 6 UARTs.
    - Hasta 2 interfaces serie maestro-esclavo(McSPI).
    - Hasta 3 puertos MMC, SD, SDIO.
    - Hasta 3 interfaces I2C.
    - Hasta 4 Bancos de pines de entrada/salida de propósito general.
    - Hasta 3 entradas DMA externa que pueden ser usadas también como entradas de interrupciones.
    - 8 temporizadores de 32 bits de propósito general.
    - 1 temporizador Watchdog
    - Motor de gráficos 3D SGX530.
    - Controlador de LCD.
    - ADC de 12 bits con un máximo de 200K muestras/segundo pudiendo ser seleccionada como entrada cualquiera de las 8 entradas analógicas de la placa.
    - Hasta 3 módulos eCAP de 32 bits.
    - Hasta 3 módulos PWM de alta resolución mejorados

    - Hasta 3 módulos de 32 bits de codificador de pulsos en cuadratura(eQEP).
- Identificación de dispositivo
- Soporte de interfaz de depuración.
    - JTAG, cJTAG para ARM y PRU-ICSS
    - Escaner de bornas
    - IEEE 5000
- DMA
- Comunicaciones entre procesadores (IPC).
- Seguridad.
    - Hardware de aceleración de encriptado(AES, SHA, RNG).
    - Arranque seguro
- Modos de arranque configurables.

Como podemos apreciar, el procesador de esta placa contiene numerosas características que ofrecen al usuario un amplio abanico de posibilidades.

2.1.4.2 Memoria

En la placa podemos encontrar tres dispositivos referentes a la memoria de esta.

- 512MB DDR3L
    - Se utilizan 16 módulos de 256Mb DDR3L que hacen el total de 4Gb(512MB). Para ello son utilizados uno de estos dos modelos:
        - MT41K256M16HA-125 de Micron.
        - D2516EC4BXGGB de Kingston.

      Este módulo de memoria opera a una frecuencia de reloj de 400Mhz.

- 4KB de memoria EEPROM
    - Un único módulo conectado mediante puerto I2C0 que contiene la información de la placa, tal como nombre de la placa, número de serie o información de revisión. El módulo utilizado en esta placa es distinto al de la BeagleBone original por razones de reducción de costes. Permite ser programada por el dispositivo mientras mantiene la protección contra escritura si no es conectada a tierra.

- 4GB eMMC
    - Podemos encontrar un único módulo de 4GB de MMC embebida en la placa. Está conectado al puerto MMC1 del procesador, permitiendo un acceso de 8 bits. Es la fuente de arranque principal, permitiendo la configuración de arranque desde la memoria MMC0, la correspondiente a la ranura MicroSD.

- Ranura MicroSD
  - La placa está equipada con esta ranura para actuar como fuente de arranque secundaria y, si así es seleccionada, como fuente primaria.

- Modos de arranque:
  - Desde eMMC. Es el modo por defecto y el más rápido de los 4 disponibles
  - Desde SD. Con este modo se arrancará desde la memoria disponible en la ranura microSD. Este modo puede usarse para sobreescribir o programar el dispositivo eMMC.
  - Arranque serie. Se utilizará el puerto serie para la descarga directa del software. Se requiere un cable USB to Serial para el uso de este puerto.
  - Desde USB. En este modo arrancará desde el puerto USB.

2.1.4.3 Gestión de la energía.

La BeagleBoard Black utiliza el dispositivo gestor de energía TPS65217C en conjunto con un regulador de voltaje de continua (LDO o Low-DropOut) para abastecer de energía al sistema. Es el mismo dispositivo que el usado en la placa original a excepción de la configuración del riel de energía, que será cambiado de la EEPROM interna al dispositivo TPS65217C con motivo de soporte de los nuevos voltajes, ya que la memoria DDR3L requiere 1.5V en lugar de los 1.8V de la DDR2, como es el caso de la BeagleBone original. El regulador externo LDOTLV70233 provee al resto de la placa del riel de 3.3V .

2.1.4.4 Interfaz USB PC

La placa contiene un conector miniUSB que conecta el puerto USB0 con el procesador, el cual no se ha visto modificado desde la BeagleBone original.

2.1.4.5 Puerto de depuración serie

La depuración serie del procesador está disponible a través de la UART0 en cabezal de 1x6 pines. Para hacer uso de ella es necesario utilizar un adaptador USB a TTL.

2.1.4.6 Puerto USB1

En la placa existe un puerto USB Type A hembra que conecta con el USB1 del procesador. Este puerto puede ser también utilizado como fuente de alimentación siendo capaz de dar hasta 500mA y 5V.

2.1.4.7 Fuentes de alimentación

La BeagleBone Black puede ser alimentada desde distintas fuentes:

- Puerto USB de un PC.
- Fuente de alimentación de 5V y 1ª conectado al conector de DC de la placa.
- Fuente de alimentación con conector USB.
- Conectores de expansión.

2.1.4.8 Botón de reset

La placa dispone un botón de reset que al ser pulsado y soltado provoca un reset de la placa. El botón de la BeagleBone Black es un poco mayor en tamaño que el de la placa original, y se ha desplazado hacia el borde de la placa con idea de hacerlo más accesible al usuario.

2.1.4.9 Botón de encendido/apagado

Se proporciona al usuario un botón de encendido/apagado situado cerca del botón de reset y el puerto Ethernet. Dicho botón se decide añadir para darle ciertas características al usuario tales como:

- Una interrupción es enviada al procesador para facilitarle el apagado ordenado salvando así los archivos y desmontando las unidades.
- Provee al procesador de la capacidad de poner la placa en modo durmiente para salvaguardar la energía.
- Aviso al procesador para despertar del modo *sleep* y restaurar el estado anterior a dicho modo.

Si este botón se mantiene presionado durante un periodo mayor a 8 segundos la placa se apagará por completo, encendiéndose la próxima vez que sea pulsado.

2.1.4.10 Indicadores

La placa contiene 5 LEDs azules, uno para indicarle al usuario si la placa está encendida, parpadeando si detecta que la energía proporcionada a esta es mayor que la máxima permitida, y cuatro más que pueden ser controlados vía software mediante los GPIOs de la placa.

Adicionalmente existen dos LEDs en el conector RJ45 que indican el estado de la conexión Ethernet.

2.1.4.11 Cabezal CTI JTAG

En la BeagleBone Black podemos encontrar un cabezal adicional de 20 pines para facilitar el desarrollo software y la depuración de este usando emuladores JTAG, aunque este cabezal no se proporciona con la placa.

2.1.4.12 Interfaz HDMI

Una interfaz HDMI está conectada a la interfaz LCD de 16 bits del procesador. Las siguientes resoluciones son las soportadas vía software:

- 1280 x 1024
- 1440 x 900
- 1024 x 768
- 1280 x 720

2.1.5 Conclusiones

Se han estudiado con detenimiento diferentes placas de desarrollo para tratar de determinar cuál es la óptima para el proyecto, fijándonos para ello tanto en las necesidades como en los objetivos de este. Han sido realmente determinantes dos aspectos principales, precisión y costes.

Después de este estudio, en el cual ha quedado patente la funcionalidad y potencia de cada una de las placas, nos ha quedado claro que la de mayor potencia y quizás precisión es la BeagleBoard-X15 pero, como hemos comentado anteriormente, el coste del proyecto es un factor a tener en cuenta porque recordemos que el objetivo de éste es obtener un electrocardiógrafo de bajo coste. Por esta razón se ha optado por su hermana pequeña, la BeagleBoard Black rev C ya que, aparte de cubrir nuestras necesidades de precisión, es del orden de 5 veces más barata, como se puede apreciar en la Fig. 4 Modelos BeagleBoard.org .

2.2 Electrocardiograma y electrocardiógrafo

Para el desarrollo del proyecto se ha realizado un estudio tanto sobre las señales del electrocardiograma como de los electrocardiógrafos. Comenzaremos explicando un poco las características del electrocardiograma, así como las de las señales implicadas, para después explicar las propiedades y características que debe tener un electrocardiógrafo.

2.2.1 Electrocardiograma

El electrocardiograma, ECG/EKG, es una prueba que registra la actividad eléctrica del corazón que se produce en cada latido cardiaco. Esta actividad eléctrica se registra desde la superficie corporal del paciente y se dibuja en un papel mediante una representación gráfica o trazado, donde se observan diferentes ondas que representan los estímulos eléctricos de las aurículas y los ventrículos.

Para la recogida de la actividad eléctrica por el electrocardiógrafo, se necesita que sobre la piel del paciente se coloquen una serie de electrodos (normalmente 10), que irán unidos hasta el electrocardiógrafo por unos cables. Con 10 electrodos se consiguen obtener 12 derivaciones, es decir, se dibujan en el papel 12 trazados de los impulsos eléctricos del corazón desde diferentes puntos del cuerpo. Se pueden obtener derivaciones extra si se añaden más electrodos a la superficie corporal, pero el electrocardiograma básico debe constar como mínimo de 12 derivaciones. El electrocardiograma de una persona sana presenta un trazado particular; cuando aparecen cambios en ese trazado el médico puede determinar si existe un problema.

Se usa para medir el ritmo y la regularidad de los latidos, el tamaño y posición de las aurículas y ventrículos, cualquier daño al corazón y los efectos que sobre él pueden tener ciertos fármacos o dispositivos implantados en el corazón (como marcapasos). Las alteraciones en el trazado son imprescindibles para la detección y análisis de las arritmias cardiacas. También resulta muy útil en los episodios agudos de enfermedad coronaria, como el infarto de miocardio.

Es una prueba sencilla, disponible, rápida, que no produce ninguna molestia (es indoloro) y no tiene ningún riesgo para el paciente (no se envía ningún tipo de electricidad a través del cuerpo, solo detecta la actividad eléctrica que se genera en el propio corazón).

2.2.1.1 ECG normal

Para entender mejor el texto desarrollado en este apartado, podemos fijarnos en la siguiente figura.

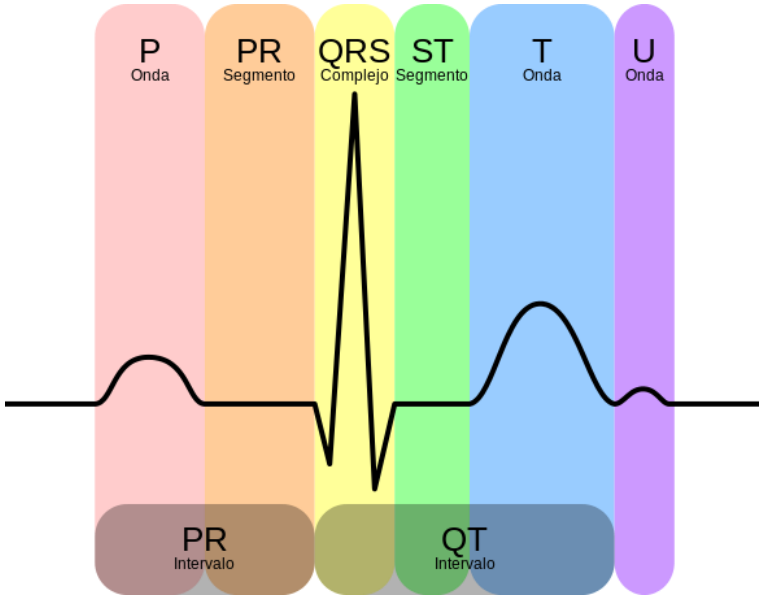

Fig. 12 Ondas ECG

El trazado típico de un electrocardiograma registrando un latido cardíaco normal consiste en una onda P, un complejo QRS y una onda T. La pequeña onda U normalmente es invisible. Estos son eventos eléctricos que no deben ser confundidos con los eventos mecánicos correspondientes, es decir, la contracción y relajación de las cámaras del corazón. Así, la sístole mecánica o contracción ventricular comienza justo después del inicio del complejo QRS y culmina justo antes de terminar la onda T. La diástoles, que es la relajación y rellenado ventricular, comienza después que culmina la sístole correspondiendo con la contracción de las aurículas, justo después de iniciarse la onda P.

- Onda P

La onda P es la señal eléctrica que corresponde a la despolarización auricular. Resulta de la superposición de la despolarización de la aurícula derecha (parte inicial de la onda P) y de la izquierda (final de la onda P). La repolarización de la onda P (llamada onda T auricular) queda eclipsada por la despolarización ventricular (Complejo QRS).

- Complejo QRS

El complejo QRS corresponde a la corriente eléctrica que causa la contracción de los ventrículos derecho e izquierdo (despolarización ventricular), la cual es mucho más potente que la de las aurículas y compete a más masa muscular, produciendo de este modo una mayor deflexión en el electrocardiograma.

