ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO ELECTRÓNICO EN INSTRUMENTACIÓN

TEMA: "DISEÑO Y CONSTRUCCIÓN DE UN CALIBRADOR-DOCUMENTADOR DE PROCESOS UTILIZANDO MICROCONTROLADORES Y SOFTWARE LIBRE PARA LA CALIBRACIÓN DE INSTRUMENTOS DE TEMPERATURA EN EL LABORATORIO DE REDES INDUSTRIALES Y CONTROL DE PROCESOS DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE EXTENSIÓN LATACUNGA"

AUTORES: GERMÁN ISRAEL ONCE SUCUZHAÑAY
JONATHAN JAVIER RIVERA COMINA

DIRECTOR: ING. EDWIN PRUNA

LATACUNGA

2017

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

CERTIFICACIÓN

Certifico que el trabajo de titulación "**DISEÑO Y CONSTRUCCIÓN DE UN CALIBRADOR-DOCUMENTADOR DE PROCESOS UTILIZANDO MICROCONTROLADORES Y SOFTWARE LIBRE PARA LA CALIBRACIÓN DE INSTRUMENTOS DE TEMPERATURA EN EL LABORATORIO DE REDES INDUSTRIALES Y CONTROL DE PROCESOS DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE EXTENSIÓN LATACUNGA**" realizado por el Sr. **GERMÁN ISRAEL ONCE SUCUZHAÑAY** y el Sr**. JONATHAN JAVIER RIVERA COMINA**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar a **GERMÁN ISRAEL ONCE SUCUZHAÑAY** y **JONATHAN JAVIER RIVERA COMINA** para que lo sustenten públicamente.

Latacunga, 14 de marzo de 2017

Ing. Edwin Pruna

DIRECTOR

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

AUTORÍA DE RESPONSABILIDAD

Nosotros, **GERMÁN ISRAEL ONCE SUCUZHAÑAY** y**. JONATHAN JAVIER RIVERA COMINA**, con cédula de identidad N° 0104069273 y N° 0503182016, respectivamente, declaramos que este trabajo de titulación "**DISEÑO Y CONSTRUCCIÓN DE UN CALIBRADOR-DOCUMENTADOR DE PROCESOS UTILIZANDO MICROCONTROLADORES Y SOFTWARE LIBRE PARA LA CALIBRACIÓN DE INSTRUMENTOS DE TEMPERATURA EN EL LABORATORIO DE REDES INDUSTRIALES Y CONTROL DE PROCESOS DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE EXTENSIÓN LATACUNGA**", ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance de la investigación mencionada.

Latacunga, 14 de marzo de 2017

Germán Israel Once Sucuzhañay
C.C.: 0104069273

Jonathan Javier Rivera Comina
C.C.: 0503182016

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

AUTORIZACIÓN

Nosotros, **GERMÁN ISRAEL ONCE SUCUZHAÑAY** y **JONATHAN JAVIER RIVERA COMINA** autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la Institución el presente trabajo de titulación "**DISEÑO Y CONSTRUCCIÓN DE UN CALIBRADOR-DOCUMENTADOR DE PROCESOS UTILIZANDO MICROCONTROLADORES Y SOFTWARE LIBRE PARA LA CALIBRACIÓN DE INSTRUMENTOS DE TEMPERATURA EN EL LABORATORIO DE REDES INDUSTRIALES Y CONTROL DE PROCESOS DE LA UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE EXTENSIÓN LATACUNGA**", cuyo contenido, ideas y criterios son de nuestra autoría y responsabilidad.

Latacunga, 14 de marzo de 2017

Germán Israel Once Sucuzhañay

C.C.: 0104069273

Jonathan Javier Rivera Comina

C.C.: 0503182016

DEDICATORIA

Dedico esta tesis de manera especial al Creador de mi vida, DIOS, que puso en mí la semilla del amor, paciencia y perseverancia.

A mis madres: AIDA, por su coraje, amor y ejemplo en mi caminar, por nunca rendirse.

BEATRIZ, que desde el cielo me dio su bendición, con sus palabras y consejos marco mi vida.

Las amo y admiro infinitamente por ser esas mujeres de guerra y lucha.

A ustedes mi motivación, esfuerzo y felicidad.

"Mira que te mando que te esfuerces y seas valiente; no temas ni desmayes, porque Jehová tu Dios esta contigo a donde quiera que vayas"

ISRAEL

Este trabajo está dedicado a mi familia, a mi madre Patricia, a mi padre Iván y a mis abuelito Victor y Rosario quienes han estado junto a mi en mis días de claridad y oscuridad y a pesar de todo nunca dejaron de creer en mí me mostraron que el esfuerzo tarde o temprano entregará su recompensa y mi hermana Lisbeth quien pronto empezará este camino.

"Jamás te detengas hasta que consigas tus objetivos."

JONATHAN

AGRADECIMIENTO

Agradezco con todo mi corazón a Dios por brindarme la oportunidad de llegar a la meta y cumplir uno de muchos logros en mi vida.

Le doy gracias infinitas a toda mi familia por su apoyo incondicional en mi caminar, a mis padres German y Aida porque de una u otro manera hicieron el ser que hoy existe.

A mami Chis por haberme dado su amor y fortaleza, desde el cielo recibo su bendición.

A mis hermanos Maribel, Tania, Doris, Eulalia y Andrés por compartir nuestros caminos juntos y cada uno poner su granote de arena para alcanzar mis metas.

A Juan Carlos y Byron por ser esas personas incondicionales que siempre me brindaron de su ayuda, como otros hermanos más.

A todas esas personas que fueron apareciendo y formando parte de vida en todo este transcurso de tiempo.

A la Institución, maestros y compañeros por brindarme sus conocimientos permitiéndome conocer a grandes personas compartiendo alegrías y penas, de manera especial a mi tutor Ing. Edwin Pruna por su paciencia entregada.

No a sido fácil el camino, pero gracias a todos ustedes por sus aportes, amor y apoyo.

ISRAEL

Agradeaco a DIOS y al Niñito de Isinche por darme siempre la fuerza para nunca rendirme.

A la Universidad de las Fuerzas Armadas ESPE extensión Latacunga por permitirme cultivar mis conocimientos tanto morales como académicos.

Al Ing. Edwin Pruna por su apoyo incondicional por depositar su confianza en mí y en este proyecto y a todos los ingenieros de la carrera de Electrónica e Instrumentación, quienes me brindaron sus amplios conocimientos tanto éticos como académicos.

JONATHAN

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El trabajo de titulación consiste en la construcción de un calibrador-documentador de temperatura, basada en las características de un instrumento comercial. Este calibrador será capaz de generar, simular y medir parámetros físicos y eléctricos como voltaje, corriente, resistencia, termopares y RTD´s, así como también se podrá realizar la alimentación de lazo o bucle, La interacción entre el usuario y el equipo se realiza mediante una pantalla touch screen de fácil lectura y uso, la cual permite ver simultáneamente los valores de entrada y de salida de las variables físicas, además realiza la comunicación con un software para la documentación y procesamiento de los datos obtenidos en una calibración de un instrumento de temperatura y a la vez genera un documento digital con los resultados finales. Para lo descrito anteriormente se emplea un software libre: se utiliza como gestor principal del sistema una placa electrónica Arduino de altas prestaciones; y para la generación, simulación, y medición de las variables eléctricas, dispositivos o elementos electrónicos compatibles para esta plataforma. Para la comprobación se utiliza un calibrador e instrumentos de temperatura como transmisores, termopares y RTD´s existentes en el Laboratorio de Redes Industriales y Control de Procesos.

