

UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA ELÉCTRICA



**DISEÑO E IMPLEMENTACIÓN DE UN ELECTROCARDÍOGRAFO, UTILIZANDO
UNA TARJETA FPGA DE 1-SOC PROGRAMADA EN C**

PRESENTADO POR:

OLIVER RABÍN PADILLA JIMÉNEZ

PARA OPTAR AL TÍTULO DE:

INGENIERO ELECTRICISTA

CIUDAD UNIVERSITARIA, SEPTIEMBRE DE 2021.

UNIVERSIDAD DE EL SALVADOR

RECTOR:

MSc ROGER ARMANDO ARIAS ALVARADO

SECRETARIA GENERAL:

ING. FRANCISCO ANTONIO ALARCÓN SANDOVAL

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DECANO:

PhD. EDGAR ARMANDO PEÑA FIGUEROA

SECRETARIO:

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERÍA ELÉCTRICA

DIRECTOR:

ING. ARMANDO MARTÍNEZ CALDERÓN

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA DE INGENIERÍA ELÉCTRICA

Trabajo de Graduación previo a la opción al Grado de:

INGENIERO ELECTRICISTA

Titulo

**DISEÑO E IMPLEMENTACIÓN DE UN ELECTROCARDIÓGRAFO, UTILIZANDO
UNA TARJETA FPGA DE1-SOC PROGRAMADA EN C**

Presentado por:

OLIVER RABÍN PADILLA JIMÉNEZ

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSc. e Ing. SALVADOR DE JESÚS GERMAN

San Salvador, Septiembre de 2021

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSc. e Ing. SALVADOR DE JESÚS GERMAN

NOTA Y DEFENSA FINAL

En esta fecha, miércoles 21 de julio de 2021, en la Sala de Lectura de la Escuela de Ingeniería Eléctrica, a las 2:00 p.m. horas, en presencia de las siguientes autoridades de la Escuela de Ingeniería Eléctrica de la Universidad de El Salvador:

1. Ing. Armando Martínez Calderón
Director


Firma



Firma



2. MSc. José Wilber Calderón Urrutia
Secretario

Y, con el Honorable Jurado de Evaluación integrado por las personas siguientes:

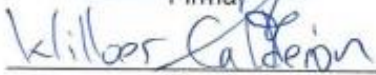
- MSC. SALVADOR DE JESÚS GERMAN
(Docente Asesor)


Firma

- DR. CARLOS OSMIN POCASANGRE JIMENEZ


Firma

- MSC. JOSE WILBER CALDERON URRUTIA


Firma

Se efectuó la defensa final reglamentaria del Trabajo de Graduación:

DISEÑO E IMPLEMENTACIÓN DE UN ELECTROCARDIÓGRAFO, UTILIZANDO UNA TARJETA FPGA DE1-SOC, PROGRAMADA EN C

A cargo del Bachiller:

- OLIVER RABÍN PADILLA JIMÉNEZ

Habiendo obtenido en el presente Trabajo una nota promedio de la defensa final:

8.9

(ocho. nueve)

AGRADECIMIENTOS

Después de mucho tiempo invertido en mi carrera universitaria estoy agradecido con Dios que me ha dado sabiduría, salud y las herramientas necesarias para culminar la carrera y este trabajo de graduación.

A mis padres Elsy Jiménez de Padilla y Bartolo Padilla que siempre me apoyaron a lo largo de toda mi formación académica, brindaron consejos y todo lo que necesité para completar tareas y proyectos durante la carrera, también porque siempre han tenido confianza en mí, en todas las decisiones que tomo y todo lo que hago por más complicado que parezca. Han sido mis bases de cómo me he formado, educado y estoy agradecido por eso.

A mi familia, hermanos, primos y abuelos que han mostrado interés en los pequeños proyectos que desarrolle durante la carrera y en mi trabajo de graduación, eso ha servido de motivación para completar con éxito todo lo que me propongo.

Al M. Sc. Ing. Salvador German que me dio la oportunidad de desarrollar este trabajo de graduación y confiarme parte del equipo utilizado en este trabajo, también la paciencia que tuvo ya que el proceso se dificulto debido a la pandemia pero el igual siguió asesorándome hasta finalizar el trabajo.

A Reina Vides que ha sido una parte fundamental para terminar con éxito este trabajo ya que me ayudo y facilito todo el proceso que conlleva la realización del trabajo y no dudo en ayudarme cuando lo necesité.

A mis amigos que estuvieron conmigo en las buenas y en las malas a lo largo de la carrera ayudándome cuando no entendía algo y brindando su apoyo cuando las cosas no parecían ir bien.

ÍNDICE GENERAL

INTRODUCCIÓN	1
CAPITULO 1. PLANTEAMIENTO DEL PROBLEMA.....	2
1.1 Definición del problema.	2
1.2 Objetivos:.....	3
1.2.1 Objetivo General:	3
1.2.2 Objetivos Específicos:.....	3
1.3 Justificación	4
1.4 Antecedentes.....	4
CAPITULO 2. MARCO TEORICO	5
2.1 El Miocardio	5
2.1.1 Funcionamiento eléctrico del corazón.....	6
2.1.2 Derivaciones del Electrocardiograma	10
2.1.2.1 Derivaciones de plano frontal.....	10
2.1.2.2 Derivaciones bipolares estándar.....	10
2.1.2.3 Derivaciones monopolares aumentadas.....	11
2.1.3 Derivaciones del plano horizontal.....	12
2.2 Muestreo de una señal	14
2.3 Filtrado digital de una señal.....	16
CAPITULO 3. DISEÑO DEL ECG	20
3.1 Diagrama conceptual del ECG.	20
3.2 Sistema de Adquisición de señal.	20
3.3 Fuente Bipolar	21
3.4 Pre-amplificación: Amplificador de instrumentación	21
3.4.1: Buffer.....	21

3.4.2: Terminal Central de Wilson.....	22
3.4.3 AD620AN	23
3.4.4 Filtrado	23
3.4.4.1 Filtro paso bajo	24
3.4.4.2 Filtro paso alto	24
3.5 Circuito de pierna derecha.....	25
3.5.1 AD7533	25
3.6 Inversor.....	26
3.7 Circuito implementado.	27
3.8 Tecnología utilizada	29
3.8.1 FPGA	29
3.8.1.1 DE1-SoC.	29
3.8.1.2 Componentes de la placa DE1-SoC.....	30
3.8.1.2.1 Temporizadores.	30
3.8.1.2.2 Temporizadores de intervalos (Interval Timer).....	31
3.8.1.2.3 Puertos paralelos.	33
3.8.1.2.4 Interrupciones	33
3.8.1.2.5 Puerto paralelo de LED rojos.	34
3.8.1.2.6 Puerto paralelo de Pantallas de 7 segmentos.....	34
3.8.1.2.7 Puerto paralelo de interruptores deslizables.	35
3.8.1.2.8 Puerto paralelo de KEY (pulsadores).	35
3.8.1.2.9 Puerto VGA.....	36
3.8.1.2.9.1 Pixel Buffer	36
3.8.1.2.9.2 Character Buffer	38
3.8.1.2.10 ADC	38

3.8.1.2.11 Memoria	39
3.8.1.2.11.1 SDRAM.....	39
3.8.1.2.11.2 On-Chip Memory	39
3.8.1.2.11.3 On-Chip Memory Character Buffer	39
3.8.1.2.12 Tabla resumen de direcciones base de todos los componentes en la placa DE1-SoC.....	40
3.9 Formas de programar la placa DE1-SoC.....	41
3.10 Instalación de Linux en la placa DE1-SoC (FPPx32).....	43
3.10.1 Linux en la placa DE1-SoC.	43
3.10.2 Montando la imagen de Linux en tarjeta microSD.....	43
3.10.3 Booteando Linux en placa DE1-SoC.....	44
3.10.4 Accesando a la interfaz de línea de comando vía terminal UART.....	45
3.10.4.1 En una PC con Windows.	45
3.10.5 Cargado el <i>Computer System</i>	47
3.10.5.1 Pasos para crear archivo *.rbf	47
3.10.5.2 Cargando *.rbf al sistema operativo.....	49
3.11 Herramienta Qsys.....	50
3.11.1 Computer System.....	53
CAPITULO 4. IMPLEMENTACION	57
4.1 Sistema de Procesado de señal	57
4.1.2 Sistema de adquisición de datos.....	57
4.2 Filtrado digital	57
4.3 Procesado.....	59
4.3.1 Almacenamiento.....	59
4.3.2 Detección de complejo QRS	59

4.4 Pruebas utilizando periféricos de placa DE1-SoC	61
4.5 Experimentación en breadboard.....	67
4.6 Experimentación e Implementación en PCB (Parte I).....	69
4.7 Lógica de programación.	71
4.8 Experimentación e Implementación en PCB (Parte II).....	72
4.9 Resultados.....	75
4.10 Experimentación e Implementación en PCB (Parte III).....	80
4.11 Resultados.....	81
Capítulo V: PRESUPUESTO	87
CONCLUSIONES	88
RECOMENDACIONES	89
BIBLIOGRAFÍA	90
ANEXOS.	93
Anexo A: Esquemático completo.	93
Anexo B: Mascara de componentes.	96
Anexo C: PCB Top	97
Anexo D: PCB Bottom	98

ÍNDICE DE FIGURAS

Figura 1: Estructura del Corazón [3].....	5
Figura 2: Onda de despolarización y repolarización que un ECG registra. [4]	6
Figura 3: Sistema de conducción del corazón [5].....	7
Figura 4: Generación de la onda P. [4].....	7
Figura 5: Nódulo AV [4].....	8
Figura 6: Complejo QRS [4]	8
Figura 7: Etapa de Repolarización Ventricular [4]	9
Figura 8: Medidas del papel de electrocardiograma [6].....	9
Figura 9: Ubicación de las derivaciones: plano frontal y horizontal. [7].....	10
Figura 10: Derivaciones de extremidades y precordiales. [7].....	11
Figura 11: Derivaciones precordiales y ubicación de electrodos. [7].....	12
Figura 12: Ejemplo de muestreo de una onda sinusoidal. [8].....	15
Figura 13: muestreo de una señal a diferentes resoluciones. [8]	15
Figura 14: Ejemplo de una señal muestreada. a) original, b) digitalizada. [8]	16
Figura 15: Esquema de una función de transferencia $F(s)$. [Elaboración propia]. .	17
Figura 16: Filtro paso bajo primer orden. [Elaboración propia].....	18
Figura 17: Diagrama Conceptual del ECG. [Elaboración propia].	20
Figura 18: Diagrama de sistema de adquisición de datos. [Elaboración propia]. ..	20
Figura 19: Circuito fuente bipolar, utilizando LM7X12. [9]	21
Figura 20: seguidor de tensión para IRA, entrada mano derecha. [Elaboración propia].	21
Figura 21: Terminal central de Wilson con resistencias de $10k \pm 2\%$. Fuentes: elaboración propia.....	22
Figura 22: Amplificador de instrumentación AD620AN. [10]	23
Figura 23: Diagrama de filtro paso bajo. [Elaboración propia].....	24
Figura 24: Diagrama de filtro paso alto. [Elaboración propia].....	24
Figura 25: circuito de pierna derecha. [Elaboración propia]	25
Figura 26: DAC AD7523. [11].....	25
Figura 27: Etapa amplificadora. [Elaboración propia].....	26
Figura 28: Etapa de referencia. [Elaboración propia]	27

Figura 29: Etapa de buffer, WTC y pierna derecha. [Elaboración propia]	27
Figura 30: Etapa pre-amplificadora. [Elaboración propia]	28
Figura 31: Etapa amplificadora. [Elaboración propia]	28
Figura 32: Diagrama en bloques de placa DE1-SoC [12].....	29
Figura 33: Diagrama en bloques de DE1-SoC [13]	30
Figura 34: Registros del reloj privado del ARM A9. [13].....	31
Figura 35: Registros del reloj de intervalos del HPS. [13]	32
Figura 36: Registros del puerto paralelo en DE1-SoC. [13]	34
Figura 37: Puerto paralelo de Salida para LEDR. [13]	34
Figura 38: Localización de bits para pantalla 7-Segmentos. [13]	35
Figura 39: Registro de datos en SW9-0. [13]	35
Figura 40: Registros utilizados en pulsadores. [13].....	36
Figura 41: Coordenadas del Pixel Buffer. [13].....	36
Figura 42: Direccionamiento del pixel y valores de colores. [13].....	37
Figura 43: Registros de control del Pixel Buffer. [13]	37
Figura 44: Direcciones y coordenadas del Character Buffer. [13].....	38
Figura 45: Direcciones del ADC y disposición de pines en DE1-So. [13].....	38
Figura 46: Switches para MSEL. [13].....	41
Figura 47: WIN32 Disk Imager. [Elaboración propia].	43
Figura 48: Win32 Disk Imager, una vez instalado el SO. [Elaboración propia].	44
Figura 49: Configuración de switches del MSEL en DE1-SoC. [13].....	44
Figura 50: Determinación de asignación de COM para USB a UART en el Administrador de Dispositivos. [Elaboración propia]	45
Figura 51: Ventana de Putty, configuración para comunicación serial. [Elaboración propia]	46
Figura 52: Inicio de sección como administrador de Linux en la interfaz de comandos. [Elaboración propia]	47
Figura 53: Conversion de *.sof a *.rbf. [Elaboración propia].....	48
Figura 54: Conversion de *.sof a *.rbf (2). [Elaboración propia]	48
Figura 55: carga de *.rbf y archivos visibles de la SD con Linux instalado. [Elaboración propia].	49

Figura 56: sustitución de *.rbf. [Elaboración propia].	49
Figura 57: *.rbf sustituido. [Elaboración propia].	50
Figura 58: Sistema brindado por Intel. [13]	51
Figura 59: Sistema a implementar en DE1-SoC. [Elaboración propia].	52
Figura 60: Computer System Qsys. [Elaboración propia].	53
Figura 61: VGA Subsystem Qsys. [Elaboración propia]	55
Figura 62: Esquema de procesamiento de señal. [Elaboración propia]	57
Figura 63: Caracterización de filtro FIR a implementar. [Elaboración propia]	58
Figura 64: Exportado de coeficientes. [Elaboración propia]	58
Figura 65: archivo de texto generado por FIR Filter Designer. [Elaboración propia]	58
Figura 66: Registro típico de una señal de ECG. [15]	59
Figura 67: Espectro en frecuencia de actividad eléctrica del corazón. [16].	60
Figura 68: Ejecución de código ejemplo de uso de puerto VGA. [Elaboración propia]	61
Figura 69: División de pantalla en 4 partes iguales. [Elaboración propia]	62
Figura 70: Generación de líneas en la fila y columna de preferencia. [Elaboración propia]	63
Figura 71: Medición del estado de los periféricos de la FPGA, LEDs, Display, y pulsadores y contador 60s. [Elaboración propia].	64
Figura 72: Lecturas del conversor analógico digital de la FPGA. [Elaboración propia]	65
Figura 73: Circuito de fuente de voltaje de $\pm 12V$. [Elaboración propia]	65
Figura 74: Implementación de ECG en Breadboard. [Elaboración propia].	66
Figura 75: Electrodo a utilizar cuando el sistema esté completo. [Elaboración propia]	66
Figura 76: Cables a utilizar para transmitir las señales desde los electrodos al sistema de adquisición de datos. [Elaboración propia].	67
Figura 77: Frecuencia de corte filtro paso bajo 39 Hz. [Elaboración propia]	67
Figura 78: Señal obtenida derivación DI implementado en breadboard. [Elaboración propia]	68

Figura 79: Circuito en PCB Finalizado. [Elaboración propia].....	69
Figura 80: Primer prueba de obtención de señal 50 muestras/s, graficas de arriba hacia abajo: DIII, DI y DII. [Elaboración propia].....	70
Figura 81: Segunda prueba de obtención de seña 100 muestras/s, graficas de arriba hacia abajo: DI, DII y DIII. [Elaboración propia].....	71
Figura 82: archivos del programa ECG. [Elaboración propia]	71
Figura 83: direccionamiento de la base de los ADC. [Elaboración propia].....	72
Figura 84: Disposición de conexiones para experimentación. [Elaboración propia]	74
Figura 85: Disposición de pines de salida de PCB. [Elaboración propia]	74
Figura 86: Disposición de pines del ADC de DE1-SoC. [Elaboración propia]	75
Figura 87: Grafica ADC1 - Derivación V2. [Elaboración propia].....	75
Figura 88 Grafica ADC2 - Derivación V3. [Elaboración propia].....	76
Figura 89 Grafica ADC3 - Derivación V4. [Elaboración propia].....	76
Figura 90 Grafica ADC4 - Derivación V5. [Elaboración propia].....	77
Figura 91 Grafica ADC5 - Derivación V6. [Elaboración propia].....	77
Figura 92 Grafica ADC6 - Derivación V1. [Elaboración propia].....	78
Figura 93: Formas de onda de un ECG normal. [Elaboración propia].....	78
Figura 94: Interfaz CLI durante la ejecución del programa. [Elaboración propia]..	79
Figura 95: contenido del archivo *.csv generado. [Elaboración propia].....	79
Figura 96: contenido del archivo *.csv generado. [Elaboración propia].....	81
Figura 97: Grafica ADC3 (DII) filtrada (rosa) y elevada al cuadrado (cyan). [Elaboración propia]	82
Figura 98: Grafica ADC3 (DII) filtrada (rosa) y elevada al cuadrado (cyan), pico detectado (azul). [Elaboración propia].....	82
Figura 99: Gráfica de AVF en ambos colores. [Elaboración propia].....	83
Figura 100: Gráfica de AVL en ambos colores. [Elaboración propia].....	84
Figura 101: Grafica de AVR en ambos colores. [Elaboración propia]	84
Figura 102: Gráfica de DI en ambos colores. [Elaboración propia]	85
Figura 103: Gráfica de DII en ambos colores. [Elaboración propia]	85
Figura 104: Gráfica de DIII en ambos colores. [Elaboración propia]	86

ÍNDICE DE TABLAS

Tabla 1: Cálculos efectuados por derivación. [Elaboración propia].	14
Tabla 2: Promedios de potenciales entre extremidades. [Elaboración propia].	22
Tabla 3: Ubicación de Registros de FPGA DE1-SoC. [13].....	40
Tabla 5: Conexiones entre DE1-SoC y PCB. [Elaboración propia]	75
Tabla 6: Valores promedio de frecuencia cardiaca. [17]	80
Tabla 7: Conexiones entre DE1-SoC y PCB. [Elaboración propia]	80
Tabla 8: Presupuesto. [Elaboración propia]	87

GLOSARIO

ADC: del inglés Analog to Digital Converter o convertidor analógico digital es un dispositivo electrónico que convierte una señal analógica ya sea de tensión o corriente, en una señal digital tomando muestras de la señal analógica a intervalos regulares con el propósito de facilitar su procesamiento.

CSV: del inglés Comma Separated Values o valores separados por comas es un archivo de texto en el cual sus caracteres están separados por comas, formando un tipo de tabla con filas y columnas, siendo las columnas separadas por comas y las filas por un salto de línea (enter).

DC: del inglés Direct Current o corriente continua es un flujo unidireccional de carga eléctrica, en el cual los electrones siempre viajan en la misma dirección y la cantidad de electrones es constante en el tiempo.

DE1-SOC: es una placa de desarrollo diseñada por Terasic que presenta una plataforma de diseño robusto utilizando un Chip de Altera, el SoC (System on a chip) que permite correr sistemas operativos ligeros, cuenta con variedad de periféricos que pueden ser controlados desde el sistema operativo en tiempo real.

ECG: Electrocardiogram o electrocardiograma, es un dispositivo que registra las señales eléctricas del corazón para luego graficar estas señales en un papel milimetrado. Se utiliza para determinar si existe algún problema eléctrico con el corazón de la persona viendo el ritmo y fuerza de los impulsos que este genera.

FPGA: del inglés Field Programmable Gate Arrays o matriz de puertas lógicas programables, son circuitos integrados reconfigurables que cuentan con interconexiones que pueden ser programadas mediante la combinación de bloques lógicos programables.

HPS: del inglés Hard Processor system, es una unidad de procesamiento dedicado incluido en la placa DE1-SoC (procesador ARM Cortex-A9), en este se, ejecutan las funciones, controlan periféricos, corre el sistema operativo e interactúa con la FPGA incluida en el chip Cyclone V.

QRS: conocido como complejo QRS es la representación gráfica de la despolarización de los ventrículos del corazón formando una estructura picuda en el ECG (electrocardiograma).

VGA: del inglés Video Graphics Array o matriz de gráficos de video es un estándar de puerto para dispositivos de video tales como: monitores, proyectores y televisores. Proporciona una visualización en color con una resolución de 640 x 480 pixeles, con una tasa de refresco de 60 Hz y 16 colores.

WCT: del inglés Wilson Central Terminal o terminal central del Wilson, es el promedio de los potenciales de cada una de las extremidades, es decir, obtiene el potencial absoluto en el que se encuentra el corazón y permite obtener las derivaciones precordiales V1 a V6.

INTRODUCCIÓN

En el periodo de enero a junio de 2020 en la Red de Hospitales MINSAL se registraron 462 muertes por enfermedades relacionadas al corazón [1] esto puede estar directamente relacionado a la capacidad que tienen los hospitales de estar equipado con instrumentos que permitan obtener información valiosa del órgano en cuestión (el corazón).

El tener o no tener estos aparatos de medición puede ser la diferencia entre que el paciente posea mejor pronóstico de vida o que sin el diagnóstico oportuno las enfermedades relacionadas al corazón tiendan a empeorar y atraer consecuencias fatales para el que lo padece.

La red de Hospitales de El Salvador posee equipo como los electrocardiógrafos que brindan los electrocardiogramas y a partir de allí poder dar un diagnóstico, el problema de esto es el elevado costo que conlleva la adquisición de un dispositivo de estos a lo cual se presenta una opción que permita obtener los mismos datos que un electrocardiógrafo usado en la red medica pero con la diferencia de que su costo se vea reducido en gran manera para que este sea asequible y poder equipar las zonas rurales que normalmente son las más afectadas por la falta de equipamiento médico.

El proyecto consiste en diseñar un ECG (electrocardiógrafo/electrocardiograma) que funcione con 10 electrodos y a partir de estos obtener los datos de las distintas derivaciones que se procesan habitualmente, para su implementación se debe entender el funcionamiento básico eléctrico del corazón, así como la forma en que se obtienen estos datos y se procesan.

CAPITULO 1. PLANTEAMIENTO DEL PROBLEMA

1.1 Definición del problema.

De los datos obtenidos del portal de transparencia del Ministerio de Salud, se puede observar que las enfermedades cardiacas ocupan el 8º y 9º [2] lugar de causas de mortalidad en el país, estas estadísticas han sido afectadas ya que antes ocupaba un lugar más alto, pero es debido a la pandemia que se ha vivido a lo largo del 2020.

Estos decesos podrían ser reducidos de alguna manera si se pudiera diagnosticar el problema con más anticipación para evaluar la posibilidad de una operación o algún tratamiento del problema, el inconveniente hasta este momento siempre es la escases de equipo especializado para monitorear el corazón, esto debido a la capacidad de adquisición que tienen los hospitales y las unidades de salud es baja lo que se refleja en el presupuesto asignado de las mismas se limitan a poseer unos cuantos aparatos disponibles por lugar, siendo estos muy pocos para atender a la cantidad de pacientes y en un periodo corto de tiempo, donde se sabe que el tiempo es lo más valioso al momento de detectar enfermedades relacionadas al corazón.

1.2 Objetivos:

1.2.1 Objetivo General:

- Implementar un electrocardiógrafo utilizando sistemas embebidos como lo es una FPGA (Field programmable gate arrays o matriz de puertas lógicas programables) DE1-SoC para registrar los impulsos eléctricos del corazón.

1.2.2 Objetivos Específicos:

- Diseñar las etapas de un electrocardiógrafo desde la obtención de datos hasta el procesado de las señales obtenidas tomando en cuenta los parámetros estándar de un ECG ordinario.
- Utilizar el puerto VGA (video graphics array o matriz grafica de videos) para controlar los parámetros del ECG y a su vez mostrar en tiempo real las mediciones realizadas.
- Mostrar de forma impresa en pantalla el registro de señales procesadas de la FPGA en un electrocardiograma.
- Hacer uso de diversas herramientas para procesar la señal tales como filtrado digital y uso de funciones para detectar el complejo QRS y poder mostrar el ritmo cardiaco de un paciente.
- Instalar una versión de Linux “DE1-SoC_UP_Linux, Kernel: 3.18” para poder explotar de la mejor manera las características que provee la FPGA y contar con el software más actual disponible para la implementación del ECG.

1.3 Justificación

Universidades de diferentes países del mundo desarrollan proyectos de gran envergadura en sistemas embebidos como microcontroladores y FPGA, La Escuela de Ingeniería Eléctrica de Universidad de El Salvador busca explorar e implementar estos sistemas en el área médica (en este caso) y actualizar la tecnología presente en los laboratorios.

1.4 Antecedentes

El uso de las FPGA en trabajos de graduación se concentra en dos “Principios del FPGA y aplicaciones en el control de procesos industriales (Autor: Benjamín Antonio Robles Rivas, 2016)” y “Diseño de un controlador PID, utilizando una tarjeta FPGA Cyclone V GX Starter, programada en VHDL (Autores: José Raúl Mejía Nuila y Guillermo Alfonso Maximiliano Narváez Henríquez, 2017)”. Existe un trabajo de graduación titulado “Desarrollo de un dispositivo confiable y de bajo costo para adquirir señales electro cardíacas, con capacidad de almacenamiento de datos y generación de registros (Autor: Alvaro Enrique Mártir Rodríguez, 2016)” donde se realiza un electrocardiograma utilizando tecnología Arduino con 3 electrodos siendo necesario realizar varias mediciones para visualizar el corazón en sus tres derivaciones bipolares, de este modo en este trabajo se realizara con 10 electrodos donde será necesaria una sola medición para obtener datos de todos los ángulos para dar un diagnóstico.

CAPITULO 2. MARCO TEORICO

2.1 El Miocardio

El corazón (miocardio) es el encargado de distribuir la sangre en todo el cuerpo, para lograrlo el corazón debe ser polarizado y despolarizado continuamente para lograr su fin. Para poder polarizarse y despolarizarse el corazón es excitado a través de pequeños impulsos eléctricos que pueden ser medidos como pequeñas variaciones de voltajes utilizando instrumentos como el electrocardiógrafo.

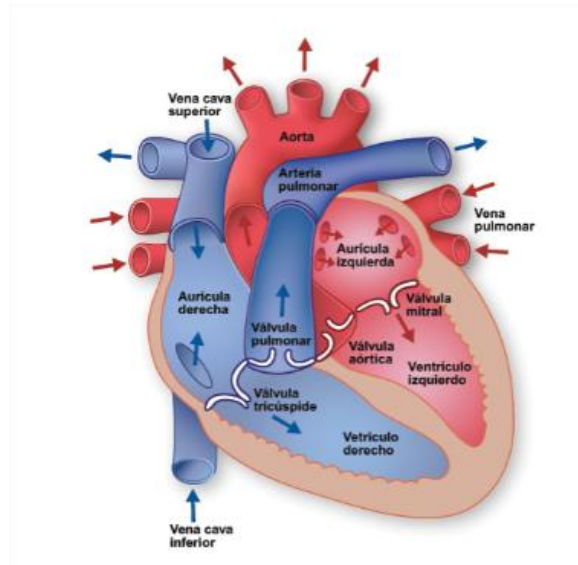


Figura 1: Estructura del Corazón [3]

Dentro del corazón existen 4 cavidades y 4 válvulas encargadas de generar el flujo de sangre (Ver Figura 1), son 2 aurículas o atrios (izquierda y derecha) y 2 ventrículos (izquierdo y derecho) también las válvulas tricúspide, pulmonar, aortica y mitral que permiten el paso de sangre en una sola dirección.

El corazón puede ser visto como dos partes que trabajan en conjunto:

Corazón derecho (Ver Figura 1): siendo sus partes la aurícula y ventrículo derecho se encargan de recibir la sangre (aurícula) que acaba de recorrer el cuerpo y se envía a través del ventrículo y la arteria pulmonar hacia los pulmones para ser oxigenada nuevamente.

Corazón izquierdo (Ver Figura 1): siendo sus partes la aurícula y ventrículo izquierdo se encargan de recibir la sangre (aurícula) ya oxigenada que proviene de los pulmones y el ventrículo los reparte a través de la aorta hacia el resto del cuerpo.

2.1.1 Funcionamiento eléctrico del corazón.

Las células que conforman el musculo del corazón son los miocitos estas poseen carga negativa cuando están en reposo (polarizadas) cuando ocurre la despolarización su carga se vuelve positiva debido a los iones de Na^+ , en este momento los miocitos se contraen lo que causa la contracción en el corazón. El fenómeno de despolarización y repolarización es medible y se puede obtener el perfil eléctrico característico de este proceso que se muestra en la Figura 2.

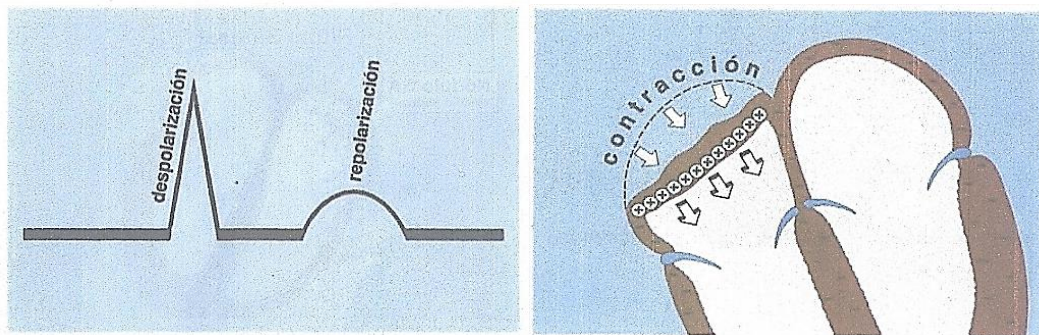


Figura 2: Onda de despolarización y repolarización que un ECG registra. [4]

Al ser una onda de despolarización, avanza a lo largo del corazón haciendo que el músculo se contraiga progresivamente más adelante se detallara como es que se dispersa la onda a través del corazón.

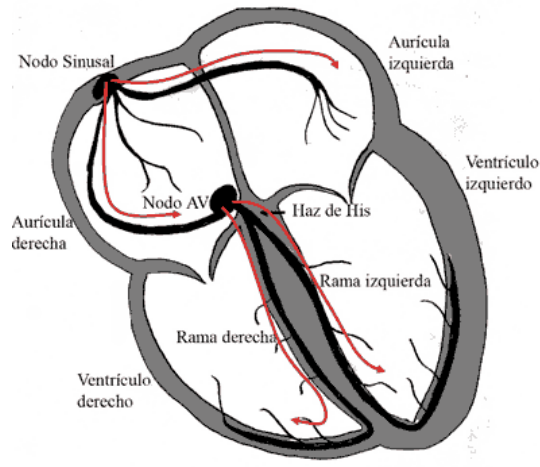


Figura 3: Sistema de conducción del corazón [5].

El ritmo cardiaco es controlado a través del Nódulo Sinusal (SA) (Ver Figura 3) éste es el origen de la onda de despolarización y su actividad se conoce como Ritmo Sinusal, este está localizado en la parte superior posterior de la aurícula derecha. Al iniciar la onda de despolarización esta contrae ambas aurículas y al repolarizar las relaja. Este proceso de contracción se refleja en el ECG como una flexión hacia arriba y se le conoce como “Onda P” (Ver Figura 4).

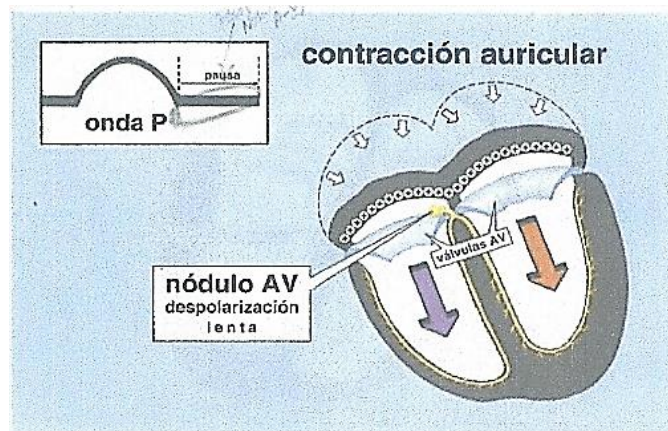


Figura 4: Generación de la onda P. [4]

El impulso eléctrico generado en el Nodo Sinusal SA viaja a través de tres tractos o conductores hacia el nodo AV: tracto intermodal anterior medio y posterior (Ver Figura 5).