- Onda T

La onda T representa la repolarización de los ventrículos. Durante la formación del complejo QRS, generalmente también ocurre la repolarización auricular que no se registra en el ECG normal, ya que es tapado por el complejo QRS. Eléctricamente, las células del músculo cardíaco son como muelles cargados; un pequeño impulso las dispara, despolarizan y se contraen. La recarga del muelle es la repolarización (también llamada potencial de acción).

En la siguiente imagen podemos ver un electrocardiograma real.

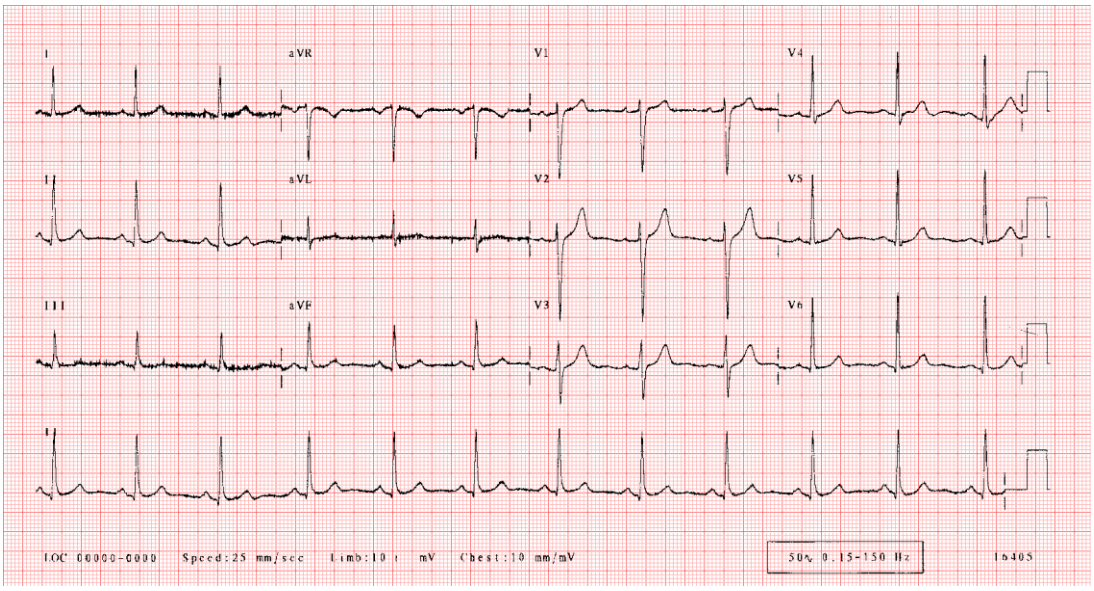

Fig. 13 ECG real normal

Gracias al estudio de estas señales, se pueden determinar ciertas enfermedades y problemas cardiacos.

2.2.1.2 Electrocardiograma fetal

Las técnicas explicadas en el apartado anterior se realizan desde hace mucho tiempo tanto en adultos como en jóvenes, incluso en recién nacidos, pero es relativamente frecuente el electrocardiograma fetal.

El estudio de la actividad cardiaca fetal ha sido durante muchos años el principal instrumento para evaluar el estado de salud del feto. El corazón humano comienza a latir, tres semanas después del momento de la concepción, con una frecuencia aproximada de 65 pulsaciones por minuto, la cual aumenta día a día hasta estabilizarse, en la semana 12, en torno a los 110-160 latidos por minuto. El que el ritmo cardiaco fetal no se encuentre entre estos límites es indicio de hipoxia (falta de oxígeno). Contamos en la actualidad con diversos procedimientos e instrumentos, todos ellos complementarios (ecógrafos, magnetocardiogramas, etc) para la monitorización cardiográfica prenatal, pero no están implantadas técnicas de electrocardiografía fetal *in utero*, con la que se puede registras la actividad eléctrica del corazón del feto mediante la colocación de electrodos en la superficie del vientre materno. Esta técnica es denominada *electrocardiograma fetal no invasivo*, y como el propio nombre indica, no resulta perjudicial ni para la madre ni para el feto. El principal inconveniente es que los electrodos capturan no solo el electrocardiograma fetal, sino también el de la madre, cuya amplitud es mucho mayor, así como la superposición de otros ruidos (actividad eléctrica del musculo uterino, interferencia de la red eléctrica, etc.). Es por ello que el primer paso en el estudio de las señales fetales corresponde a la separación de las señales de interés mediante técnicas de procesamiento de señal.

2.2.2 Electrocardiógrafo

La técnica del electrocardiograma se viene realizando desde hace bastante tiempo, y es por eso que existen equipos muy precisos para realizarla en todos los hospitales. Pero aquí nos centraremos en los electrocardiogramas portátiles de bajo costo.

Podemos encontrar diferentes electrocardiógrafos de bajo coste. Hablaremos principalmente de dos, uno realizado en la Universidad de Perú, cuyos autores son Carlos A. Alva, Wilfredo Reaño y Joel O. Castillo, y de otro realizado por los componentes de www.cooking-hacks.com.

2.2.2.1 Electrocardiógrafo Universidad de Perú

En la publicación basan todo el artículo en la adquisición de señal más que en el procesado digital posterior. Más adelante indican las características que debe tener la parte de la digitalización de la señal resultante, pero sin entrar en más detalle.

Para dicha adquisición se diferencian diferentes etapas que comentaremos a continuación.

- Amplificador de instrumentación

Para el trato con la señal diferencial que tendríamos a la entrada(los electrodos), eligen un amplificador de instrumentación de alto CMRR(Common-mode reject ratio o Relación de Rechazo de Modo Común). La razón de dicha elección es que el modo común será el ruido blanco proveniente del exterior o del ambiente que rodea al sistema.

El esquema final del amplificador, basado en el integrado TL084, se puede apreciar en la siguiente figura.

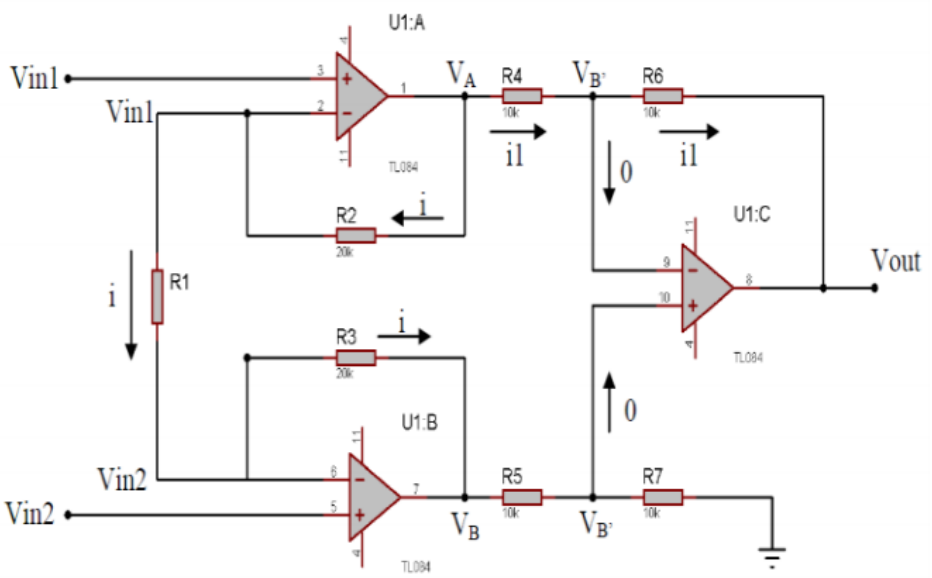

Fig. 14 Esquema Amplificador basado en TL084

- Filtrado paso-bajo

En la siguiente etapa realizan un filtrado paso bajo de orden 4, que a su vez es producto de dos filtros de orden 2 en cascada tipo Butterworth. Este tiene como fin, ser un filtro antialiasing con el propósito de evitar la banda de trabajo del sistema. Ya que la señal de entrada será de baja frecuencia, establecen la frecuencia de corte en 200Hz. En la siguiente figura podemos apreciar el esquema final de dicho filtro.

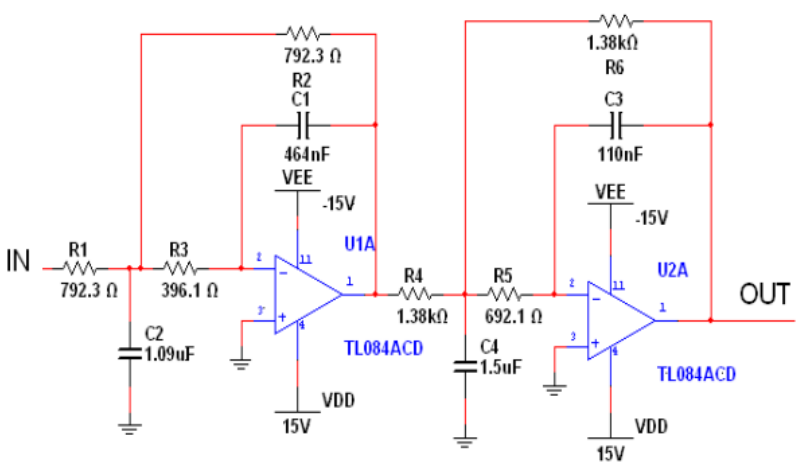

Fig. 15 Esquema final filtro paso-bajo 4to orden

- Filtrado paso-alto

A continuación del filtro paso-bajo nos encontramos con un filtro pasa-alto también de orden 4. Este filtro tiene la función de filtro anti señal DC, con el propósito de evitar la señal continua en la salida final. En total la unión del filtro pasa-bajo y el pasa-alto resulta en un filtro paso de banda. Podemos apreciar el esquema final del filtro pasa-alto en la siguiente figura.

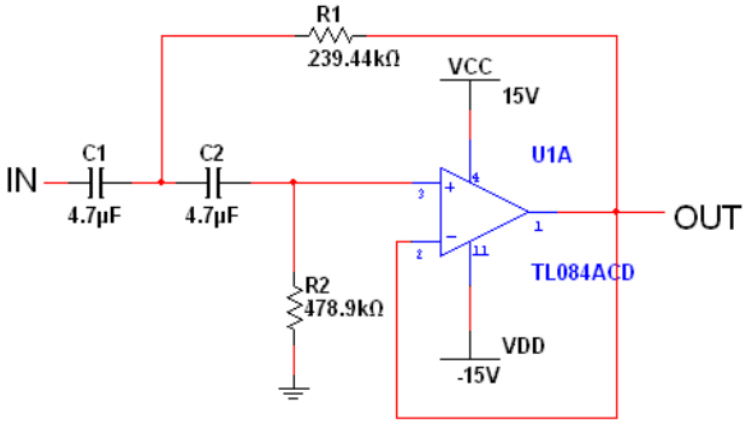

Fig. 16 Esquema final filtro pasa-alto de 4to orden

- Filtro Notch

Se realiza un filtro Notch centrado en la frecuencia de rechazo de 60Hz, ya que esta es la frecuencia de mayor potencia producto de la inducción de la corriente eléctrica 110/220AC. Para una mayor efectividad se emplea un filtro Notch de orden 4, el cual puede apreciarse en la siguiente figura.

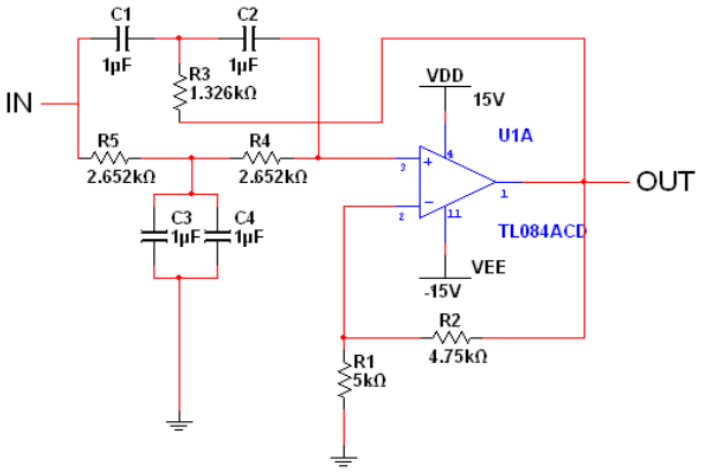

Fig. 17 Esquema final Filtro Notch 4to orden

- Proceso final de adquisición de datos

A continuación de muestra el diagrama de bloques de la unión de todas las etapas, más el tratamiento digital.