PALABRAS CLAVE:

- **CALIBRADOR DE TEMPERATURA**
- **SOFTWARE PYTHON**
- **TECNOLOGÍA ARDUINO**
- **LABORATORIO DE CONTROL DE PROCESOS**

ABSTRACT

The titration work consists of the construction of a calibrator-temperature documenter, based on the characteristics of a commercial instrument. This calibrator will be able to generate, simulate and measure physical and electrical parameters such as voltage, current, resistance, thermocouples and RTDs, as well as loop or loop feed. The interaction between the user and the equipment is performed Through a touch screen screen that is easy to read and use, which allows simultaneous viewing of the input and output values of the physical variables, and also communicates with software for the documentation and processing of the data obtained in a calibration of an instrument Of temperature and at the same time generates a digital document with the final results. As described above, free software is used: a high performance Arduino electronic board is used as the main system manager; And for the generation, simulation, and measurement of the electrical variables, devices or electronic elements compatible for this platform. For the test, a calibrator and temperature instruments such as transmitters, thermocouples and RTDs are used in the Laboratory of Industrial Networks and Process Control.

KEYWORDS:

- **TEMPERATURE CALIBRATOR**
- **PYTHON SOFTWARE**
- **ARDUINO TECHNOLOGY**
- **PROCESS CONTROL LABORATORY**

CAPÍTULO I

1. PROBLEMÁTICA

1.1 Antecedentes

La calibración de instrumentos se ha desarrollado desde muchos años atrás para comparar las unidades fundamentales de medida del instrumento con otro instrumento patrón, basada en la medición de los valores de parámetros físicos y eléctricos, ésta comparación de instrumentos permite determinar un error, el cual se verifica si está dentro de la tolerancia permitida de funcionamiento; además se obtiene diagnóstico, verificación y documentación del trabajo del instrumento de procesos. Los resultados de una calibración se compactan en un documento digital, que permite tomar decisiones al técnico sobre el estado de funcionamiento del equipo calibrado, ya sea para realizar un ajuste o remplazarlo.

En las últimas décadas la evolución de la tecnología aplicada a la industria ha generado la necesidad de obtener un producto de alta calidad optimizando recursos materiales y humanos, debido a este problema surge la necesidad de poseer un control exacto de los instrumentos que intervienen en los lazos de control, utilizados según su función, dicho control se lo realiza de forma periódica, considerando las responsabilidades inherentes que existen por los posibles riesgos que puedan presentarse a las personas. Para realizar este proceso se utilizan instrumentos patrón calibrados y regulados por un centro metrológico a cargo de un grupo de técnicos que tienen la responsabilidad de mantener en óptimas condiciones dichos dispositivos de calibración y emitir un certificado de calibración con los datos más relevantes de los resultados obtenidos y condiciones a la que fue sometido.

1.2 Planteamiento del problema

El laboratorio de redes industriales y control de procesos, no posee un equipo capaz de cumplir las funciones de un calibrador-documentador realizado en hardware y software libre para la conexión con la PC. En el laboratorio existe una gran variedad de calibradores cada uno de ellos con diferentes características, funcionalidades y prestaciones lo que hace que el precio de dicho equipo varié, estos equipos al pertenecer a marcas mundiales tienen un precio alto, en la actualidad estos equipos son indispensables en las industrias, dado que es una herramienta fiable y resistente, ideal para la calibración y el mantenimiento de instrumentos, así como para solucionar problemas relacionados con éstos, para la documentación, captura de datos, procedimiento de calibración cumpliendo normas rigurosas como las ISO 9000, FDA, EPA y OSHA, el software que utilizan los calibradores comerciales tienen un alto costo.

1.3 Justificación e importancia

En la actualidad la tendencia es el uso de hardware y software libre por las características que posee, ya que se encuentra disponible a bajos costos o gratis, esto hace que se pueda modificar sin ningún límite, además la libertad de estudiarlo y adaptarlo, distribuir copias, mejorar y el uso para cualquier propósito.

La construcción del calibrador-documentador de temperatura usando hardware y software libre ayudará de manera positiva al aprendizaje del estudiante, además permitirá que el estudiante pueda calibrar instrumentos de temperatura y documentar los resultados obtenidos, con los cuales tendrá un criterio profesional, si dicho elemento se encuentra en óptimas condiciones de funcionamiento o si es necesario el ajuste del mismo, lo que normalmente se realiza en una industria, fomentando una formación global del futuro profesional.

Este proyecto permitirá que el estudiante pueda realizar prácticas de calibración, medición, generación y simulación de parámetros eléctricos y físicos y a la vez determinar la importancia del uso de un calibrador-documentador de temperatura en la industria, dado que la variable temperatura interviene en muchos procesos industriales, la cual es ampliamente usada y estudiada en carreras técnicas, además el laboratorio de control de procesos y redes industriales tiene a disposición transmisores y sensores de temperatura, lo cual facilita el uso que el estudiante pueda dar a este equipo.

Además de proporcionar conocimientos de calibración y ajuste de instrumentos de temperatura, se aportará conocimientos en temas como, el lenguaje de programación de alto nivel Processing de la plataforma Arduino la cual es similar a C++, valores de las señales estándares de los transmisores, comportamiento y simulación de termopares y RTD´s, utilización de elementos, módulos o dispositivos electrónicos para la generación y medición de corriente, tensión, resistencia, el uso de transmisores industriales, las conexiones respectivas y el lenguaje de programación del software libre. Todos estos temas son de vital importancia y gran utilidad, ayudando a cumplir con el perfil del estudiante de carrera de ingeniería Electrónica en Instrumentación.

Cabe mencionar que no hay proyectos que abarquen todas las características de un calibrador de temperatura realizado el hardware y software libre.

A partir de lo descrito anteriormente, la innovación, y el aporte que se brindará al estudiante, se observa la necesidad del diseño e implementación de un calibrador-documentador de procesos (temperatura), contribuyendo de esta manera a la parte práctica que es fundamental para lograr una educación integral.

1.4 Objetivos

1.4.1 Objetivo General

- Diseñar y construir un calibrador-documentador de procesos utilizando microcontroladores y software libre para la calibración de instrumentos de temperatura en el Laboratorio de Redes Industriales y Control de Procesos de la Universidad de las Fuerzas Armadas-ESPE extensión Latacunga.

1.4.2 Objetivos Específicos

- Obtener conocimientos acerca de la utilización, configuración y programación de la plataforma Arduino, los elementos o módulos y software libre compatible con este hardware.
- Implementar circuitos para medir tensión, corriente, resistencia eléctrica, RTD y termopares.
- Implementar circuitos para generar tensión, corriente, resistencia eléctrica y alimentación al bucle.
- Implementar circuitos para simular RTD y termopares.
- Desarrollar el programa que permita controlar todas las variables medidas, generadas y simuladas en la plataforma Arduino del microcontrolador.
- Diseñar un HMI para la visualización y control de las variables medidas, generadas y simuladas por medio de la pantalla táctil.
- Realizar las pruebas respectivas para verificar el correcto funcionamiento del calibrador-documentador, tomando como patrón un calibrador de temperatura.
- Realizar la comunicación entre el hardware y software para obtener un documento con los datos de la calibración.
- Realizar un manual de uso.

1.5 Hipótesis

¿El diseño y construcción de un calibrador-documentador de procesos utilizando microcontroladores y software libre permitirá la calibración

eficiente de instrumentos de temperatura, proporcionando la respectiva documentación?