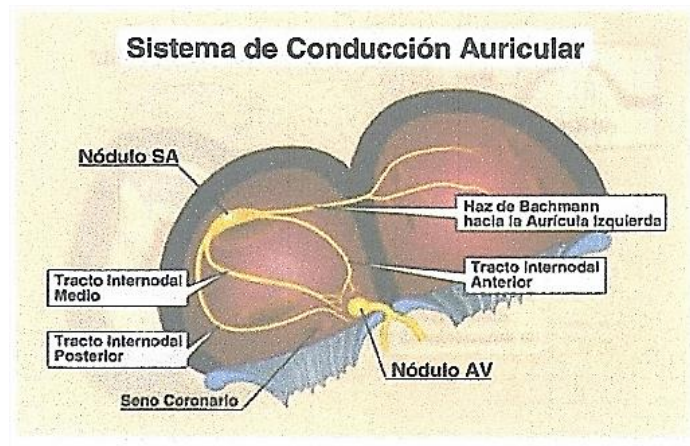


Figura 5: Nódulo AV [4]

Al llegar la onda al nodo AV ocurre una breve pausa que permite que la sangre de las aurículas llegue a los ventrículos para continuar el proceso de despolarización a través del “Haz de His” que despolariza los ventrículos rápidamente debido a que los haces de rama izquierda y derecha son partes del corazón muy conductivas gracias a las fibras de Purkinje.

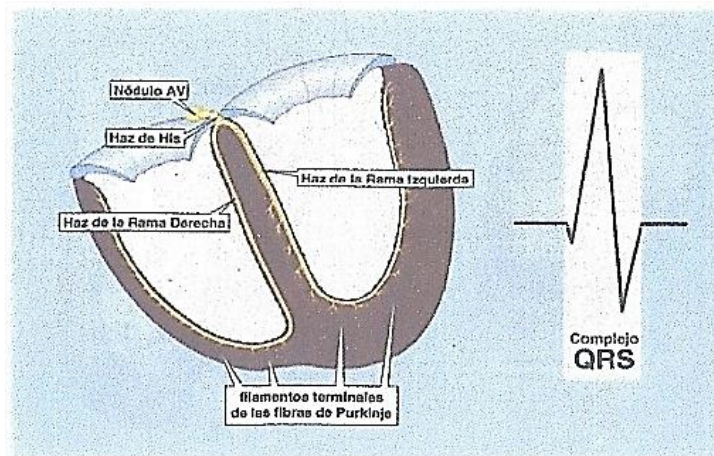


Figura 6: Complejo QRS [4]

Al ocurrir la despolarización completa a través de los haces de rama se produce lo que se conoce como complejo QRS que representa la despolarización ventricular (Ver Figura 6).

Finalmente, el corazón debe repolarizarse, esta repolarización inicia después del complejo QRS y finaliza hasta que termina la onda T (Ver Figura 7).

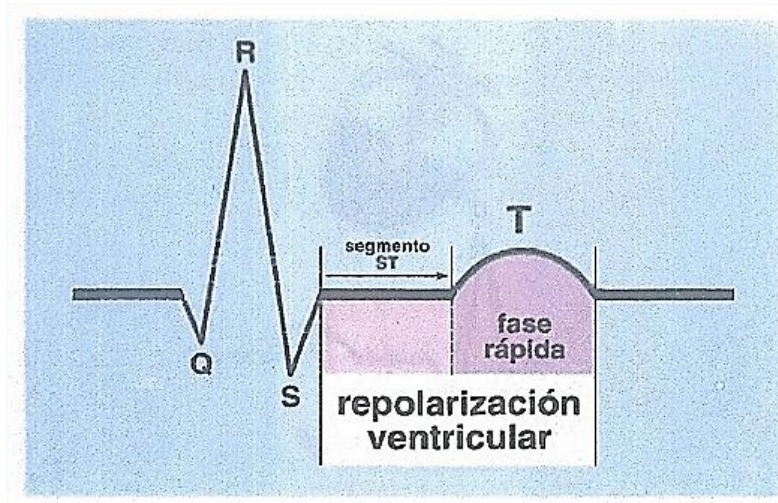


Figura 7: Etapa de Repolarización Ventricular [4]

El ECG se registra en una tira de papel cuadrículado, esta cuadrícula es de 5mm x 5mm con divisiones de 1mm (Ver Figura 8).

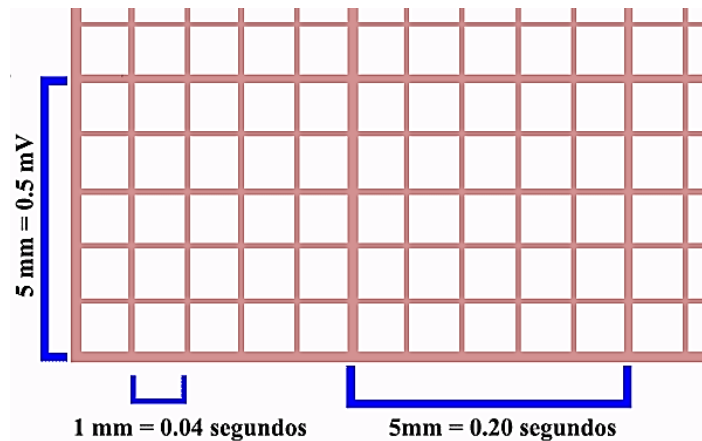


Figura 8: Medidas del papel de electrocardiograma [6].

En el eje vertical se mide la amplitud de la corriente eléctrica que se mide indirectamente a través del voltaje en una escala de milivoltios, la norma específica que 10 mm de altura equivale a 1 mV es decir; 1 mm representa 0.1 mV.

En el eje horizontal se mide el tiempo, donde 1 mm horizontal equivale a 0.04 s

Para el caso de que se mostrara en una pantalla VGA también, se debe realizar un escalado, donde se representara la cuadrícula en los pixeles de la pantalla, la escala

que se utilizara dependerá de la resolución que se mostrara, estando esto indeterminado hasta la presentación de este reporte.

2.1.2 Derivaciones del Electrocardiograma

Las derivaciones son el registro de la diferencia de potencial eléctrico entre dos puntos (electrodos) ya sean derivaciones bipolar o monopolar. Con un ECG de 10 electrodos se pueden obtener las 12 derivaciones, 6 de extremidades y 6 de pecho.

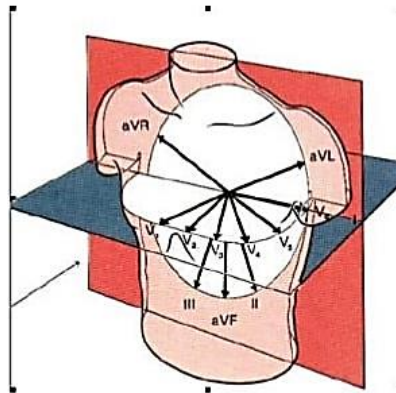


Figura 9: Ubicación de las derivaciones: plano frontal y horizontal. [7].

2.1.2.1 Derivaciones de plano frontal

Se les denomina así a las derivaciones que se obtienen de las extremidades, estas derivaciones se subdividen en: derivaciones bipolares (clásicas o Einthoven) y derivaciones monopoles aumentadas.

2.1.2.2 Derivaciones bipolares estándar

Registran la diferencia de potencial entre dos electrodos ubicados en extremidades diferentes:

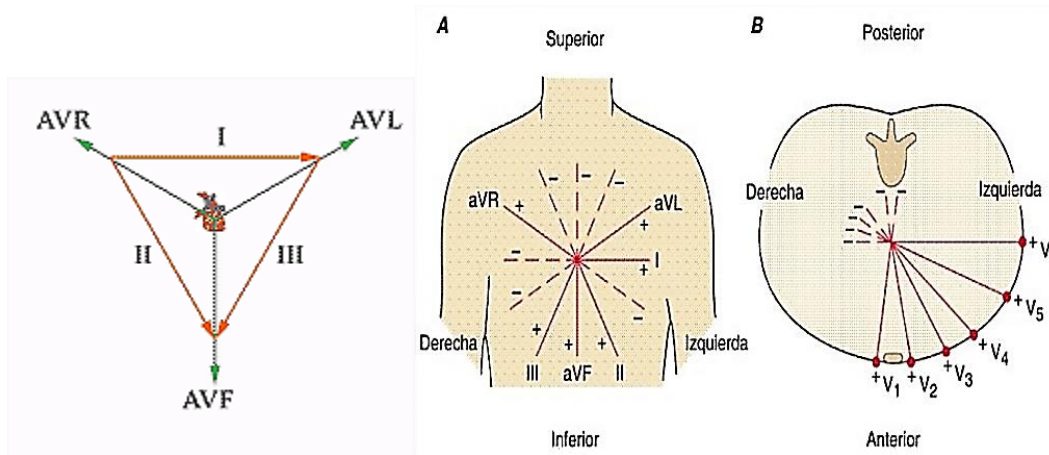


Figura 10: Derivaciones de extremidades y precordiales. [7].

- **D1 o I**: diferencia de potencial entre brazo derecho y brazo izquierdo, su vector está en dirección 0° .
- **D2 o II**: diferencia de potencial entre brazo derecho y pierna izquierda. Su vector está en dirección a 60° .
- **D3 o III**: diferencia de potencial entre brazo izquierdo y pierna izquierda. Su vector está en dirección a 120° .

2.1.2.3 Derivaciones monopares aumentadas

En el electrocardiograma, las derivaciones monopares de las extremidades, registran la diferencia de potencial entre un punto teórico en el centro del triángulo de Einthoven, con valor de 0 y el electrodo de cada extremidad, permitiendo conocer el potencial absoluto en dicho electrodo.

A estas derivaciones en un inicio se les nombró VR, VL y VF. La V significa Vector, y R, L, F: derecha, izquierda y pie.

Luego se encontró una forma de obtener las mismas derivaciones amplificadas y se renombraron a aVR, aVL y aVF, donde la "a" significa aumentada.

- **aVR**: potencial absoluto del brazo derecho. Su vector está en dirección a -150° .
- **aVL**: potencial absoluto del brazo izquierdo. Su vector está en dirección a -30° .

- **aVF**: potencial absoluto de la pierna izquierda. Su vector está en dirección a 90°.

2.1.3 Derivaciones del plano horizontal.

Las derivaciones precordiales del electrocardiograma son seis. Se denominan con una V mayúscula y un número del 1 al 6.

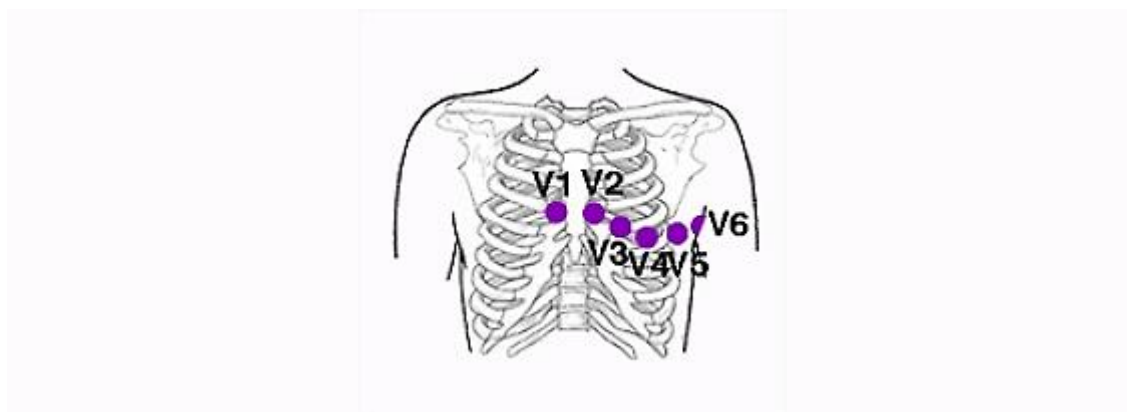


Figura 11: Derivaciones precordiales y ubicación de electrodos. [7]

Son derivaciones monopolares, registran el potencial absoluto del punto donde está colocado el electrodo del mismo nombre.

En el electrocardiograma normal, en las derivaciones precordiales, los complejos QRS son predominantemente negativos en las derivaciones V1 y V2 y predominantemente positivos en V4 a V6.

Derivaciones precordiales:

- **V1**: esta derivación registra los potenciales de las aurículas, de parte del tabique y de la pared anterior del ventrículo derecho. El complejo QRS presenta una onda R pequeña (despolarización del septo interventricular) seguida de una onda S profunda (ver morfología del complejo QRS).
- **V2**: el electrodo de esta derivación precordial, está encima de la pared ventricular derecha, por tanto, la onda R es ligeramente mayor que en V1, seguida de una onda S profunda (activación ventricular izquierda).

- **V3:** derivación transicional entre potenciales izquierdos y derechos del ECG, por estar el electrodo sobre el septo interventricular. La onda R y la onda S suelen ser casi iguales (complejo QRS isobifásico).
- **V4:** el electrodo de esta derivación está sobre el ápex del ventrículo izquierdo, donde es mayor el grosor. Presenta una onda R alta seguida de una onda S pequeña (activación de ventrículo derecho).
- **V5 y V6:** estas derivaciones están situadas sobre el miocardio del ventrículo izquierdo, cuyo grosor es menor al de V4. Por ello la onda R es menor que en V4, aunque sigue siendo alta. La onda R está precedida de una onda q pequeña (despolarización del septo).

Para obtener cada derivación se debe hacer una operación la cual se muestra en la siguiente tabla. La forma de obtención de estas ecuaciones se muestra más adelante con su respectivo circuito.

Tabla 1: Cálculos efectuados por derivación. [Elaboración propia].

Derivación	Tipo	Cálculos
I	Extremidad	LA – RA
II	Extremidad	LL – RA
III	Extremidad	LL – LA
aVR	Aumentada	RA – (LA+LL)/2
aVL	Aumentada	LA – (RA+LL)/2
aVF	Aumentada	LL – (RA + LA)/2
V1	Precordial	V1-(RA+LA+LL)/3
V2	Precordial	V2-(RA+LA+LL)/3
V3	Precordial	V3-(RA+LA+LL)/3
V4	Precordial	V4-(RA+LA+LL)/3
V5	Precordial	V5-(RA+LA+LL)/3
V6	Precordial	V6-(RA+LA+LL)/3

2.2 Muestreo de una señal

Una señal identifica a algo que lleva información del estado o comportamiento de un valor físico. En el tiempo discreto estas señales se representan como secuencias de números.

Para muestrear una señal es necesario el uso de un convertidor analógico digital que se encarga de evaluar el estado de la señal en intervalos regulares para que pueda ser almacenado en una memoria. El intervalo con el que el ADC (Analog to digital converter o convertidor analógico digital) toma las muestras se denomina frecuencia de muestreo f_s . Entre más alta sea la frecuencia de muestreo, se obtiene una mejor captura de la señal analógica.

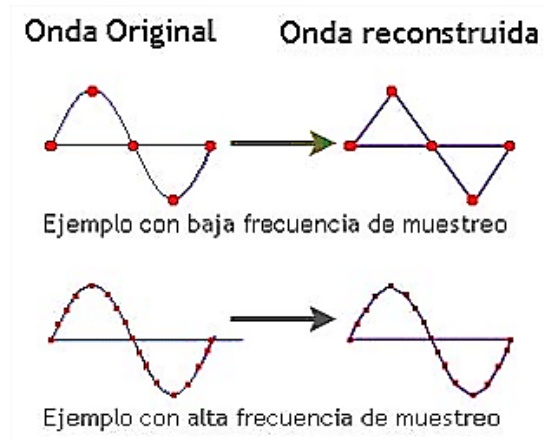


Figura 12: Ejemplo de muestreo de una onda sinusoidal. [8]

Se observa que, a mayor frecuencia de muestreo, hay mayor número de puntos para dibujar la señal original y la señal muestreada es mucho más parecida a la original.

Otro factor que influye en la digitalización de una señal analógica es la resolución o cuantificación, que son los escalones o valores de voltaje adjudicados al momento en que se toma la señal, esto sirve para determinar la altura de la señal en algún instante de tiempo determinado.

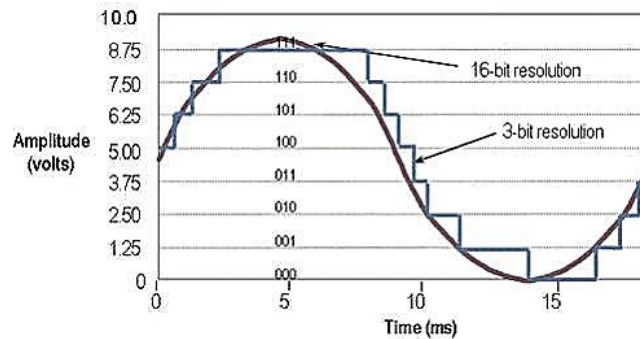


Figura 13: muestreo de una señal a diferentes resoluciones. [8]

Entre mayor cantidad de escalones se tengan, la señal muestreada también es más parecida a la señal real. El número de escalones viene dado por la formula $escalones = 2^n$, donde n es el número de bits.

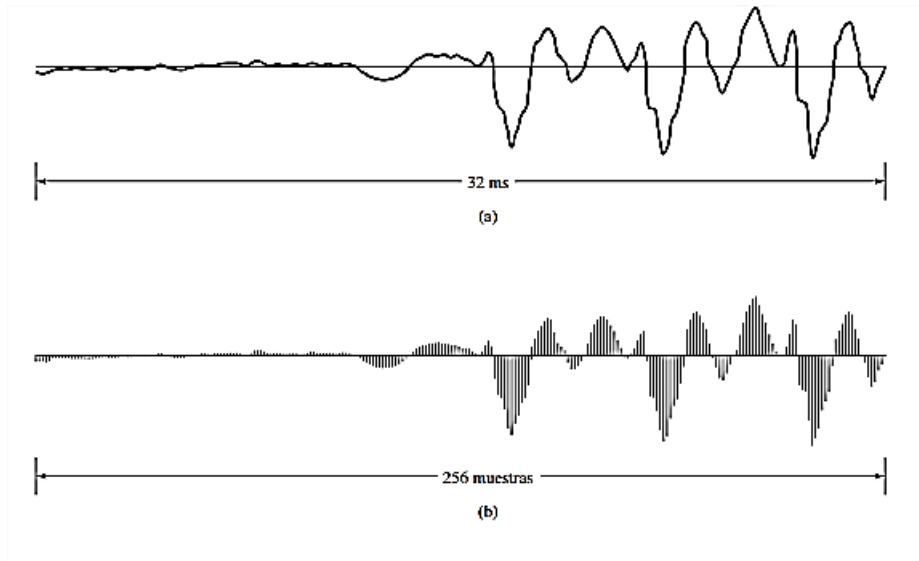


Figura 14: Ejemplo de una señal muestreada. a) original, b) digitalizada. [8]

Hay que tener mucho cuidado al momento de muestrear una señal dado que también hay criterios para muestrear la señal, como el criterio de Nyquist que dicta:

$$f_s \geq 2f_{max}$$

La frecuencia de muestreo debe ser al menos dos veces mayor que la frecuencia máxima que se desea muestrear.

2.3 Filtrado digital de una señal.

Un filtro digital es un sistema que, dependiendo de las variaciones de una señal de entrada en el tiempo y amplitud, realiza un proceso matemático sobre dicha señal para obtener una señal de salida con el objetivo de resaltar o atenuar características de la señal de entrada, se utilizaran para reducir el ruido al muestrear la señal en el convertidor analógico digital.

Ya sea un filtro analógico o digital, se denota a través de una función de transferencia que caracteriza la respuesta del filtro a cualquier tipo de señal de entrada.

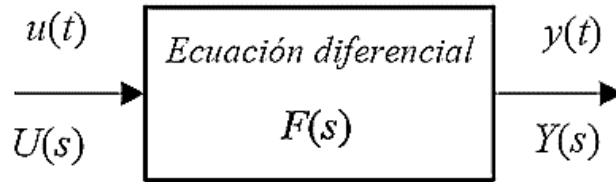


Figura 15: Esquema de una función de transferencia $F(s)$. [Elaboración propia].

Para una señal $u(t)$ se obtiene una salida $y(t)$, esto se logra con la función de transferencia $F(s)$. Donde la señal $u(t)$ se lleva al dominio de la frecuencia $U(s)$ para multiplicarse con $F(s)$ y así obtener $Y(s)$ para luego devolverla al dominio del tiempo $y(t)$.

$$y(t) = u(t) * f(y)$$

$$Y(s) = U(s)F(s)$$

Esto puede ser aplicado a señales discretas (digitales) para eliminar ruido presente en la señal digitalizada, realizando las siguientes operaciones utilizando el método de aproximación invariante al escalón:

1. Calcular o definir la función de transferencia en el dominio de la frecuencia utilizando la transformada de Laplace.
2. Integrar la función en el tiempo, lo que se traduce multiplicarla por $\frac{1}{s}$ en la frecuencia.
3. Aplicar fracciones parciales si es posible a la función de transferencia discretizada.
4. Discretizar la función
5. Multiplicar por $A \left(\frac{z-1}{z} \right)$
6. Aplicar límites cuando $Z \rightarrow 1$ y cuando $s \rightarrow 0$ para $G(Z)$ y $G(s)$ respectivamente para despejar A.
7. Aplicar transformada Z inversa.

Se demostrará estos pasos para un filtro de primer orden. El proceso para filtros de mayor orden es el mismo solo que más complejo.

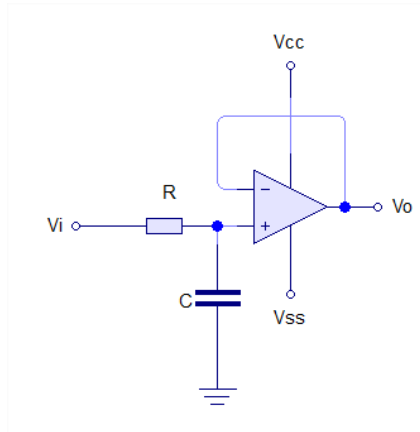


Figura 16: Filtro paso bajo primer orden. [Elaboración propia].

Aplicando LCK en V_o se obtiene:

$$\frac{V_o - V_i}{R} + \frac{V_o}{1/SC} = 0$$

$$G(s) = \frac{V_o}{V_i} = \frac{1}{1 + RSC} = \frac{1}{RC \left(\frac{1}{RC} + s \right)}$$

Sustituyendo:

$$\frac{1}{RC} = 2\pi f_c; \quad \text{Donde } f_c = \text{frecuencia de corte del filtro.}$$

$$G_e(s) = \frac{1}{s} \frac{2\pi f_c}{2\pi f_c + s}$$

Aplicando fracciones parciales:

$$G_e(s) = \frac{1}{s} \frac{2\pi f_c}{2\pi f_c + s} = \frac{A}{s} + \frac{B}{s + 2\pi f_c}$$

$$2\pi f_c = A(s + 2\pi f_c) + Bs$$

$$2\pi f_c = s(A + B) + A2\pi f_c$$

$$A = 1$$

$$A + B = 0$$

$$B = -1$$

Despejando A y B se obtienen $A = 1; B = -1$

$$G_e(s) = \frac{1}{s} - \frac{1}{s + 2\pi f_c}$$

Discretizando $G_e(s)$:

$$G_e(Z) = \frac{Z}{Z-1} - \frac{Z}{Z-e^{-2\pi f_c T}}; \quad \text{donde } T = \text{periodo de muestreo}$$

$$G_e(Z) = \frac{Z(1 - e^{-2\pi f_c T})}{(Z-1)(Z - e^{-2\pi f_c T})}$$

Multiplicando por $A \left(\frac{z-1}{z}\right)$:

$$G(Z) = A \left(\frac{z-1}{z}\right) \frac{Z(1 - e^{-2\pi f_c T})}{(Z-1)(Z - e^{-2\pi f_c T})}$$

$$G(Z) = A \frac{(1 - e^{-2\pi f_c T})}{(Z - e^{-2\pi f_c T})}$$

Aplicando límites:

$$\lim_{Z \rightarrow 1} \left[A \frac{(1 - e^{-2\pi f_c T})}{(Z - e^{-2\pi f_c T})} \right] = \lim_{s \rightarrow 0} \left[\frac{2\pi f_c}{2\pi f_c + s} \right]$$

Despejando A se obtiene $A = 1$

$$G(Z) = \frac{(1 - e^{-2\pi f_c T})}{(Z - e^{-2\pi f_c T})} = \frac{Y(Z)}{X(Z)}$$

Aplicando transformada Z inversa:

$$Z^{-1}\{X(Z)(Z^{-1} - Z^{-1}e^{-2\pi f_c T})\} = Z^{-1}\{Y(Z)(1 - Z^{-1}e^{-2\pi f_c T})\}$$

$$y[n] = e^{-2\pi f_c T} y[n-1] + (1 - e^{-2\pi f_c T}) x[n-1]$$

Si se presta atención el número de desplazamientos de la muestra es 1, esto corresponde al orden del filtro, es decir, si fuese un filtro orden 3, habrían 3 desplazamientos $[n-3]$ y la función de transferencia contaría con 3 elementos.

CAPITULO 3. DISEÑO DEL ECG

3.1 Diagrama conceptual del ECG.



Figura 17: Diagrama Conceptual del ECG. [Elaboración propia].

A grandes rasgos el funcionamiento del ECG se reduce a la figura anterior, se conectan electrodos al paciente, hay un circuito encargado de adquirir y adecuar la señal para ser introducida a la FPGA donde se hará todo el procesamiento de la misma para poder ser visualizada en la salida ya sea a la pantalla VGA o una salida impresa.

3.2 Sistema de Adquisición de señal.

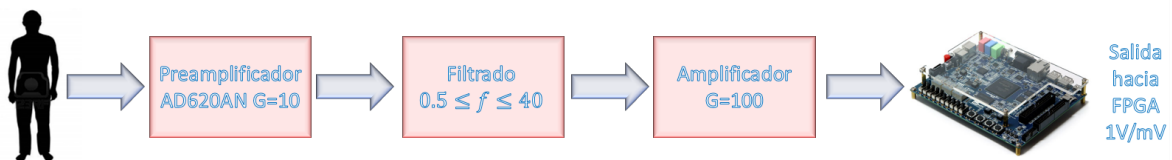


Figura 18: Diagrama de sistema de adquisición de datos. [Elaboración propia].

La etapa de adquisición de datos consta de una primera etapa de pre-amplificación con una ganancia de 10V/V, un filtrado analógico de segundo orden que disminuye señales parasitas de la red y así como otros ruidos de alta frecuencia, una segunda etapa inversora con una ganancia de 100V/V para que la señal resultante posea una ganancia general de 1V/mV y este montada en DC a 2.5V para ser leída por el convertidor análogo digital de la FPGA.

3.3 Fuente Bipolar

La alimentación del electrocardiógrafo es una fuente regulada aislada mediante el transformador, esta proveerá un voltaje constante de $\pm 12V$. Esta fuente tiene capacidad de hasta 3 A y alimentara todo el circuito.

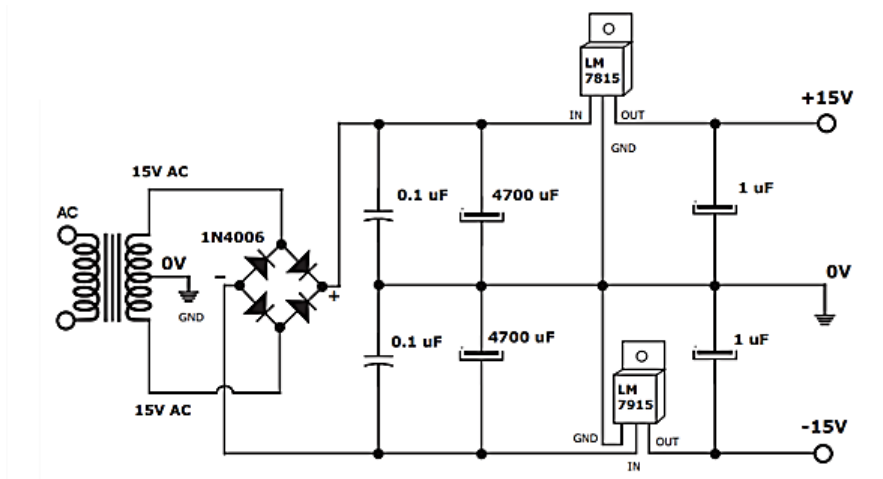


Figura 19: Circuito fuente bipolar, utilizando LM7X12. [9]

3.4 Pre-amplificación: Amplificador de instrumentación

3.4.1: Buffer.

El seguidor de tensión se utiliza ya que posee una alta impedancia de entrada y una baja impedancia de salida, esto permite aislar los electrodos conectados al cuerpo del circuito. Al ser un buffer su ganancia de voltaje es de 1 V/V. Todos los electrodos menos el de pierna derecha están conectados a un buffer.

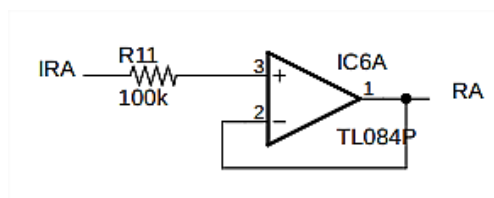


Figura 20: seguidor de tensión para IRA, entrada mano derecha. [Elaboración propia].

3.4.2: Terminal Central de Wilson.

Es el promedio de los potenciales de cada miembro, esto permite medir el potencial absoluto de los electrodos conectados al pecho (V1-V6).

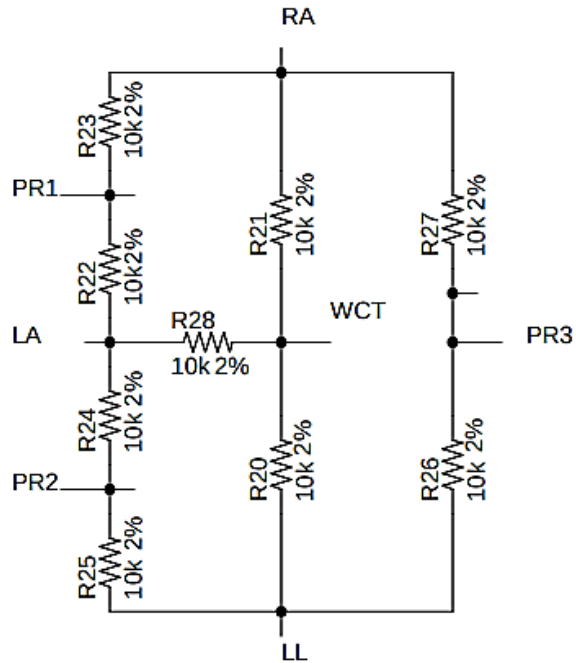


Figura 21: Terminal central de Wilson con resistencias de $10k \pm 2\%$. Fuentes: elaboración propia.

Tabla 2: Promedios de potenciales entre extremidades. [Elaboración propia].

Promedio	Terminales
PR1	$(RA+LA)/2$
PR2	$(LA+LL)/2$
PR3	$(RA+LL)/2$

3.4.3 AD620AN

El AD620AN es un amplificador de instrumentación de bajo costo y gran precisión el cual se utiliza normalmente en amplificación de señales biomédicas tales como los impulsos eléctricos del corazón. Su uso se debe a que pueden obtenerse grandes ganancias con un alto nivel de rechazo de ruido de modo común.

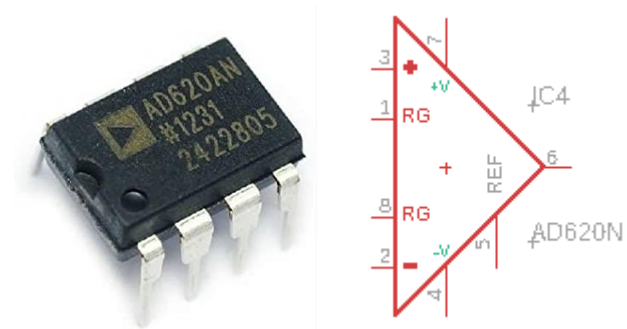


Figura 22: Amplificador de instrumentación AD620AN. [10]

La resistencia R_G conectada en los pines 1 y 8 del AD620N controla la ganancia del amplificador siguiendo la siguiente formula:

$$G = \frac{49.4}{R_G} + 1$$

Para este caso la ganancia se estableció de 10V/V lo que conlleva a una $R_G=5.5$ k Ω , este valor no se encuentra comercialmente, el valor más cercano que se encuentra es 5.6 k Ω , se prefirió resistencias de valor fijo a un potenciómetro ya que de esta forma no da lugar a posible descalibración si lo manipula una persona ajena a la electrónica.