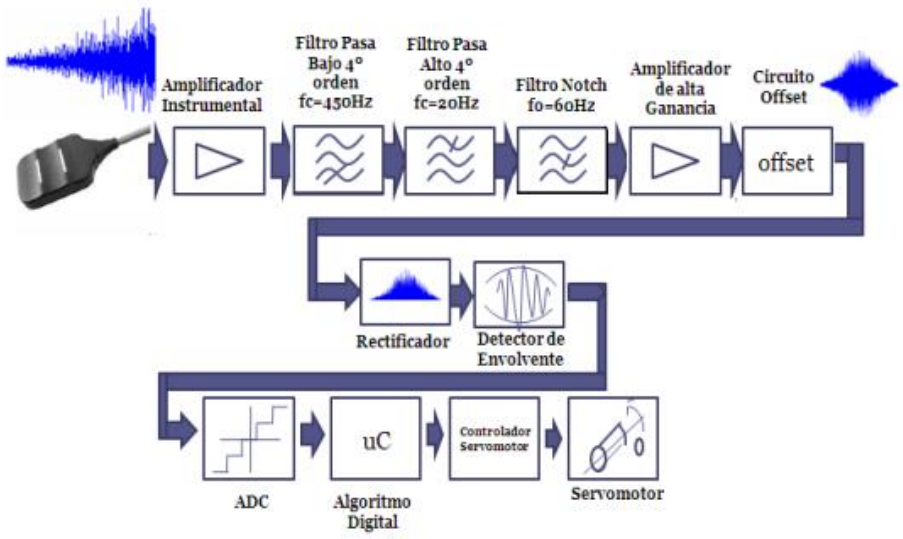

Fig. 18 Diagrama de bloques completo

Como resultado final nos muestran la señal obtenida en el osciloscopio.

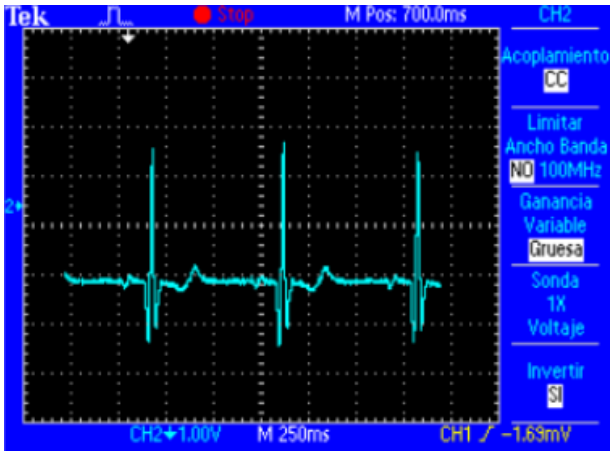

Fig. 19 Señal ECG sistema final

2.2.2.2 Electrocardiógrafo Cooking Hacks

Los compañeros de www.coocking-hacks.com han desarrollado una plataforma para Arduino y Raspberry Pi llamada *e-health Sensor Platform* para aplicaciones biométricas o médicas. Es una capa(*cape*) compatible con ambas placas de desarrollo que dispone de diferentes etapas de adquisición de señal para las diferentes pruebas médicas. Ofrece diversas posibilidades como pueden ser presión sanguínea, pulso y oxigeno en sangre, temperatura de la sangre, GSR(Galvanic Skin Response), electromiografía, electrocardiograma y alguna otra prueba más, pero nosotros nos centraremos en la parte referente al electrocardiograma. La plataforma se puede apreciar en la siguiente figura.

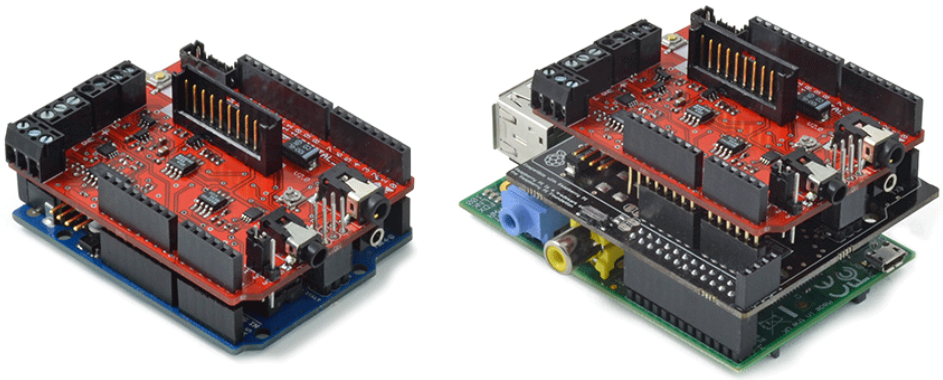

Fig. 20 e-health Platform

Este es un proyecto open-source, por lo que ofrecen al usuario toda la información respecto al diseño, desde los esquemáticos hasta el código de las librerías utilizadas para la gestión de las señales recogidas.

Veamos el esquemático de la adquisición de señal del electrocardiograma.

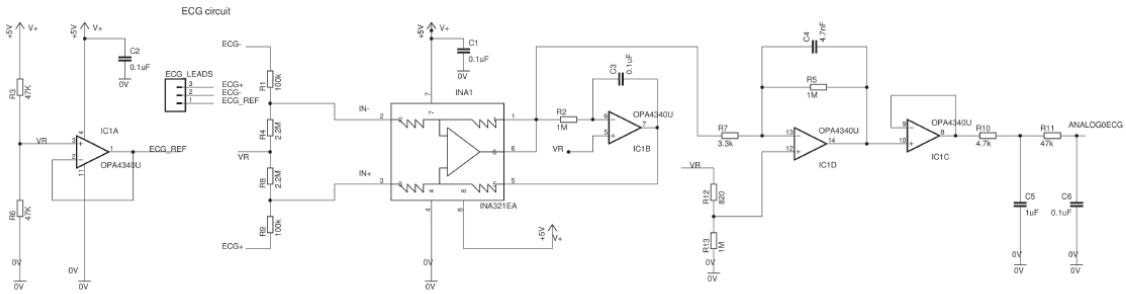

Fig. 21 Esquematico ECG e-health

Como se puede apreciar en la figura, hacen uso de un amplificador de instrumentación para la conversión de la señal diferencial de entrada proveniente de los electrodos en la unión de las dos amplificada. Una vez convertida, se utiliza un amplificador operacional para la etapa de amplificación, realizando después un filtrado de tercer orden eliminando así altas frecuencias no deseadas.

Además de esto, ofrecen las librerías utilizadas, tanto para Arduino como para Raspberry Pi, así como ejemplos de códigos de como emplear dichas librerías. Aquí se encuentra incluida la representación de los datos en una pantalla LCD conectada a la placa de desarrollo y como aportación extra han desarrollado una aplicación móvil para poder ver los resultados de las pruebas en el smartphone. En la siguiente figura podemos ver los resultados finales, mostrados en un LCD.

Fig. 22 Representacion ECG en LCD

3 Diseño amplificador

3.1 Especificaciones

Para las especificaciones de nuestra PCB es necesario tener en cuenta diferentes aspectos, entre los más importantes se encuentran las características de la señal de entrada a la placa y las características que debe tener la señal de salida de ésta, ya que nuestra señal de salida será la entrada del ADC de la BeagleBone Black.

3.1.1 Señal de entrada

La señal de entrada de la placa será la proveniente de los electrodos conectados al cuerpo humano, por lo tanto debemos prepararnos para distintos aspectos. Uno de ellos es el rango, tanto en amplitud como en frecuencia, que presenta este tipo de señales. En la siguiente tabla puede observarse el rango con claridad.

Amplitud(mV)		**Frecuencia(Hz)**	
MAX	MIN	MAX	MIN
0.05	3	0.01	300

Si miramos los valores de la tabla, podemos deducir el problema principal de esta señal, el ruido. Al ser una señal con una amplitud tan reducida, es muy probable que algún tipo de ruido nos influya en la forma de esta señal e incluso en algunos casos enmascare su valor sin que podamos diferenciar la señal con claridad.

Al tener estas características hay que tener en cuenta que todo puede influir, una mala conexión del cable al que están conectados los electrodos hacia la BeagleBone Black, la mala adhesión de los electrodos al cuerpo(esto es especialmente crítico), el movimiento del propio ser humano al que se le conectan los electrodos, sujeto del cual queremos obtener el ECG, una contracción de un músculo, respiración, etc.

Un ruido que aparece notablemente es el inherente a la red eléctrica. Es un ruido concentrado en la frecuencia de 50Hz en España, en Estados Unidos por ejemplo la frecuencia es 60Hz, que es omnipresente y presenta niveles altos de energía, pero eso lo eliminaremos por software más adelante.

3.1.2 Señal de salida

Para las especificaciones de la placa debemos tener en cuenta hacia dónde va nuestra señal. En nuestro caso, la señal será conectada al ADC de la BeagleBone Black, cuyas limitaciones serán claves para nuestro diseño.

Las propiedades del ADC son restrictivas en cuando al rango en amplitud de la señal, siendo el rango permitido de 0 a 1.8V, lo cual deberá ser tenido en cuenta a la hora de establecer la ganancia total de nuestro circuito. El hecho de que no lea valores negativos es también una restricción importante dentro del rango, ya que deberemos establecer un modo común sobre el cual irá oscilando la señal. Lo ideal en nuestro caso es que ese punto medio quede establecido en 0.9V, para aprovechar de la manera más ampliamente posible el rango de entrada disponible.

Teniendo en cuenta las restricciones explicadas en el apartado anterior, junto con las características de la señal de entrada, podemos establecer la ganancia máxima del circuito en 300, ya que la máxima amplitud de nuestra señal es de 3mV, por lo que al centrar la señal en 0.9V nos daría un máximo de 0.9V+3mV*300 = 1.8V, el máximo permitido por el ADC.

Por otra parte, existen limitaciones en cuando a la corriente de entrada al ADC, siendo el máximo permitido de 2uA, lo que será otro aspecto a tener en cuenta.

3.1.3 Componentes

Nuestro circuito va a estar compuesto de resistencias, condensadores, amplificadores operacionales y amplificadores de instrumentación. Tras el estudio realizado en el estado del arte, podemos observar que el componente de un uso más extendido para este tipo de aplicaciones es el INA321 de Texas Instruments.

Al leer detenidamente el datasheet del componente, descubrimos que ellos mismos nos indican como conformar el circuito para la creación de un amplificador de ECG de bajo coste, que es precisamente nuestro objetivo.

Basándonos en el trabajo desarrollado por www.cooking-hacks.com comentado también en el estado del arte, elegiremos el amplificador operacional OPA4340, también fabricado por Texas Instruments.

Tanto los condensadores como las resistencias han sido elegidos del tipo SMD(Surface Mount Device), llamados dispositivos de montaje superficial, por las ventajas ofrecidas por esta tecnología. Entre estas ventajas podemos encontrar:

- Reducción del peso y dimensiones del circuito.

- Reducción de costes de fabricación.
- Reducción de taladrados en el PCB.
- Integración de en ambas caras del PCB.
- Valores de componentes pasivos(precisamente nuestro caso) muchos más precisos.

Se puede encontrar más información respecto a los componentes en sus respectivos datasheets, cuyo enlace se encuentra en las referencias.

3.2 Diseño esquemático

En la siguiente figura podemos ver el circuito de partida, sobre el cual se han realizado cambios que se comentaran a continuación.

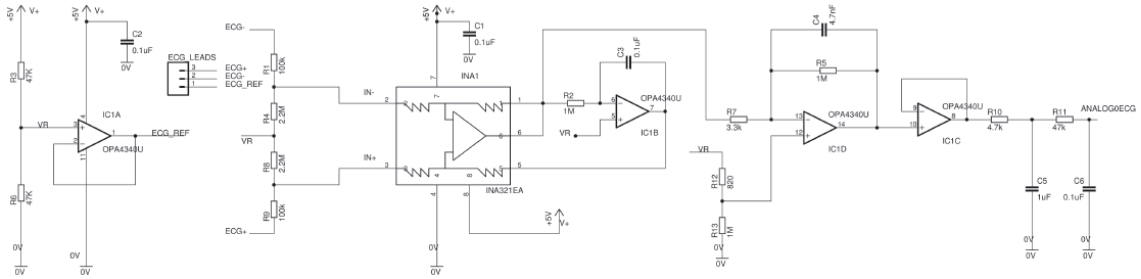

Fig. 23 Esquemático circuito ECG

Vamos a abordar el circuito por sus diferentes etapas para poder analizarlo y comprenderlo de una manera más detallada.