CAPÍTULO II

2. MARCO CONTEXTUAL Y TEÓRICO

2.1 Marco Teórico

2.1.1 Variable temperatura

Esta variable es considerada como una de las de mayor importancia en los procesos industriales siendo una magnitud o variable física que representa la cantidad de calor expresada en grados centígrados, Fahrenheit, o Kelvin que posee un cuerpo, la cual es medida para controlar un proceso (Ramos, 2014).

a. Instrumentos de medición de temperatura

La medición de temperatura constituye una de las más comunes e importantes que se realiza en los procesos industriales. Existen diferentes sensores de temperatura utilizados en la industria en los cuales es necesario medir la temperatura. La selección apropiada de un instrumento de medición de temperatura depende del tipo de sensores disponibles, de sus limitaciones, de sus consideraciones prácticas y de su aplicación.

b. Sensores de temperatura

Los sensores de temperatura son dispositivos capaces de transformar los cambios de temperatura en cambios en señales eléctricas mismas que son tratadas en futuras etapas de acondicionamiento.

Existen principalmente tres tipos de sensores de temperatura, los termistores, los RTD y los termopares. Un sensor de temperatura, comúnmente está formado por un elemento primario, este puede ser:

termistor, RTD o termopar, la vaina que lo envuelve y que está rellena de un material muy conductor de temperatura, para que los cambios se transmitan rápidamente al elemento sensor y del cable al que se conectarán el equipo electrónico.

Termistor

El termistor está basado en el comportamiento que presenta la resistencia de los semiconductores, la cual es variable en función de la temperatura **Figura 1.**

Existen los termistores tipo NTC en los cuales al existir un aumento de temperatura, su valor de resistencia disminuye y los termistores tipo PTC, en los cuales, al aumentar el valor de temperatura también lo hace su valor de resistencia.

Sin embargo ya que los termistores no son lineales según la temperatura es necesario aplicar fórmulas complejas para determinar el valor de temperatura tomando en cuenta la corriente que circula, además calibrarlos es difícil por las razones antes mencionadas.

Figura 1: Termistor tipo Ntc y Ptc

Fuente: (Ruben, 2011)

Un termistor posee las siguientes características:

- El rango de temperatura de trabajo se encuentra entre -50ºC y 150ºC, aunque las unidades encapsuladas pueden alcanzar hasta los 300ºC.
- En la mayoría de las aplicaciones para una temperatura de 25ºC la resistencia varía entre 100 ohmios y 100Kohmios.
- Tiene un tamaño reducido lo que le permite reaccionar de manera rápida a los cambios de temperatura (poseen una mayor sensibilidad a variaciones de temperatura que otro sensor primario).
- Posee la propiedad de intercambiabilidad es decir la tolerancia con la que es producido un termistor, Lo que cambiar un termistor por otro sin la necesidad de calibrar nuevamente el aparato de medida.

Los termistores poseen varios tipos de configuraciones; que son tipo: perla, disco, chip, arandela y barra. Los termistores tipo perla con cubierta de cristal se caracterizan por tener una excelente estabilidad y fiabilidad en rangos de temperatura superiores a 300ºC. Los termistores tipo chip y disco tienen un tamaño mayor que los de perla, por ende poseen una potencia de disipación mayor, sin embargo esto afecta a sus tiempos de respuesta. Por su geometría, los termistores de disco suelen tener más potencia de disipación.

Termopar

Un termopar es un dispositivo usado en las mediciones de temperatura, se basa en efectos termoeléctricos, siendo uno de los medidores más importantes y más utilizados debido a su buen rendimiento, ya que se usan tanto para temperaturas bajas y altas. Los termopares se utilizan principalmente cuando es necesario enviar la información de la medida a sitios remotos. Constan principalmente de dos metales distintos, unidos en un extremo (Guerrero, 2002), **figura 2.**

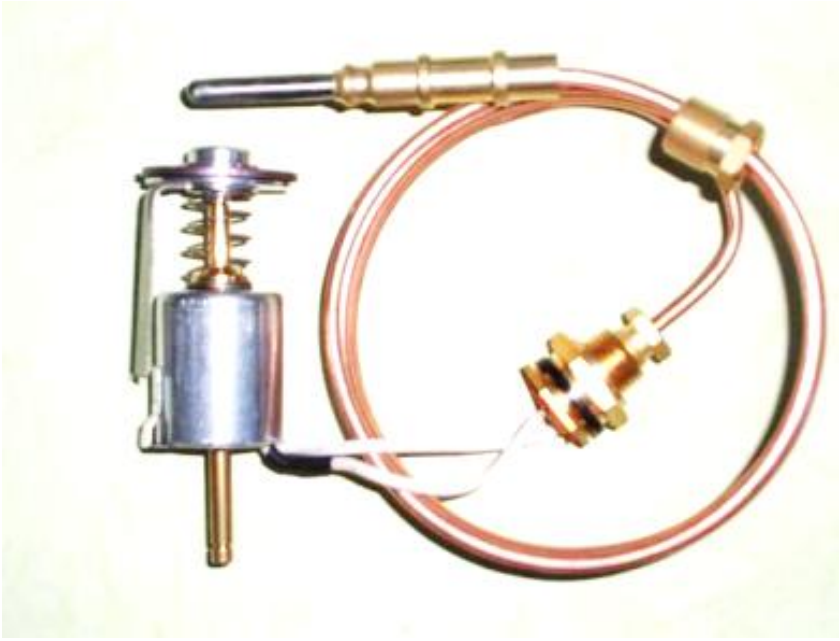


Figura 2: Instrumento de medida de temperatura termopar

Fuente: (Guerrero, 2002)

Al escoger los materiales con los cuales se construyen los termopares se debe tener en cuenta ciertos factores para garantizar su tiempo de vida útil y manteamiento por lo que de esta forma se desarrollan los siguientes tipos mostrados en la tabla 1.

Existen distintos tipos de combinaciones o aleaciones de metales para formar un termopar. Las cuatro combinaciones más comunes son J, K, T y E. También existen combinaciones capaces de soportar temperaturas altas como: R, S, C y GB.

Cada aleación funciona en un distinto rango de temperatura, aunque la temperatura máxima varía con el diámetro del alambre con el que se construye el termopar. A pesar de que la aleación del termopar proporciona el rango de temperatura, el rango máximo también se ve limitado por el diámetro del alambre del termopar. Es decir, un termopar construido con un alambre delgado posiblemente no alcance todo el rango de temperatura para el que fue construido.

Tabla 1.
Tipos de termopares de acuerdo a su composición y rango

Tipo de termopar		Material	Rango
B		+ Platino - 30% Rodio - Platino - 6% Rodio	0 + 1700 °C
C		+ Tungsteno - 5% Renio - Tungsteno - 26% Renio	0 + 2320 °C
E		+ Níquel - Cromo - Cobre - Níquel	-200 + 900 °C
J		+ Hierro - Cobre - Níquel	0 + 750 °C
K		+ Níquel - Cromo - Níquel - Aluminio	0 + 1700 °C
N		+ Níquel - Cromo - Silicio - Níquel - Silicio - Magnesio	-200 + 1250 °C
R		+ Platino - 13% Rodio - Platino	0 + 1450 °C
S		+ Platino - 10% Rodio - Platino	0 + 1450 °C
T		+ Cobre - Cobre - Níquel	-200 +350 °C

Fuente: (Rosa, 2014)

En el mercado existen varios tipos de termopares, los cuales varían su precio de acuerdo a la aleación con la que están construidos y el rango de temperatura que son capaces de medir:

- **Cobre – constantano (tipo t):** Este termopar está formado por un alambre de cobre como conductor positivo y una aleación de 60% de cobre y 40% de níquel como elemento conductor negativo. Se utiliza para medir temperaturas bajo 0 °C hasta los 350 ºC.