3.4.4 Filtrado

La señal base que se obtiene después de la pre-amplificación contiene mucho ruido y componentes de baja frecuencia que distorsionaran la lectura en la FPGA para esto se pretende reducir un poco este ruido ya que el convertidor analógico digital solo trabaja en el rango de 0 a +5V, de cierta forma es una limitación al momento de implementar, pero la señal puede ser mejorada aún más implementando filtrado digital que se explica en el apartado 4.3.2 *Detección de complejo QRS*.

3.4.4.1 Filtro paso bajo

Este filtro se diseña con frecuencia de corte de 40Hz, se diseña de segundo orden para que posea solamente una etapa. Se utiliza el software de **Analog Filter Wizard** por mayor facilidad.

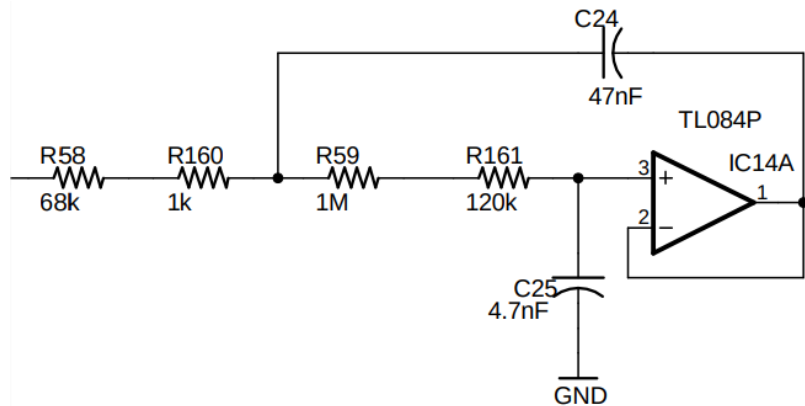


Figura 23: Diagrama de filtro paso bajo. [Elaboración propia]

3.4.4.2 Filtro paso alto

Nuevamente se utiliza Analog Filter Wizard y se diseña un filtro paso alto con frecuencia de corte 0.5 Hz, esto se hace para eliminar los componentes de DC que pueda tener la señal, ya que cuando pase por la ganancia del amplificador inversor de la siguiente etapa puede saturar la salida.

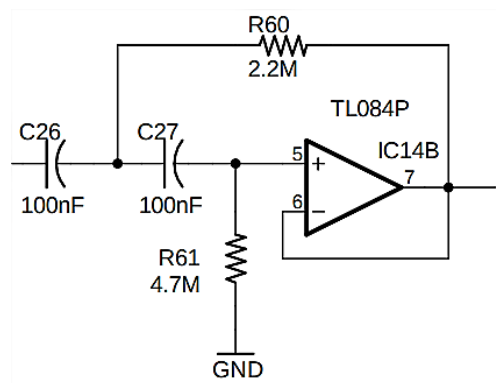


Figura 24: Diagrama de filtro paso alto. [Elaboración propia]

3.5 Circuito de pierna derecha

Este circuito es el más importante ya que en esta etapa se aísla completamente al paciente del polo a tierra, debido a que al conectar los electrodos en la piel en funcionamiento normal no pasaría nada, pero si existe alguna falla a tierra se puede estar seguro que la descarga no pasara por el paciente, ya que eso puede traer consecuencias fatales. Para esto se obtiene el potencial total en el corazón utilizando la Terminal Central de Wilson (WCT), así la pierna derecha tendrá el potencial absoluto impidiendo así que se formen corrientes que circulen en el cuerpo.

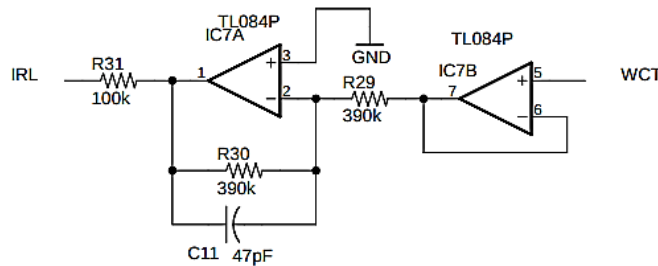


Figura 25: circuito de pierna derecha. [Elaboración propia]

3.5.1 AD7533

Es un convertor digital analógico (DAC) de 10 bits de resolución, es decir puede generar $2^{10} - 1$ niveles de tensión desde 0 V a V_{ref} en este caso $V_{ref} = 5 V$.

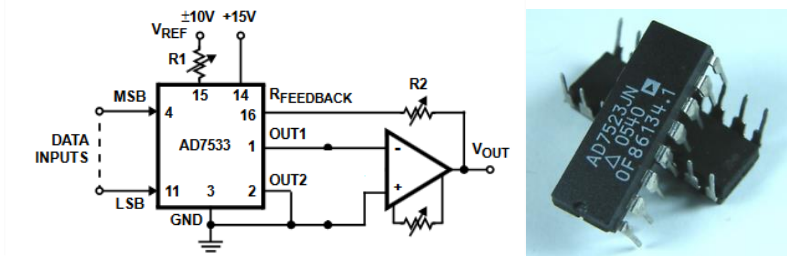


Figura 26: DAC AD7523. [11]

Para obtener el valor binario correspondiente a la tensión que se desea obtener se utiliza la siguiente ecuación.

$$\# \text{Binario requerido} = \frac{V_{\text{requerido}}}{V_{\text{Ref}}} * 2^n$$

Donde n es el número de bits del DAC.

De tal manera que para obtener 2.5V se necesita un 128 en binario que equivale a 10000000'b.

3.6 Inversor

La última etapa es para lograr la ganancia de $1 V/mV$ que es lo estándar para un ECG además de montar la señal en DC ya que se requiere que tenga un offset de $+2.5 V$.

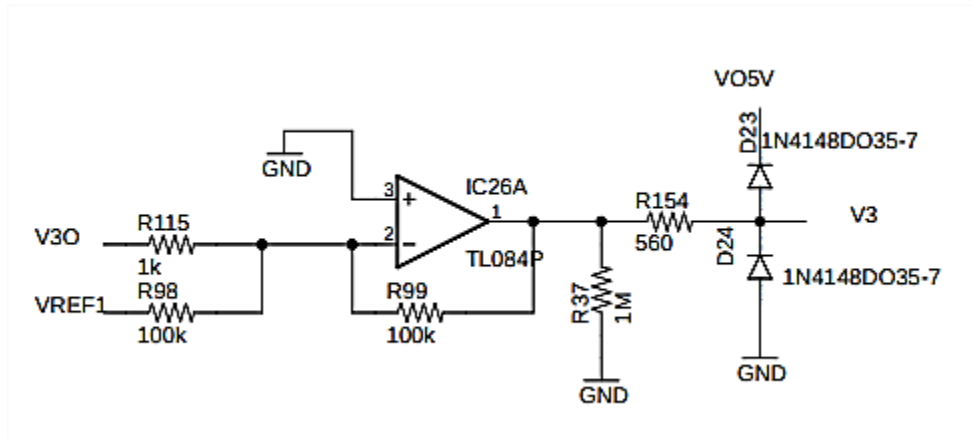


Figura 27: Etapa amplificadora. [Elaboración propia]

El detalle de esta etapa consiste en que la señal DC de $+2.5 V$ no será amplificada por lo cual $R98$ debe ser igual a $R99$, y la ganancia que se busca de $100 V/V$ se maneja con $R115$ y $R99$ con la siguiente ecuación.

$$V_{out} = -R99 \left(\frac{V_{30}}{R115} + \frac{V_{ref1}}{R98} \right)$$

Donde $R99 = R98$ y $R99/R115 = 100$

Los diodos utilizados al final de la etapa son para mantener los voltajes dentro del rango que la FPGA pueda manejar, ya que la señal se desea que oscile entre 0 y $+5V$, en dado caso algún operacional fallara y en su salida hubiese tensión superior el diodo entrara en funcionamiento y mantendrá el voltaje en la salida lo más cercano a $+5V$ o a $0V$.

Se utilizan amplificadores TL084P, debido a que tienen una muy buena velocidad de respuesta y posee en un solo encapsulado cuatro amplificadores operacionales lo que permite que el diseño en PCB (Printed Circuit Board o circuito impreso) sea más compacto.

3.7 Circuito implementado.

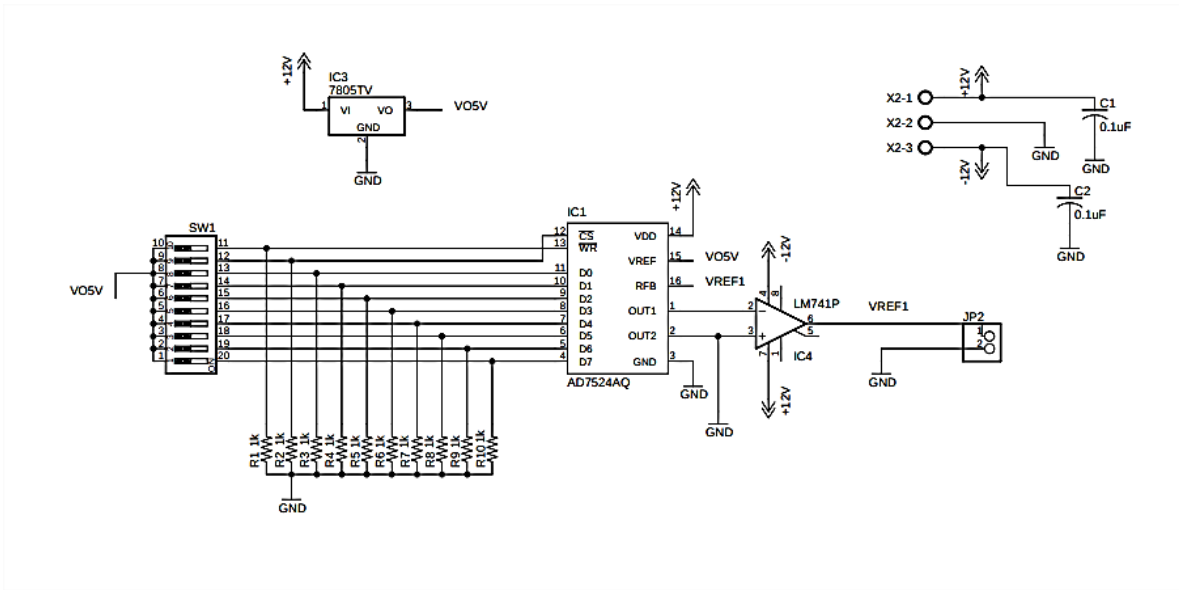


Figura 28: Etapa de referencia. [Elaboración propia]

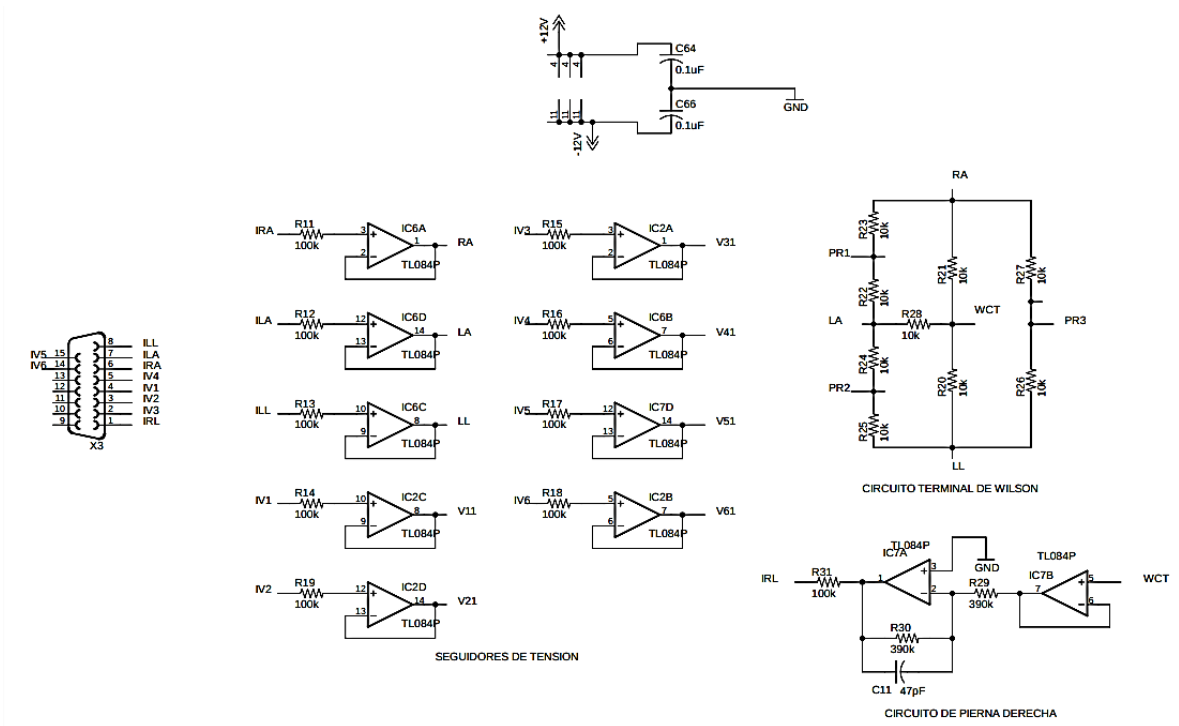


Figura 29: Etapa de buffer, WTC y pierna derecha. [Elaboración propia]

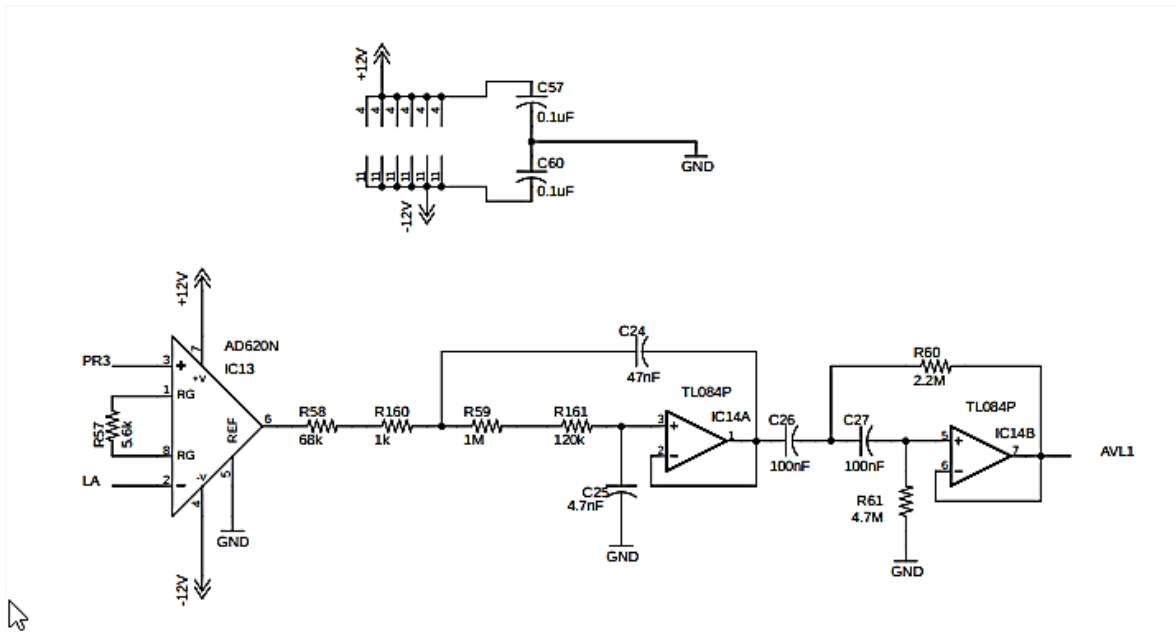


Figura 30: Etapa pre-amplificadora. [Elaboración propia]

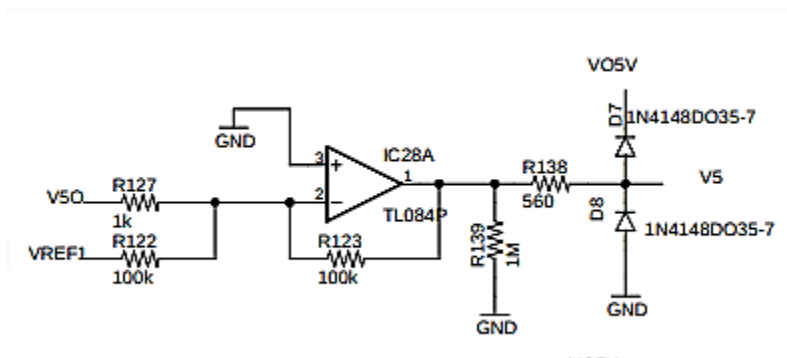


Figura 31: Etapa amplificadora. [Elaboración propia]

Cabe destacar que la etapa pre-amplificadora y amplificadora se repite para cada derivación.

3.8 Tecnología utilizada

3.8.1 FPGA

Una FPGA (field-programmable gate array o matriz de puertas lógicas programables), es un dispositivo que nos permite describir un circuito digital usando un lenguaje de descripción de hardware, los más comunes son VHDL y Verilog; que tras cargarlo en el integrado, el circuito descrito es creado físicamente en el chip. Con esto se pueden implementar funciones sencillas como compuestas, flipflops o un sistema combinacional complejo.

3.8.1.1 DE1-SoC.

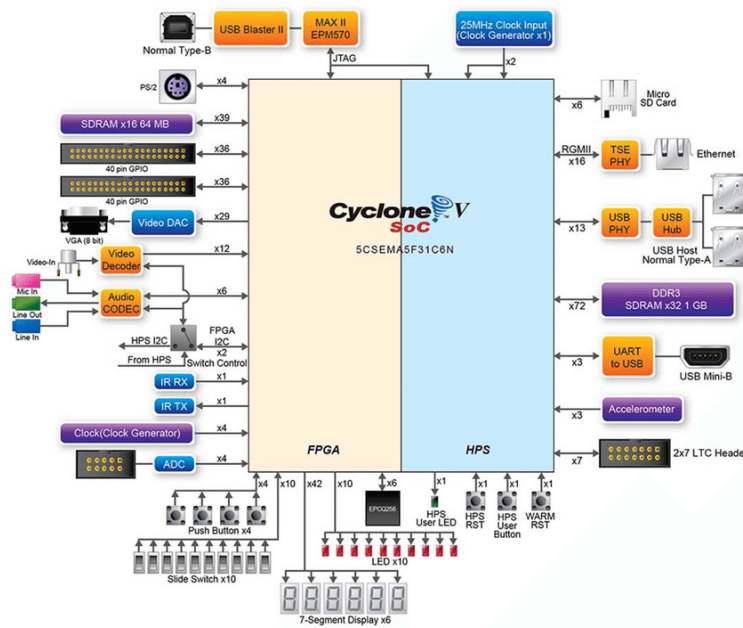


Figura 32: Diagrama en bloques de placa DE1-SoC [12]

La placa de desarrollo DE1-SoC posee un Cyclone V SoC, el cual le permite instalar sistema operativo a la placa. Se puede observar que el chip Cyclone V se divide en dos grupos, FPGA y HPS, cada uno posee periféricos asociados a cada parte del chip, esto indica cuales periféricos pueden ser controlados desde cada parte de la placa, es decir, si se desea manipular los display se deberá manipular desde la parte FPGA del chip, no se puede modificar los estados directamente desde la otra parte del chip pero existe una forma que se pueden controlar utilizando puentes entre

Address	31	...	16	15	...	8	7	3	2	1	0	Register name
0xFFFE600	Load value											Load
0xFFFE604	Current value											Counter
0xFFFE608	Unused				Prescaler			Unused	I	A	E	Control
0xFFFE60C	Unused										F	Interrupt status

Figura 34: Registros del reloj privado del ARM A9. [13]

La figura anterior muestra los registros para configurar el reloj privado. Estos registros tienen la base 0xfffec600, se accesa a estos utilizando words (32 bits). Para utilizar este temporizador es de tener en cuenta lo siguiente:

- La frecuencia de reloj es de 200 MHz.
- Es necesario agregar un valor de conteo inicial en el registro *Load value* de 32 bits máximo.
- El registro E controla que el contador funcione o este detenido, E=1 Start, E=0 Stop.
- El registro I controla las interrupciones, se estableció en I=0 (sin interrupciones).
- El registro A determina se el conteo se resetea al llegar a cero (A=1) o se detiene (A=0).
- El registro Prescaler si vale 0 decrementa cada ciclo, si vale 1 decrementa cada dos ciclos. Se deja en 0 para no hacer uso de esta función.
- El registro F indica si se ha llegado hasta a 0 haciendo que F=1, para resetearlo se sobrescribe F=1.

3.8.1.2.2 Temporizadores de intervalos (Interval Timer)

La computadora DE1-SoC incluye un módulo temporizador implementado en la FPGA que puede ser usado por el procesador A9. Este temporizador puede ser cargado con un valor preestablecido, y luego hacer una cuenta regresiva hacia cero usando un reloj de 100MHz. La interfaz programadora para el temporizador incluye seis registros de 16 bit, como se ilustra en la figura siguiente. El registro de 16 bit en la dirección 0xFF202000 proporciona información de estado acerca del

temporizador, y el registro en la dirección 0xFF202004 habilita las configuraciones de control. Los campos de bit en estos registros se describen a continuación:

- TO proporcionar una señal de tiempo fuera que se establece en 1 por el temporizador cuando ha alcanzado un valor de conteo de cero. El bit TO puede restablecerse al escribir 0 en él.
- RUN se establece en 1 por el temporizador mientras este contando. Escribir operaciones en el estado 'halfword' no afecta el valor del bit RUN.
- ITO es usado para generar interrupciones.
- CONT afecta la operación continua del temporizador. Cuando el temporizador alcanza un valor de cuenta de cero, automáticamente recarga el valor de inicio de conteo especificado. Si CONT está en 1, entonces el temporizador continúa contando hacia atrás automáticamente. Pero si CONT está en 0, entonces el temporizador se detiene después de que ha alcanzado un valor de conteo de cero.
- (START/STOP) es usado para empezar/suspender la operación del temporizador al escribir 1 en el bit respectivo.

Address	31	...	17	16	15	...	3	2	1	0		
0xFF202000	Unused						RUN		TO		Status register	
0xFF202004	Unused				STOP		START		CONT		ITO	Control register
0xFF202008	Not present (interval timer has 16-bit registers)										Counter start value (low)	
0xFF20200C											Counter start value (high)	
0xFF202010											Counter snapshot (low)	
0xFF202014											Counter snapshot (high)	

Figura 35: Registros del reloj de intervalos del HPS. [13]

Los dos registros de 16 bit en las direcciones 0xFF202008 y 0xFF20200C permite que el periodo del temporizador sea cambiado al configurar el valor inicial del conteo. La configuración por defecto proporcionado en la computadora DE1-SoC da un periodo de temporizador de 125ms. Para alcanzar este periodo, el valor inicial del conteo es $100\text{MHz} \times 125\text{ms} = 12.5 \times 10^6$. Es posible capturar una instantánea del valor del contador en cualquier tiempo al realizar una escritura en la dirección

0xFF202010. Esta operación de escritura causa que el valor actual del contador de 32 bit sea almacenado en los dos registros de 16 bit del temporizador en las direcciones 0xFF202010 y 0xFF202014. Estos registros pueden ser leídos entonces para obtener el valor del conteo. Un segundo temporizador de intervalos también está disponible en la FPGA, iniciando en la dirección base 0xFF202020.

3.8.1.2.3 Puertos paralelos.

Hay varios puertos paralelos implementados en la FPGA que soportan transferencias de datos de entrada, salida y bidireccionales entre el procesador ARM A9 y los periféricos I/O. A cada puerto paralelo se le asigna una dirección base y contiene hasta 4 registros de 32 bits. Los puertos que tienen capacidad de salida incluyen un registro de escritura de datos, y los puertos con capacidad de entrada tienen un registro de lectura de datos. Los puertos bidireccionales también incluyen un registro de direccionamiento que tiene el mismo ancho de bit que el registro de datos. ***Cada bit en el registro de datos puede ser configurado como una entrada al configurar el bit correspondiente en el registro de direccionamiento a 0, o como una salida al configurar la posición de este bit a 1.*** Al bit de direccionamiento se le asigna la dirección base +4.

3.8.1.2.4 Interrupciones

Los puertos paralelos que soportan interrupciones incluyen dos registros relacionados en las direcciones Base+8 y Base+C. El registro de máscara de interrupción, el cual tiene la dirección Base+8, especifica si una señal de interrupción debe o no ser enviada al GIC cuando los datos presentes en un puerto de entrada cambian valor. Estableciendo una localización de bit en este registro a 1, permite generar interrupciones, así como estableciendo el bit a 0 previene la interrupción. Finalmente, el puerto paralelo puede contener un registro de captura de flanco en la dirección Base+C. Cada bit en este registro tiene el valor de 1 si la localización de bit correspondiente en el puerto paralelo ha cambiado su valor de 0 a 1 desde que fue leído por última vez. Realizar una operación de escritura al registro de captura de flanco establece todos los bits en el registro a 0, y borra cualquier interrupción asociada.

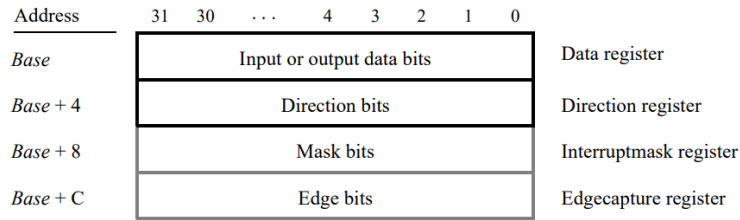


Figura 36: Registros del puerto paralelo en DE1-SoC. [13]

3.8.1.2.5 Puerto paralelo de LED rojos.

Los LEDs rojos LEDR₉₋₀ en la placa de la DE1-SoC son manejados por un puerto paralelo de salida, como se ilustra en Figura 37. El puerto contiene un registro de datos de 10 bit, el cual tiene la dirección 0xFF200000. Este registro puede escrito o leído por el procesador usando accesos de *word*, y los bits superiores no usados en el registro son ignorados.

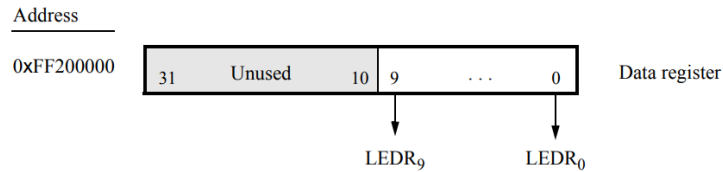


Figura 37: Puerto paralelo de Salida para LEDR. [13]

3.8.1.2.6 Puerto paralelo de Pantallas de 7 segmentos.

Hay dos puertos paralelos conectados a las pantallas de 7 segmentos en la placa de la DE1-SoC, cada uno comprende un registro de datos de 32 bit solo de escritura. Como se indica en la figura siguiente, el registro en la dirección 0xFF200020 maneja los dígitos de HEX3 a HEX0, y el registro en la dirección 0xFF200030 maneja los dígitos de HEX5 y HEX4. Los datos pueden ser escritos en estos dos registros, y leídos de nuevo, al usar las operaciones de *Word (32 Bits)*. Estos datos controlan directamente los segmentos de cada pantalla, de acuerdo a la localización dada en la Figura 38. La localización de segmentos del 6 al 0 en cada pantalla de 7 segmentos en la placa de la DE1-SoC se ilustra en la parte derecha de la figura.

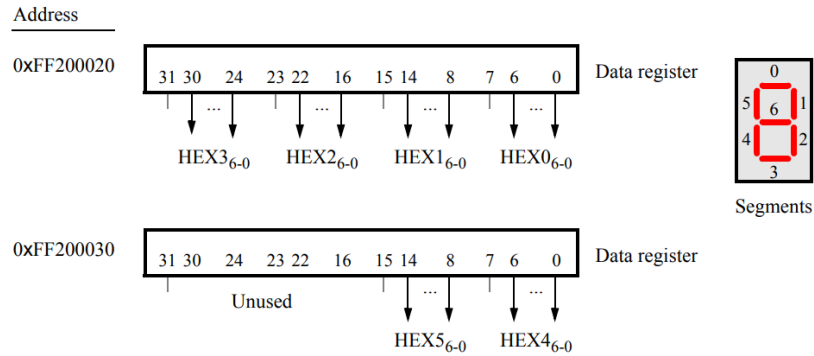


Figura 38: Localización de bits para pantalla 7-Segmentos. [13]

3.8.1.2.7 Puerto paralelo de interruptores deslizables.

Los interruptores SW₉₋₀, en la placa de la DE1-SoC están conectados a un puerto de entrada paralelo. Como se ilustra en la figura siguiente, este puerto comprende un registro de datos de solo lectura de 10 bit, el cual está ubicado en la dirección 0xFF200040.

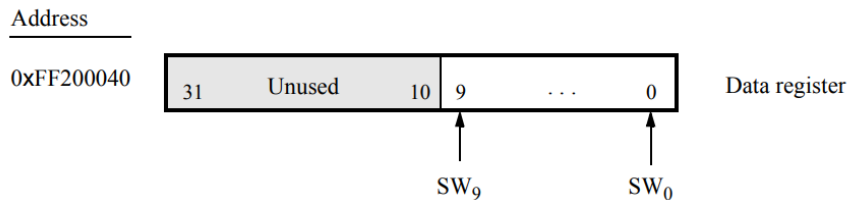


Figura 39: Registro de datos en SW9-0. [13]

3.8.1.2.8 Puerto paralelo de KEY (pulsadores).

El puerto paralelo conectado a los interruptores pulsadores KEY₃₋₀ en la placa de la DE1-SoC comprende de 3 registros de 4 bits, como se muestra en la Figura 40. Estos registros tienen una dirección base 0xFF200050 y pueden ser accesados usando operaciones *word*. El registro de datos de solo lectura provee los valores de los interruptores KEY₃₋₀. Los otros dos registros en la figura siguiente, en las direcciones 0xFF200058 y 0xFF20005C sirven para generar interrupciones al ser presionados y reconocer si el flanco es de subida o bajada, siendo 1 en flanco de subida.

Address	31	30	...	4	3	2	1	0	
0xFF200050	Unused				KEY ₃₋₀				Data register
Unused	Unused								
0xFF200058	Unused				Mask bits				Interruptmask register
0xFF20005C	Unused				Edge bits				Edgecapture register

Figura 40: Registros utilizados en pulsadores. [13]

3.8.1.2.9 Puerto VGA

La FPGA incluye un puerto de salida de video, que puede ser conectado a un monitor, soporta una resolución de 640x480. La imagen de video es generada a partir de dos fuentes, el pixel buffer y el character buffer.

3.8.1.2.9.1 Pixel Buffer

Este se encarga de mantener la información del color para cada pixel que es cargado en la pantalla VGA, esta resolución por defecto trabaja a 320x240 pixeles pero puede ser cambiada a la versión estándar de 640x480 utilizando la herramienta Qsys como se mostrara más adelante.

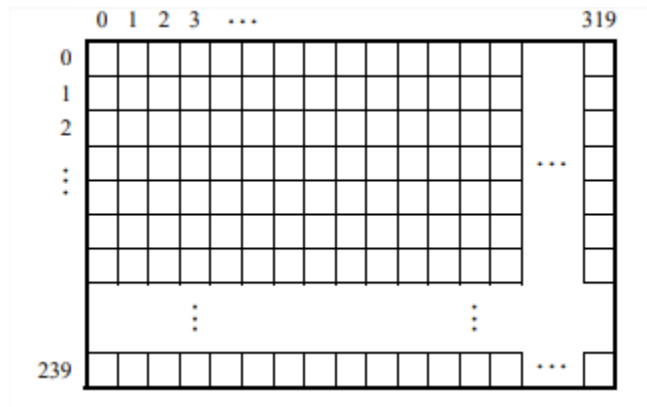


Figura 41: Coordenadas del Pixel Buffer. [13]

Como se ve en la imagen anterior se puede controlar el color de cada pixel ubicándose en la coordenada que se requiera, e insertando el valor del color que se desea, el color debe ser ingresado en formato de 16bits, es decir 5 bits para rojo, 6 bits para verde y 5 bits para azul, siendo el rojo con más peso y el azul con el de menor peso.