- Referencia a partir de alimentación

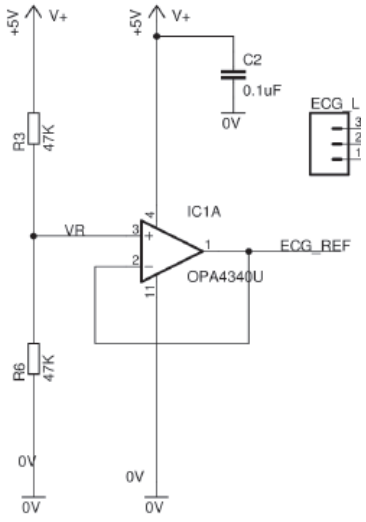

Fig. 24 Referencia y ECGref

En esta parte del circuito tratamos de obtener un valor de referencia que nos valdrá para el resto del circuito y el punto que obtendrá dicho valor esta denominado por la etiqueta “VR”. Si nos fijamos en la figura podemos observar que dicho punto está determinado por un divisor resistivo determinado por las resistencias R3 y R6. A continuación podemos apreciar la fórmula de la cual se obtiene su valor.

$$VR = V_+ * \frac{R_6}{R_3 + R_6}$$

En nuestro esquemático podemos observar que el punto "VR" esta entre dos resistencias iguales, por lo que obtendríamos en el punto medio un voltaje de 2.5V. Durante el desarrollo del proyecto se modificaron los valores de dichas resistencias con el fin de establecer este valor en 0.9V y cumplir con las especificaciones requeridas. Para conseguirlo se establecieron los valores de las resistencias R3 y R6 de 200Kohm y 43Kohm respectivamente.

Ese voltaje pasa a través de un buffer con el objetivo de tener el flujo de corriente controlado y su salida será utilizada para el ECGref. Éste será el electrodo de referencia, el cual se conectará al cuerpo humano con el objetivo de tener el mismo valor de referencia en todas las partes y obtener unos valores correctos en la medición.

- Amplificador de instrumentación

En la siguiente figura podemos apreciar la parte de nuestro circuito referente al INA, cuyo contenido es orientativo.

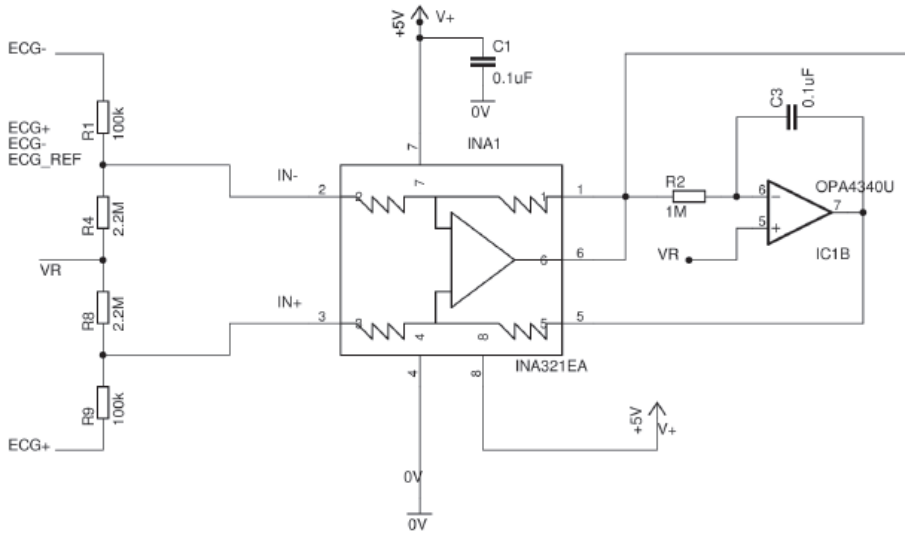

Fig. 25 INA321 conexión

Antes de explicar la figura debemos conocer el contenido real del recuadro correspondiente al INA321, mostrado en la siguiente figura. Nótese que la enumeración de las conexiones se mantiene.

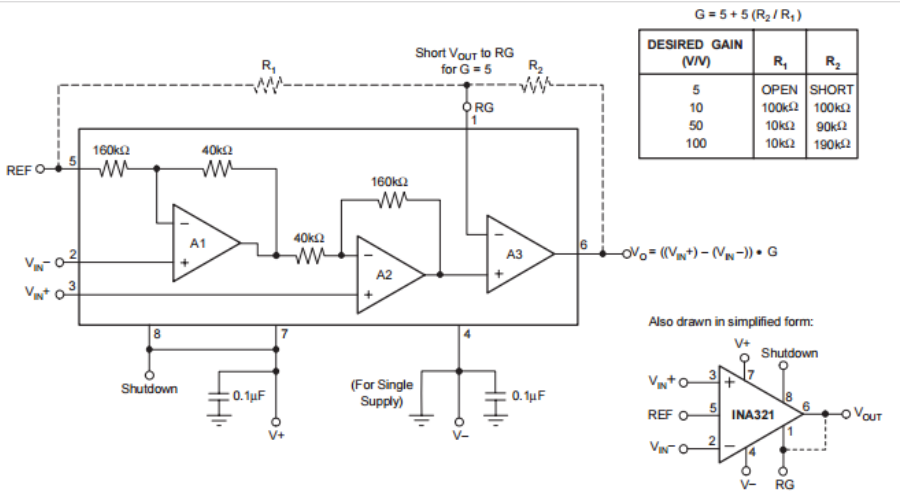

Fig. 26 INA321 contenido

A priori ya podemos obtener la ganancia de esta parte del circuito, puesto que la configuración de las resistencias que la determinan es la primera de las indicadas en la segunda figura y en consecuencia se obtiene una ganancia igual a 5.

Esta parte del circuito es utilizada para crear la señal diferencial a partir de las obtenidas desde los electrodos ECG+ y ECG-, los cuales estarán conectados al cuerpo humano en sus correspondientes lugares. La salida obtenida vendrá dada por la fórmula indicada en la segunda figura.

En el pin número 5 tenemos conectado la salida de uno de los componentes del OPA4340, cuya salida establece la referencia del INA321, y a cuya entrada se encuentra la referencia obtenida gracias al divisor resistivo comentado al principio del apartado. Esto establecerá el modo común de la señal diferencial de salida del amplificador de instrumentación.

- Amplificador operacional

La siguiente parte que vamos a explicar corresponde a uno de los componentes del OPA4340 y es utilizado para aumentar la ganancia del circuito una vez obtenida la señal diferencial deseada. Antes de esto veamos que contiene el componente indicado en la siguiente figura. Nótese que la “x” de la denominación nos indica el número de componentes iguales a este que contiene el empaquetado.

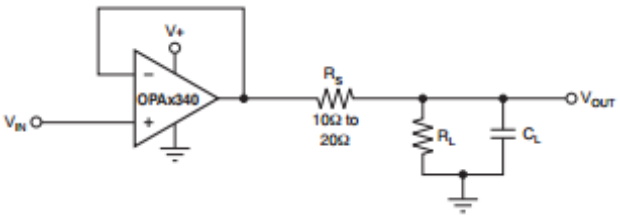

Fig. 27 OPAx340

Y ahora veamos la parte del circuito a explicar.

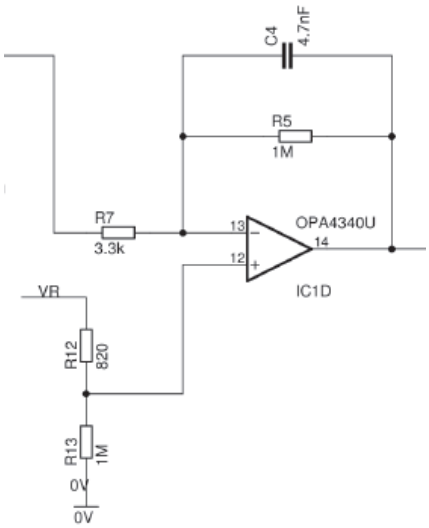

Fig. 28 Amplificador mayor ganancia

Este componente del circuito va a ser nuestra principal fuente de la ganancia total. Es un amplificador cuya ganancia viene dada por el cociente de las resistencias R5 y R7, que con los valores de la figura nos proporcionaría una ganancia de 1M/3.3K = 300. Este valor ha tenido que ser modificado durante el desarrollo del proyecto, puesto que si recordamos los apartados anteriores, el amplificador operacional nos proporcionaba una ganancia de 5, que multiplicada por la de éste, tendríamos una ganancia total de 1500, lo cual sobrepasa la restricción del circuito establecida en apartados anteriores de 300. Con motivo de cumplir esta restricción se han modificado los valores de estas resistencias R5 y R7 a 600Kohm y 10Kohm respectivamente, consiguiendo una ganancia de 600K/10K = 60, que multiplicada por la ganancia del INA resultaría en los 300 deseados. Por otra parte, la existencia de la capacidad nos indica que además de amplificar, se realiza un filtrado paso bajo de la señal. El valor de dicha capacidad ha tenido que ser modificado ya que con el valor original, la frecuencia de corte se establecía en aproximadamente 33Hz, lo que no cuadraba con nuestras especificaciones. El valor final se establece en 1nF, situándose el polo en torno a los 270Hz, lo cual está más cercano a las especificaciones a cumplir.

- Filtro final

Pasamos a explicar la parte final del circuito, formado por un buffer y dos filtros. Veamos cómo está compuesto en la siguiente figura.

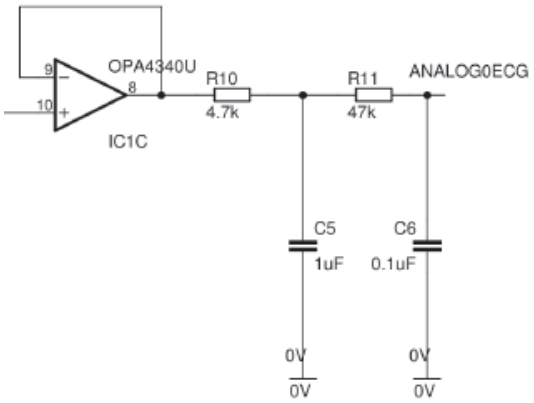

Fig. 29 Buffer y filtro final

La primera parte se compone de un buffer cuya función mantener el flujo de intensidad controlado. La segunda parte está formada por dos filtros RC paso de baja de primer orden. Estos filtros se establecen con el objetivo eliminar las componentes de altas frecuencias que no nos interesan. Con los valores de la figura obtendríamos un filtrado con una frecuencia de corte de aproximadamente 30Hz por lo que durante el desarrollo del proyecto se han tenido que modificar el valor de la resistencia R11 y el condensador C5 a 4.7Kohm y 0.1uF respectivamente, para al final obtener una frecuencia de corte de aproximadamente 330Hz y cumplir con las restricciones establecidas. La fórmula para la obtención de la frecuencia de corte se muestra a continuación.

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

Como apunte final, se incluye una tabla con los valores de todos los componentes.

ELEMENTO	VALOR
R1	100Kohm
R2	1Mohm
R3	200Kohm
R4	2Mohm
R5	600Kohm
R6	43kohm
R7	10Kohm
R8	2Mohm
R9	100Kohm
R10	4.7kohm
R11	4.7kohm
R12	820ohm
R13	1Mohm
C1	0.1uF
C2	0.1uF
C3	0.1uF
C4	1nF
C5	0.1uF
C6	0.1uF

3.3 Diseño PCB

Para el diseño del PCB utilizaremos el programa Altium Designer. Éste es un software de diseño de PCBs, FPGAs y software embebido muy completo, gracias al cual nos será más fácil el desarrollo de nuestra placa PCB. Para más información se puede visitar su página web, www.altium.com .

El programa dispone de varias ventanas según en qué parte del desarrollo estemos en ese momento. El primer paso es la creación del esquemático, el cual ha sido explicado en el apartado anterior.

Podemos ver el esquemático terminado en la siguiente figura.