- **Chromel – alumel (tipo k):** Es capaz de medir temperaturas de hasta 1200º C. Puesto que está elaborado con níquel lo hace resistente a la oxidación. Se lo utiliza frecuentemente en los hornos de tratamientos térmicos. Su costo es considerable por lo que su uso es limitado.
- **Platino rodio – platino (tipo r):** Se desarrolla con materiales de alta pureza (**ver figura 3**) son capaces de medir hasta 1500 ºC tomando en cuenta las debidas precauciones. Son resistentes a la oxidación pero su uso se limita en atmósferas reductoras por su fácil contaminación con el hidrógeno y nitrógeno que modifican la respuesta del instrumento.

Figura 3: Termopar tipo R

Fuente: (Rosa, 2014)

RTD

En una termo resistencia o RTD a medida que varía la temperatura, su resistencia también se modifica, dicha variación es directamente proporcional a la variación de temperatura, Estos sensores son fabricados con alambres finos separados por un material aislante para posteriormente ser encapsulados, después se inserta dentro de una vaina o tubo metálico cerrado en un extremo que se llena con un polvo aislante y se sella con cemento para impedir que absorba humedad como se muestra en la **figura 4.**

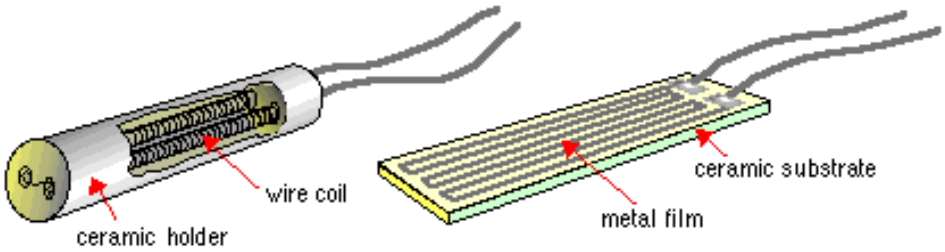


Figura 4: Rtd de construcción encapsulado

Fuente: (CV, 2016)

En la **tabla 2** se resume los tipos de RTD más utilizados en la industria de acuerdo a su material de fabricación

.

Tabla 2:
Tipos de Rtd de acuerdo su material y rango

Metal	Rango de temperatura	Características
Níquel - Hierro	-200 ° C a 200 ° C	Bajo costo
Platino	-240 ° C a 660 ° C	Buena precisión.
Níquel	-80 ° C a 260 ° C	Bajo costo, rango limitado de temperaturas
Cobre	-200 ° C a 260 ° C	Bajo coste

Fuente: (CV, 2016)

c. Calibración de temperatura

La Calibración se define como la comparación de la salida de un instrumento, con el valor entregado por un instrumento patrón que sirve para determinar su error de lectura.

Si se desea realizar una calibración de la variable temperatura de tipo industrial para el control de un proceso, los pasos a seguir son los siguientes: (Antioquia, 2010):

- **Conocer el intervalo a calibrar**. Es necesario verificar que nuestro equipo es capaz de cubrir el intervalo de calibración del instrumento bajo prueba (UUT por sus siglas en inglés).
- **Analizar incertidumbres**. Se recomienda que la incertidumbre total del equipo de referencia (termómetro de referencia, indicador y fuente de temperatura) tenga una relación de 4:1 con respecto a la exactitud del instrumento bajo prueba.
- **Definir puntos de medición**. Dividir de manera equidistante en temperatura el intervalo de calibración en al menos 5 puntos de medición cubriendo la mayor parte de dicho intervalo.
- **Llevar a cabo las mediciones**. Programar la fuente de temperatura para cada uno de los diferentes puntos de medición, una vez que la fuente de temperatura esté estable se deben tomar lecturas del instrumento de referencia y los instrumentos a calibrar. Se recomienda que se tomen varias lecturas en cada punto con lo cual se mejora la incertidumbre.
- **Realizar cálculos**. Una vez tomadas las mediciones se obtienen los promedios de las lecturas en cada punto, se calcula la incertidumbre de cada punto de medición y se determina si el instrumento a calibrar se encuentra dentro de las especificaciones del fabricante o su norma correspondiente.

2.1.2 Variables eléctricas

Las magnitudes eléctricas fundamentales son la tensión, resistencia, corriente y potencia siendo estas variables físicas, además se encuentran otras magnitudes como el caudal, flujo, temperatura, nivel entre otras que son medidas a través de una variable física como el voltaje resistencia o

corriente según el instrumento que se utilice. (Juan Carlos Álvarez Antón, 2007).

A continuación se muestra en la **tabla 3** las variables eléctricas y sus respectivas unidades:

Tabla 3:
Variables eléctricas básicas

Nombre	Símbolo	Unidad	Simbología
Resistencia	R	Ohmios	[Ω]
Corriente	I	Amperio	[A]
Voltaje	V	Voltio	[V]

a. Resistencia

Es la resistencia eléctrica que existe entre dos puntos de un conductor cuando al aplicar una diferencia de potencial de un voltio dichos puntos provocan una corriente eléctrica debido a que esta se opone al paso o desplazamiento de la corriente eléctrica.

b. Corriente

Es la corriente que circula por un conductor con una resistencia de un ohmio cuando entre sus extremos se aplica una diferencia de potencial de un voltio, por convención la corriente se denomina el pase de cargas eléctricas desde un terminal o punto positivo a uno negativo.

c. Voltaje

Es el potencial que existe entre dos puntos de un circuito y teniendo la diferencia se denomina diferencia de potencial o tensión entre ambos

puntos siendo la energía necesaria para mover los electrones de la carga negativa a la positiva.

2.1.3 Calibrador de temperatura

Es un equipo o dispositivo con el cual se realiza la calibración de instrumentos de medición de temperatura (**ver figura 5**), es decir permite la calibración o verificación de las sondas de temperatura, indicadores de temperatura o de los termómetros de lectura directa y transmisores de temperatura, utilizados en el ámbito industrial.

Figura 5: Calibrador de temperatura Fluke

Fuente: (FLUKE, 1995-2016)

Un calibrador de temperatura, se utiliza para corregir las imprecisiones en los termómetros y sensores de temperatura instalados en un proceso industrial, dado que la medición de temperatura es un parámetro fundamental en muchos procesos industriales y la mayoría de transmisores trabajan con la señal estándar de corriente existen los calibradores de lazo

que son fabricados con un propósito expreso, solucionar problemas en lazos de corriente 4 – 20 mA. Estos instrumentos versátiles son generalmente capaces no solo de medir corriente, sino que también de generar corriente hacia dispositivos pasivos en un lazo además de simular presión, temperatura, y también simular la operación de un transmisor (loop – powered) 4 - 20 mA. (FLUKE, 1995-2016).

2.1.4 Funciones de los calibradores

a. Medir

Un calibrador sirve para medir corriente de un transmisor de temperatura, presión, caudal nivel y otras variables de un proceso. En la **figura 6** *se* muestra como el calibrador sería usado para medir corriente en un lazo de entrada de señal vivo o en operación (Villajulca, 2011).