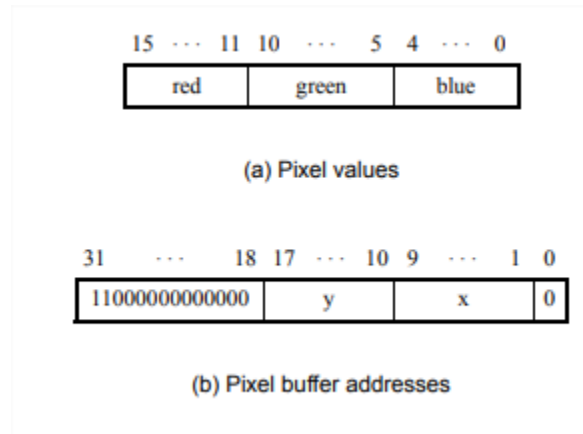


Figura 42: Direccionamiento del pixel y valores de colores. [13]

La manera en que se maneja la cuadrícula VGA es la siguiente:

Asumiendo la manera estándar 320x240, se utilizan 9 bits para el eje X y 8 bits para el eje Y, de esta forma si se quisiera direccionar al pixel (0,2) sería dándole a X=000000000'b e Y=00000010, siendo el valor de direccionamiento la unión 0000001000000000000'b.

Address	31 ... 24	23 ... 16	15 ... 8	7 ... 4	3	2	1	0	
0xFF203020	front buffer address								Buffer register
0xFF203024	back buffer address								Backbuffer register
0xFF203028	Y				X				Resolution register
0xFF20302C	m	n	Unused	B	Unused	A	S		Status register

Figura 43: Registros de control del Pixel Buffer. [13]

Cuando uno modifica el pixel buffer lo hace en tiempo real, lo que significa que todas las modificaciones que se hagan en la imagen serán mostradas en la pantalla, para evitar esto se puede utilizar otro buffer de pixeles, donde se precargara una imagen o cuadro y cuando esté lista para ser mostrada se realiza un intercambio entre el buffer secundario y el primario, esto permite que se muestre la imagen modificada, y poder realizar el mismo proceso, precargando la imagen antes de mostrarla.

3.8.1.2.9.2 Character Buffer

Este tiene un funcionamiento similar al del pixel buffer, solo que las dimensiones son 80x60 caracteres. Donde para mostrar el carácter debe escogerse la coordenada y luego darle el valor del carácter en ASCII que se desee mostrar.

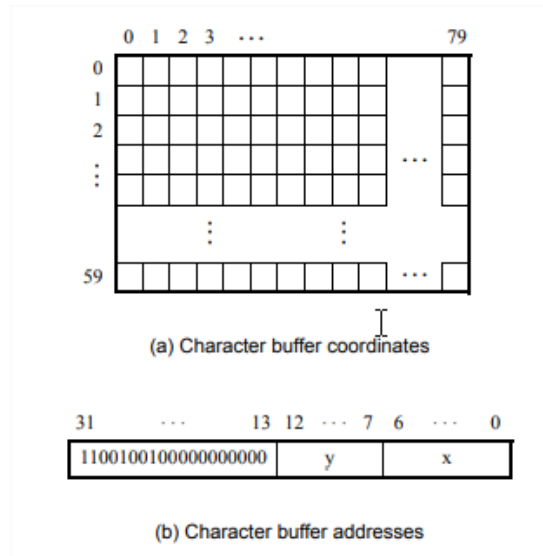


Figura 44: Direcciones y coordenadas del Character Buffer. [13]

3.8.1.2.10 ADC

El puerto de convertidores analógicos-digital permite el acceso a 8 canales de 12bits de convertidores analógico-digital. Los ADC empiezan desde la dirección 0xFF204000 y entre canal se debe hacer un desplazamiento de 0x04 por registro.

Se puede definir si se desea que el ADC realice la conversión automáticamente o solo cuando se quiera leer información en el puerto. Para lograrlo se escribe 1 en el bit R del canal 0. De esta manera se actualizan los valores de los ADC cada vez que se escriba en el canal 0.

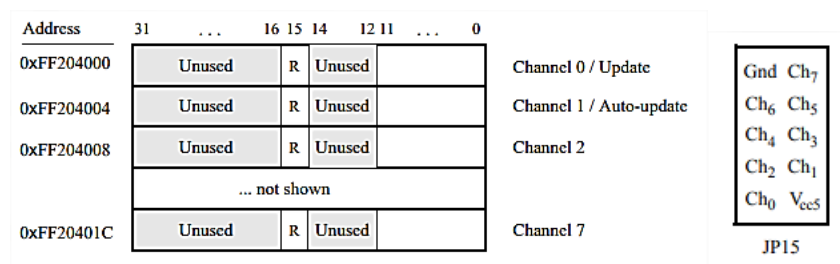


Figura 45: Direcciones del ADC y disposición de pines en DE1-So. [13]

3.8.1.2.11 Memoria

El HPS posee una memoria conectada al ARM MPCORE de 1GB DDR3, esta memoria es utilizada solamente para ejecutar el sistema operativo de la SD, así que todas las variables internas utilizadas en el código se almacenan en esta memoria. Esta memoria RAM de 1GB es la principal, pero también posee otros dos módulos de memoria del lado FPGA del chip: la SDRAM y On-Chip Memory.

3.8.1.2.11.1 SDRAM

Esta SDRAM provee una interfaz de 64MB de memoria que está organizada en 32M x16 bits. Su dirección dentro del “Computer System” es 0xC0000000 a 0xC3FFFFFF.

3.8.1.2.11.2 On-Chip Memory

La placa en la parte interna de la FPGA posee una memoria de 256kB, esta memoria está organizada en 64K x 32bits. Sus direcciones se encuentran en el rango de 0xC8000000 a 0xC803FFFF. Usualmente se utiliza como *pixel buffer* en los puertos de video (VGA y Video-in).

3.8.1.2.11.3 On-Chip Memory Character Buffer

La DE1-SoC incluye un espacio en memoria interna de 8kB, esta es utilizada para almacenar los caracteres para la salida VGA. Esta memoria está organizada en 8K x 8bits, su direccionamiento está en el rango de 0xC9000000 a 0xC9001FFF.

3.8.1.2.12 Tabla resumen de direcciones base de todos los componentes en la placa DE1-SoC.

Tabla 3: Ubicación de Registros de FPGA DE1-SoC. [13]

Base Address	End Address	I/O Peripheral
0x00000000	0x3FFFFFFF	DDR3 Memory
0xFFFF0000	0xFFFFFFFF	A9 On-chip Memory
0xC0000000	0xC3FFFFFF	SDRAM
0xC8000000	0xC803FFFF	FPGA On-chip Memory
0xC9000000	0xC9001FFF	FPGA On-chip Memory Character Buffer
0xFF200000	0xFF20000F	Red LEDs
0xFF200020	0xFF20002F	7-segment HEX3–HEX0 Displays
0xFF200030	0xFF20003F	7-segment HEX5–HEX4 Displays
0xFF200040	0xFF20004F	Slider Switches
0xFF200050	0xFF20005F	Pushbutton KEYs
0xFF200060	0xFF20006F	JP1 Expansion
0xFF200070	0xFF20007F	JP2 Expansion
0xFF200100	0xFF200107	PS/2
0xFF200108	0xFF20010F	PS/2 Dual
0xFF201000	0xFF201007	JTAG UART
0xFF201008	0xFF20100F	Second JTAG UART
0xFF201020	0xFF201027	Infrared (IrDA)
0xFF202000	0xFF20201F	Interval Timer
0xFF202020	0xFF20202F	Second Interval Timer
0xFF203000	0xFF20301F	Audio/video Configuration
0xFF203020	0xFF20302F	Pixel Buffer Control
0xFF203030	0xFF203037	Character Buffer Control
0xFF203040	0xFF20304F	Audio
0xFF203060	0xFF203070	Video-in
0xFF204000	0xFF20401F	ADC
0xFF709000	0xFF709063	HPS GPIO1
0xFFC04000	0xFFC040FC	HPS I2C0
0xFFC08000	0xFFC08013	HPS Timer0
0xFFC09000	0xFFC09013	HPS Timer1
0xFFD00000	0xFFD00013	HPS Timer2
0xFFD01000	0xFFD01013	HPS Timer3
0xFFD0501C	0xFFD0501F	FPGA Bridge
0xFFFE0100	0xFFFE01FC	GIC CPU Interface
0xFFFE0D00	0xFFFE0DFFC	GIC Distributor Interface
0xFFFE0600	0xFFFE060F	ARM A9 Private Timer

3.9 Formas de programar la placa DE1-SoC.

Utilizando los switches MSEL se puede preconfigurar la FPGA de 3 formas. Cada una tiene ciertas particularidades para poder programar la placa.

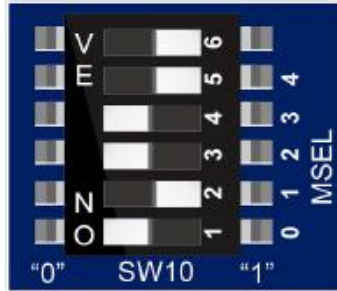


Figura 46: Switches para MSEL. [13]

MSEL[4:0]	CONFIGURACION	DESCRIPCION
10010	AS	FPGA configurada desde EPCQ (QUARTUS)
10100	FPPx32	FPGA configurada desde software HPS desde CLI
00000	FPPx16	FPGA configurada desde software HPS con entorno gráfico.

- **AS:**
Esta configuración por defecto de la FPGA, con esta configuración se puede manipular la FPGA utilizando un programa externo llamado QUARTUS II, en los trabajos de graduación anteriores: “Principios del FPGA y aplicaciones en el control de procesos industriales (2016)” y “Diseño de un controlador PID, utilizando una tarjeta FPGA Cyclone V GX Starter, programada en VHDL(2017)”, se ha trabajado la FPGA de esta forma, cargando el programa desde la computadora a la placa, cabe destacar que la forma de manipular los periféricos es utilizando VHDL, esta configuración es utilizada para la descripción de circuitos lógicos complejos.
- **FPPx16:**
Esta configuración se carga un Linux en la microSD con entorno gráfico, las por defecto son las versiones LXDE de Linux. Con esta forma solo se puede utilizar la FPGA como si fuese una computadora, se dispone de, entorno gráfico pero solamente se pueden utilizar los periféricos del lado HPS del Cyclone V dado que los puentes están deshabilitados.

- **FPPx32:**

Esta configuración es la utilizada para el desarrollo de este proyecto, con esta configuración se instala una imagen de Linux “liviana” desarrollado para la placa DE1-SoC, la placa al ser encendida boteará el sistema operativo instalado en la microSD, dado que es una configuración intermedia entre AS y FPPx16, se debe controlar la placa desde el sistema operativo utilizando una interfaz de comandos entre la placa y computadora, esto debido a que se dispondrán de todos los periféricos de la placa DE1-SoC incluyendo el puerto VGA. La comunicación entre la computadora y FPGA es vía UART, que es una comunicación serial a una velocidad estándar de 115,200 bits/s. De esta forma se puede programar la FPGA desde el propio sistema operativo y bastara con ejecutar el programa como cualquier programa en C desde la placa.

3.10 Instalación de Linux en la placa DE1-SoC (FPPx32).

3.10.1 Linux en la placa DE1-SoC.

Linux es un sistema operativo encontrado en una amplia variedad de dispositivos de computadoras tales como las computadoras personales, servidores, y smartphones. Una ventaja clave que se explotará son los controladores incorporados de Linux que soportan una amplia gama de dispositivos, incluyendo muchos de los dispositivos encontrados en la placa DE1-SoC.

3.10.2 Montando la imagen de Linux en tarjeta microSD.

La placa DE1-SoC está diseñada para arrancar Linux *DE1-SoC_UP_Linux, Kernel: 3.18* [14] desde una tarjeta microSD insertada. Las tecnologías Altera y Terasic proveen un número de imágenes de tarjetas microSD con Linux que pueden ser usadas para obtener rápidamente Linux en operación en la DE1-SoC. Estas imágenes de Linux van desde una simple distribución de Linux de una única línea de comando, hasta la distribución más completa de Ubuntu Linux con una interfaz gráfica (GUI). Para correr una de estas distribuciones de Linux, se debe escribir la imagen (formato de archivo .img) en una tarjeta microSD lo suficientemente grande con 8GB bastara. En este caso se usará la imagen DE1-SoC-UPLinux.img. Para escribir la tarjeta microSD, se usará la herramienta de libre uso 'Win32 Disk Imager'. Los pasos para la escritura de la tarjeta microSD se describen a continuación.

1. Insertar la tarjeta microSD en la computadora usando un lector de tarjetas, luego ejecutar ins32 Disk Imager.

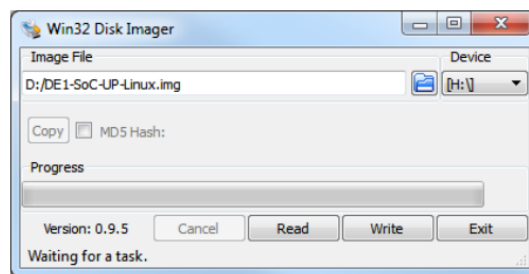


Figura 47: WIN32 Disk Imager. [Elaboración propia].

2. Seleccionar la letra de manejo correspondiente a la tarjeta microSD, bajo la pestaña Device, como se muestra en la figura anterior.
3. Seleccionar la imagen DE1-SoC-UPLinux.img bajo la pestaña Image File, como se muestra en la figura anterior.

4. Click en 'Write' para escribir la tarjeta microSD. Si se solicita confirmar la sobrescritura, presionar 'Yes'. Una vez la escritura se ha completado, se observará el cuadro de éxito mostrado en la figura siguiente.



Figura 48: Win32 Disk Imager, una vez instalado el SO. [Elaboración propia].

3.10.3 Booteando Linux en placa DE1-SoC.

Ahora que la tarjeta microSD está cargada con Linux, se puede insertar en la ranura para microSD en la DE1-SoC. Antes de encender la placa, se deben configurar los interruptores MODE SELECT (MSEL) localizados en la parte baja de la placa al valor $MSEL[4:0]=5'b01010$, como se muestra en la figura siguiente. Esto configura el chip Cyclone V SoC para permitirle al procesador ARM la programación en la FPGA, lo cual es necesario mientras la imagen de Linux programará la FPGA como parte de su proceso de arranque. Además, se necesitará esta configuración MSEL cuando se programe manualmente la FPGA desde la línea de comando de Linux.

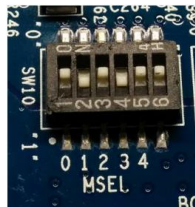


Figura 49: Configuración de switches del MSEL en DE1-SoC. [13]

Una vez la tarjeta microSD está insertada y MSEL está configurado, se puede encender la placa para tener Linux. En este punto se requiere alguna manera de interacción con el sistema operativo Linux. Dado que la versión de Linux utilizada no proporciona interfaz GUI, en su lugar, soporta la interacción a través de la interface de línea de comandos (CLI).

3.10.4 Accesando a la interfaz de línea de comando vía terminal UART.

La distribución en uso ha sido configurada para dirigir todos los textos de entrada y de salida (en los estándares stdout, stdin, stderr) al serial UART de la Cyclone V HPS. El serial UART es un dispositivo que facilita la comunicación serial de caracteres, a menudo a través de un cable serial. En la DE1-SoC se conecta a un chip USB-UART el cual envía y recibe los textos a través de un cable USB hacia un computador anfitrión. En este computador, se puede usar cualquiera de los programas terminales fácilmente disponibles que son capaces de comunicación serial para acceder a la CLI del Linux operando en la DE1-SoC. Para este caso se usará la herramienta de libre uso 'Putty', la cual está disponible tanto para Windows como para Linux. Iniciar Putty, luego conectar la USB-to-UART de la DE1-SoC al PC usando un cable microUSB.

3.10.4.1 En una PC con Windows.

Los dispositivos de comunicación serial como los USB-to-UART son reconocidos como puertos COM. Como puede haber múltiples puertos COM conectados a la PC, cada puerto COM es asignado con un número de identificación único. La Figura 50 muestra la lista de puertos COM disponibles en una PC cualquiera en su manejador de dispositivos. Aquí, solo existe un puerto COM (el USB-to-UART) al cual le fue asignado el número 9 (**COM9**). En el caso donde hay muchos puertos COM disponibles, el número asignado al USB-to-UART puede ser determinado al desconectar y reconectar el cable para ver cuál puerto COM desaparece y luego reaparece en la lista.

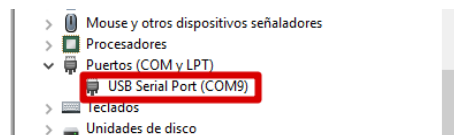


Figura 50: Determinación de asignación de COM para USB a UART en el Administrador de Dispositivos. [Elaboración propia]

Una vez que el dispositivo serial (COM) correspondiente al USB-to-UART se ha determinado, Putty puede ser configurado para conectarse a él. La Figura 51 muestra la ventana principal de Putty. En esta ventana, el tipo de conexión serial

debe ser elegida, y el puerto COM debe ser ingresado en el campo de línea serial, como se muestra en la figura.

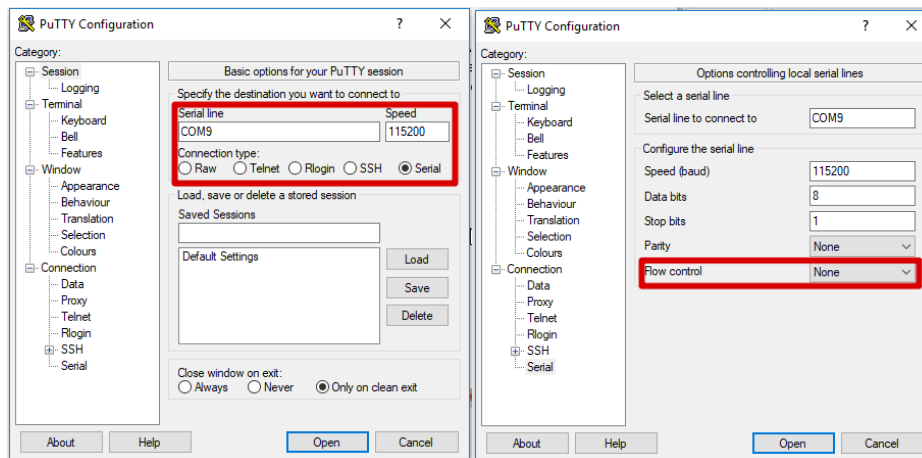
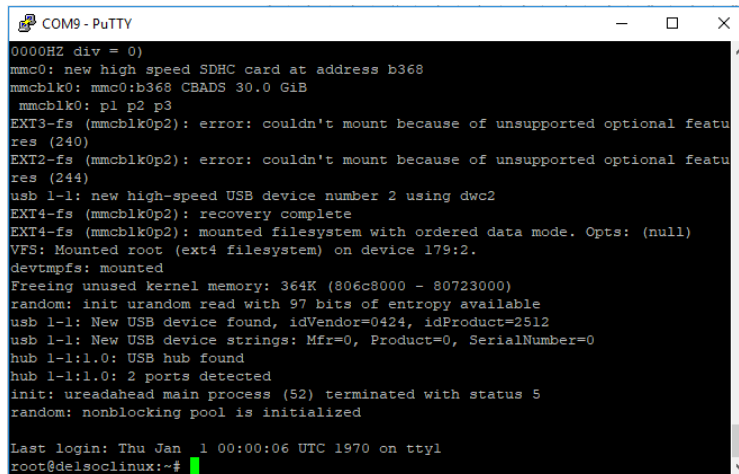


Figura 51: Ventana de Putty, configuración para comunicación serial. [Elaboración propia]

Algunos detalles adicionales acerca del USB-to-UART deben ser ingresados seleccionando la pestaña: Serial en el cuadro de Categoría en el lado izquierdo de la ventana. La pestaña Serial se muestra en la Figura 51. Estas especificaciones deben ser configuradas para que concuerden con las especificaciones mostradas, con la velocidad (ritmo de baud) establecida a **115,200 bits por segundo**, los bits de datos establecidos a 8 bits, los bits de stop establecidos a 1 bit, y paridad y control de flujo establecido a ninguno.

Una vez todas las especificaciones de la línea serial han sido ingresadas, presionar Open para iniciar la terminal. Se ha conectado a la CLI y se puede iniciar el uso de Linux al ingresar los comandos de Linux (intentar presionar Enter para dibujar la línea de espera de comando). Una vez Linux ha finalizado su arranque, se ingresará en la CLI como el usuario 'root', lo que significa que se tiene privilegios de nivel de administrador permitiendo modificar configuraciones y ejecutar programas que se deseen.



```
COM9 - PuTTY
0000HZ div = 0)
mmc0: new high speed SDHC card at address b368
mmcblk0: mmc0:b368 CBADS 30.0 GiB
  mmcblk0: p1 p2 p3
EXT3-fs (mmcblk0p2): error: couldn't mount because of unsupported optional featur
es (240)
EXT2-fs (mmcblk0p2): error: couldn't mount because of unsupported optional featur
es (244)
usb 1-1: new high-speed USB device number 2 using dwc2
EXT4-fs (mmcblk0p2): recovery complete
EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode. Opts: (null)
VFS: Mounted root (ext4 filesystem) on device 179:2.
devtmpfs: mounted
Freeing unused kernel memory: 364K (806c8000 - 80723000)
random: init urandom read with 97 bits of entropy available
usb 1-1: New USB device found, idVendor=0424, idProduct=2512
usb 1-1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
hub 1-1:1.0: USB hub found
hub 1-1:1.0: 2 ports detected
init: ureadahead main process (52) terminated with status 5
random: nonblocking pool is initialized

Last login: Thu Jan  1 00:00:06 UTC 1970 on tty1
root@delsoclinux:~#
```

Figura 52: Inicio de sección como administrador de Linux en la interfaz de comandos. [Elaboración propia]

3.10.5 Cargado el **Computer System**.

Por defecto el sistema operativo trae una preconfiguración de todos los dispositivos y periféricos con los que cuenta la DE1-SoC, esta configuración puede ser sustituida para cambiar parámetros que no pueden ser modificados a través del código en C, como lo son la configuración de video del puerto VGA, por defecto la placa trabaja con una resolución de 320x240 pixeles, en el caso de este proyecto se desea trabajar con una resolución de 640x480 pixeles, además del tipo de RGB a utilizar ya sea de 8 o 16 bits, también la forma de direccionamiento consecutivo o x,y que se explica en la sección del Pixel Buffer de este trabajo.

Cambiando el computer system se pueden modificar las direcciones de memoria mostradas en la tabla de “Ubicación de registros de FPGA DE1-SoC”.

Para cargar el sistema a utilizar, debe ser creado y compilado en Quartus y la herramienta Qsys, de la compilación se obtendrá un archivo *.sof, este archivo será convertido a su equivalente en *.rbf el cual será copiado a la carpeta /home/root/ del sistema Linux. La creación del archivo *.sof se mostrará en el siguiente apartado.

3.10.5.1 Pasos para crear archivo *.rbf

1. Abrir “*Quartus Prime 15.1 Lite Edition*” o su equivalente.
2. Click en la pestaña *File*, luego *Convert programming files*.

3. Se abrirá una ventana, en *Programming file type* se selecciona *Raw Binary File(.rbf)*, en *Mode* se escoge *Passive Parallel x16* y en *File name* se selecciona la carpeta donde se guardara el archivo convertido.

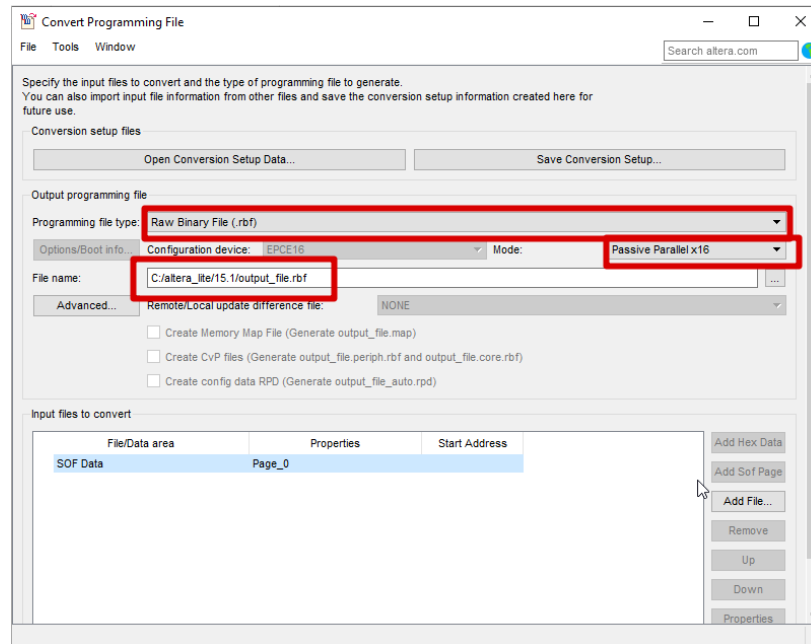


Figura 53: Conversion de *.sof a *.rbf. [Elaboración propia]

4. Luego se selecciona el archivo *.sof que se encuentra en la carpeta del proyecto de Quartus.

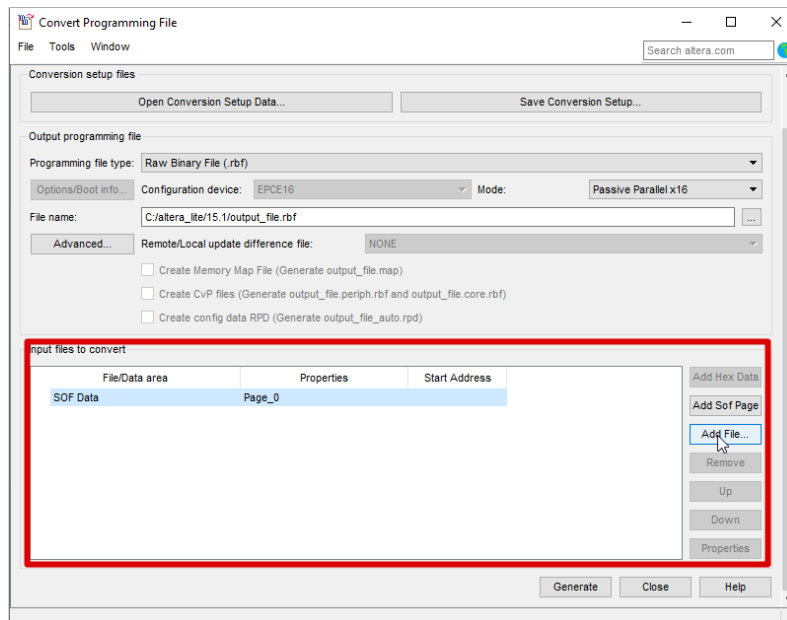


Figura 54: Conversion de *.sof a *.rbf (2). [Elaboración propia]

5. Finalmente se da click en *Generate*.

3.10.5.2 Cargando *.rbf al sistema operativo.

De la sección anterior se mostró como se crea el *.rbf. Este archivo se copia a la SD donde se ha instalado el SO. Los otros tres archivos mostrados en la Figura 55 no deben eliminarse ya que son parte de la configuración de la placa que se ejecutan al encenderla.

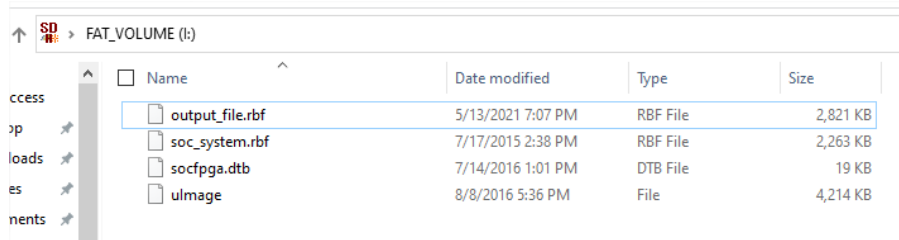


Figura 55: carga de *.rbf y archivos visibles de la SD con Linux instalado. [Elaboración propia].

Para lograr que se ejecute este archivo al encender la placa se debe modificar un archivo de arranque desde la interfaz CLI siguiendo los siguientes pasos.

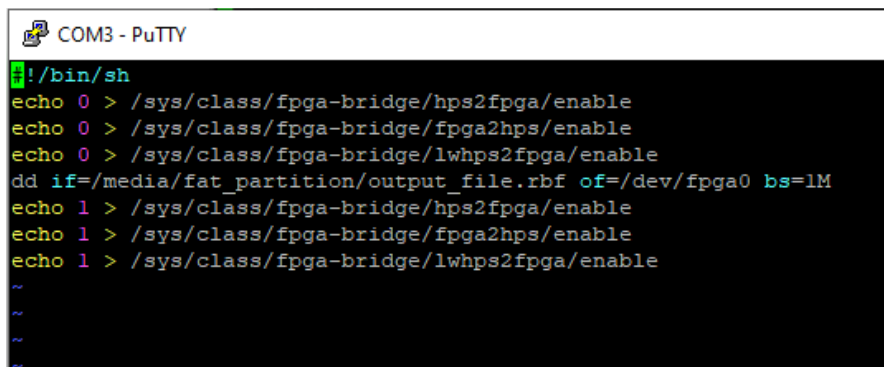
1. Colocar la SD en la placa DE1-SoC y encenderla.
2. Abrir la interfaz Putty.
3. Para poder editar el documento debe utilizarse un editor con los que cuenta Linux, en este caso se utiliza Vi (ya se da por entendido que el usuario sabe cómo se utiliza este editor). Escribir el siguiente comando `vi /etc/init.d/programfpga` y dar enter. Se observara lo siguiente:

```
COM3 - PuTTY
#!/bin/sh
echo 0 > /sys/class/fpga-bridge/hps2fpga/enable
echo 0 > /sys/class/fpga-bridge/fpga2hps/enable
echo 0 > /sys/class/fpga-bridge/lwhps2fpga/enable
dd if=/home/root/DE1_SoC_Computer.rbf of=/dev/fpga0 bs=1M
echo 1 > /sys/class/fpga-bridge/hps2fpga/enable
echo 1 > /sys/class/fpga-bridge/fpga2hps/enable
echo 1 > /sys/class/fpga-bridge/lwhps2fpga/enable
```

Figura 56: sustitución de *.rbf. [Elaboración propia].

Se debe editar la ruta del *.rbf que se carga al encender la placa por la ruta siguiente:
`/media/fat_partition/output_file.rbf`

Debe quedar como la Figura 57



```
COM3 - PuTTY
~/bin/sh
echo 0 > /sys/class/fpga-bridge/hps2fpga/enable
echo 0 > /sys/class/fpga-bridge/fpga2hps/enable
echo 0 > /sys/class/fpga-bridge/lwhps2fpga/enable
dd if=/media/fat_partition/output_file.rbf of=/dev/fpga0 bs=1M
echo 1 > /sys/class/fpga-bridge/hps2fpga/enable
echo 1 > /sys/class/fpga-bridge/fpga2hps/enable
echo 1 > /sys/class/fpga-bridge/lwhps2fpga/enable
~
~
~
```

Figura 57: *.rbf sustituido. [Elaboración propia]

4. Guardar los cambios en el archivo y reiniciar la placa.

3.11 Herramienta Qsys

Para la creación de las interconexiones de la placa DE1-SoC el único método posible es a través de la descripción de las conexiones entre los diferentes periféricos y el procesador principal utilizando VHDL, para poder codificar toda la descripción de cómo van interconectados los componentes se hace un proceso realmente tedioso y teniendo el cuidado de realizar bien las conexiones, para evitar esto se utiliza Qsys que es una herramienta que viene incluida en el software Quartus II la cual permite describir el circuito de manera gráfica a través de una interfaz más amigable y no es necesario tener conocimientos avanzados de VHDL.

En Qsys uno selecciona componentes que viene incluidos en las placas de aprendizaje como la DE1-SoC, Intel ya provee estas librerías y las descripciones de pines que se necesitan para cada componente, lo único que se debe hacer es realizar las conexiones necesarias para que exista comunicación entre el procesador y el periférico. Estas interconexiones son posibles gracias al Avalon Bridge, que es un ruteo automático entre los pines del procesador y periférico que realiza la herramienta Qsys; por ejemplo, para controlar la memoria SDRAM es necesario tener un bus de comunicación entre el procesador y la misma de 16 bits

estos bits de comunicación son administrados por el Avalon que automáticamente crea puentes entre el la SDRAM y el procesador a través de la FPGA.

En el sitio de Intel se puede descargar una imagen que contiene el sistema descrito en la siguiente figura, para propósitos de este trabajo de graduación se necesitaba contar con la mejor resolución en el puerto VGA, por defecto utiliza 320x240 pixeles. Pero dado que es una resolución muy baja, se rediseño el sistema para que trabaje a 640x480 pixeles.