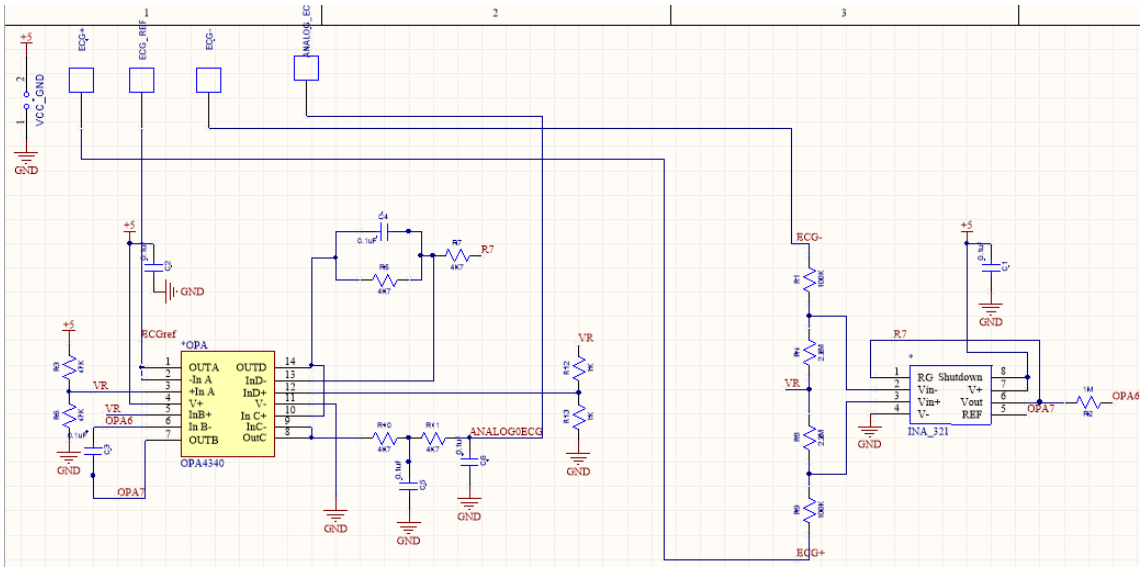

Fig. 30 Esquemático Altium Designer

Una vez terminado comenzaremos con el diseño del PCB, tal como será nuestra placa una vez revelada. Para ello se nos abrirá una pantalla en la cual tendremos disponibles todos los componentes que hayamos incluido en nuestro esquemático, y tendremos que diseñar las conexiones entre estos. Hay que tener en cuenta el tipo de PCB que vamos a fabricar, en nuestro caso uno a doble cara, en el que podremos realizar conexiones tanto por la cara superior como por la cara inferior de este. Una vez realizadas todas las conexiones y establecido los planos de tierra tendremos el PCB listo para revelar. El resultado final puede verse en la siguiente figura.

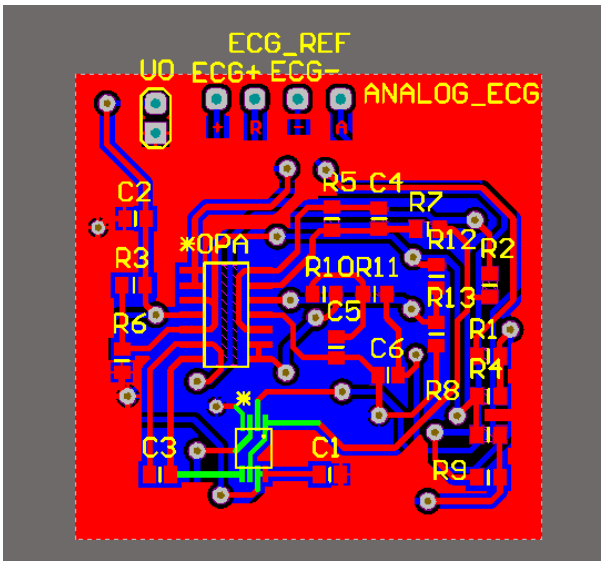

Fig. 31 PCB

En la figura puede apreciarse el OPA, el componente de mayor tamaño, y el INA, el componente con las conexiones en verde. El resto de componentes, como hemos visto en el apartado del esquemático, son resistencias y condensadores SMD. Por otra parte, podemos apreciar en la parte superior, de izquierda a derecha, la conexión de alimentación y tierra, ECG+, ECGref, ECG- y por último la salida del circuito, que será la que se conectará al ADC de la BeagleBone Black.

Una vez realizado el diseño, debemos revelar nuestro PCB. Para ello imprimiremos este diseño en un papel de cebolla y, una vez hecho esto, nos iremos a la máquina de revelado.

Para un correcto revelado tendremos la PCB en la máquina durante 2 minutos y medio, en una disolución que estará formada por 2 partes de agua, 1 parte de agua oxigenada y otra de ácido. Una vez terminado esto debemos realizar los taladros necesarios para las vías y pines y, una vez hecho esto, ya podemos soldar nuestros componentes a la placa siempre que antes hayamos limpiado los restos de resina que tiene la placa con acetona. El resultado final podemos verlo en la siguiente figura.

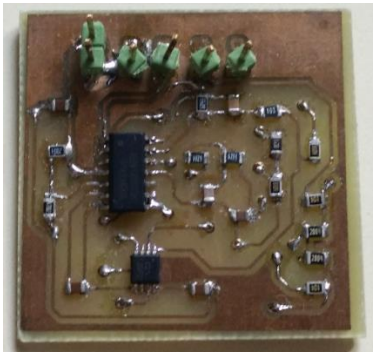

Fig. 32 PCB

3.4 Medidas experimentales

Es necesario realizar distintas pruebas a nuestra placa con el fin de caracterizarla, es decir, ver cómo funciona realmente. Para ello se han llevado a cabo diferentes pruebas cuyo resultado nos proporcionaran dicha información, más concretamente se obtienen su diagrama de bode, para conocer su ganancia a diferentes frecuencias, y su fft(Fast Fourier Transform) para conocer cómo afecta la distorsión a nuestro circuito o mejor dicho, como nuestro circuito distorsiona la señal una vez ésta lo ha atravesado.

Comenzaremos con el diagrama de bode.

Para conseguirlo nos encontramos con algunas dificultades, la más destacada de ellas radica en la señal tan pequeña que nos vemos obligados a utilizar para conseguir que el amplificador no sature y podamos apreciar realmente la ganancia máxima del circuito. Hemos utilizado una señal diferencial de 1mVpp con un offset de 0.9 voltios, ya que así será nuestra señal real. Recordar que el circuito tiene una ganancia máxima de aproximadamente 300.

Nótese que a partir de que la ganancia comienza a disminuir, nos resulta muy complicado poder medir con precisión la señal de salida, ya que aunque estemos hablando de una ganancia de 15, nuestra señal de entrada es de 1mVpp, por lo que la salida correspondería a 15mVpp, lo cual es complicado medir con precisión en el osciloscopio. Además de eso hay que tener en cuenta que en señales tan pequeñas el ruido puede ser muy significativo. Para obtener el bode se ha variado la frecuencia de la señal de 1 hercio a 1000 hercios, siendo imposible seguir avanzando por el motivo comentado anteriormente.

El resultado puede apreciarse en la siguiente figura.

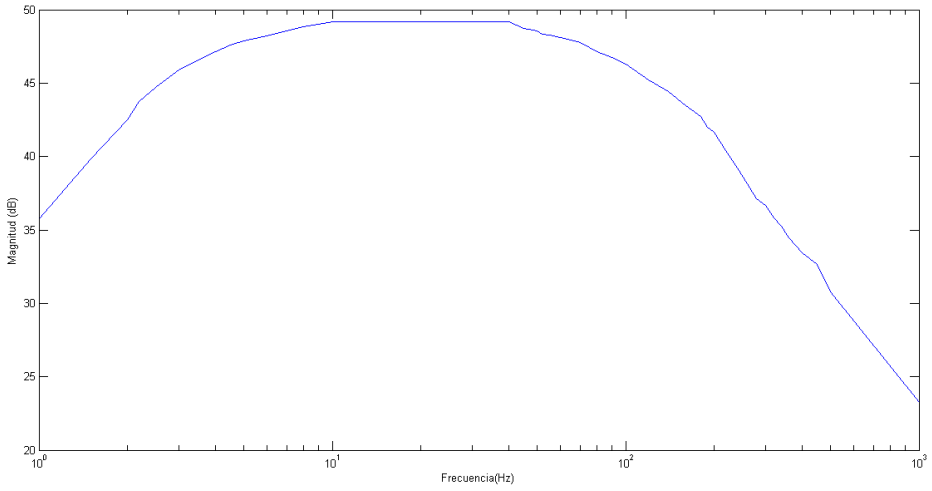

Fig. 33 Diagrama de bode. Magnitud

Pasamos a representar el espectro, con ayuda de Matlab para realizar la fft. Los datos sobre los que trabajamos han sido obtenidos directamente del osciloscopio, con una señal de entrada a 20 Hz, para aprovechar la máxima ganancia de nuestro circuito.

El espectro puede apreciarse en la siguiente figura.

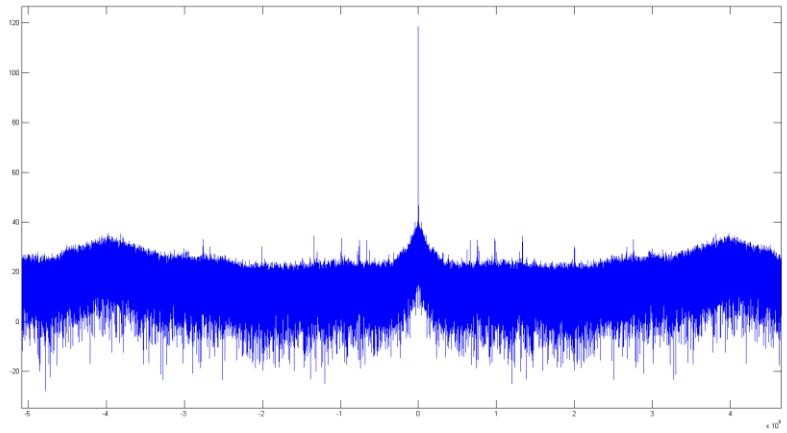

Fig. 34 Espectro

Ahora ampliaremos en la zona que nos interesa.

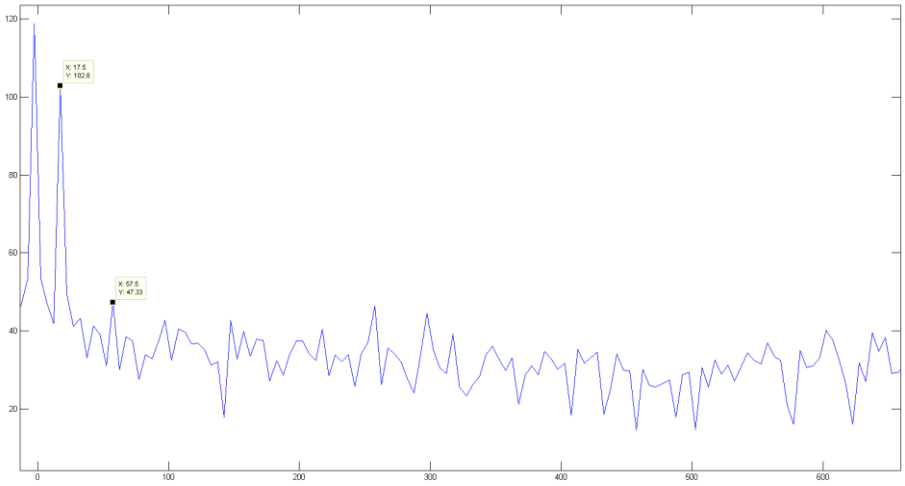

Fig. 35 FFT zona de mayor interés

En la imagen podemos apreciar el tono fundamental en aproximada mente 20 Hz, y el tercer armónico, en aproximadamente 60 Hz, como era de esperar. Entre ambos existe una diferencia de aproximadamente 55 dB, lo que nos indica que nuestro circuito no distorsiona mucho la señal al pasar por él.

4 Placa de desarrollo

4.1 Preparación

4.1.1 Primeros pasos

La BeagleBone Black se proporciona con un sistema Linux precargado y un cable miniUSB para conectar esta al ordenador y empezar a utilizarla. Al conectarla al ordenador la placa recibirá alimentación, por lo que se encenderá, y al mismo tiempo se nos proporcionará una interfaz de configuración. La BeagleBone Black arrancara desde la memoria interna de 2GB de la que dispone la placa o desde los 4GB de eMMC, aunque también es posible arrancar desde una tarjeta microSD.

La placa aparecerá en nuestro ordenador como una unidad flash, proporcionándonos una copia local de la documentación y de los drivers. No se debe utilizar esta interfaz para reconfigurar la microSD, pero si es posible actualizar los parámetros de arranque haciendo uso del archivo *uEnv.txt* .

Es necesario instalar los drivers en nuestro ordenador para habilitar el acceso "network-over-USB", que no es más que dar acceso a la placa a través de la conexión USB pero emulando una red IP. La IP 192.168.7.2 será la utilizada para acceder, y podremos acceder por ssh o vía navegador introduciendo en este la dirección http://192.168.7.2 . No es recomendable utilizar el navegador Internet Explorer.

Nosotros utilizaremos la imagen de Linux que viene por defecto con la placa, pero es posible descargar la imagen que se quiera desde la url https://beagleboard.org/latest-images e instalarla siguiendo el tutorial indicado en http://beagleboard.org/getting-started.

4.1.2 Librerías necesarias

Para el muestreo de la señal utilizaremos el ADC de la placa haciendo uso del PRU-ICSS(Programmable Real-Time Industrial Communication SubSystem). Para ello existen varias posibilidades, como utilizar código ensamblador proporcionado por el fabricante pero, nosotros utilizaremos una librería ya desarrollada que, a parte del acceso al ADC, nos proporciona acceso a todas las características del PRU.