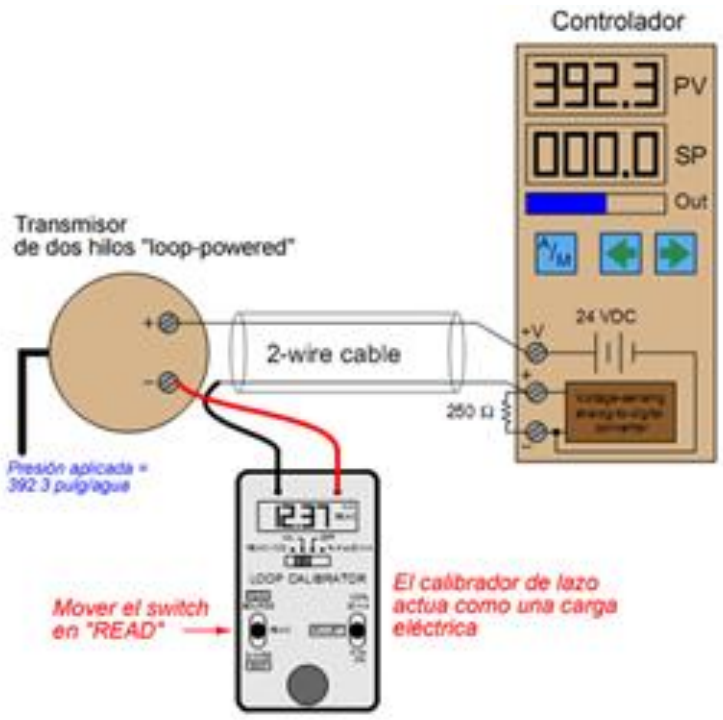


Figura 6: Medición de corriente en lazo de un transmisor de temperatura

Fuente: (Villajulca, 2011)

El cableado es interrumpido en el hilo negativo del transmisor, y el calibrador es conectado en serie para medir corriente; debemos tener en cuenta la polaridad del calibrador de modo que esté en relación con el lazo: el calibrador está actuando como un dispositivo pasivo (como una carga, mas no como una fuente), el hilo positivo del lazo debe conectarse al cable de prueba rojo y el hilo negativo con el cable de prueba negro.

En la **figura 7** se muestra al calibrador como una fuente de corriente hacia un transmisor.

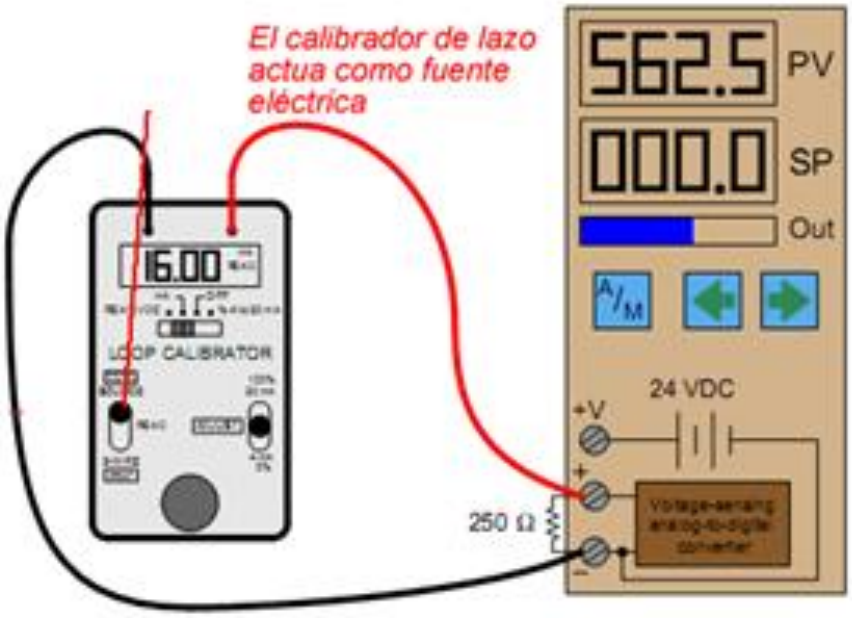


Figura 7: Medición de temperatura de un calibrador

Fuente: (Villajulca, 2011)

b. Generar

El calibrador de lazo puede ser usado como generador de señal de 4-20mA en lugar de un instrumento para probar las funciones de un instrumento independientemente. En la **figura 8** se observa un calibrador

usado como fuente de corriente enviando una señal de 16.00mA como PV (variable de proceso) de entrada del controlador. (cecuamaq, 2016)

No hace falta transmisor, dado que el calibrador ha ocupado su lugar, la función del calibrador ahora es una fuente activa de corriente que puede simular presión temperatura y no una carga pasiva. No solo provee la información (regular corriente), sino que también provee la energía al lazo **figura 8**. La fuente de energía DC dentro del controlador no es usada para este lazo, por el calibrador está en "modo fuente" que provee la energía necesaria para controlar la corriente a través de la resistencia de 250 ohm.

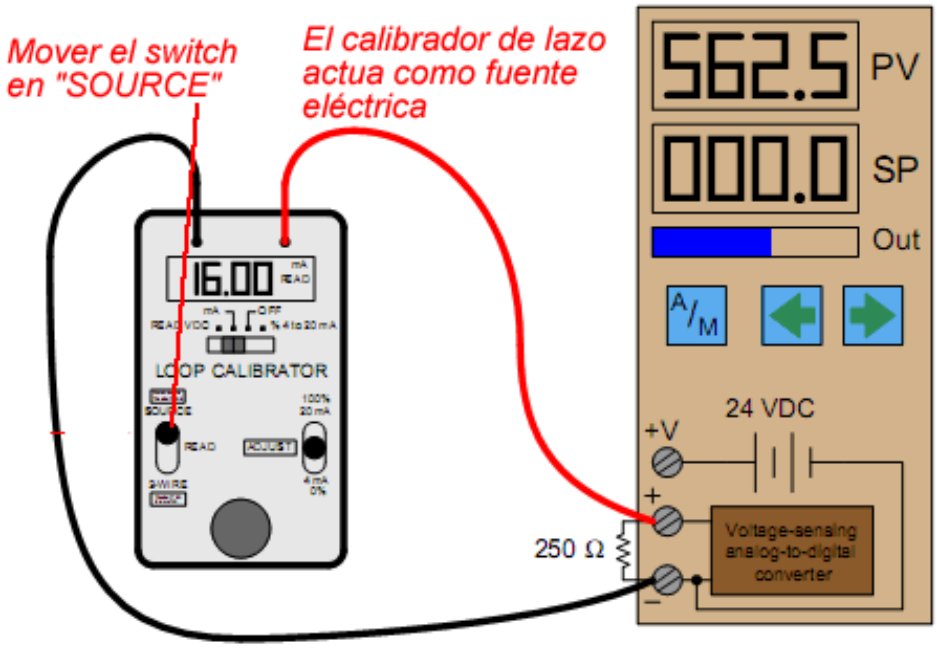


Figura 8: Calibrador como fuente y generador en un lazo

Fuente: (Villajulca, 2011)

Un método alternativo de proveer una señal de corriente (temperatura, presión) conocida y emular el comportamiento de un transmisor es poner el calibrador de lazo en modo simulación (**ver figura 9**). En este modo, el calibrador sirve para regular la corriente del lazo a un determinado valor, pero no genera algún voltaje para controlar la corriente. En vez de ello, es un elemento pasivo que necesita de algún voltaje externo en el lazo de corriente. (MARTINEZ, 2015)