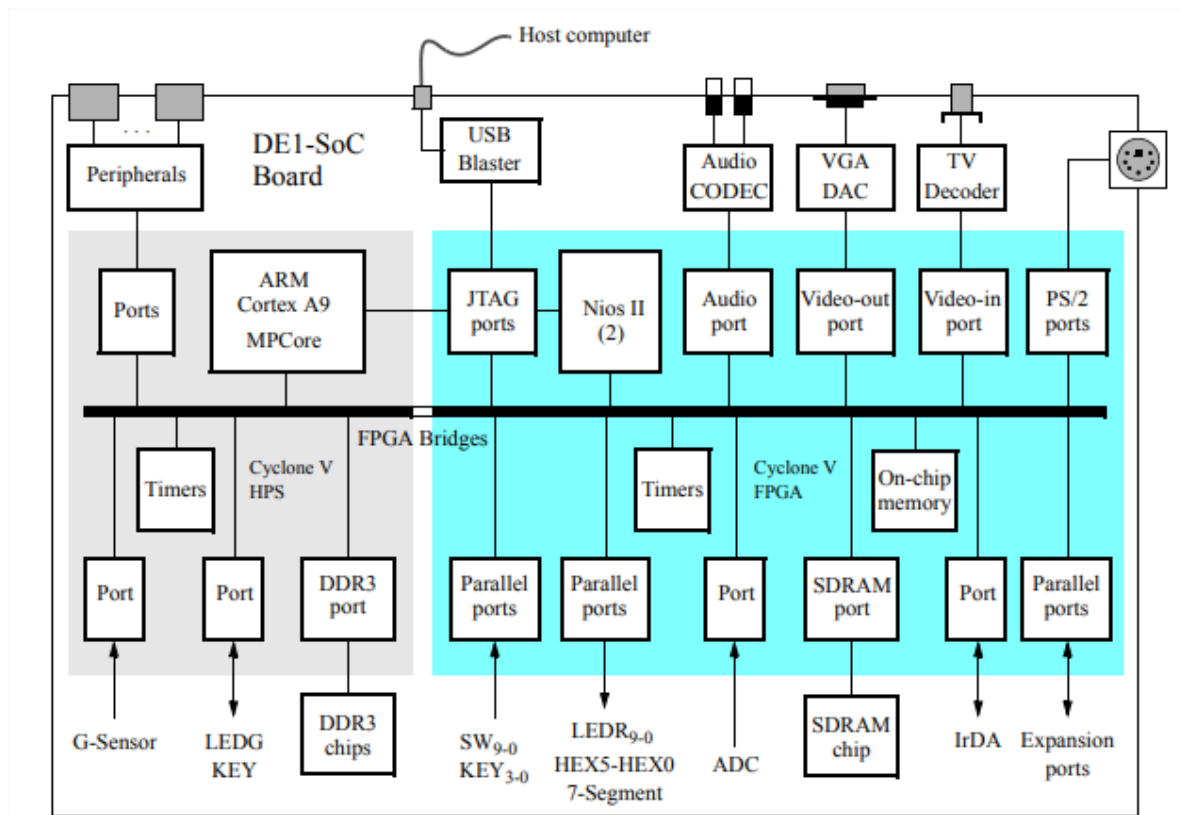


Figura 58: Sistema brindado por Intel. [13]

Una de las ventajas de utilizar Qsys es que se le pueden asignar direcciones de memoria de manera arbitraria para poder controlar los componentes del sistema, estas direcciones son utilizadas en el código en C al momento de hacer el mapeo de memoria y modificar las direcciones físicas utilizando `mmap()`. Para poder decidir si se desea obtener una etiqueta externa para poder acceder a pines físicos del componente solo se debe exportar y asignar un nombre y dirección.

Para la implementación del ECG es necesario el siguiente sistema:

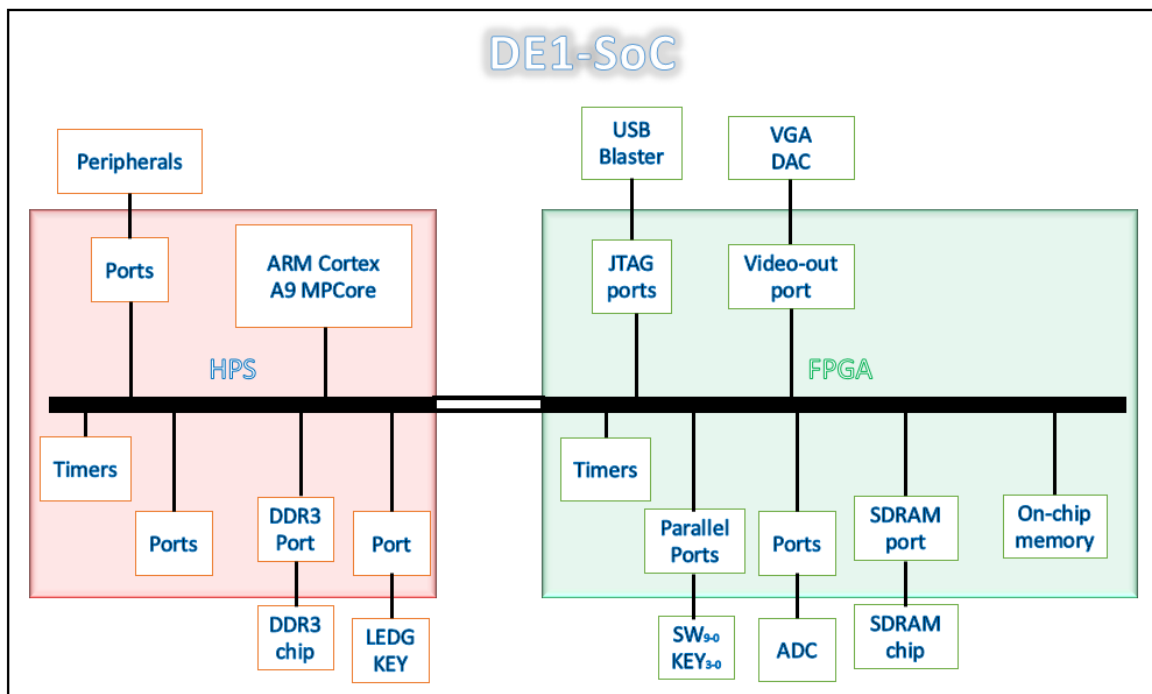


Figura 59: Sistema a implementar en DE1-SoC. [Elaboración propia]

En el sistema de la Figura 59 se observan los periféricos a utilizar, ADC, SW, KEY, la SDRAM, puerto VGA del lado de la FPGA y del lado del HPS la DDR3, LEDG y KEY. Existe una diferencia respecto al sistema mostrado en la figura Figura 58 ya que en este trabajo no se utilizan todos los componentes que la placa DE1-SoC provee.

Para la correcta configuración de cada componente fue necesario leer el funcionamiento básico de los mismos. Por ejemplo para poder utilizar el puerto VGA se creó un pequeño sistema dedicado a video que luego fue anexado al sistema principal, siguiendo la guía “*Making Qsys Components*” de Altera. A continuación se brindará una breve descripción de todos los componentes utilizados.

- **ARM_9_HPS:** es el procesador y es el encargado de controlar los periféricos utilizando las direcciones elegidas para el bus Avalon. También es donde se ejecutará el código compilado en C.
- **SDRAM:** en esta memoria se almacenara la matriz VGA que contiene los valores de color y posición de cada pixel, para poder modificar lo que se ve en pantalla, se direccionara un puntero hacia la SDRAM donde se podrán modificar estos valores. Posee 32 Mbit de almacenamiento por 16 bit de espacio. La matriz VGA posee dimensiones de 640x480 pixeles.
- **ON-CHIP:** es otra memoria RAM, pero con menos espacio de almacenamiento pero con 32 bits de ancho. En esta memoria se almacenaran los caracteres para mostrar mensajes en el puerto VGA. De manera similar los caracteres se almacenan en una matriz de 80x60 caracteres.
- **Pixel_DMA_Addr_Translation:** es el puntero que por defecto se configuro en Qsys para que apunte a la dirección base de la SDRAM que corresponde a 0xc0000000. Cabe destacar que esta dirección puede ser modificada también al momento de ejecutar el código.
- **VGA_Subsystem:** es un subsistema donde se encuentra todo lo relacionado con el puerto VGA, donde se definen parámetros como el modo de direccionamiento (consecutivo o X e Y), bits de color (16 bits u 8 bits), la resolución por defecto VGA 320x240, Remuestreo RGB donde se cambian los 16 bits de color en 1 canal a 30 bits de color con 3 canales de 10 bits cada uno.
- **ADC:** es un periférico que direcciona al convertidor analógico digital, se define la frecuencia a la que trabajara el convertidor y los canales, esto en caso que se cambie el chip que trae por defecto.
- **Interval_Timer:** la placa DE1-SoC cuenta con relojes de propósito general, en el cual uno puede determinar en base la señal de reloj del PLL el tiempo que se necesite que cuente. Por defecto se dejó la frecuencia a 100 MHz.

- **VGA_Char_Buffer:** se encarga de renderizar los caracteres ASCII a una representación de píxeles para que sea entendido por el controlador VGA.
- **VGA_Alpha_Blender:** es el encargado de combinar dos señales de video en una sola, la señal se encuentra en dos partes: una para los píxeles del puerto VGA y otra para mostrar los caracteres. Estas dos señales se combinan en el Alpha_Blender.
- **VGA_Dual_Clock_FIFO:** se utiliza para transferir la señal de video entre dos dispositivos que trabajan a distintas frecuencias de reloj.
- **VGA_Controller:** genera las señales requeridas que entiende el DAC VGA que tiene la placa.

CAPITULO 4. IMPLEMENTACION

4.1 Sistema de Procesado de señal

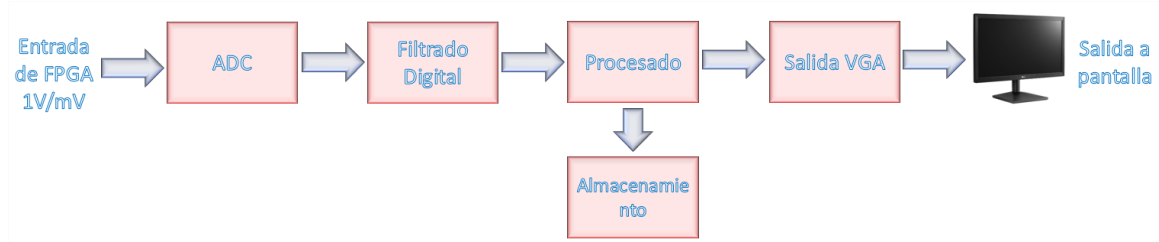


Figura 62: Esquema de procesamiento de señal. [Elaboración propia]

4.1.2 Sistema de adquisición de datos

Para la adquisición de datos se utiliza el convertidor analógico digital que posee la placa DE1-SoC, este es un convertidor de 12 bits de resolución, la frecuencia de muestreo utilizada es $f_s = 250 \text{ muestras/s}$, se debe hacer énfasis en que se cumple el criterio de Nyquist ya que el espectro en frecuencia de la señal generada por el corazón no supera los 50 Hz (Ver Figura 67 "Espectro en frecuencia de actividad eléctrica del corazón").

4.2 Filtrado digital

A pesar de haber utilizado un filtro analógico en la etapa *Sistema de adquisición de señal* se presentan ruidos parásitos de la red eléctrica y actividad de baja frecuencia que desplazan la señal y entorpecerían el procesado de la misma, para resolver este problema se hace uso de técnicas de filtrado digital, se ha aplicado un filtro pasabanda con las mismas frecuencias de corte que el filtro analógico, solo que de mayor orden, 50 para ser específico, como se demostró en el apartado de Teoría de filtrado, se sabe que el número de coeficientes de la función de transferencia es igual al orden del filtro.

Para obtener los coeficientes del filtro se hizo uso del programa *FIR Filter Designer* en el cual se puede escoger el tipo de filtro, frecuencias de corte, ventana y pueden ser exportados los coeficientes a un archivo *.txt.

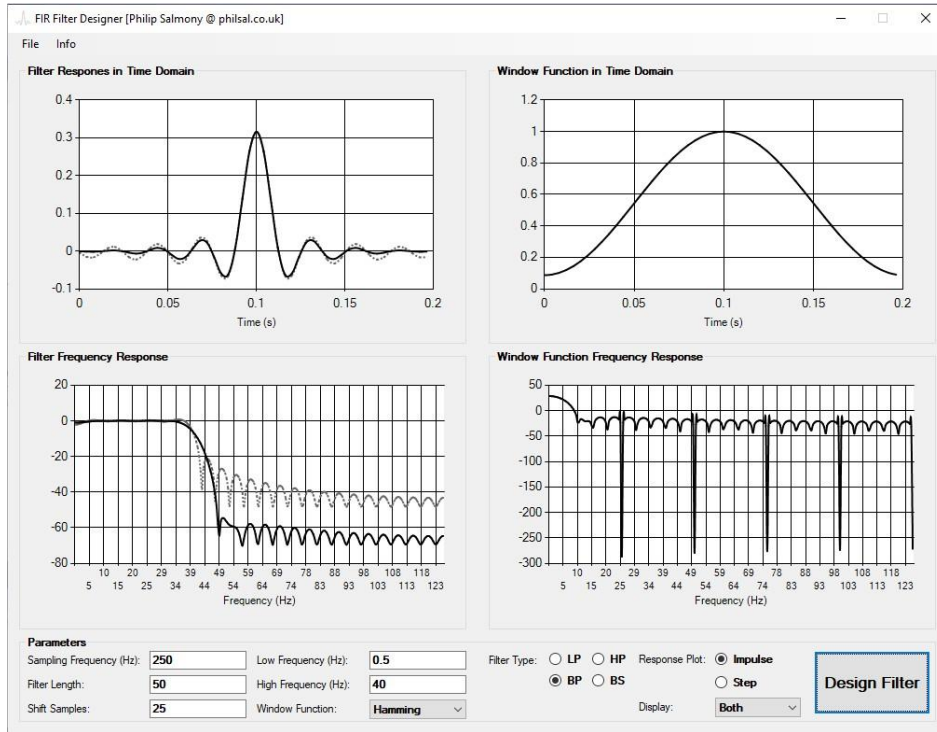


Figura 63: Caracterización de filtro FIR a implementar. [Elaboración propia]

Se llena la información con el tipo de filtro que se desea obtener. Luego de esto exportar los coeficientes como se muestra en la siguiente imagen.

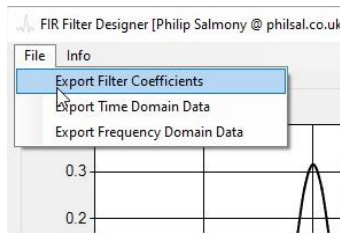


Figura 64: Exportado de coeficientes. [Elaboración propia]

Del cual se guarda en la ubicación que se desee y se le asigna un nombre para identificarlo.

```

250con50.txt - Notepad
File Edit Format View Help
Filter Order: 50 Sampling Frequency (Hz): 250.000000 Cut-Off Frequency Lo (Hz): 0.500000 Cut-Off Frequency Hi (Hz): 40.000000

-0.0003421,-0.001370831,-0.001668085,-0.000685841,0.001108018,0.001946675,0.000043030,-0.004057720,-0.006683175,-0.003943690,0.003419171,0.008568945,0.004022588,-0.009591282,-0.020674337

float coeff[] = {-0.0003421f,-0.0013708f,-0.0016681f,-0.0006858f,0.0011080f,0.0019467f,0.0000430f,-0.0040577f,-0.0066832f,-0.0039437f,0.0034192f,0.0085689f,0.0040226f,-0.0095913f,-0.0206

```

Figura 65: archivo de texto generado por FIR Filter Designer. [Elaboración propia]

La última línea es la de interés ya que formara parte del código a implementar.

4.3 Procesado

4.3.1 Almacenamiento

Una vez la señal ya ha sido filtrada se procede a almacenar los datos en un archivo *.csv (Comma Separated Values o valores separados por coma) el cual permite cargar estos datos ya sea a Excel, Octave, MATLAB o cualquier software que soporte la lectura de archivos con esta extensión. Este archivo es guardado en la ubicación donde se ejecute el programa, con el nombre del paciente y contiene los valores tal como se obtienen en el ADC y ya filtradas.

4.3.2 Detección de complejo QRS

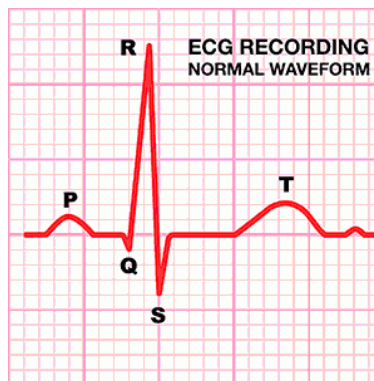


Figura 66: Registro típico de una señal de ECG. [15]

La detección del complejo QRS se reduce a la detección de la onda R dado que su amplitud es la más alta en toda la señal, puede ser detectada identificando los puntos más altos del ECG. Debido a esto es importante eliminar el offset de la señal y eliminar las bajas frecuencias que provocan un movimiento periódico en la altura de las muestras.

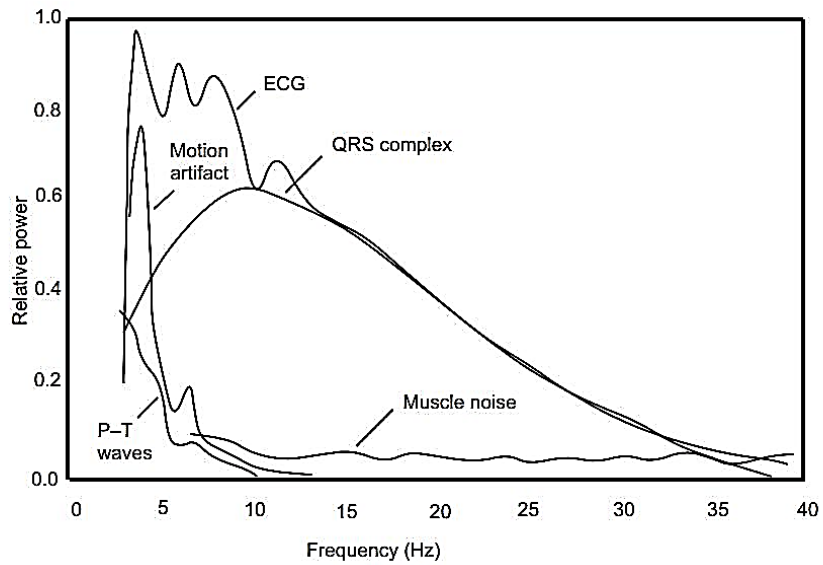


Figura 67: Espectro en frecuencia de actividad eléctrica del corazón. [16]

En la imagen anterior se puede ver el rango de frecuencias atribuido a las tres partes de una onda ECG, la onda P, T y QRS. Utilizando esta información de las frecuencias atribuidas a las ondas se pueden aplicar filtros específicos que permitan atenuar las señales que no son de interés.

4.4 Pruebas utilizando periféricos de placa DE1-SoC

Se han creado dos códigos donde se interactúa con los parámetros del puerto VGA. Dada la complejidad que posee la manipulación de los píxeles se deben crear rutinas para poder dibujar la interfaz gráfica.

Se han podido dibujar líneas de colores en la pantalla además de poder dibujar un cuadrado de las dimensiones y colores que se deseen.

```
COM8 - PuTTY
root@delsoclinux:~# ls
DE1_SoC_Computer.rbf  drawbox      init_openc1.sh  openc1.rbf
OpenCL_Examples      gsensor     leds            openc1_arm32_rte
animation_bouncy     helloworld  mem_rw         pushbutton_irq_handler
root@delsoclinux:~# cd /
root@delsoclinux:/# ls
bin  dev  home  media  opt  root  sbin  srv  tmp  var
boot  etc  lib  mnt  proc  run  selinux  sys  usr
root@delsoclinux:/# cd media/
root@delsoclinux:/media# cd fat_partition/
root@delsoclinux:/media/fat_partition# ls
OLIVER          allfpga      drawbox        reloj.c        vga
System Volume Information  allfpga.c    drawbox.c      soc_system.rbf  vga.c
adc             bouncy      pixelbuffer.c  socfpga.dtb
adc.c           bouncy.c    reloj          uImage
root@delsoclinux:/media/fat_partition# ./drawbox 10 30 10 40 0xFFFF
root@delsoclinux:/media/fat_partition# ./drawbox
Error. Invalid arguments.
Usage: drawbox <color (2-byte HEX)> <x1> <y1> <x2> <y2>
root@delsoclinux:/media/fat_partition# ./drawbox 0xffff 10 30 10 40
root@delsoclinux:/media/fat_partition# ./drawbox 0xffff 10 10 30 40
root@delsoclinux:/media/fat_partition# ./drawbox 0xfff0 20 40 60 90
root@delsoclinux:/media/fat_partition# ./drawbox 0x000f 90 90 200 200
root@delsoclinux:/media/fat_partition#
```

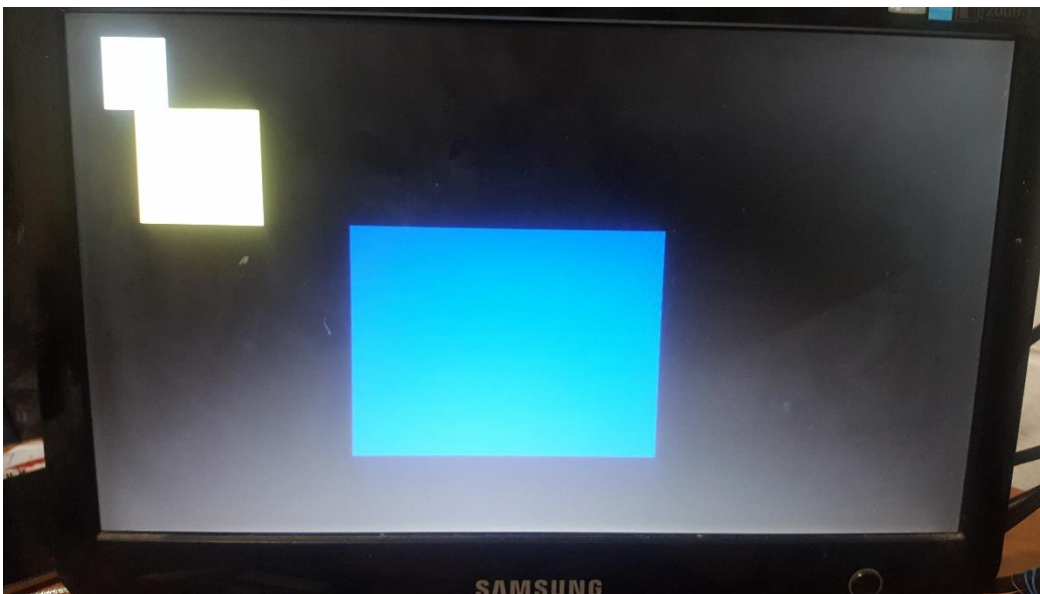


Figura 68: Ejecución de código ejemplo de uso de puerto VGA. [Elaboración propia]

```
COM8 - PuTTY
gsensor.c
halo ce by the king of MP4.zip
instruc.txt
mem_rw
msxmlenu.msi
pixelbuffer
pixelbuffer.c
raton.c
reloj.c
relojed.c
relojinterval.c
relojito
relojito2
relojitoo
root
vgacolor
vgacolor.c
vgares
vgares.c
root@delsoclinux:/media/fat_partition/OLIVER# ./vgares
RES f00140
RES f00140
STATUS 8090020
```

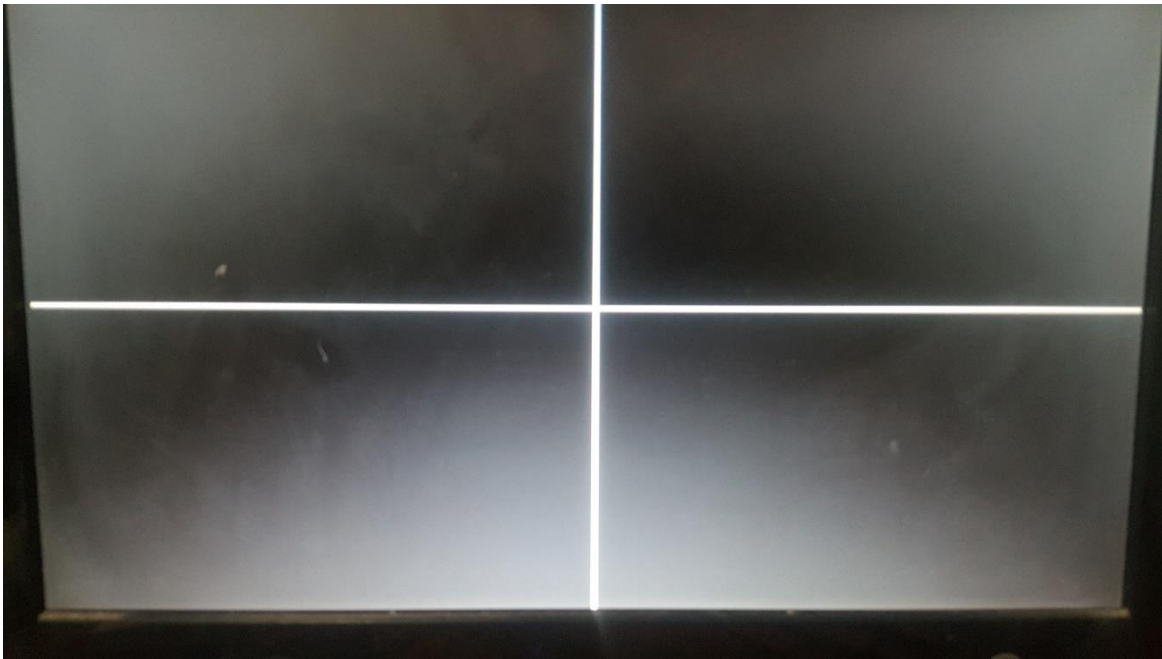


Figura 69: División de pantalla en 4 partes iguales. [Elaboración propia]

```
COM8 - PuTTY
mem_rw
msxmlenu.msi
pixelbuffer
pixelbuffer.c
raton.c
reloj.c
relojed.c
relojinterval.c
relojito
relojito2
relojitoo
root
vgacolor
vgacolor.c
vgares
vgares.c
root@delsoclinux:/media/fat_partition/OLIVER# ./vgacolor
RES f00140
STATUS 8090020
RES f00140
STATUS 8090020
Ingrese fila 50
Ingrese columna 60
Ingrese fila █
```

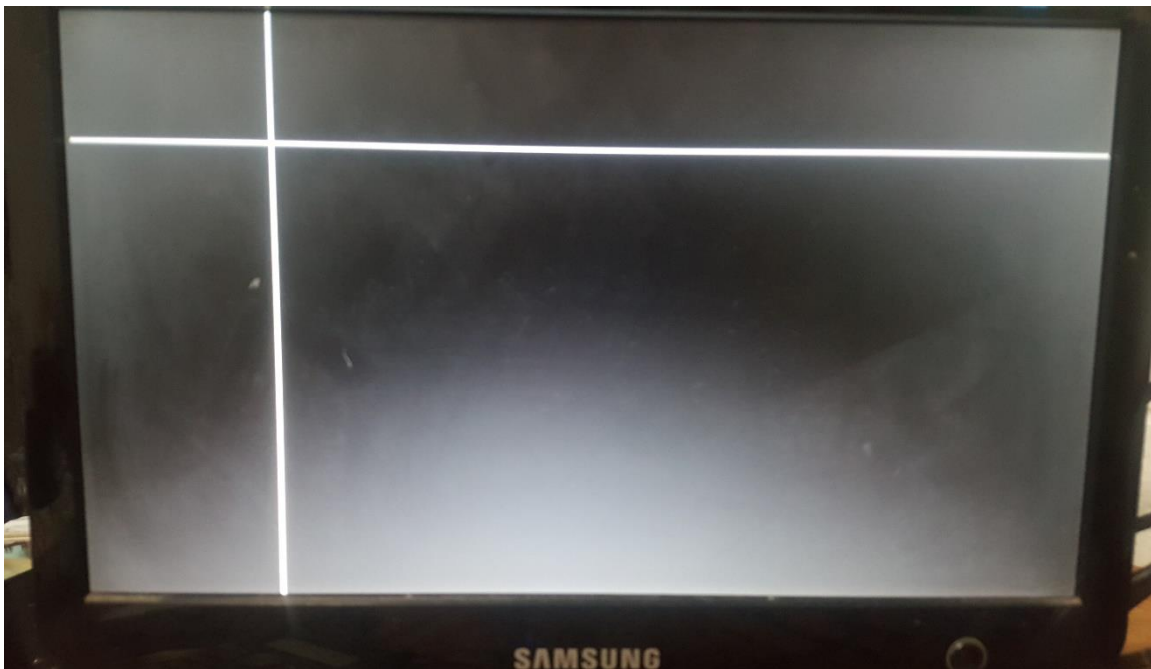


Figura 70: Generación de líneas en la fila y columna de preferencia. [Elaboración propia]


```
COM8 - PuTTY
vgacolor
vgacolor.c
vgares
vgares.c
root@delsoclinux:/media/fat_partition/OLIVER# gcc allfpga
allfpga    allfpga.c
root@delsoclinux:/media/fat_partition/OLIVER# ./allfpga
LED = 0 SW = 0 HE30 = 0      HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16191  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16134  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16219  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16207  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16230  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16237  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16253  HE54 = 0      KEY = 0
-----
LED = 0 SW = 0 HE30 = 16135  HE54 = 0      KEY = 0
```

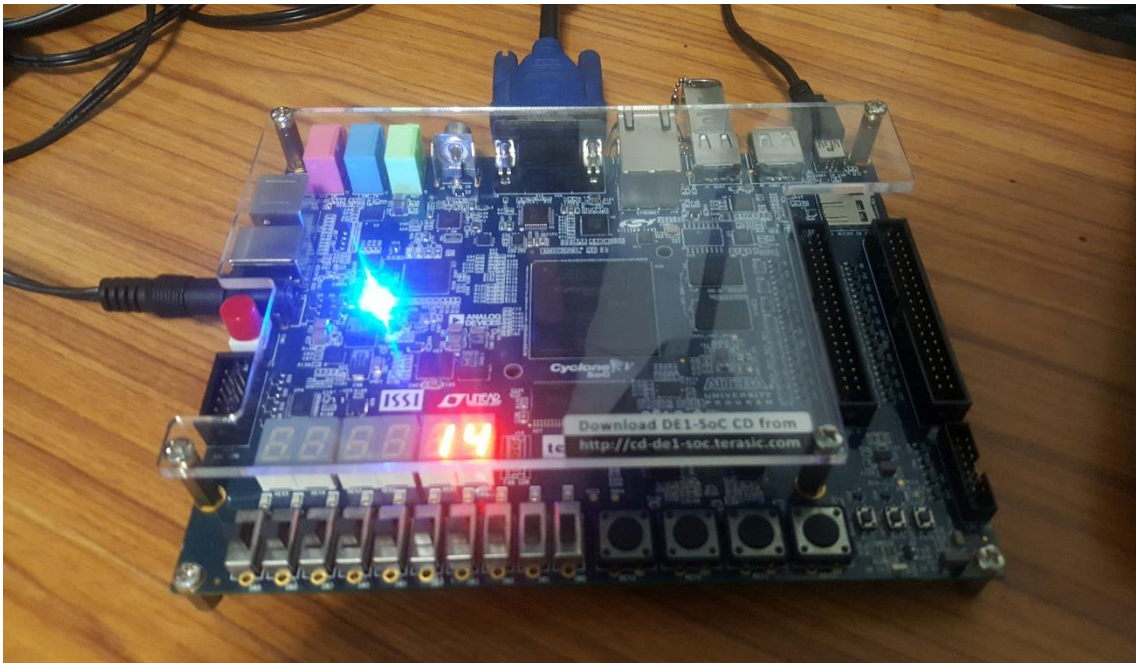


Figura 71: Medición del estado de los periféricos de la FPGA, LEDs, Display, y pulsadores y contador 60s. [Elaboración propia]

```
COM8 - PuTTY
root@delsoclinux:/media/fat_partition# ./adc
adc0= 1343
adc0= 2049
adc0= 2040
adc0= 2039
adc0= 2041
adc0= 2041
adc0= 2032
adc0= 2047
adc0= 2033
adc0= 2033
adc0= 2030
adc0= 2040
adc0= 2049
adc0= 2042
adc0= 2039
adc0= 2038
adc0= 2048
adc0= 2044
adc0= 2034
adc0= 2033
adc0= 2039
adc0= 2047
adc0= 2047
```

Figura 72: Lecturas del conversor analógico digital de la FPGA. [Elaboración propia]



Figura 73: Circuito de fuente de voltaje de $\pm 12V$. [Elaboración propia]

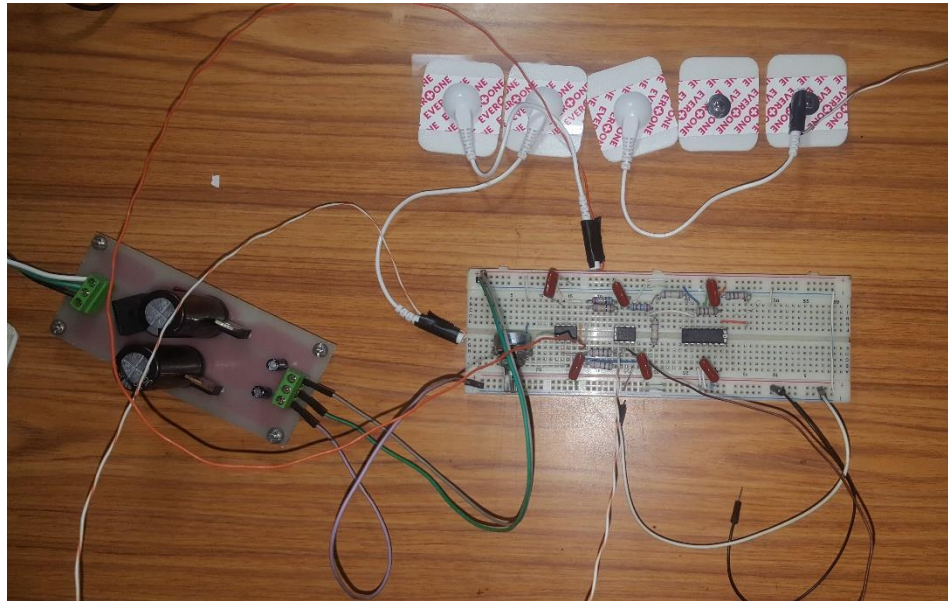


Figura 74: Implementación de ECG en Breadboard. [Elaboración propia]

De la figura anterior se puede notar que están tres electrodos conectados al circuito, hasta cierto punto del proyecto la fuente de referencia fue un divisor de voltaje, se optó por utilizar un DAC para obtener valores de tensión más estable, ya que anteriormente variaba de entre 2.48V y 2.54V esto sucede por los cambios de temperatura y otras condiciones fuera del control.