Es digno de mención que es posible acceder al ADC de la BeagleBone Black sin hacer uso de este subsistema. No es una tarea difícil pero, por las características de este proyecto

los resultados del muestreo no son válidos. El problema de este tipo de acceso es el hecho de que hacemos uso del procesador principal de la placa. La explicación es la siguiente: el sistema operativo corre sobre este procesador y, cuando ejecutamos un programa para que realice el muestreo, el proceso que se crea puede no ser tan prioritario como otros procesos internos que estén sucediéndose en la placa. Es el sistema operativo el que le da prioridad a los procesos y el resultado de esto es que no tenemos la precisión necesaria, debido a que éste puede en un momento dado parar dicho proceso y atender a otro, resultando un tiempo entre muestras distinto en cada ejecución. Se debe tener en cuenta que tratamos de muestrear una señal analógica y no cualquier señal, la señal del corazón humano, por lo que una buena precisión es necesaria si queremos obtenerla correctamente, ya que existen muchos tipos de ruido que pueden interferir en esta, desde el ruido inherente a la línea eléctrica, hasta una pequeña contracción de los músculos del cuerpo. En el apartado experimentos se hace mención de las librerías utilizadas previas a la utilización del PRU de la placa.

La ventaja de utilizar este subsistema, es que opera de una manera totalmente independiente al procesador en el que se ejecuta el sistema operativo ya que cuenta con su propio microprocesador, por lo este problema queda solucionado.

La librería que nosotros vamos a utilizar se denomina *libpruio*. Es un driver que nos sirve para tratar con los procesadores ARM33xx, desarrollado sobre la BeagleBone, y que cubre el trato con las entradas analógicas y las entradas/salidas digitales. Está diseñada para una fácil configuración y el tratamiento de datos a alta velocidad. Esta librería corre sobre el *host*, el ARM, y en paralelo sobre el PRU, controlando los siguientes subsistemas:

- Módulo de control
- GPIO
- PWMSS(PWM SubSystem)
- TSC_ADC_SS(Touch Screen Controler and Analog-to-Digital Convertor SubSystem o simplemente ADC)

Esta librería ofrece tres posibles modos de ejecución:

- Modo IO: Para tratar con líneas analógicas y digitales, pero permitiendo que el ARM controle los tiempos.
- Modo RB: Para tratar con líneas analógicas y digitales, pero con el manejo de los tiempos y la precisión del muestreo del ADC controlado por el PRU.
- Modo MM: Para líneas analógicas y triggers opcionales, con los tiempos y la precisión del muestreo del ADC controlados por el PRU.

A continuación vamos a ver los requisitos que necesitamos para hacerla funcionar sobre nuestra placa. Aunque hay que tener en cuenta que el driver esta testeado con éxito en una placa BeagleBone Black bajo Ubuntu 13.10 y una imagen Debian 2014-08-05, hemos comprobado que funciona correctamente en nuestro sistema con el Linux que trae la

placa instalado. Pasamos a ver los pasos necesarios para la instalación. Nótese que algunos archivos serán descargados desde internet, por lo que es necesario conectar la Beagle a internet mediante el conector Ethernet.

- Instalación del compilador FreeBasic

Primero debemos descargarnos los archivos necesarios y descomprimirlos

```
wget http://www.freebasic-portal.de/dlfiles/452/BBB_fbc-1.00.tar.bz2
tar xjf BBB_fbc-1.00.tar.bz2
```

con eso ya tendremos en el directorio en el cual nos encontremos en ese momento una carpeta llamada BBB_fbc-1.00. Ahora debemos copiar estos archivos al directorio de librerías.

```
cd BBB_fbc-1.00
cp usr/local/bin/fbc /usr/local/bin/
cp -R usr/local/lib/freebasic /usr/local/lib/
```

Por último comprobamos la versión del compilador con la orden:

```
fbc -versión
```

En el momento en el que está escribiendo este documento la salida del comando es la siguiente:

```
FreeBASIC Compiler - Version 1.01.0 (10-14-2014), built for linux-arm (32bit)
Copyright (C) 2004-2014 The FreeBASIC development team.
```

- Instalar el driver de pruss

Antes de nada instalamos el paquete de ARM335x original:

```
apt-get install am335x-pru-package
```

Seguidamente seguimos con la instalación del paquete FB de FB prussdvr kit:

```
wget http://www.freebasic-portal.de/dlfiles/539/FB_prussdrv-0.0.tar.bz2
tar xjf FB_prussdrv-0.0.tar.bz2
```

A continuación copiamos los archivos necesarios para su funcionamiento en el directorio de librerías:

```
cd FB_prussdrv-0.0
mkdir /usr/local/include/freebasic/BBB
cp include/* /usr/local/include/freebasic/BBB
cp bin/pasm /usr/local/bin
```

- Instalación de la librería libpruio

Nos descargamos y descomprimimos el paquete de libpruio:

```
wget http://www.freebasic-portal.de/dlfiles/539/libpruio-0.2.tar.bz2
tar xjf libpruio-0.2.tar.bz2
```

Copiamos los archivos:

```
cd libpruio-0.2
cp src/c_wrapper/libpruio.so /usr/local/lib
ldconfig
cp src/c_wrapper/pruio*.h* /usr/local/include
cp src/config/libpruio-0A00.dtbo /lib/firmware
cp src/pruio/pruio*.bi /usr/local/include/freebasic/BBB
cp src/pruio/pruio.hp /usr/local/include/freebasic/BBB
```

El siguiente paso es activar el módulo PRUSS en la Beagle, ya que por defecto esta desactivado. Esto debemos realizarlo en cada arranque de la placa o antes de ejecutar un programa que haga uso de la librería, resultando en un fallo de segmentación si no se ha realizado este paso:

```
echo libpruio > /sys/devices/bone_capemgr.slots
```

Si esta orden fallara, es decir, al ejecutar nuestro código resultara en un fallo de segmentación o simplemente nos apareciese un fallo al ejecutar el comando, es debido a la versión de Linux que tenemos instalada. En ese caso el comando a utilizar es el siguiente:

```
sudo echo BB-BONE-PRU-01 > /sys/devices/bone_capemgr.*/slots
```

Con esto tendremos la placa lista para utilizar la librería libpruio. Nótese que algunos de los comandos utilizados en esta preparación requieren de acceso root al sistema operativo(apt-get, ldconfig, mkdir …/usr/…, cp …/usr/…, cp …/lib/…, echo …/sys/…).

Opcionalmente se utilizan otras librerías en un programa Python utilizado para mostrar gráficamente las capturas del ADC de la BeagleBone Black. La librería elegida es *matplotlib*. Esta es una librería o paquete de Python diseñada para mostrar gráficas 2D capaz de producir figuras de calidad en variedad de formatos.

Para su instalación podemos proceder de distintas maneras:

- Desde código fuente

Primero debemos descargarnos desde Github el código fuente y para ello existen dos caminos(siempre desde línea de comandos en la placa y con el cable Ethernet conectado):

```
git clone git@github.com:matplotlib/matplotlib.git
```

o

```
git clone git://github.com/matplotlib/matplotlib.git
```

Y una vez hecho, compilamos la librería con el siguiente comando:

```
cd matplotlib
python setup.py install
```

Si estamos bajo Debian/Ubuntu(como sería nuestro caso si no hemos modificado la imagen que viene preinstalada en la placa) podemos proceder con el siguiente comando:

```
sudo apt-get build-dep python-matplotlib
```

Una vez concluida la instalación podremos utilizar en nuestro código la librería importándola como cualquier paquete de Python.

4.2 Programas

Se han desarrollado dos programas principales. El primero de ellos se ha desarrollado en código C, utilizando la librería libpruio. Esta ofrece la posibilidad de utilizar más de una línea analógica como entrada del ADC, lo que nos será de utilidad para líneas futuras, pero para el alcance de este proyecto solo utilizamos una de ellas, que será la entrada analógica AIN0. El hilo de ejecución es sencillo:

- Incluimos las librerías necesarias
- Dentro del main:
    - Establecemos el número de muestras que vamos a escribir en el archivo de salida.
    - Establecemos la tasa de muestreo
    - Establecemos el número de entradas analógicas que vamos a tener en cuenta.
    - Establecemos el número de archivos de salida que vamos a crear y su nombre.
    - Creamos un nuevo driver.
    - Configuramos el acceso al ADC.
    - Configuramos el driver.
    - Comenzamos la ejecución de éste en modo RB(Mirar en la descripción de la librería).
    - Tomamos las muestras en los registros que tienen acceso a la línea analógica.
    - Creamos el archivo de salida.
    - Escribimos estas muestras en el archivo de salida.
    - Destruimos el driver.

- Cerramos el archivo de salida.
- Fin ejecución.

El resultado de esta ejecución es la creación de un archivo en los cuales se han guardado los datos obtenidos de los registros que tienen acceso al ADC, en binario. Estos datos serán tratados más tarde.

El código es el siguiente:

```
/** \file ADCsampling.c
\brief Example: fetch ADC samples in a ring buffer and save to file.
*/

#include "unistd.h"
#include "time.h"
#include "stdio.h"
#include "../c_wrapper/pruio.h"

//main function
int main(int argc, char **argv)
{
  const uint32 tSamp = 327680;  //!< The number of samples in the files (per step).
  const uint32 tmr = 50000;     //!< The sampling rate in ns (5000 -> 200 kHz).*/
  const uint32 NoStep = 1;       //!< The number of active steps (must match setStep calls and mask).
  const uint32 NoFile = 1;      //!< The number of files to write.
  const char *NamFil = "output.%u"; //!< The output file names.
  struct timespec mSec;
  mSec.tv_nsec=1000000;
  pruIo *io = pruio_new(PRUIO_DEF_ACTIVE, 0x98, 0, 1); //! create new driver
  if (io->Errr){
              printf("constructor failed (%s)\n", io->Errr); return 1;}

  do {
    if (pruio_adc_setStep(io, 9, 1, 0, 0, 4)){ //          step 9, AIN-0
```

```
        printf("step 9 configuration failed: (%s)\n", io->Errr); break;}


    uint32 mask = 1 << 9;         //!< The active steps (9 to 11).
    uint32 tInd = tSamp * NoStep; //!< The maximum total index.
    uint32 half = ((io->ESize >> 2) / NoStep) * NoStep; //!< The maximum index
of the half ring buffer.
    printf("mask:%u\n",mask);
    if (half > tInd){ half = tInd;}  //       adapt size for small files
    uint32 samp = (half << 1) / NoStep; //!< The number of samples (per step).
    printf("samp:%u\n", samp);
    if (pruio_config(io, samp, mask, tmr, 0)){ //       configure driver
                      printf("config failed (%s)\n", io->Errr); break;}


    if (pruio_rb_start(io)){
                    printf("rb_start failed (%s)\n", io->Errr); break;}


    uint16 *p0 = io->Adc->Value;  //!< A pointer to the start of the ring buffer.
    uint16 *p1 = p0 + half;         //!< A pointer to the middle of the ring
buffer.
    uint32 n;  //!< File counter.
    char fName[20];
    for(n = 0; n < NoFile; n++){
      sprintf(fName, NamFil, n);
      printf("Creating file %s\n", fName);
      FILE *oFile = fopen(fName, "wb");
      uint32 i = 0;               //!< Start index.
      while(i < tInd){
        i += half;
        if(i > tInd){             // fetch the rest(no complete chunk)
          uint32 rest = tInd + half - i;
          uint32 iEnd = p1 >= p0 ? rest : rest + half;
          while(io->DRam[0] < iEnd) nanosleep(&mSec, NULL);
          printf("  writing samples %u-%u\n", tInd -rest, tInd-1);
          fwrite(p0, sizeof(uint16), rest, oFile);
          uint16 *swap = p0;
```

```
          p0 = p1;
          p1 = swap;
        }
        if(p1 > p0) while(io->DRam[0] < half) nanosleep(&mSec, NULL);
        else        while(io->DRam[0] > half) nanosleep(&mSec, NULL);
        printf("  writing samples %u-%u\n", i-half, i-1);
        fwrite(p0, sizeof(uint16), half, oFile);
        uint16 *swap = p0;
        p0 = p1;
        p1 = swap;
      }
      fclose(oFile);
      printf("Finished file %s\n", fName);
    }
  } while(0);
  pruio_destroy(io);
  return 0;
}
```

De esta forma, en un solo archivo C conseguimos acceder de manera precisa al ADC de la BeagleBone Black, estableciendo la frecuencia de muestreo, hasta 200 KHz, de una manera sencilla.