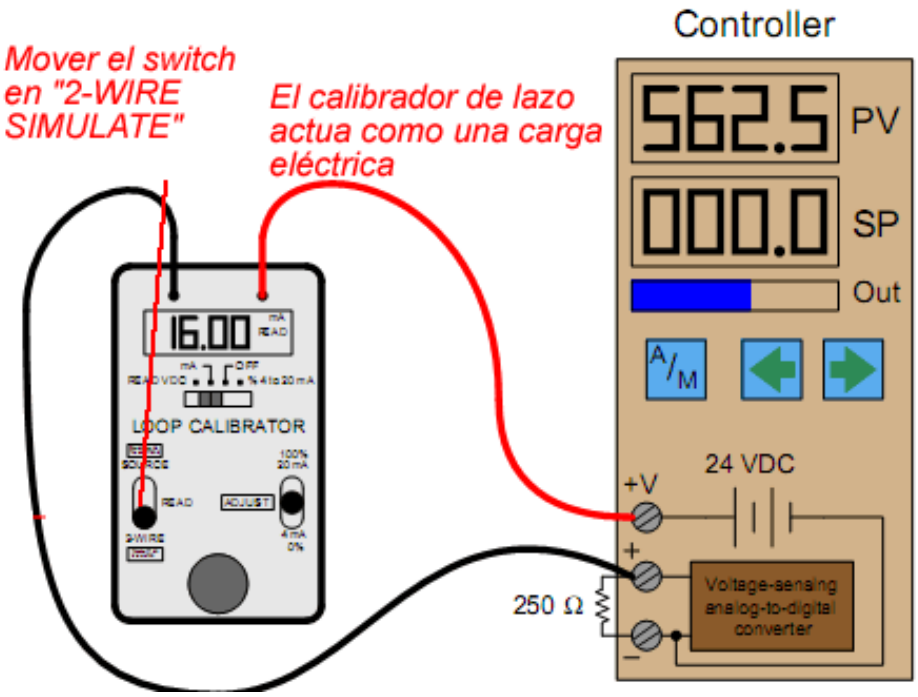


Figura 9: Calibrador en modo simulador

Fuente: (Villajulca, 2011)

El calibrador actúa como una carga, exactamente como un transmisor. La única fuente de energía en este circuito es los 24 voltios DC dentro del controlador. De este modo se simula un transmisor que es especialmente útil para testear un lazo 4-20 mA en la entrada del controlador. (Danica Schwarzkopf, 2012)

2.1.5 Software libre

El término Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software (hispalinux, 2010):

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.

- La libertad de distribuir copias, con lo que puedes contribuir a la mejora del software en uso. (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Un programa pertenece a software libre si los usuarios tienen todas estas libertades. Es decir la libertad de distribuir copias, sea con o sin modificaciones, gratis o pagando una cantidad por su distribución, a cualquier institución y en cualquier lugar. Esto significa (entre otras cosas) que no se tiene que pedir o pagar permisos (Galindo, 2010).

También existe la libertad de hacer modificaciones y utilizar las mismas de manera privada en tu trabajo u ocio, sin ni siquiera tener que anunciar que dichas modificaciones existen. Si publicas tus cambios, no existe la obligación de avisar a alguien en particular.

La libertad para usar un programa significa que cualquier persona u organización puede usar dicho programa en cualquier tipo de sistema informático, para cualquier clase de trabajo, y sin tener la obligación de comunicárselo al desarrollador o a alguna otra entidad específica.

La libertad de distribuir copias debe incluir tanto las formas binarias o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar (distribuir programas de modo ejecutable es necesario para que los sistemas operativos libres sean fáciles de instalar). Si no hay posibilidad de producir un binario o ejecutable de un programa concreto (ya que algunos lenguajes no tienen esta capacidad), Existe la libertad de distribuir estos formatos en caso de encontrar o desarrollar una manera de crearlos.

Para que las libertades de hacer modificaciones y de publicar versiones mejoradas tengan sentido, se debe tener acceso al código fuente del programa. Por lo tanto, la posibilidad de acceder a este código fuente es una condición necesaria para el software libre.

Para que estas libertades sean reales, deben ser irrevocables; si el desarrollador del software tiene el poder de revocar la licencia aunque no existan motivos comprobados, el software no es libre.

a. Características del software libre

De acuerdo a la definición, el software es libre si se garantizan las 4 libertades descritas en la tabla 4

Tabla 4:
Libertades del software libre

Libertad 0	Libertad 1	Libertad 2	Libertad 3
La libertad de usar el programa, con cualquier tipo de propósito (privado, educativo, público, comercial, militar, etc.)	La libertad de estudiar el funcionamiento del programa, y adaptarlo a las necesidades, para lo cual es necesario poder acceder al código fuente	La libertad de distribuir copias, con lo que puedes ayudar a quien sea	La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie

Fuente: (clerus, 2010)

2.1.6 Software Arduino

Los entornos de desarrollo de software de Arduino son compatibles para la gran mayoría de placas de Arduino ya que contienen toda la documentación relativa a los dos proyectos para programar y subir los archivos en las tarjetas, son: el nuevo Arduino Studio y el clásico Arduino IDE **figura 10**.

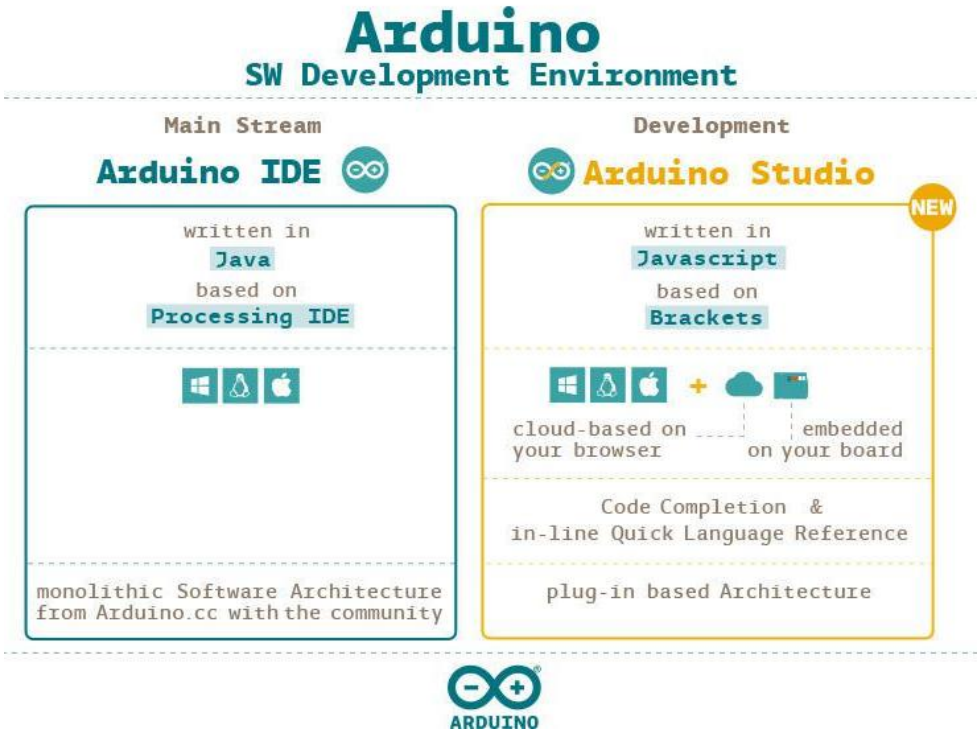


Figura 10: Software Arduino Studio y Arduino IDE

Fuente: (srl, 2016)

a. Arduino IDE

Es el software estándar de Arduino para escribir el código de programación y subirlo a la tarjeta. Se ejecuta en Windows, Mac OS X y Linux. El entorno está escrito en Java y otro software de código abierto. Este software se puede utilizar con cualquier placa Arduino.

b. Arduino Studio

Es un nuevo entorno de desarrollo de código abierto para el lenguaje de programación de Arduino. En lugar de una arquitectura monolítica y un modelo de desarrollo centralizado, Arduino Estudio toma ventaja del sistema extensible del editor, además se puede permitir a los usuarios utilizar Arduino Studio como autónomo. Sólo existe un editor para todos los entornos de Arduino Studio el cual puede funcionar correctamente en las plataformas Linux, Mac OS X y Windows.