Figura 75: Electrodo a utilizar cuando el sistema esté completo. [Elaboración propia]

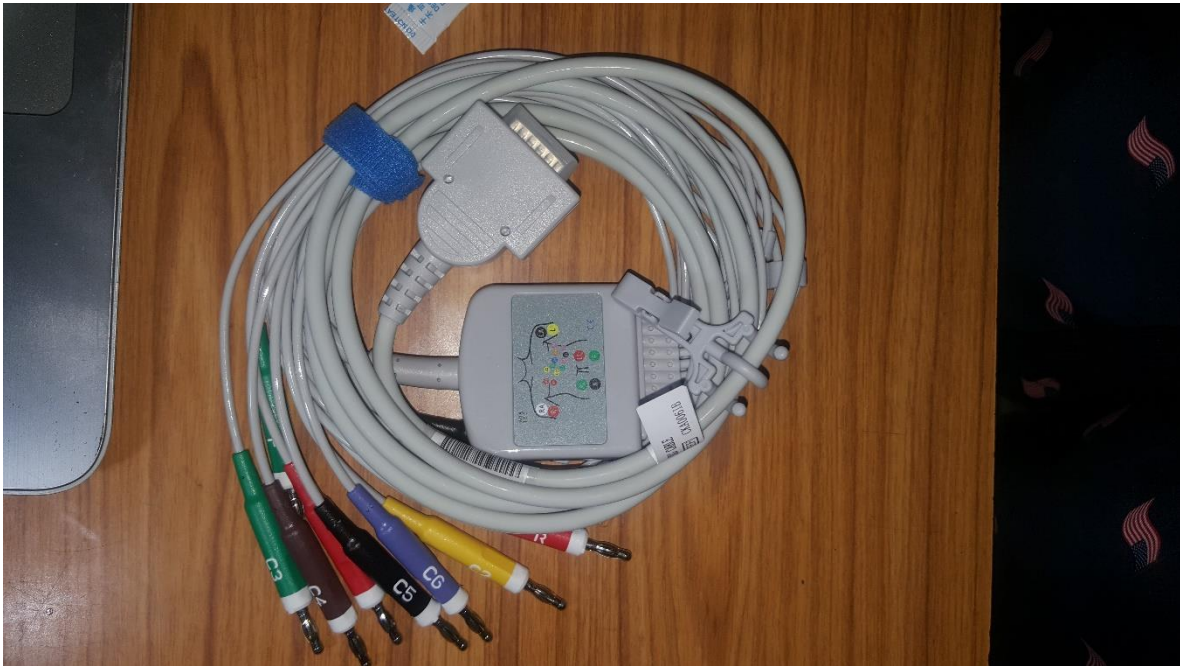


Figura 76: Cables a utilizar para transmitir las señales desde los electrodos al sistema de adquisición de datos. [Elaboración propia]

4.5 Experimentación en breadboard.

Lo primero que se puso a prueba fue que el filtro funcionara bajo las condiciones para las que se diseñó, con frecuencia de corte de paso bajo en 40 Hz y en paso alto de 0.5 Hz. A lo cual se hizo pasar una señal senoidal a través del circuito para determinar sus frecuencias de corte.

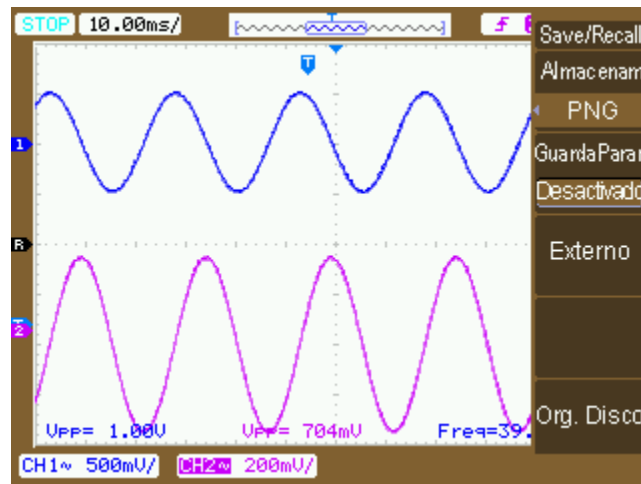


Figura 77: Frecuencia de corte filtro paso bajo 39 Hz. [Elaboración propia]

Debido a las limitaciones del equipo utilizado la frecuencia de corte del filtro paso alto no pudo ser determinada.

Luego de determinar que el circuito estaba trabajando, se procedió a inyectar las señales por medio de electrodos ubicados de la siguiente manera: referencia en pierna derecha, electrodo positivo en mano izquierda, electrodo negativo en mano derecha. De esta forma se obtiene la derivación DI.

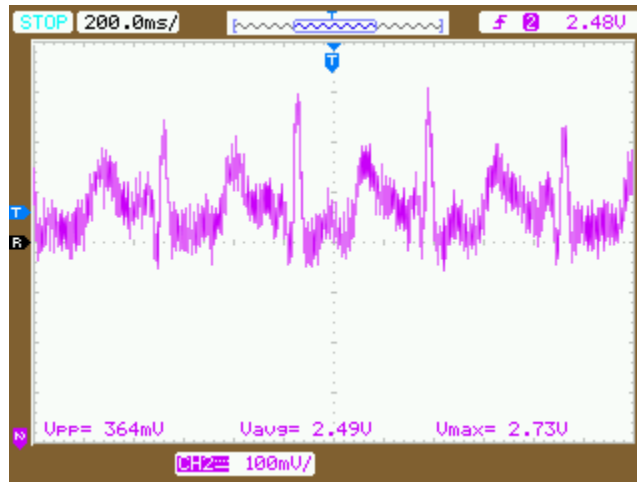


Figura 78: Señal obtenida derivación DI implementado en breadboard. [Elaboración propia]

De la figura anterior se puede determinar que la señal obtenida se encuentra montada en una señal DC a 2.5V como se diseñó. Y que no se ve involucrada señal alterna de baja frecuencia ya que no oscila y se mantiene entorno a la señal DC.

4.6 Experimentación e Implementación en PCB (Parte I).

Al haber obtenido resultados positivos en la experimentación, se procedió al diseño y armado del circuito final utilizando el software Eagle.

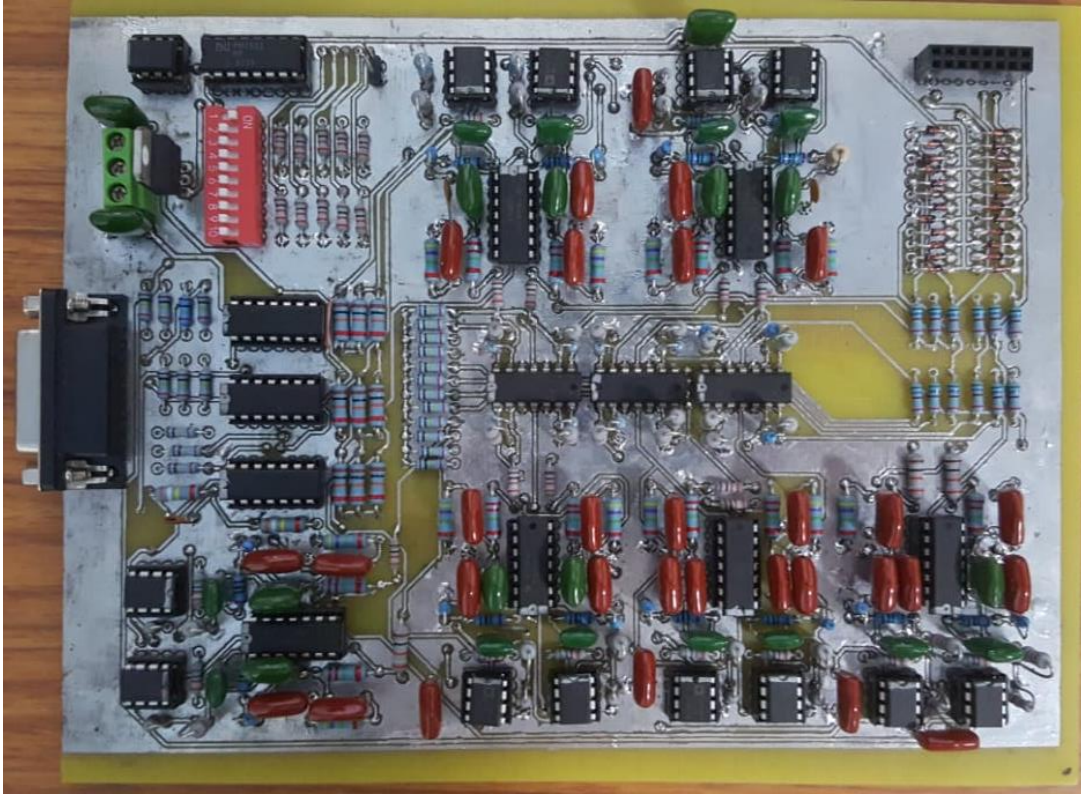
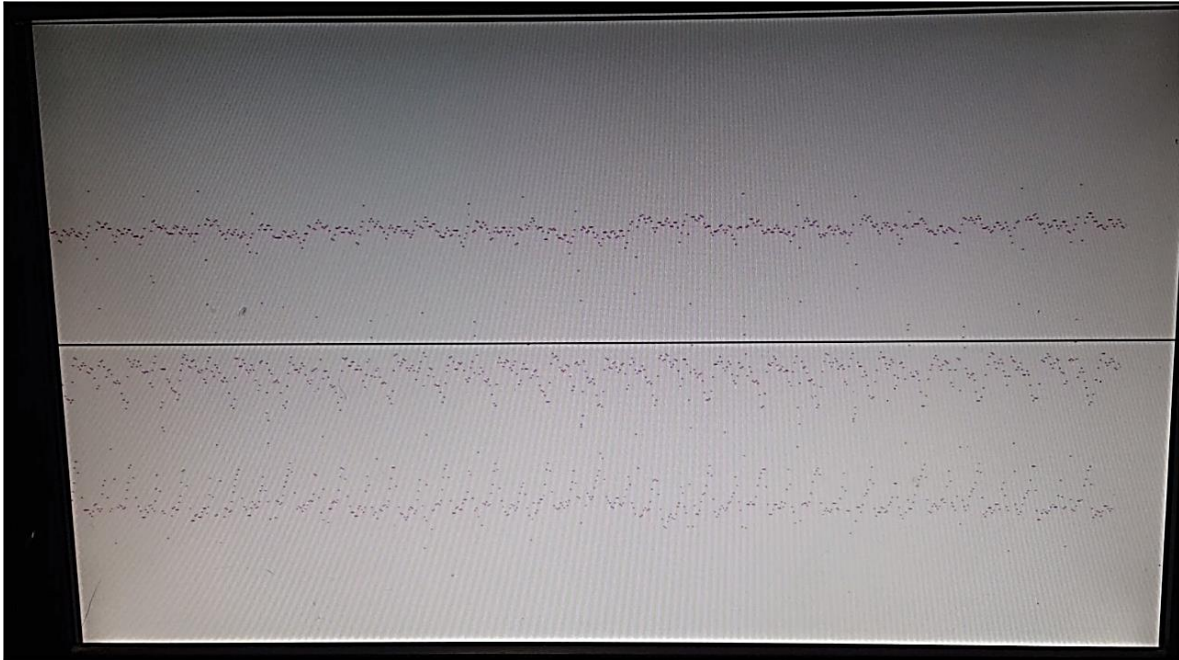


Figura 79: Circuito en PCB Finalizado. [Elaboración propia]



*Figura 80: Primer prueba de obtención de señal 50 muestras/s, graficas de arriba hacia abajo: DIII, DI y DII.
[Elaboración propia]*

Se puede apreciar que la velocidad de muestreo utilizada es baja y debe ser aumentada para obtener una mejor resolución, esto implica que el tiempo asignado a cada pixel debe disminuir, a lo cual hay que modificar partes del código donde se envía al puerto VGA la imagen.

Se ha tomado una muestra cada 20 ms, es decir 50 muestras por segundo, cada pixel corresponde a un intervalo de 20ms, la resolución del puerto VGA es 640x480 pixeles, a lo cual se completa un ciclo cada 12.8 segundos.

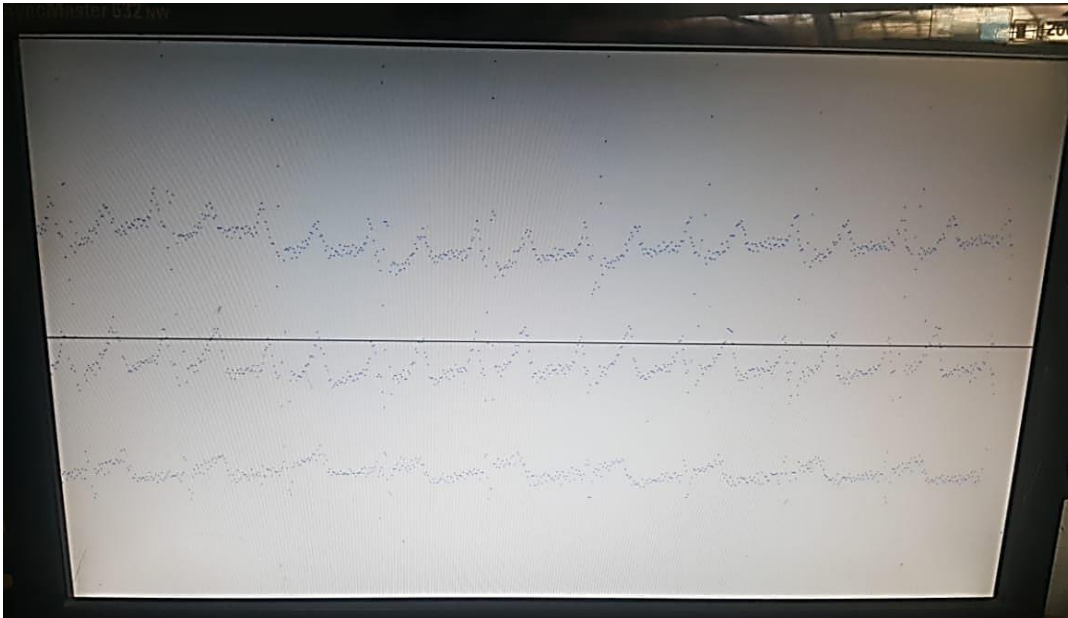


Figura 81: Segunda prueba de obtención de señal 100 muestras/s, graficas de arriba hacia abajo: DI, DII y DIII. [Elaboración propia]

Luego se aumentó la frecuencia de muestreo al doble, 100 muestras por segundo, se puede apreciar un leve balanceo en la señal DI en la figura anterior, este ruido de baja frecuencia será removida digitalmente utilizando el algoritmo presentado en la sección “Detección de complejo QRS”.

4.7 Lógica de programación.

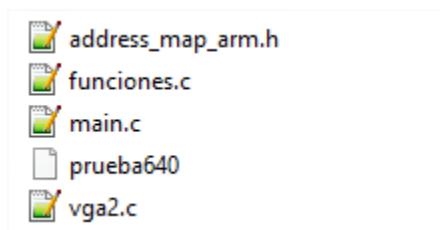


Figura 82: archivos del programa ECG. [Elaboración propia]

Se cuenta con 4 archivos que conforman el programa del electrocardiógrafo, se detalla un poco acerca de cada uno.

- **address_map_arm.h:** este archivo contiene todas las direcciones que se obtienen utilizando la herramienta Qsys al momento de compilar todos los periféricos que se utilizan para este proyecto.



```

#define LDATA          2
#define RDATA          3
#define VIDEO_IN_BASE 0x00203060
#define ADC_BASE       0x00204000
#define ADC1_BASE      0x00204001
#define ADC2_BASE      0x00204008

```

Figura 83: direccionamiento de la base de los ADC. [Elaboración propia]

Se puede notar que la dirección posee otra base adicional que es 0x00200000, esta corresponde al direccionamiento del bus de datos hacia LW_BASE que es lo que conecta el procesador con el periférico en este caso el ADC.

- **funciones.c:** esta contiene todos los procesos que se ejecutaran en el programa, tales como el control de leds, displays, botones y ADC's.
- **main.c:** este contiene el orden en que serán ejecutadas las funciones como leer los pines del ADC o enviar al VGA la imagen generada, controlar los relojes que se utilizan, detección de picos, etc.
- **vga2.c:** este contiene todo el segmento de código que controla el puerto VGA, tales como: limpiar la pantalla, generar mensajes o alertas, cambiar la forma de visualización, generar el eje de coordenadas, etc.

4.8 Experimentación e Implementación en PCB (Parte II).

Se han logrado cambios significativos en la forma en que se procesa, almacena y presenta la señal, algunos de los cambios son:

- Anteriormente la frecuencia de muestreo máxima alcanzada fue de 100 muestras/segundo, si bien cumple con el criterio de Nyquist la presentación en el puerto VGA carecía de nitidez y se notaban ciertos recortes en toda la señal, ahora se ha optimizado el algoritmo de impresión en pantalla ahorrando ciclos de ejecución de código, pudiendo aumentar la frecuencia hasta 1000 muestras/ segundo.

- La frecuencia de muestreo actual es de 250 muestras/segundo, se definió este valor para poder visualizar al menos 2 complejos QRS durante un ciclo dado que un ritmo cardiaco normal en reposo ronda los 60 latidos por segundo, se genera un complejo QRS cada 1 segundo, y el tiempo de refresco en pantalla es de 2.56 segundos.
- Se le aplico un primer filtrado digital a los datos de entrada.
- Ya se implementó un segmento de código dedicado al almacenamiento de los datos medidos en bruto y los filtrados, para poder ver el contraste entre estas dos señales.

Dado que solo se cuenta con 8 canales y de estos 8 solo funcionan 6, se midieron las derivaciones precordiales para su posterior comparación con lo que se esperaría ver de un ECG normal.

Para la experimentación la interconexión de la placa y el PCB es de la siguiente manera:

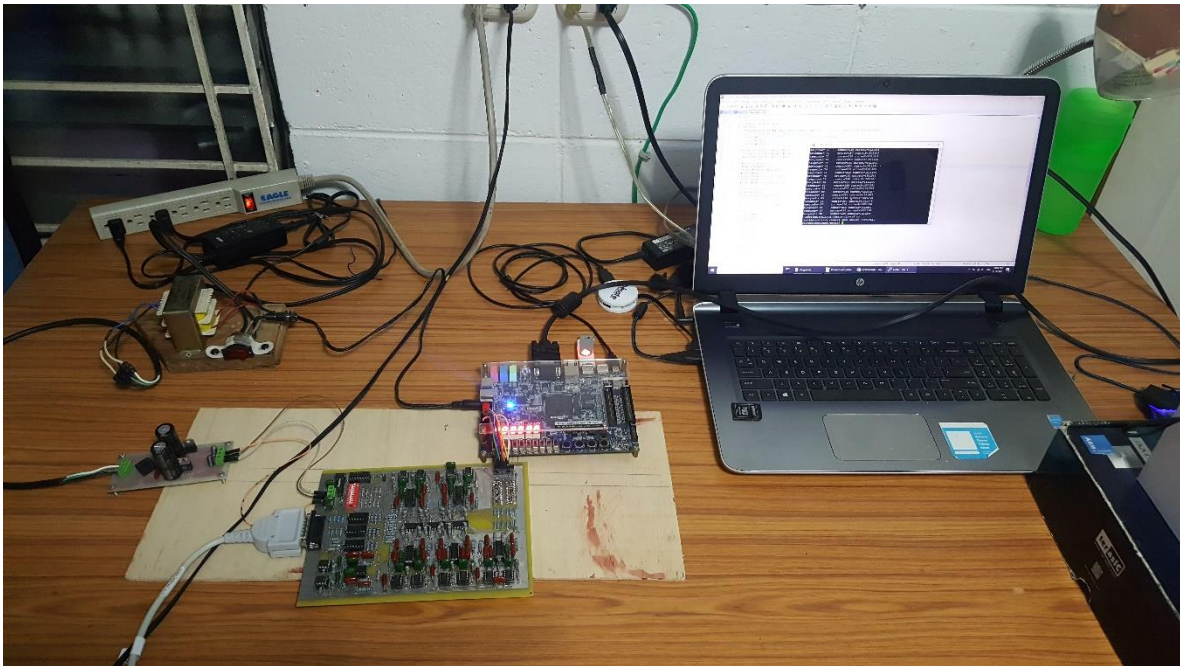


Figura 84: Disposición de conexiones para experimentación. [Elaboración propia]

El cable VGA está conectado hacia un televisor que cumple la función de monitor.

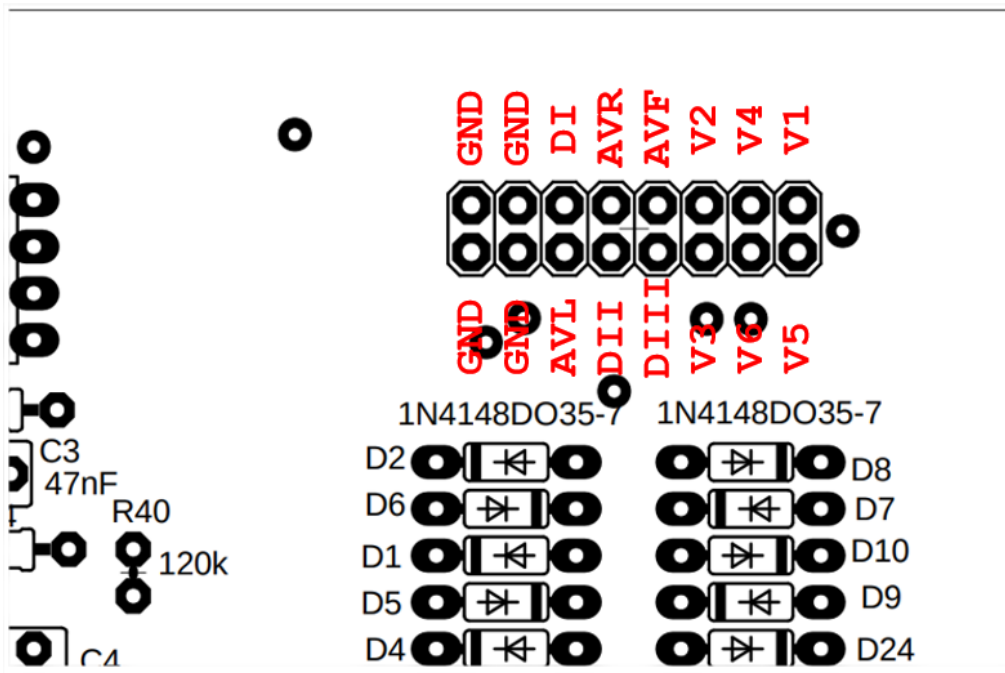


Figura 85: Disposición de pines de salida de PCB. [Elaboración propia]

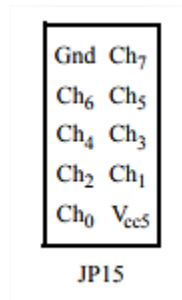


Figura 86: Disposición de pines del ADC de DE1-SoC. [Elaboración propia]

Teniendo las dos figuras anteriores para comprender la conexión, se muestra la tabla de interconexión entre el PCB y la placa DE1-SoC.

Tabla 4: Conexiones entre DE1-SoC y PCB. [Elaboración propia]

Canal	Derivación
ADC1	V2
ADC2	V3
ADC3	V4
ADC4	V5
ADC5	V6
ADC6	V1

4.9 Resultados

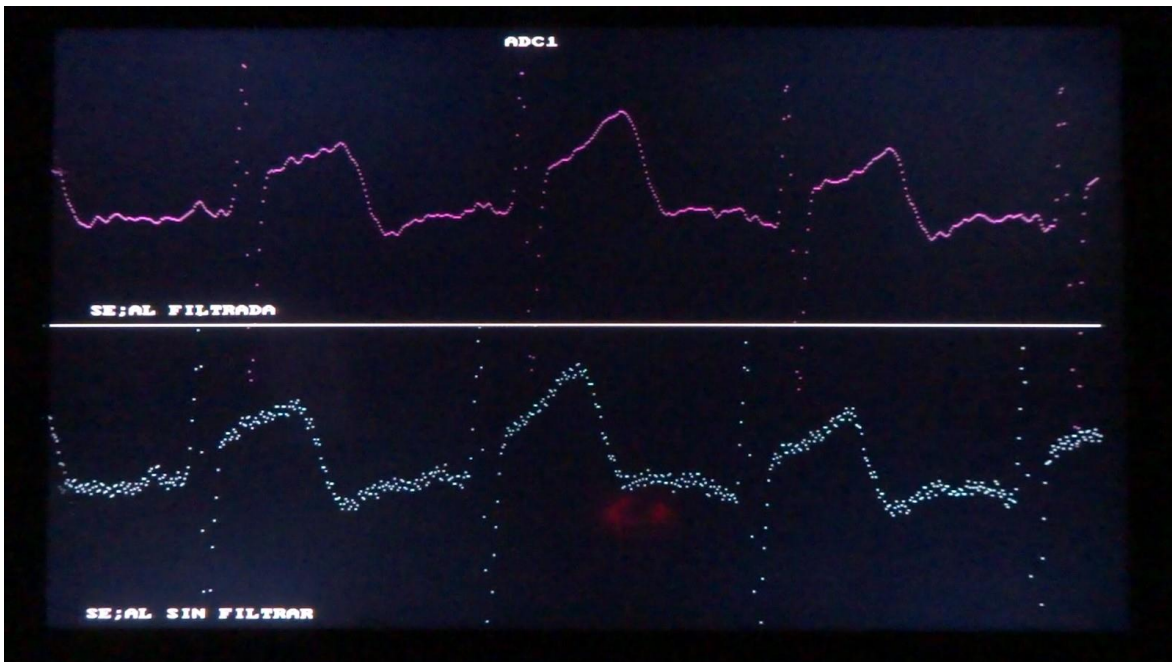


Figura 87: Grafica ADC1 - Derivación V2. [Elaboración propia]

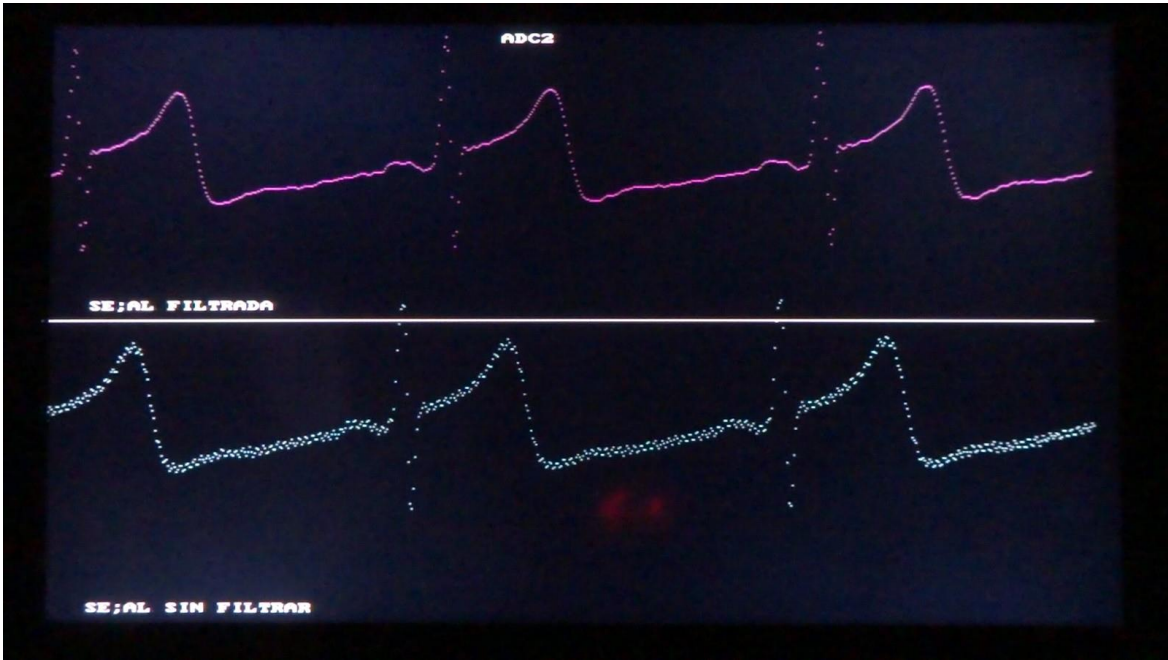


Figura 88 Grafica ADC2 - Derivación V3. [Elaboración propia]

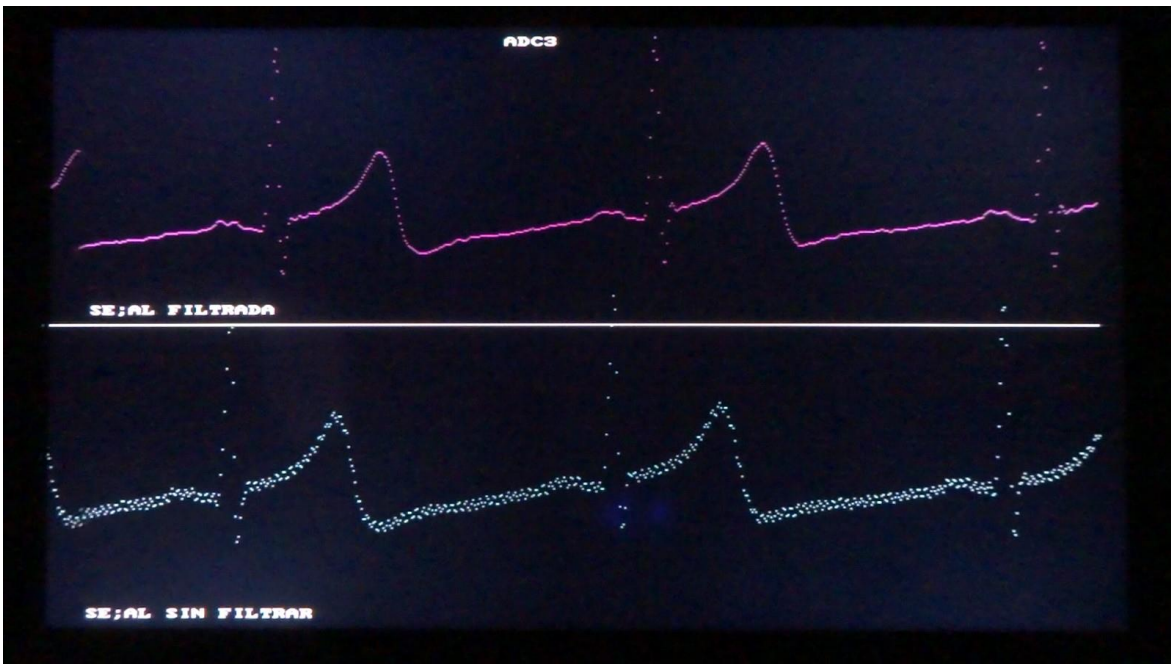


Figura 89 Grafica ADC3 - Derivación V4. [Elaboración propia]