Evidentemente estos datos no nos dan mucha información, de manera que hay que transformarlos de binario a un tipo de dato legible o al menos representable. Para este fin se desarrolla un pequeño programa en código Python. Dicho programa realiza dos tareas principales, una de ellas opcional:

- Crear un archivo, a partir del original en binario, en el que cada fila tenga dos valores, el número de muestra y su valor en decimal.
- Crea una imagen PNG en el cual representa en el eje X el número de muestra, y en el eje Y el valor correspondiente a la muestra(opcional).

El código del programa es el siguiente:

```
import struct  # Module to deal with binary data
import matplotlib as mpl  # Module for 2D graphics
```

```
mpl.use('Agg') ## To not plot directly. Using a non-interactive backend
import matplotlib.pyplot as plt  # pyplot function.For graphics plot


f = open("/route/to/binary_input_file","rb") #open input file
fcsv = open("/route/to/output_file","w") #create and open output file

binario = f.read() # reading input file

## 2 dictionaries. Values and sample number

valores = []
t = []


## Saving hexadecimal values converted from binary values

for i in range(len(binario)/2):
       valores.append(struct.unpack_from('H',binario[i*2]+binario[i*2+1]))

n = 0

## Saving hexadecimal and sample number

for i in range(len(valores)):
       valores[i] = valores[i][0]
       t.append(n)
       n += 1


## Writing values to output file

for i in range(len(valores)):
       fcsv.write(str(i) + "," + str(valores[i]) + "\n")
```

```
## Closing output file

fcsv.close()
f.close()

## Creating graphic figures and saving

plt.plot(t,valores)
plt.xlabel('x axis name')
plt.ylabel('y axis name')
plt.title('Set_title to figure')
plt.savefig("/route/to/fig_name")
plt.clf()  # Useful if you want to make more than one figure

print "Imagen guardada con exito.\nNumero de muestras: ",str(len(t))
```

4.3 Experimentos

Durante el desarrollo del proyecto se han ido realizando diferentes pruebas para alcanzar el objetivo. En este apartado se explicarán algunos de los caminos elegidos para al final mostrar la solución adoptada.

El objetivo era ser capaces de acceder al ADC de la BeagleBone Black para muestrear la señal proveniente de nuestra placa. En primera instancia eso no es algo difícil, ya que existen diversas librerías para realizar dicha función, pero el problema es que no están diseñadas para hacerlo con precisión, no son capaces de establecer un tiempo de muestreo, sino que ofrecen la capacidad de leer de una de los pines definidos como entradas analógicas. Otro inconveniente es que la mayoría de ellas están desarrolladas en Python, que al ser un lenguaje interpretado, es más lento, y menos preciso en tiempo de ejecución, aunque es cierto que bajo ese código Python puede encontrarse otro lenguaje, como el C, pero para el sistema operativo sigue siendo un lenguaje interpretado.

Otro motivo por el que no nos resultan validas esas librerías es que si realizamos un bucle de duración 5 segundos, por ejemplo, y contamos el número de iteraciones que se realizan, leyendo del ADC, podemos comprobar que como máximo conseguimos llegar a una frecuencia de 400 muestras por segundo, o lo que es lo mismo, seríamos

capaces de muestrear a unos 400Hz, lo que nos es totalmente insuficiente si queremos ser capaces de muestrear la señal proveniente de nuestro cuerpo, la cual si recordamos tenía una frecuencia máxima de 300Hz, con un número de muestras suficientes como para obtener toda la información necesaria de la señal y que pueda ser reconstruida. Esto último lo sabemos gracias al criterio de Nyquist, el cual nos dice que para poder reconstruir una señal y no perder información, es necesario que la frecuencia de muestreo sea como mínimo 2 veces mayor a la máxima frecuencia de la señal a muestrear. Además de esto, también debemos tener en cuenta el problema ya mencionado en la explicación de la instalación de la librería libpruio, referente a los diferentes tiempos de ejecución de los programas según la prioridad que el sistema operativo les concediera a los procesos que se ejecutan en ese momento.

No vamos a entrar en detalle en las librerías mencionadas, pero si vamos a nombrar las que hemos usado, ya que aunque para este proyecto no son las más adecuadas, pueden ser útiles cuando no sea necesaria tanta precisión en el tiempo de muestreo. Nótese que es posible acceder a las entradas analógicas de la Beagle, con los valores ya en digital, accediendo simplemente a un "archivo de texto", el cual es la salida de la conversión que realiza el ADC de las diferentes líneas analógicas, por lo que ejecutando el comando *cat* de Linux se nos muestra por pantalla el valor en digital que contiene dicha línea en el momento de ejecutar el comando. Algunas de las librerías de las que hablamos son Adafruit_BBIO o .

Ahora empezaremos con las pruebas realizadas gracias al programa en C desarrollado utilizando la librería *libpruio* y el script de Python.

Primero debemos conocer donde están situadas las entradas analógicas en la placa. Podemos apreciarlo en la siguiente figura.

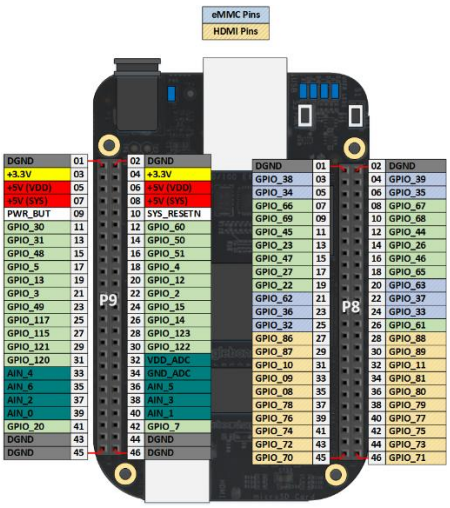

Fig. 36 BeagleBone Black pinout

Una vez que sabemos la posición de estas, debemos elegir una de ellas. Nosotros vamos a utilizar la línea AIN_0. A continuación nos aseguraremos de que en el programa en C se encuentre elegida la misma entrada analógica a la cual hayamos conectado la salida de nuestra placa PCB, en este caso la AIN_0 y ejecutaremos el programa. Para una mejor adquisición es conveniente que estemos aislados, para que no resulte ninguna interferencia en la señal.

Una vez terminada la ejecución del programa el resultado será un archivo con los datos de la adquisición en binario. Este archivo debemos convertirlo en datos que podamos ver y representar. Para ello utilizaremos el script Python indicado anteriormente. El resultado de dicha ejecución será un archivo csv con los datos por número de muestra y una imagen en la que se representan en el eje horizontal el número de muestra y en el eje vertical el valor de cada muestra.

En el siguiente ejemplo se ha muestreado la señal a 1KHz y tomando una muestra de 4 segundos. En la siguiente figura se puede apreciar la muestra original.

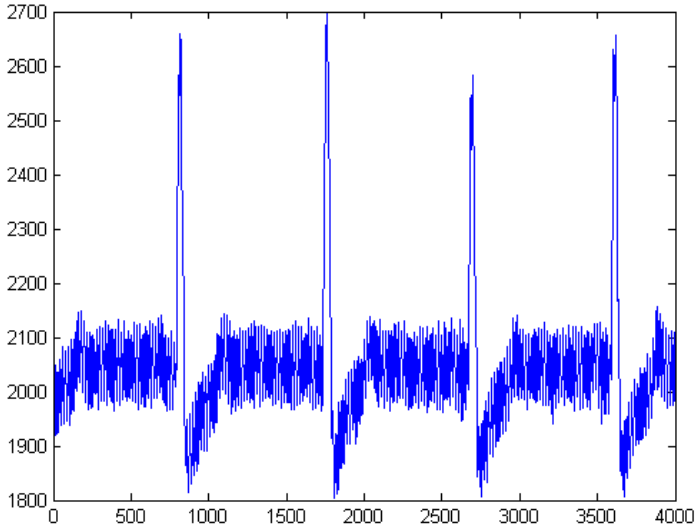

Fig. 37 1KHz 4 segundos

Podemos tratar de limpiar la señal un poco, ya que como puede apreciarse el ruido de la línea eléctrica a 50Hz está muy presente en la muestra. Con ayuda de Matlab y sus herramientas de DSP se han desarrollado 3 scripts, con los que haremos 3 cosas, una fft para ver el espectro de la señal, un filtro Notch a la frecuencia indicada para eliminarla de la muestra, y un filtro paso bajo de un orden suficiente para eliminar del todo las altas frecuencias, ya que como recordaremos del apartado del esquemático mediante hardware solo se ha implementado un filtro de orden 2(en total) con lo que no somos capaces de eliminar todas las componentes de alta frecuencia.

A continuación podemos ver el código que genera el espectro.

```
function espectro(senal,color)

if nargin < 2
    color =  'b';
end
pxx = pwelch(senal);
plot(10*log10(pxx),color)
end
```

Al aplicar esto a nuestro conjunto de datos originales el espectro resultado es el siguiente.

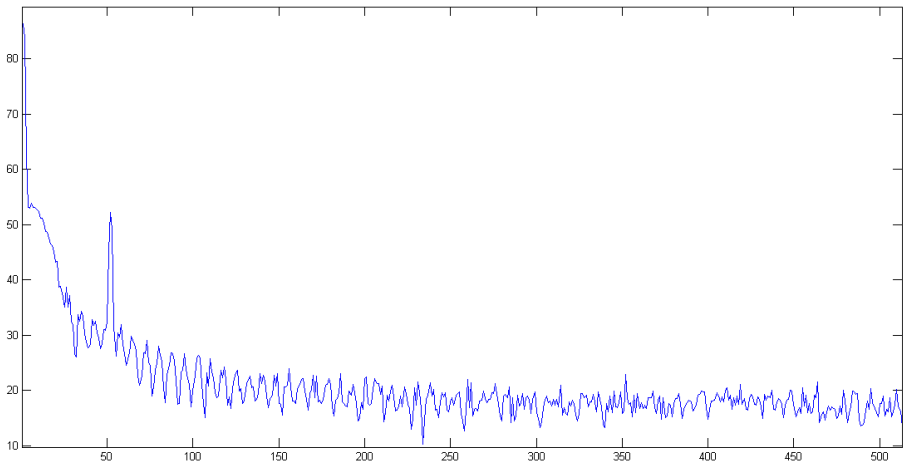

Fig. 38 Espectro señal original

En el espectro podemos apreciar perfectamente el pico de potencia a la frecuencia de 50Hz correspondiente al ruido de la línea eléctrica.

Para eliminarlo usaremos el siguiente código de Matlab.

```
function [b,a] = notchFilter(Fs,Fnotch)

BW     = 2;    % Bandwidth
Apass  = 30;   % Bandwidth Attenuation

[b, a] = iirnotch(Fnotch/(Fs/2), BW/(Fs/2), Apass);
```

```
end
```

El código anterior nos proporciona los coeficientes de un filtro notch centrado en la frecuencia *Fnotch* y para la frecuencia de muestreo *Fs*. Para aplicar dichos coeficientes a nuestra muestra utilizaremos el siguiente comando.

```
[b,a] = notchFilter(1000,50);
filtrada = filter(b,a,muestraOriginal);
```

Con esto tendremos una nueva muestra. Veamos ahora su espectro superpuesto con el de la muestra anterior.

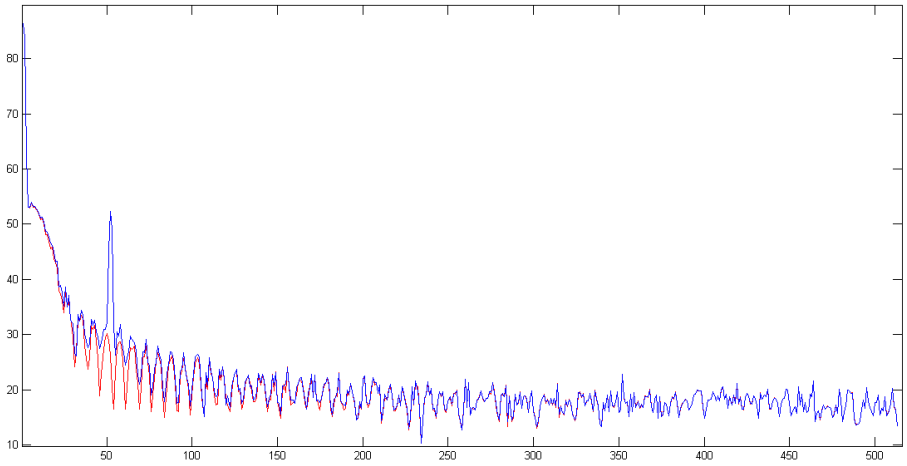

Fig. 39 Espectro Filtro Notch

En la figura puede apreciarse el espectro de la señal original(azul), con el espectro de la señal filtrada(rojo) superpuesto. Claramente hemos eliminado la potencia de la señal a 50Hz. Veamos cual es el resultado de este filtrado en la muestra.