Además de los entornos de programación que ofrecen arduino.cc y arduino.org, existe otro software, el cual es un sistema operativo basado en Linux que corre dentro de los Arduinos con un procesador MIPS Qualcomm Atheros, Yun o Tian. LininoIO es un framework (**figura 11**) capaz de integrar las capacidades de un microcontrolador dentro de un entorno Linux. Es posible escribir una aplicación en Python, Node.js, etc. usando LininiOS para controlar completamente la MCU y los dispositivos conectados. (wordpress, 2016)

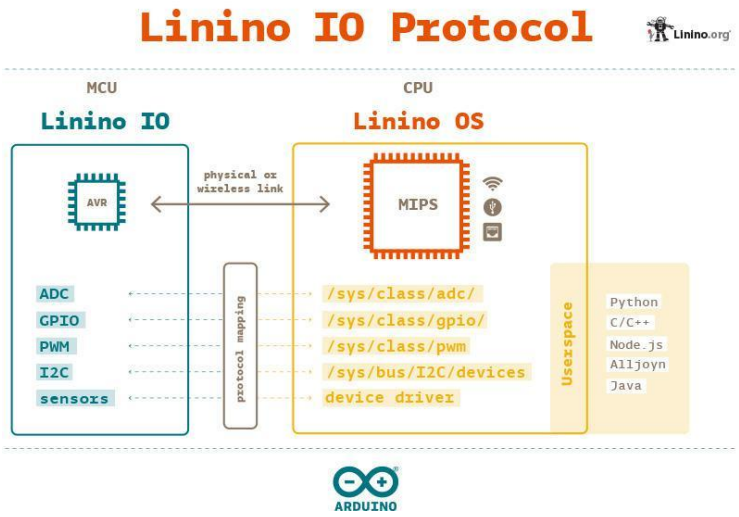


Figura 11: Linino protocolo de arduino para software libre

Fuente: (wordpress, 2016)

2.1.7 Software Python

Python **figura 12** es un lenguaje de programación multipropósito de alto nivel Su filosofía de diseño enfatiza la productividad del programador y la legibilidad del código. Tiene un núcleo de programación simple con unos pocos comandos básicos y simple semántica, pero además tiene una enorme y variada gama de librerías estándar, que incluyen una Interfaz de Programación de Aplicaciones (API) API para muchas de las funciones en el nivel del sistema operativo (OS) (Pierro, 2016).

Figura 12: Python lenguaje de programación

Fuente: (Alvarez, 2003)

a. Características de Python

Propósito general: Es decir que pueden crear todo tipo de programas. No es un lenguaje creado específicamente para la web, aunque entre sus posibilidades sí se encuentra el desarrollo de páginas (Alvarez, 2003).

Multiplataforma: Hay versiones disponibles de Python en muchos sistemas informáticos distintos. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para el mismo. (Alvarez, 2003)

Interpretado: Quiere decir que no se debe compilar el código antes de su ejecución. En realidad sí se produce una compilación, pero esta se realiza de manera transparente para el programador. En ciertos casos, cuando se ejecuta por primera vez un código, se producen unos bytecodes que se guardan en el sistema y que sirven para acelerar la compilación implícita que realiza el intérprete cada vez que se ejecuta el mismo código.

Interactivo: Python dispone de un intérprete por línea de comandos en el que se pueden introducir sentencias. Cada sentencia se ejecuta y produce un resultado visible, que puede ayudar a entender mejor el lenguaje y probar los resultados de la ejecución de porciones de código rápidamente.

Orientado a Objetos: La programación orientada a objetos está soportada en Python y ofrece en muchos casos una manera sencilla de crear programas con componentes reutilizables.

Funciones y librerías: Dispone de muchas funciones incorporadas en el propio lenguaje, para el tratamiento de strings, números, archivos, etc. Además, existen varias librerías que se pueden importar en los programas para tratar temas específicos como la programación de ventanas o sistemas en red o crear archivos comprimidos en .zip.

Sintaxis clara: Python tiene una sintaxis muy clara. En muchos lenguajes, para separar porciones de código, se utilizan elementos como las llaves o las palabras clave begin y end. Para separar las porciones de código en Python se debe tabular hacia dentro, colocando un margen al código que iría dentro de una función o un bucle. Esto ayuda a que todos los programadores adopten unas mismas notaciones y que los programas de cualquier persona tengan un aspecto muy similar (Alvarez, 2003).

2.1.8 Arduino DUE

Es la primera placa de desarrollo de Arduino basado en ARM. Esta placa está basada en un potente microcontrolador ARM CortexM3 de 32bit, programable mediante el IDE de Arduino. Aumenta la potencia de cálculo disponible para los usuarios de Arduino manteniendo el lenguaje lo más compatible posible para que muchos programas puedan migrarse en cuestión de minutos (Artero, 2013).

El Arduino Due en la **figura 13***,* dispone de 54 pines digitales de entrada / salida (de los cuales 12 pueden utilizarse para salidas PWM), 12 entradas analógicas, 4 UARTs (puertas seriales), un reloj de 84 MHz, una conexión USB OTG, 2 DAC (convertidor digital a analógico), 2 TWI, un conector de alimentación, un cabezal SPI, un cabezal JTAG, un botón de reinicio y un botón de borrado. También hay algunas características interesantes como DACs, Audio, DMA, una biblioteca multitarea experimental, etc.

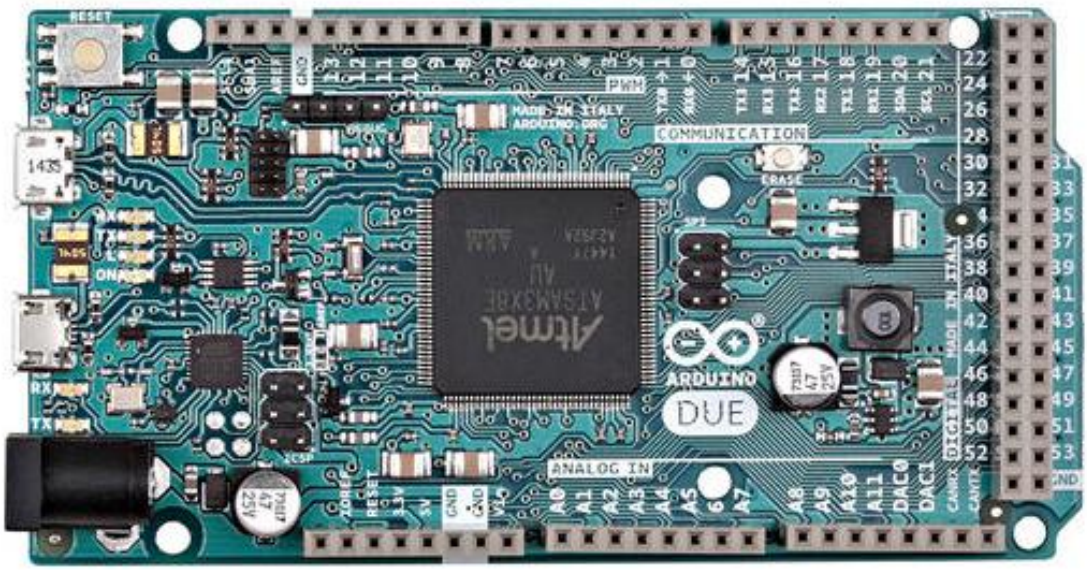


Figura 13: Placa arduino Due

Fuente: (Ojeda, 2016)

Para compilar el código para el procesador ARM, se necesita la última versión del IDE de Arduino: v1.5 Debido a las limitaciones de tensión del sistema impuesto por el Atmel SAM3X8E, por ende shields de Arduino basados en los modelos de 5v no funcionarán correctamente.

a. Comunicación Arduino

El Arduino DUE tiene una serie de elementos para la comunicación con un ordenador, otro Arduino, microcontroladores, y diferentes dispositivos, como los teléfonos, tablets, cámaras, etc. El SAM3X proporciona una UART hardware y tres hardware USARTs para TTL de comunicación en serie (3,3 V) (srl, 2016).