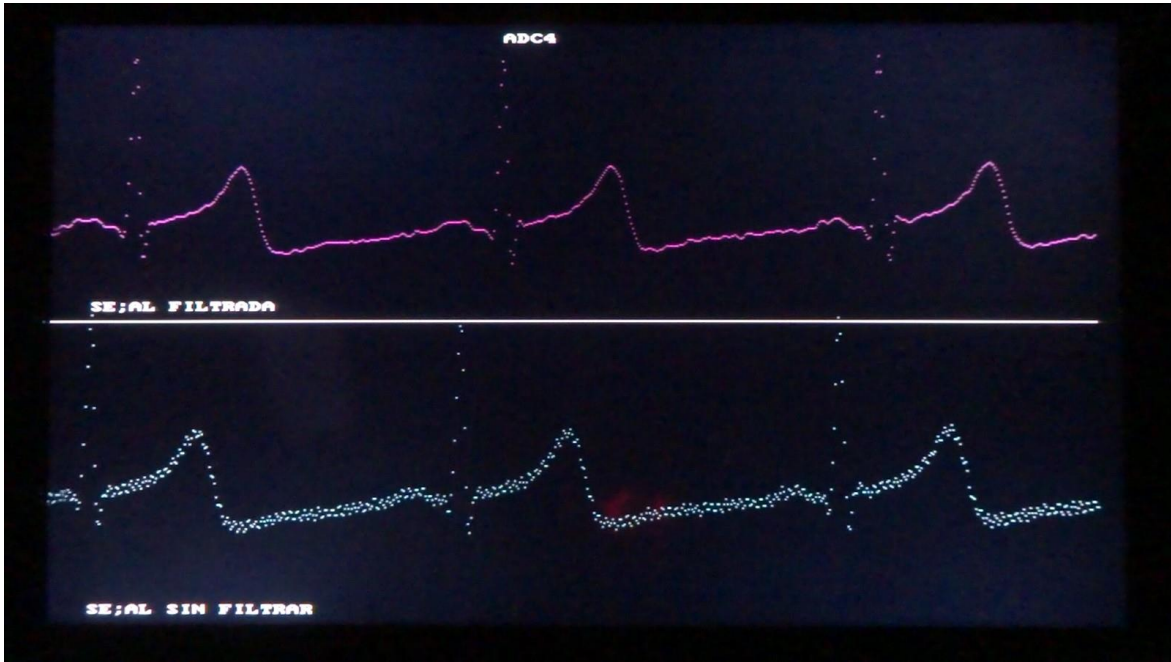


Figura 90 Grafica ADC4 - Derivación V5. [Elaboración propia]

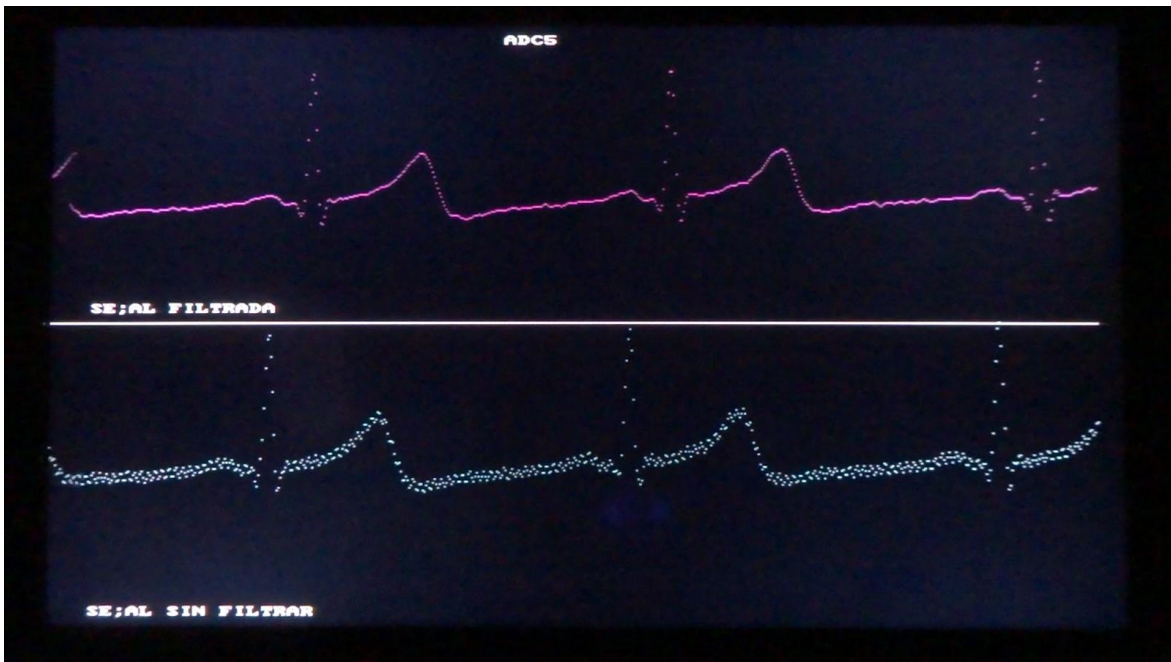


Figura 91 Grafica ADC5 - Derivación V6. [Elaboración propia]

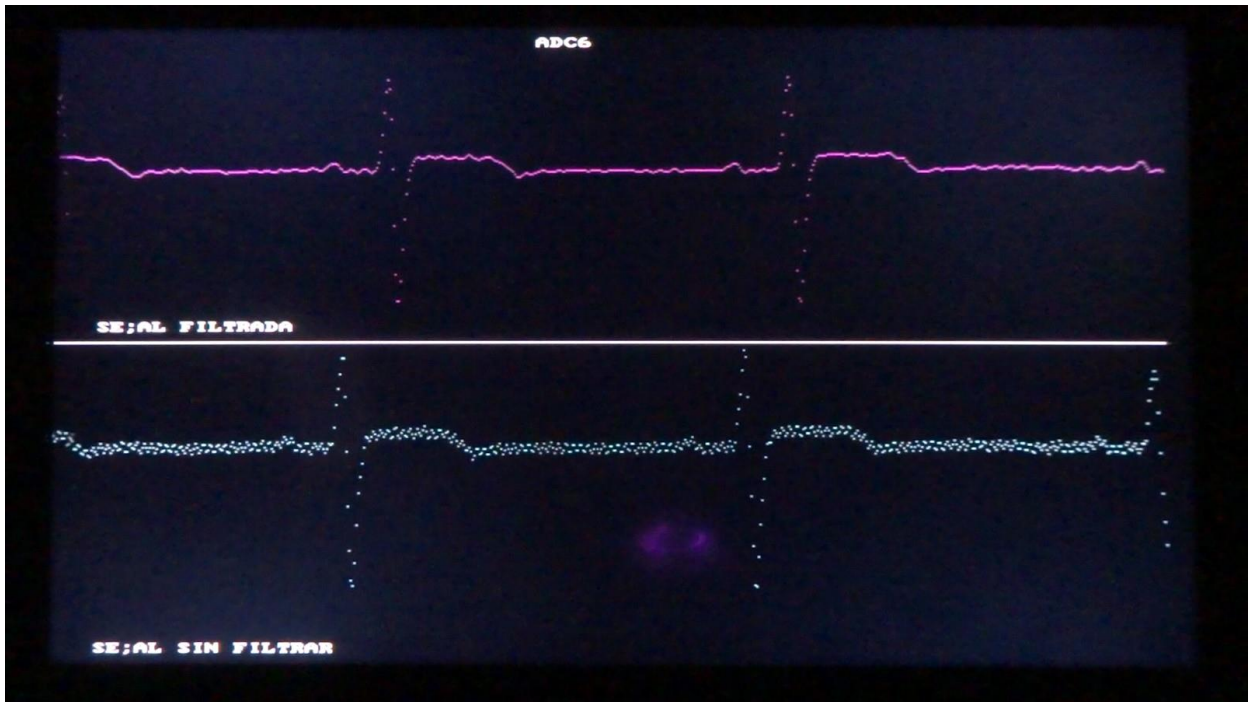


Figura 92 Grafica ADC6 - Derivación V1. [Elaboración propia]

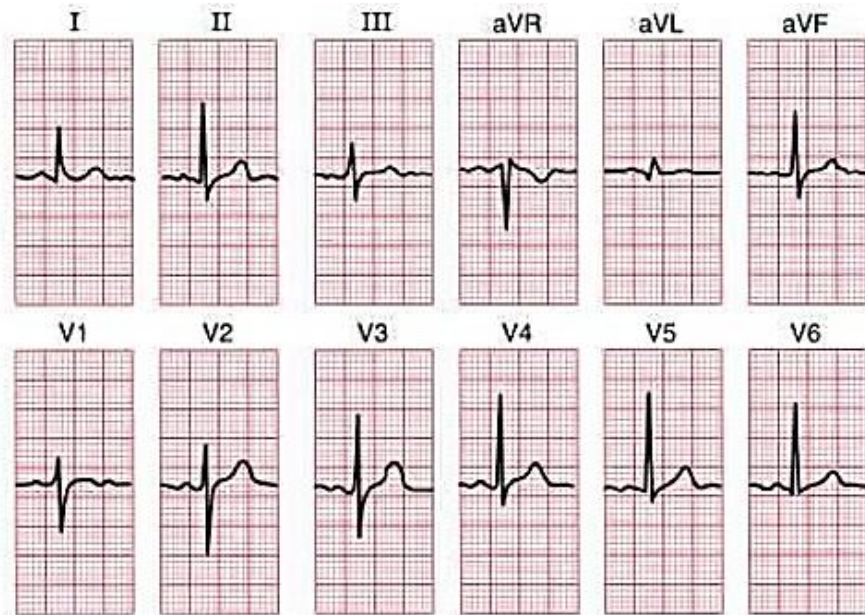


Figura 93: Formas de onda de un ECG normal. [Elaboración propia]

Por comparación, se observa que las gráficas obtenidas son muy similares a las formas de onda esperadas. Falta graficar el eje de tiempo y amplitud es decir la cuadrícula.

```

COM3 - PuTTY
^Croot@delsoclinux:~# ./prueba640
==CONSECUTIVE=1
COLOR_16=1
BUFF_ADD=1003
BBUFF_ADD=10003
DMA=130022
RESOLUTION=1e00280
fs=250  ret1=1000
inicio filtro
hsegundo= 1   conteo=249  conteo2=1512023
hsegundo= 2   conteo=250  conteo2=1522492
hsegundo= 3   conteo=250  conteo2=1521896
hsegundo= 4   conteo=250  conteo2=1522169
hsegundo= 5   conteo=250  conteo2=1522017
hsegundo= 6   conteo=250  conteo2=1522005
hsegundo= 7   conteo=250  conteo2=1522185
hsegundo= 8   conteo=250  conteo2=1521973
hsegundo= 9   conteo=250  conteo2=1522008
hsegundo= 10  conteo=250  conteo2=1522316
hsegundo= 11  conteo=250  conteo2=1522481
hsegundo= 12  conteo=250  conteo2=1522544
hsegundo= 13  conteo=250  conteo2=1522413
hsegundo= 14  conteo=250  conteo2=1522232
hsegundo= 15  conteo=250  conteo2=1522319

```

Figura 94: Interfaz CLI durante la ejecución del programa. [Elaboración propia]

Al iniciar el programa se muestra la configuración del puerto VGA así como la frecuencia de muestreo, a continuación el detalle de tiempo y cuantas muestras son tomadas en cada segundo.

```

cordialesbuena.csv - Notepad
File Edit Format View Help
t,No,ADC1,ADC2,ADC3,ADC4,ADC5,ADC6,ADC7,AD1F,ADC2F,ADC3F,AD4F,ADC5F,ADC6F,ADC7F
0,1,0.979004,3.474121,3.917236,3.846436,3.865967,2.310791,2.392578,-0.000335,-0.001188,-0.001340,-0.001316,-0.001323,-0.000791,-0.000819
0,2,0.842285,3.233643,3.709717,3.687744,3.756104,2.177734,2.357178,-0.001630,-0.005869,-0.006639,-0.006534,-0.006584,-0.003913,-0.004086
0,3,0.659180,2.531738,2.946777,3.048096,3.306885,1.964111,2.286377,-0.003013,-0.011094,-0.012628,-0.012514,-0.012729,-0.007512,-0.008004
0,4,0.837402,2.088623,2.277832,2.425537,2.827148,2.020264,2.309570,-0.003266,-0.011962,-0.013693,-0.013798,-0.014417,-0.008601,-0.009497
0,5,1.307373,2.098389,2.060547,2.148438,2.578125,2.260742,2.375488,-0.002188,-0.006172,-0.006947,-0.007412,-0.008566,-0.005752,-0.006758
0,6,1.789551,2.325439,2.086182,2.113037,2.498779,2.485352,2.467041,-0.001414,0.001454,0.002377,0.001770,0.000315,-0.001755,-0.002251
0,7,2.111816,2.520752,2.133789,2.104492,2.453613,2.607422,2.508545,-0.003519,0.000267,0.002066,0.001858,0.000638,-0.002941,-0.002562
0,8,2.347412,2.681885,2.220459,2.142334,2.464600,2.692871,2.552490,-0.009305,-0.016406,-0.016053,-0.015444,-0.016098,-0.013413,-0.012653
0,9,2.601318,2.882080,2.401123,2.281494,2.592773,2.844238,2.587891,-0.015711,-0.040302,-0.043244,-0.042114,-0.042648,-0.028476,-0.028589
0,10,2.775879,3.015137,2.526855,2.393799,2.680664,2.947998,2.657471,-0.017480,-0.050018,-0.055101,-0.054519,-0.056123,-0.035583,-0.037444
0,11,2.814941,3.045654,2.520752,2.388916,2.675781,2.954102,2.629395,-0.012615,-0.030681,-0.033513,-0.034441,-0.038078,-0.025830,-0.028693
0,12,2.817383,3.024902,2.482910,2.352295,2.634277,2.940674,2.640381,-0.006680,0.004131,0.007582,0.005544,0.000369,-0.006931,-0.008427
0,13,2.862549,3.055420,2.512207,2.382812,2.659912,2.982178,2.650146,-0.009790,0.014826,0.023276,0.022119,0.017478,-0.001984,-0.000145
0,14,2.930908,3.105469,2.572021,2.437744,2.707520,3.041992,2.674561,-0.026491,-0.028842,-0.022961,-0.021256,-0.023469,-0.028912,-0.024676
0,15,2.927246,3.096924,2.548828,2.418213,2.694092,3.032227,2.668457,-0.047809,-0.107603,-0.112123,-0.107944,-0.108444,-0.076938,-0.074428
0,16,2.888184,3.050537,2.480469,2.359619,2.639160,2.988281,2.673340,-0.055962,-0.153499,-0.167463,-0.164192,-0.166799,-0.106755,-0.109905
0,17,2.899170,3.046875,2.489014,2.357178,2.645264,2.995605,2.675781,-0.041385,-0.107679,-0.118458,-0.119851,-0.128532,-0.083823,-0.091659
0,18,2.965088,3.112793,2.565918,2.431641,2.714844,3.057861,2.694092,-0.017194,0.008284,0.015752,0.009774,-0.004636,-0.019832,-0.025323
0,19,2.965088,3.111572,2.564697,2.429199,2.706299,3.051758,2.691650,-0.013866,0.082369,0.109537,0.104492,0.090450,0.020776,0.025341
0,20,2.902832,3.061523,2.484131,2.358398,2.637939,2.978516,2.662354,-0.052724,-0.001929,0.024930,0.027918,0.021521,-0.030376,-0.015927
0,21,2.899170,3.056641,2.485352,2.352295,2.629395,2.973633,2.640381,-0.118805,-0.238525,-0.240739,-0.228333,-0.226615,-0.171987,-0.159093
0,22,2.974212,3.120117,2.569580,2.424316,2.691650,3.039551,2.662354,-0.159792,-0.443977,-0.483565,-0.470312,-0.470648,-0.298939,-0.302322
0,23,3.001709,3.148193,2.589111,2.449951,2.706299,3.065186,2.701416,-0.118600,-0.349806,-0.390646,-0.390684,-0.407250,-0.251049,-0.272578
0,24,2.957764,3.112793,2.531738,2.395020,2.659912,3.018799,2.662354,0.023098,0.202658,0.235473,0.214313,0.178648,0.077944,0.061172

```

Figura 95: contenido del archivo *.csv generado. [Elaboración propia]

4.10 Experimentación e Implementación en PCB (Parte III).

Dado que se estará comparando el ritmo cardiaco de la persona que se esté evaluando se necesita algo con que compararlo, para comparar estos datos se muestra la siguiente tabla proporcionada por la *Unión de trabajadores de farmacia de Alicante* donde muestra el rango normal de ritmo cardiaco para las diferentes edades.

Tabla 5: Valores promedio de frecuencia cardiaca. [17]

FRECUENCIA CARDIACA		
Grupo	Edad	Latidos por minuto
RN	Nacimiento – 6 semanas	120-140
Infante	7 semanas - 1 año	100-130
Lactante mayor	1 – 2 años	100-120
Pre-escolar	2 – 6 años	80-120
Escolar	6 – 13 años	80-100
Adolescente	13 – 16 años	70-80
Adulto	16 años y más	60-80

La tabla anterior se utilizara para determinar si la persona a la que se mide sufre de arritmia.

Tabla 6: Conexiones entre DE1-SoC y PCB. [Elaboración propia]

Canal	Derivación
ADC1	AVL
ADC2	DI
ADC3	DII
ADC4	AVR
ADC5	DIII
ADC6	AVF

El detector de complejo QRS consiste de 5 pasos:

- 1. Eliminación de componentes de baja frecuencia (DC):** utilizando el filtro digital pasa bandas mencionado en el capítulo 4.2, con frecuencia inferior de 0.5 Hz y superior de 40 Hz. Y además de eso, se obtiene un promedio de 500 muestras para obtener el valor base de la señal ya filtrada, y restándole este valor a las siguientes 500 muestras y repitiendo el proceso, con esto se

logra que la onda se encuentre en la línea de 0V permitiendo el uso del siguiente paso

2. **Elevar al cuadrado la señal (dada la naturaleza de la onda R):** ya que la onda R es la única que fácilmente sobrepasara la línea de 1V todos los puntos por debajo de 1V al ser elevados al cuadrado se volverán más pequeños, y los que se encuentren arriba de 1V se volverán más grandes.
3. **Funcion detectora de picos:** se detectan obteniendo diferencias entre el punto anterior y el actual, se determina un umbral mínimo ($0.5V^2$) para que se considere una diferencia grande, ya que en una onda R se genera un pico que se encuentra mucho más alto que los demás puntos.
4. **Validación de datos:** teniendo los picos detectados, puede ser que detecte dos ondas R si la señal se encuentra muy distorsionada, para evitarlo se define un tiempo de 0.12s de duración de la onda R, en este tiempo no podrá existir otro pico detectado, solo el primer pico

Para calcular el ritmo cardiaco se almacena el tiempo en el que se detectan los picos, y se realiza una diferencia de tiempos y una división, como se muestra en la siguiente formula.

$$frecuencia = \frac{60}{t_{pico_nuevo} - t_{pico_anterior}}$$

4.11 Resultados

Figura 96: contenido del archivo *.csv generado. [Elaboración propia]

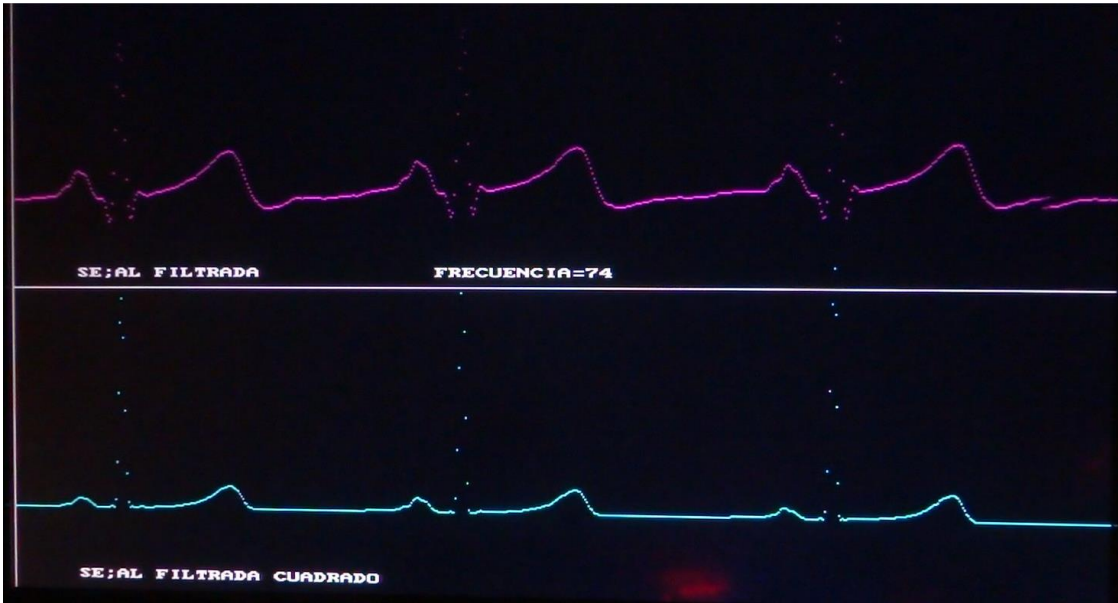


Figura 97: Grafica ADC3 (DII) filtrada (rosa) y elevada al cuadrado (cyan). [Elaboración propia]

La señal cyan se encuentra montada a 1.9V aproximadamente, ya que es el promedio obtenido al calcular la componente DC de la señal.

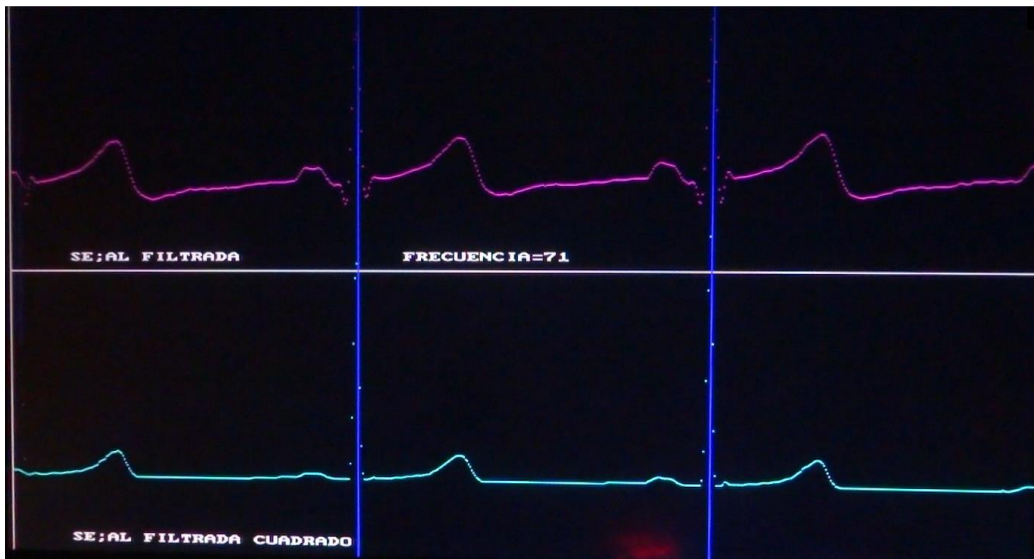


Figura 98: Grafica ADC3 (DII) filtrada (rosa) y elevada al cuadrado (cyan), pico detectado (azul). [Elaboración propia]

Como puede notarse en la figura anterior, se grafica una línea vertical en el punto más alto de la señal que corresponde a la onda R.

Ya detectados los picos se puede medir la frecuencia cardiaca, que en la imagen se puede ver que son 71 latidos/minuto.

Finalmente se grafica sobre una cuadrícula con la escala que se utiliza en un electrocardiograma que son, horizontal 25 mm/s y vertical 10 mm/mV . Se muestra la misma grafica en ambos colores desde la Figura 99: Gráfica de AVF en ambos colores. [Elaboración propia]Figura 99 hasta la Figura 104 por motivos de contraste, ya que se desea que se pueda diferenciar la cuadrícula de la gráfica. De los resultados se determinó que el mejor color es el verde.

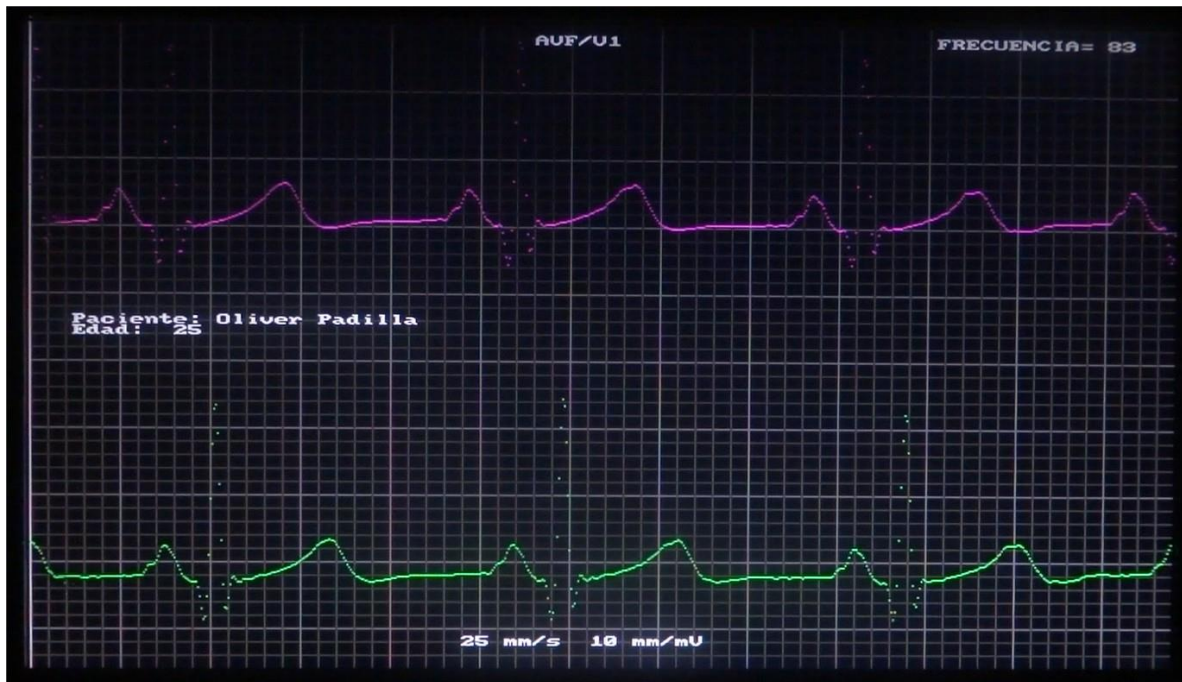


Figura 99: Gráfica de AVF en ambos colores. [Elaboración propia]

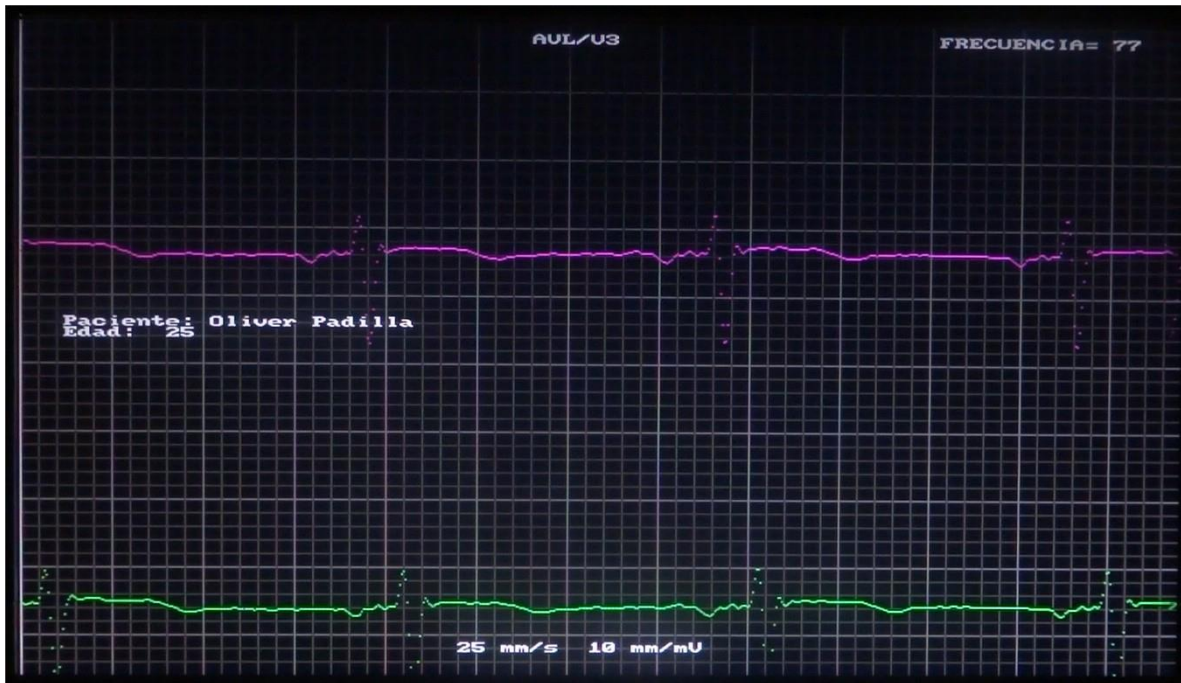


Figura 100: Gráfica de AVL en ambos colores. [Elaboración propia]

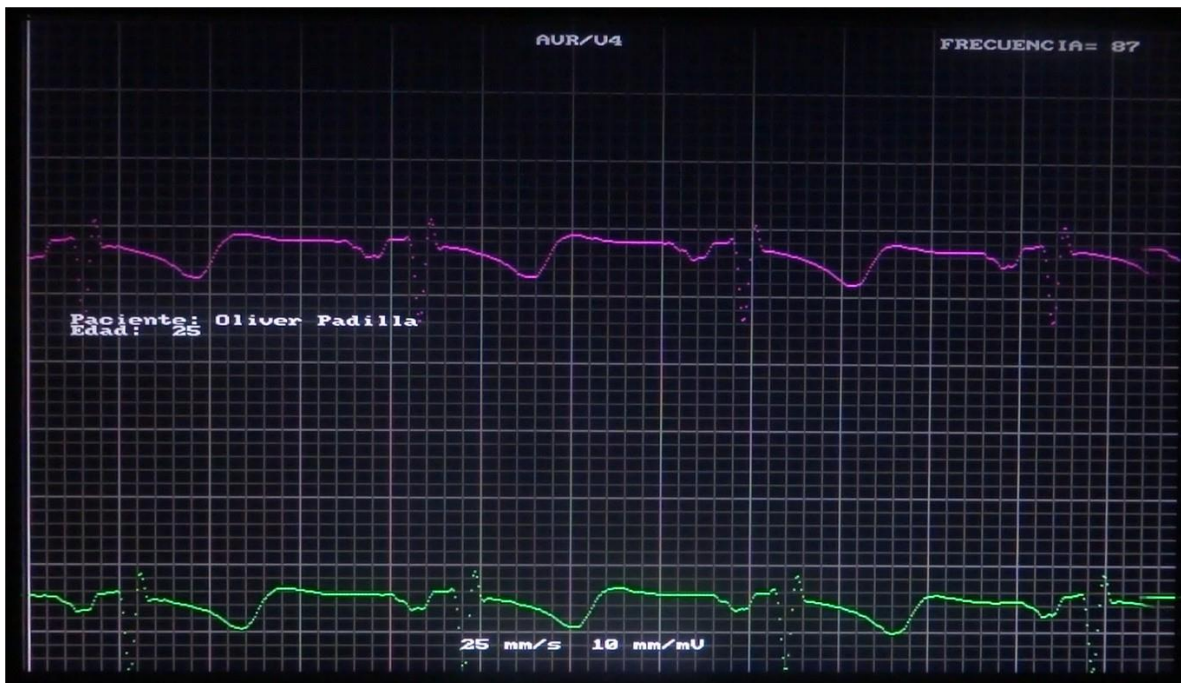


Figura 101: Gráfica de AVR en ambos colores. [Elaboración propia]