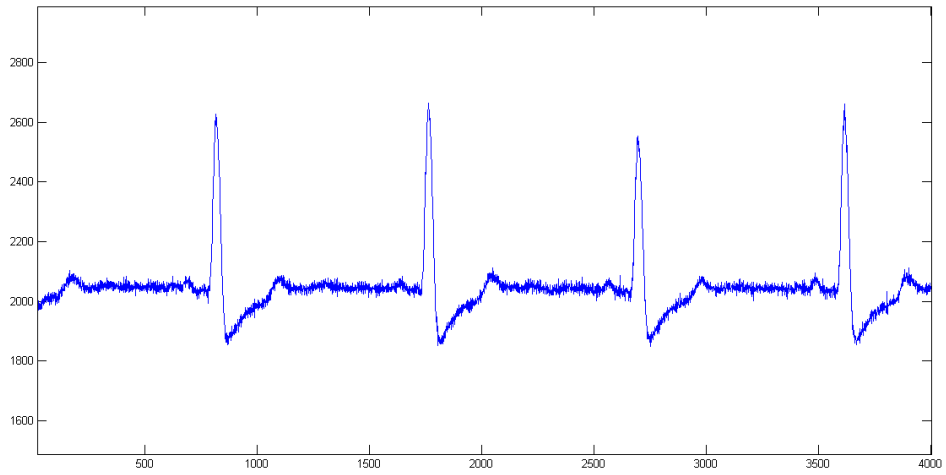

Fig. 40 1KHz 4 segundos filtro Notch 50Hz

Se puede apreciar perfectamente la diferencia con la primera muestra. Para mejorarlo aún más, decidimos aplicarle un filtro pasabaja de orden 4 para tratar de eliminar del todo las componentes de alta frecuencias. El código podemos verlo a continuación.

```
function b = pasobajo(Fs)

Fpass = 250;              % Passband Frequency
Fstop = 400;              % Stopband Frequency
Dpass = 0.057501127785;   % Passband Ripple
Dstop = 0.0001;           % Stopband Attenuation
dens  = 20;               % Density Factor

% Calculate the order from the parameters using FIRPMORD.
[N, Fo, Ao, W] = firpmord([Fpass, Fstop]/(Fs/2), [1 0], [Dpass, Dstop]);

% Calculate the coefficients using the FIRPM function.
b  = firpm(N, Fo, Ao, W, {dens});

end
```

Esta función nos generará los coeficientes del filtro que aplicando el mismo comando que para el filtro Notch, con la excepción de que los coeficientes “a” son inexistentes(a=1), podremos realizar el filtrado de la señal. Este es el resultado de dicho filtrado.

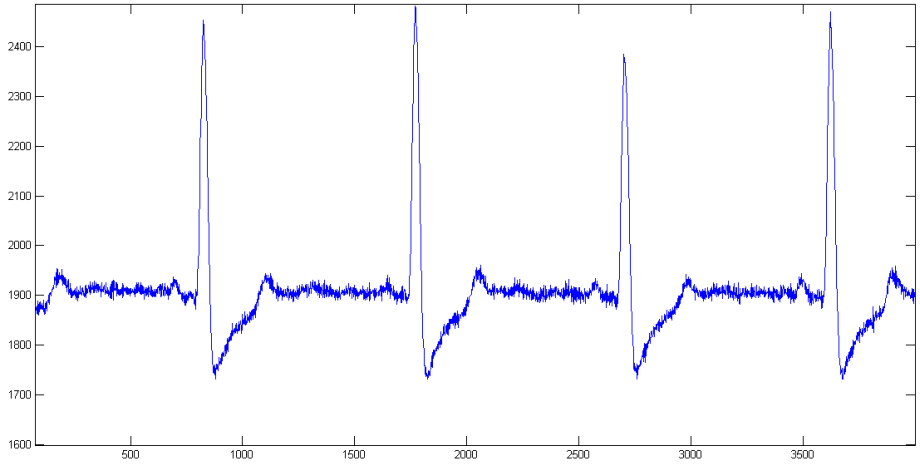

Fig. 41 1Khz 4segundos Filtro pasabaja y Notch

Se puede apreciar cierta mejoría, aunque no es demasiada significativa. Veamos ahora un ejemplo con un muestreo a 50KHz, en el cual la mejoría es más notable. En este caso se muestran las tres señales en la misma figura.

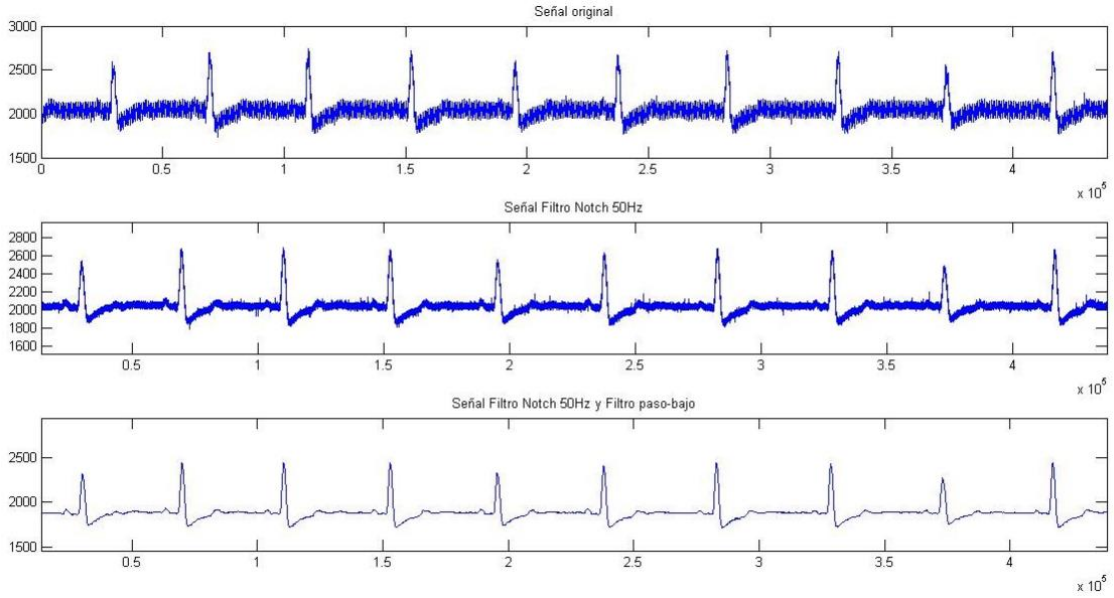

Fig. 42 50Khz Original,Filtro Notch y Filtro Pasabaja

5 Conclusiones y líneas futuras

En este proyecto de fin de carrera se ha diseñado una placa PCB de adaptación para la adquisición de la señal del electrocardiograma. Por otra parte también se ha configurado y programado la placa de desarrollo BeagleBone Black Rev C, trabajando con librerías que nos facilitan el acceso a periféricos con baja latencia, a través del microprocesador ARM3358 incluido en la placa. La idea era ser capaz de capturar la señal ECG proveniente del cuerpo humano para después representarla de una manera clara. Por consiguiente, el objetivo era crear un primer prototipo de electrocardiograma portátil para así facilitar un futuro trabajo relacionado con este tema.

Para poder abordar el tema adecuadamente se han estudiado las características eléctricas de la señal ECG así como las prestaciones ofrecidas por las diferentes placas de desarrollo del mercado. Para ello, una vez elegida la placa de desarrollo, la BeagleBone Black, se ha ido probando diferentes maneras de capturar la señal, obteniendo resultados errados al comienzo del proyecto llegando así a comprender mejor el funcionamiento tanto de la placa como de la señal ECG.

Durante el desarrollo de la PCB hemos tenido que revelar, taladrar, y soldar componentes. Con esto se ha ganado cierta experiencia con el manejo de equipos dedicados a este tema, como diferentes soldadores, métodos, accesorios, etc. Por otro lado se ha aprendido a identificar posibles fallos en el diseño, y a tener una visión del objetivo de cada una de las partes de la placa PCB.

Tras los fallos cometidos al comienzo del proyecto en la captura de la señal, el objetivo era ser capaz de acceder al PRU de la placa para así obtener una mayor precisión en dicha captura. Este es un sistema complejo difícil de abordar pero, gracias a la librería *libpruio* hemos sido capaces de obtener la precisión necesaria para nuestro proyecto.

Se deja el prototipo en manos de la Escuela Técnica Superior de Ingenieros para futuros trabajos. Algún ejemplo de estos podría ser la mejora de dicho prototipo, por ejemplo incluyendo la representación gráfica en tiempo real de la señal ECG en una pantalla LCD o mediante la instalación de un servidor web en la placa y su debida representación en un navegador. Este electrocardiógrafo puede ser la base para realizar un electrocardiógrafo fetal, incluyendo en el procesado de señal un código que implemente los algoritmos relacionados con técnicas de separación de señal comentadas en el estado del arte para el electrocardiograma fetal no invasivo.

6 Referencias

[1] http://www.atmel.com/images/atmel-8271-8-bit-avr-microcontroller-atmega48a-48pa-88a-88pa-168a-168pa-328-328p_datasheet_complete.pdf

[2] http://www.ti.com/product/AM3358#features

[3] https://beagleboard.org

[4] http://users.freebasic-portal.de/tjf/Projekte/libpruio/doc/html/

[5] http://matplotlib.org/index.html

[6] https://learn.adafruit.com/setting-up-io-python-library-on-beaglebone-black/adc

[7] http://www.urp.edu.pe/pdf/ingenieria/electronica/CIR-11_Electrocardiografo_de_Bajo_Costo.pdf

[8] https://www.arduino.cc/en/Main/ArduinoBoardUno

[9] https://www.raspberrypi.org/

[10] https://www.cooking-hacks.com/

[11] https://www.cooking-hacks.com/documentation/tutorials/ehealth-biometric-sensor-platform-arduino-raspberry-pi-medical

[11] http://www.fundaciondelcorazon.com/informacion-para-pacientes/metodos-diagnosticos/electrocardiograma.html

[12] http://www.ni.com/white-paper/5593/en/

[13] http://www.sciencedirect.com/science/article/pii/S0898122107005019

[14] http://beaglebone.cameon.net/home/reading-the-analog-inputs-adc

[15] https://www.linux.com/learn/how-get-analog-input-beaglebone-black

[16] http://beagleboard.org/pru

[17] http://exploringbeaglebone.com/chapter13/

[18] http://elinux.org/Ti_AM33XX_PRUSSv2

7 Índice de Figuras

Fig. 1 Arduino Uno 11
Fig. 2 Raspberry Pi 3 Model B 14
Fig. 3 Raspberry Pi 3 Model B 16
Fig. 4 Modelos BeagleBoard.org 18
Fig. 5 BeagleBoard-X15 19
Fig. 6 BeagleBoard-X15 TOP 19
Fig. 7 BeagleBoard-X15 BOTTOM 20
Fig. 8 Diagrama de Bloques AM5728 20
Fig. 9 BeagleBone Black Rev C 24
Fig. 10 BeagleBoard Black key components Block Diagram 24
Fig. 11 AM3358BZCZ Diagrama de Bloques 25
Fig. 12 Ondas ECG 33
Fig. 13 ECG real normal 34
Fig. 14 Esquema Amplificador basado en TL084 36
Fig. 15 Esquema final filtro paso-bajo 4to orden 37
Fig. 16 Esquema final filtro pasa-alto de 4to orden 37
Fig. 17 Esquema final Filtro Notch 4to orden 38
Fig. 18 Diagrama de bloques completo 38
Fig. 19 Señal ECG sistema final 39
Fig. 20 e-health Platform 39
Fig. 21 Esquematico ECG e-health 40
Fig. 22 Representacion ECG en LCD 40
Fig. 23 Esquemático circuito ECG 44
Fig. 24 Referencia y ECGref 44
Fig. 26 INA321 conexión 45
Fig. 27 INA321 contenido 46
Fig. 28 OPAx340 47
Fig. 29 Amplificador mayor ganancia 47
Fig. 30 Buffer y filtro final 48
Fig. 31 Esquemático Altium Designer 50
Fig. 32 PCB 51
Fig. 33 PCB 51
Fig. 34 Diagrama de bode. Magnitud 52
Fig. 35 Espectro 53
Fig. 36 FFT zona de mayor interés 53
Fig. 37 BeagleBone Black pinout 66
Fig. 38 1KHz 4 segundos 67
Fig. 39 Espectro señal original 68
Fig. 40 Espectro Filtro Notch 69
Fig. 41 1KHz 4 segundos filtro Notch 50Hz 70

Fig. 42 1Khz 4segundos Filtro pasabaja y Notch 71
Fig. 43 50Khz Original,Filtro Notch y Filtro Pasabaja..... 71