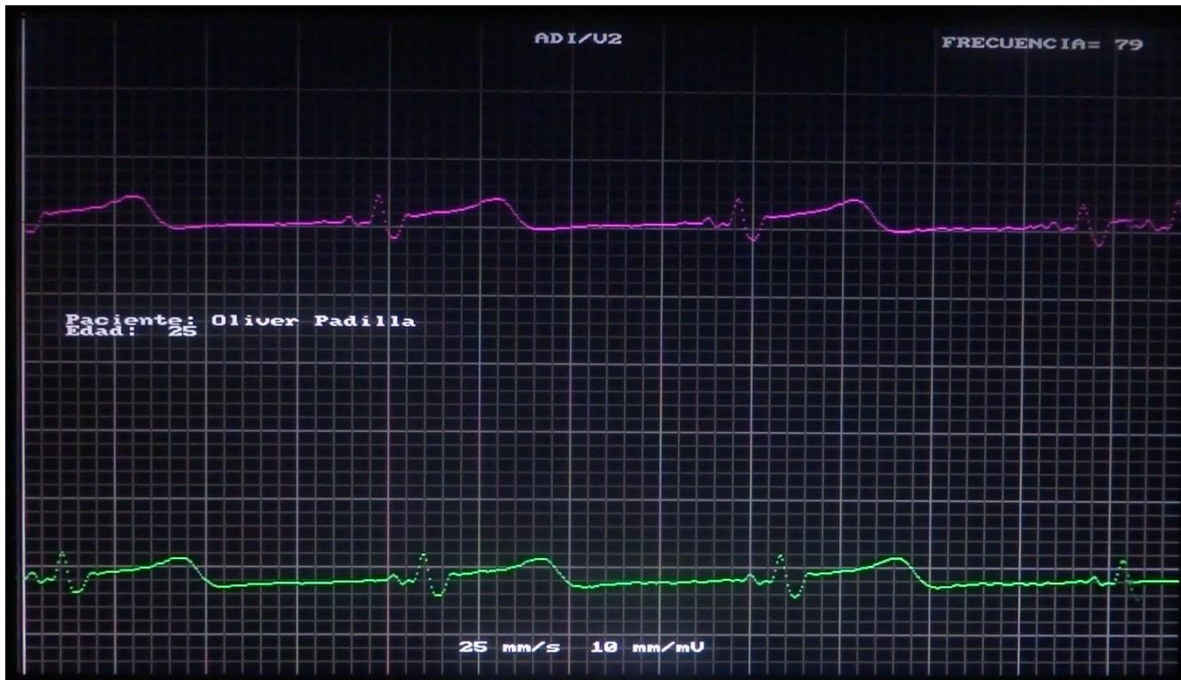


Figura 102: Gráfica de DI en ambos colores. [Elaboración propia]

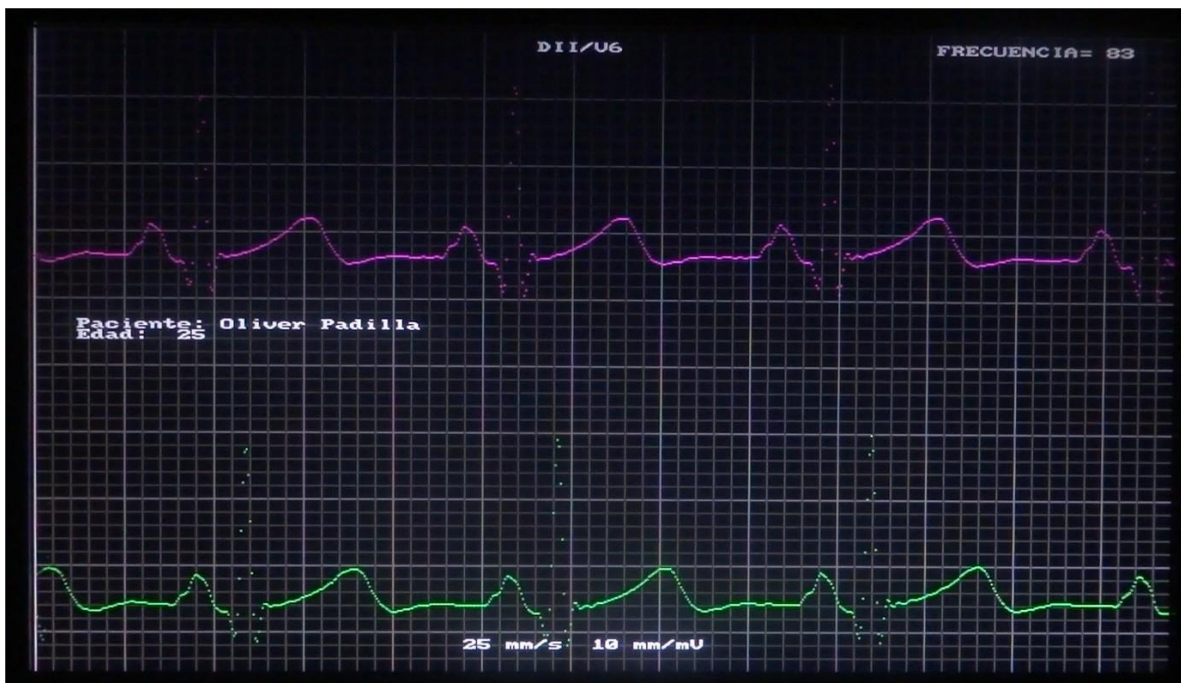


Figura 103: Gráfica de DII en ambos colores. [Elaboración propia]



Figura 104: Gráfica de DIII en ambos colores. [Elaboración propia]

Capítulo V: PRESUPUESTO

Tabla 7: Presupuesto. [Elaboración propia]

	ITEM	MATERIAL	CANTIDAD	P.U	SUBTOTAL
E L E C T R O C A R D I O G R A F O	1	DB15 joystick	1	\$ 0.70	\$ 0.70
	2	bases arduino de 16pines	2	\$ 0.50	\$ 1.00
	3	base2pin	1	\$ 0.25	\$ 0.25
	4	base3pin	1	\$ 0.30	\$ 0.30
	5	DIP 10	1	\$ 0.79	\$ 0.79
	6	47nF	12	\$ 0.45	\$ 5.40
	7	4.7nF	12	\$ 0.45	\$ 5.40
	8	0.1uF	32	\$ 0.45	\$ 14.40
	9	47pF	1	\$ 0.45	\$ 0.45
	10	1n4148	24	\$ 0.23	\$ 5.40
	11	AD7533	1	\$ 1.00	\$ 1.00
	12	7805	1	\$ 0.45	\$ 0.45
	13	lm741	1	\$ 0.53	\$ 0.53
	14	TL084	12	\$ 0.74	\$ 8.91
	15	AD620	12	\$ 7.95	\$ 95.42
	16	1k	34	\$ 0.19	\$ 6.38
	17	100k	34	\$ 0.19	\$ 6.38
	18	10k 1%	9	\$ 0.19	\$ 1.69
	19	390k	2	\$ 0.19	\$ 0.38
	20	5.6k	12	\$ 0.19	\$ 2.25
	21	2.2M	12	\$ 0.19	\$ 2.25
	22	4.7M	12	\$ 0.19	\$ 2.25
	23	560	12	\$ 0.19	\$ 2.25
	24	1M	24	\$ 0.19	\$ 4.50
	25	68k	12	\$ 0.19	\$ 2.25
	26	120k	12	\$ 0.19	\$ 2.25
	27	Tarjeta Cobre doble cara 15x20cm	1	\$ 2.00	\$ 2.00
F U E N T E	28	Transformador 120/24V 3A	1	\$ 9.00	\$ 9.00
	29	Switch 240/120 15A	1	\$ 0.75	\$ 0.75
	30	Fusible 120V/ 0.5A	1	\$ 1.10	\$ 1.10
	31	Porta Fusible	1	\$ 0.75	\$ 0.75
	32	IC 7812	1	\$ 0.45	\$ 0.45
	33	IC 7912	1	\$ 0.75	\$ 0.75
	34	C 1uF/50V	2	\$ 0.50	\$ 1.00
	35	C 4700uF/63V	2	\$ 0.75	\$ 1.50
	36	C 0.1uF/250V	2	\$ 0.45	\$ 0.90
	37	Puente Diodos 3A	1	\$ 1.00	\$ 1.00
	38	Tarjeta cobre una cara 11x5cm	1	\$ 1.00	\$ 1.00
	39	Conector cable 120V/15A	1	\$ 0.75	\$ 0.75
	40	Cable triplex	1	\$ 1.00	\$ 1.00
	41	Varios(brocas,estaño, pasta...)	1	\$ 32.00	\$ 32.00
	42	Electrodos Ecg	1	\$ 43.28	\$ 43.28
	43	Cable para Electrodo	1	\$ 43.29	\$ 43.29
	44	Electrodos desechables	1	\$ 29.43	\$ 29.43
μ	45	DE1-SoC	1	\$ 175.00	\$ 175.00
TOTAL					518.16

CONCLUSIONES

- Se logró implementar exitosamente un electrocardiógrafo utilizando la placa DE1-SoC con las siguientes características: resolución en pantalla de 640 x 480 pixeles, tiempo de graficado por cuadro 2.56s, escalas de cuadrícula de 25 mm/s en eje X y 10 mm/mV en eje Y (que concuerda con el estándar utilizado ampliamente por los aparatos comerciales), frecuencia de muestreo 250 muestras/s, los registros son guardados en un archivo extensión *.csv.
- La PCB diseñada con las entradas de los electrodos: LA, RA, LL, RL, V1, V2, V3, V4, V5 y V6, provee datos de las doce derivaciones: DI, DII, DIII, aVL, aVF, aVR, V1, V2, V3, V4, V5 y V6. A la salida de la PCB se obtiene una señal con una ganancia de 1V/mV que corresponde al estándar de ganancia utilizado en la mayoría de electrocardiógrafos.
- Para la detección del complejo QRS se hizo uso de herramientas como: una etapa de filtrado digital (filtro paso banda orden 50) con las frecuencias 0.5 Hz a 40 Hz para eliminar el ruido proveniente de la red y de la conversión analógica-digital, luego se aplicaron funciones donde se elimina la componente DC de la señal, se eleva al cuadrado la señal sin offset, se determina la ubicación de los picos con su marca de tiempo y finalmente se calcula el ritmo cardiaco.
- Se utilizó la memoria SDRAM como buffer de video ya se manejó una resolución de 640 x 480 pixeles siendo necesarias 307,200 direcciones las cuales no pueden ser contenidas en On-Chip RAM ya que solo posee espacio para 262,144 direcciones, y la SDRAM como posee 64 MB es más que suficiente para almacenar los pixeles a esta resolución.
- Con el objetivo de optimizar el código y este fuese lo más eficiente en tiempo de ejecución, se evitó llamar rutinas para controlar el puerto VGA, ya que al hacer esto, se debían almacenar los datos de los pixeles en una matriz y luego esta matriz debe ser leída desde la función que controla el puerto VGA lo cual requiere de tiempo que se traduce en una disminución o bloqueo al momento de determinar la frecuencia de muestreo.

RECOMENDACIONES

- Es de suma importancia contar con una buena puesta a tierra, ya que sin esta se obtendría mucho ruido en la toma de datos y sería muy difícil quitar ese ruido utilizando filtrado digital. Se recomienda en caso que el dispositivo sea portátil, cargar con una pequeña varilla de tierra. Se asume que un hospital cuenta con una buena puesta a tierra.
- Implementar un circuito de multiplexado, para poder elegir la fuente de señal a medir en el ECG, ya sean las derivaciones precordiales o de los miembros, siendo este controlado con la FPGA activando o desactivando un pin de salida de la placa ya sea del JP1 o JP2.
- Si se desea obtener las 12 derivaciones a la vez, implementar otro ADC en la placa, añadiendo un ADC a la interfaz Qsys y seleccionando pines del puerto paralelo JP1 o JP2, ya que para el desarrollo de este trabajo se encuentran en desuso.

BIBLIOGRAFÍA

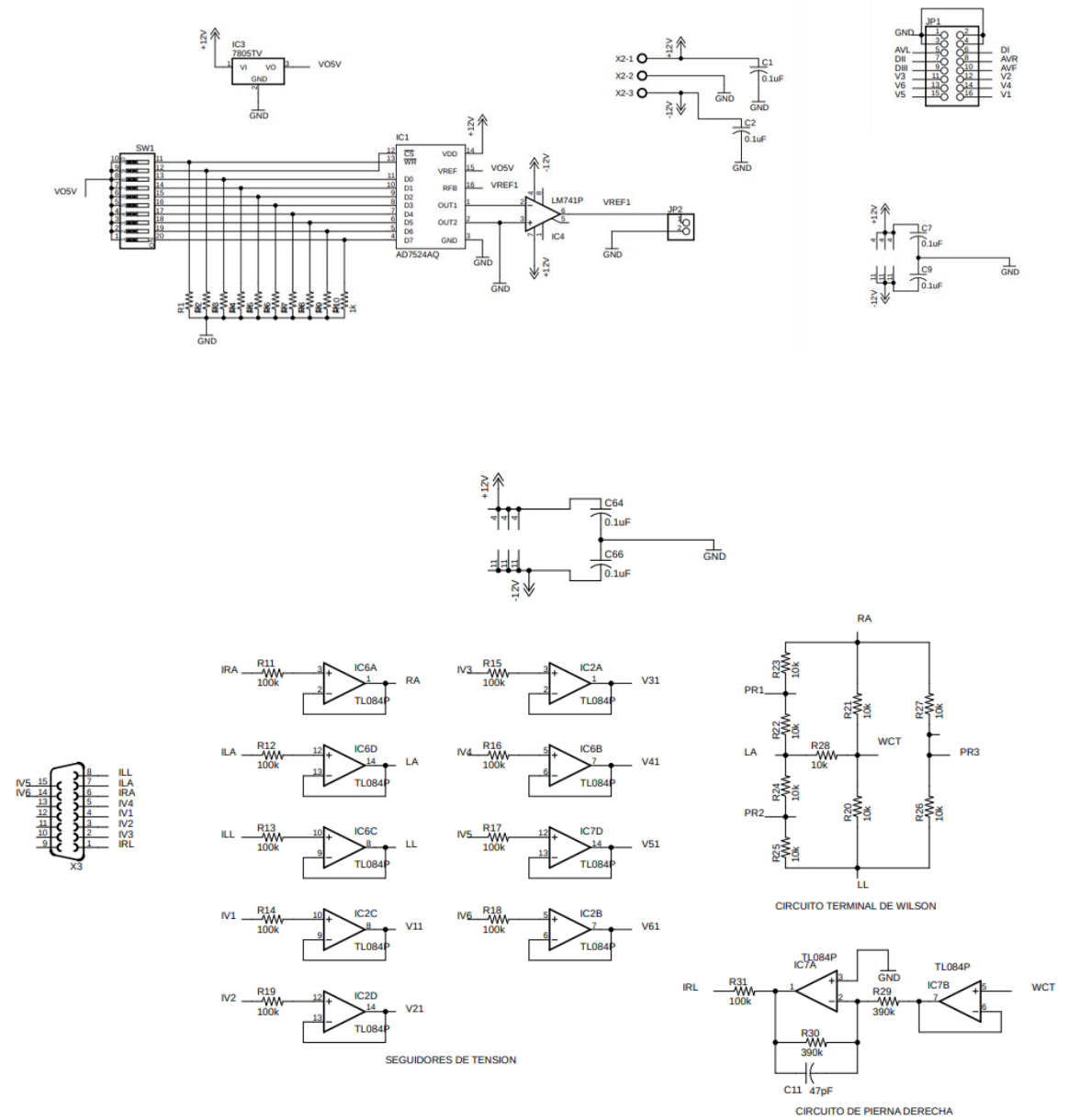
- [1] «Principales causas de mortalidad ocurridas en la Red de Hospitales del MINSAL,» [En línea]. Available: <https://www.transparencia.gob.sv/institutions/minsal/documents/377341/download>. [Último acceso: 10 08 2020].
- [2] Portal de Transparencia MINSAL, «Portal de Transparencia MINSAL,» 10 08 2020. [En línea]. Available: https://www.transparencia.gob.sv/institutions/minsal/documents/otra-informacion-de-interes?utf8=%E2%9C%93&q%5Bname_or_description_cont%5D=&q%5Byear_cont%5D=2020&button=&q%5Bdocument_category_id_eq%5D=.
- [3] Texas Heart Institute, «Texas Heart Institute,» [En línea]. Available: <https://www.texasheart.org/heart-health/heart-information-center/topics/anatomia-del-corazon/>. [Último acceso: 10 08 20].
- [4] D. Dubin, Dubin: Interpretacion de ECG, Florida: COVEE Publishing Company, 2011.
- [5] BirthLH, «BirthLH,» [En línea]. Available: https://ikastaroak.birt.eus/edu/argitalpen/backupa/20200331/1920k/es/EME/APB/APB04/es_EME_APB04_Contentidos/website_115_el_sistema_elctrico_d_el_corazn.html. [Último acceso: 8 4 2021].
- [6] My EKG, «My EKG,» [En línea]. Available: <https://www.my-ekg.com/generalidades-ekg/papel-ekg.html>. [Último acceso: 5 4 20].
- [7] My EKG, «My EKG,» [En línea]. Available: <https://www.my-ekg.com/generalidades-ekg/derivaciones-cardiacas.html>. [Último acceso: 8 1 2021].
- [8] Codigos tips y programas varios, «Codigos tips y programas varios,» [En línea]. Available: <https://blog.hackerspace.sv/2013/04/el-convertidor-analogo-digital-en.html>. [Último acceso: 23 07 2021].

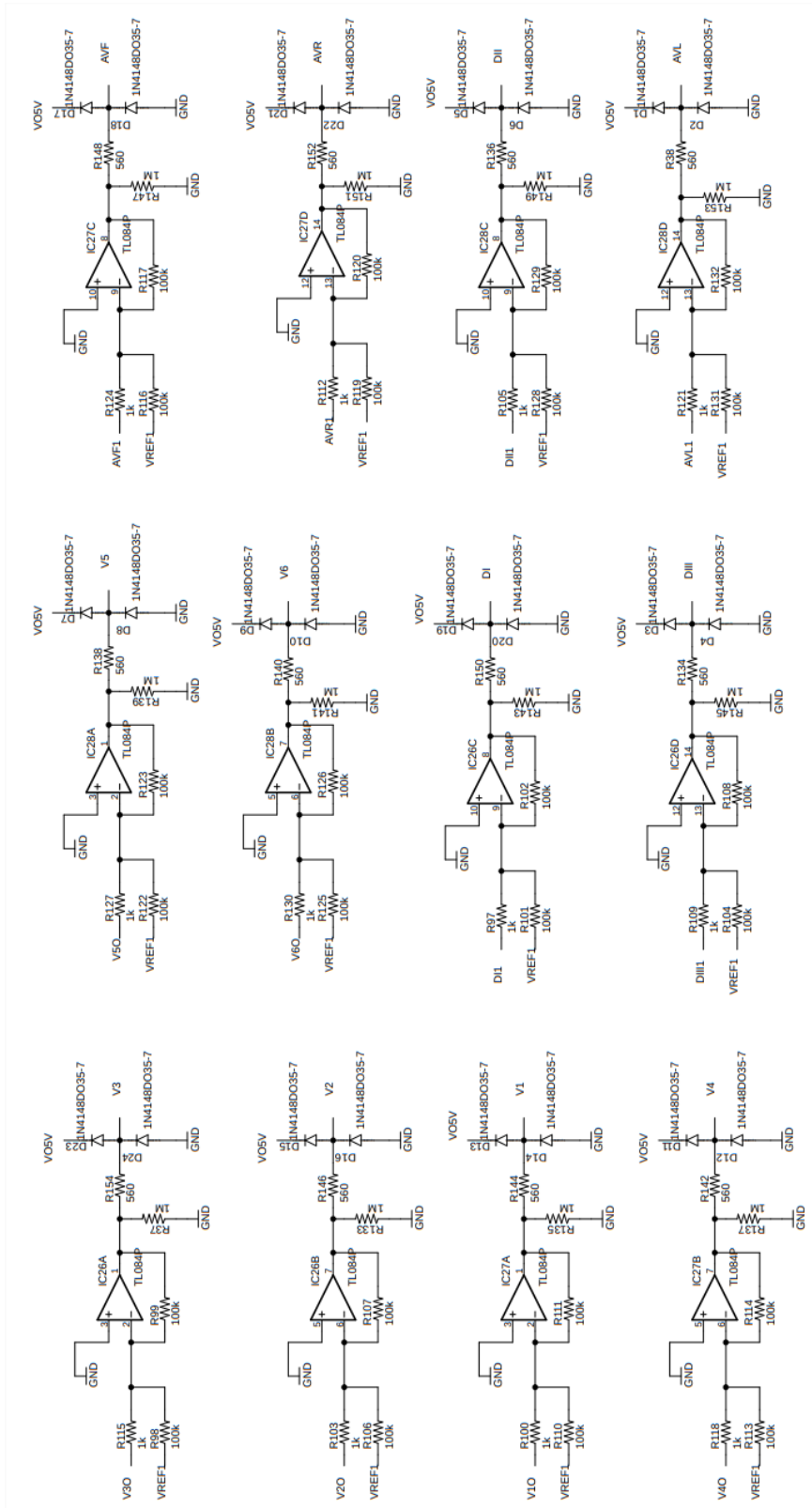
- [9] A. V. Oppenheim, «Tratamiento de señales en tiempo discreto,» de *Tratamiento de señales en tiempo discreto*, Madrid, Pearson Educación S.A, 2011, p. 1108.
- [10] C. s. videorockola, «Construya su videorockola,» [En línea]. Available: <http://www.videorockola.com/proyectos-electronicos/fuentes/construya-una-fuente-simetrica-regulada-2/>. [Último acceso: 25 10 2020].
- [11] Analog Devices, «AD620N datasheet,» [En línea]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/AD620.pdf>. [Último acceso: 10 03 2020].
- [12] A. Devices, «AD7533 datasheet,» [En línea]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/AD7533.pdf>. [Último acceso: 20 3 2020].
- [13] «Manual DE1-SoC rev E,» [En línea]. Available: <https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=836>. [Último acceso: 15 04 2020].
- [14] Altera, «DE1-SoC Computer System witch ARM* Cortex* A9,» [En línea]. Available: <https://software.intel.com/content/www/us/en/develop/topics/fpga-academic/materials-tutorials.html>. [Último acceso: 15 04 2020].
- [15] Cornell ece5760, «DE1-SoC: ARM HPS Linux,» [En línea]. Available: https://people.ece.cornell.edu/land/courses/ece5760/DE1_SOC/HPS_peripherals/linux_index.html. [Último acceso: 10 09 19].
- [16] asistenciasanitaria.com.ar, «Enfermeria-Nursing,» [En línea]. Available: <https://asistenciasanitaria.com.ar/2018/08/29/como-hacer-un-electrocardiograma-y-nociones-basicas-sobre-como-interpretarlo/>. [Último acceso: 5 04 2021].
- [17] W. J. T. Ed, Biomedical Digital Signal Processing, Prentice Hall, 1993.

- [18] U. d. T. d. Farmacia, «Union de Trabajadores de Farmacia,» [En línea]. Available: <https://www.utfalicante.com/blog/tabla-de-signos-vitales-por-edad-44.html>. [Último acceso: 20 05 2021].
- [19] J. E. González-Barajas, «Cálculo del umbral para detección de la onda R del complejo cardiaco,» Universidad Santo Tomas, Bogotá-Colombia, 2013.
- [20] Altera, «Video IP Cores for Altera DE-Series Boards,» [En línea]. Available: <https://software.intel.com/content/www/us/en/develop/topics/fpga-academic/materials-tutorials.html>. [Último acceso: 16 04 2020].
- [21] Altera, «Introduction to the Altera Qsys System Integration Tool,» [En línea]. Available: <https://software.intel.com/content/www/us/en/develop/topics/fpga-academic/materials-tutorials.html>. [Último acceso: 19 04 2020].
- [22] Altera, «Making Qsys Componets,» [En línea]. Available: <https://software.intel.com/content/www/us/en/develop/topics/fpga-academic/materials-tutorials.html>. [Último acceso: 20 04 2020].
- [23] «DE1-SoC course,» [En línea]. Available: <http://people.ece.cornell.edu/land/courses/ece5760/>. [Último acceso: 10 05 2020].
- [24] Altera, «HPS Cyclone V,» [En línea]. Available: <https://www.intel.com/content/www/us/en/programmable/hps/cyclone-v/hps.html>. [Último acceso: 10 03 2019].
- [25] L. J. M. V. F. Walter Mata, «Prototipo para la adquisición y procesamiento de bioseñales cardiacas con tecnología inalámbrica,» Mexico, 2016.
- [26] A. E. M. RODRÍGUEZ, «Desarrollo de un dispositivo confiable y de bajo costo para,» 2016.
- [27] B. A. R. RIVAS, «Principios del FPGA y aplicaciones en el control de,» 2016.
- [28] J. R. M. N. y. G. A. M. N. HENRÍQUEZ, «Diseño de un controlador PID, utilizando una tarjeta,» 2016.

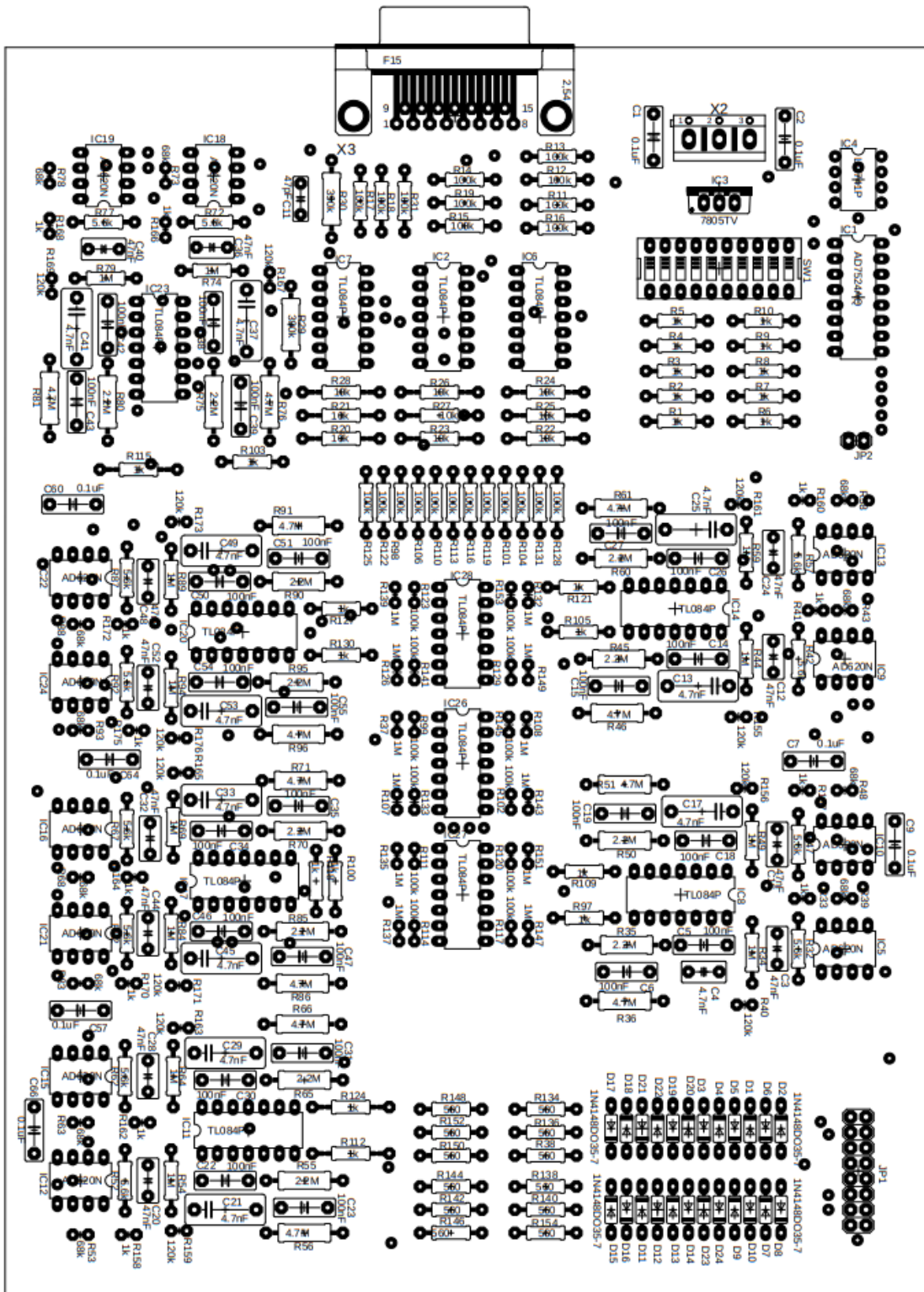
ANEXOS.

Anexo A: Esquemático completo.

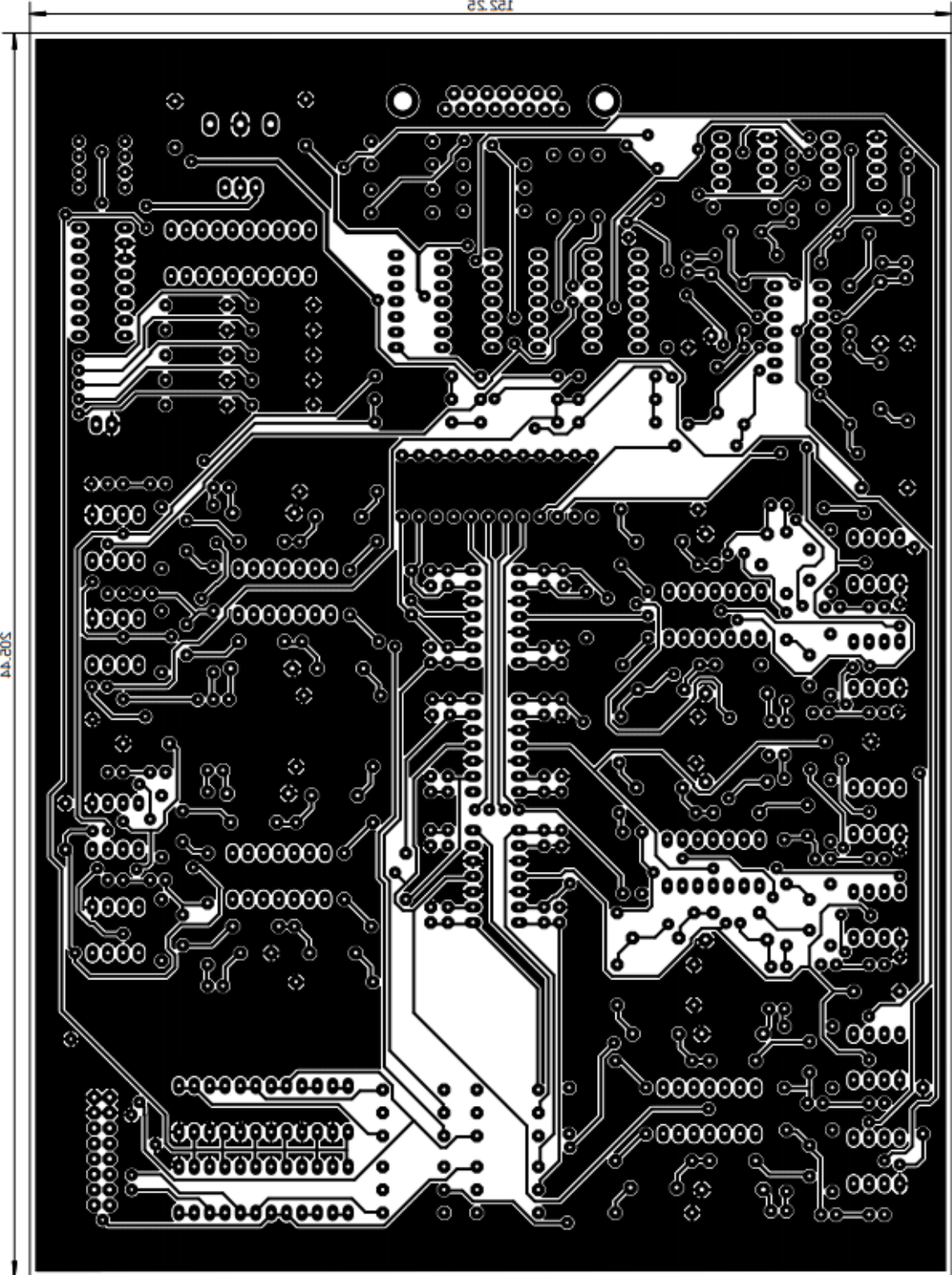




Anexo B: Mascara de componentes.



Anexo C: PCB Top



Anexo D: PCB Bottom