El puerto de programación está conectado a un ATmega16U2, que proporciona un puerto COM virtual de software en un ordenador conectado (Para reconocer el dispositivo en máquinas de Windows necesitará un archivo .inf, pero las máquinas OSX y Linux reconocerán la tarjeta como un puerto COM de forma automática.). Los pines RX0 y TX0 proporcionan la comunicación serie a USB para la programación a través del microcontrolador ATmega16U2. El software de Arduino incluye un monitor serie que permite a los datos ser transmitidos desde y hacia el Arduino.

b. Processing

Se trata de un lenguaje de programación, entorno de desarrollo y comunidad en línea, de código abierto para todas las personas que deseen crear imágenes, animaciones e interacciones abiertas. Inicialmente creado para servir como un cuaderno de bocetos de software y para enseñar los fundamentos de programación de computadoras dentro de un contexto

visual, al pasar los años este evolucionó hasta convertirse en una herramienta de desarrollo para los profesionales (Ben Fry, 2004).

c. Wiring

El wiring es un marco de programación de código abierto para los microcontroladores y placas, que permite la escritura de software multiplataforma para controlar los dispositivos conectados a una amplia gama de placas de microcontroladores para crear todo tipo de codificación, objetos interactivos, espacios, etc. De acuerdo a las limitaciones de cada microcontrolador (Barriga, 2013).

d. Características de arduino Due

Las principales características del arduino Due se resumen en la **tabla 5**.

Tabla 5:
Características técnicas de arduino Due

Microcontroladores	**AT91SAM3X8E**
Voltaje de operación	3.3V
Voltaje de entrada	7-12 V
Límites de voltaje	6-16 V
Pines I/O digital	54 (de los cuales 12 proporcionan salida PWM)
Entradas analógicas	12
Salidas analógicas	2 (DAC)
Salida máxima I/O	130 mA
Corriente máx. por pin 3.3V	800 mA
Corriente máx. por pin 5V	800 mA
Memoria Flash	512 KB disponibles para las aplicaciones de usuario
SRAM	96 KB (64+32 KB)
Frecuencia de Reloj	84 MHz

Fuente: (srl, 2016)

e. Beneficios del núcleo ARM

Es un núcleo ARM de 32 bits que puede superar las tarjetas típicas de 8 bits. Las diferencias más significativas son: (srl, 2016):

- Un núcleo de 32 bits, que permite operaciones en 4 bytes de datos de ancho dentro de un único reloj de la CPU.
- El reloj de la CPU es de 84Mhz.
- 96 Kbytes de SRAM.
- 512 Kbytes de memoria flash para el código.
- Controlador de DMA, que puede aliviar la CPU de hacer las tareas intensivas de memoria.

f. Comunicación SPI (Serial Peripheral Interface)

La Interfaz Periférica Serial (SPI) es un protocolo de datos en serie síncrono utilizado por microcontroladores para comunicarse con uno o más dispositivos periféricos rápidamente en distancias cortas. También se puede utilizar para la comunicación entre dos microcontroladores.

En una comunicación SPI siempre debe existir un dispositivo maestro (generalmente un microcontrolador) que controla los dispositivos periféricos. Normalmente hay tres líneas comunes a todos los dispositivos:

- **MISO (Master In Slave Out) -** La línea Slave para enviar datos al maestro,
- **MOSI (Master Out Slave In) -** La línea maestra para enviar datos a los periféricos,
- **SCK (Serial Clock) -** Los pulsos de reloj que sincronizan la transmisión de datos generada por el maestro

SS (Slave Select): el pin de cada dispositivo que el maestro puede usar para habilitar y deshabilitar dispositivos específicos. Cuando el pin de selección de esclavo de un dispositivo está bajo, se comunica con el maestro. Cuando está en alto, ignora al maestro. Esto le permite tener múltiples dispositivos SPI compartiendo las mismas líneas MISO, MOSI y CLK.

A diferencia de otros buses el SPI no implementa el nivel del enlace entre dispositivos, es decir no hay un campo para la dirección ni un campo para ACK, etc. El SPI se comporta como un shift register donde a cada golpe de clock se captura un bit. En parte no es necesario hacer un direccionamiento de los chips ya que mediante la señal Chip select, habilitamos al integrado al que queremos enviar los datos.

El funcionamiento para un envío de un master es el siguiente:

- Se habilita el chip al que hay que enviar la información mediante el CS (Opcional).
- Se carga en el buffer de salida el byte a enviar.
- La línea de clock empieza a generar la señal cuadrada donde normalmente por cada flanco de bajada se pone un bit en MOSI.
- El receptor normalmente en cada flanco de subida captura el bit de la línea MISO y lo incorpora en el buffer.
- Se repite el proceso 8 veces y se ha transmitido un byte. Si se ha terminado de transmitir se vuelve a poner la línea CS en reposo. Hay que tener en cuenta que a la vez que el Master está enviando un dato también lo recibe así que si el Slave ha depositado algún byte en el buffer de salida, este también será enviado y recibido por el Master. (Arduino, 2016).

2.1.9 Pantalla arduino tft lcd

La pantalla TFT de Arduino es una pantalla LCD **figura 14** retro iluminada, es capaz de dibujar texto, imágenes y formas en la pantalla haciendo uso de la biblioteca de TFT. También posee una ranura para tarjetas micro-SD en la parte posterior de la pantalla en donde se puede almacenar el mapa de bits, las imágenes entre otras cosas para mostrar, además también es posible recuperar todo tipo de datos necesarios.

Los pines de la pantalla están diseñados para encajar en la parte delantera del Arduino, se construye en formato Shield para conectarla directamente con la tarjeta, para funcionar utiliza el protocolo SPI por lo que se ahorran pines caso de que sea necesario utilizar otros accesorios. La pantalla táctil es resistiva ya que utiliza los pines analógicos de Arduino para poder recuperar directamente la posición. Está basada en el controlador ssd1963. (Arduino, 2016).

Figura 14: Pantalla lcd TFT para arduino

Fuente:(Arduino, 2016)

Las características principales de la pantalla son las siguientes:

- Chip controlador ssd1963.
- Pixeles de resolución y colores de acuerdo al tamaño de pantalla.
- Compatible con una tensión de 5 voltios puesto que posee un regulador de tensión en la placa.

2.1.10 Elevador de voltaje

El elevador de tensión o voltaje **figura 15** es un equipo cuya misión es mantener la tensión de salida dentro de un cierto límite a partir de la selección de la derivación correspondiente de un autotransformador. Esta función la realiza un circuito electrónico que es el encargado de activar una serie de relays de potencia los cuales manejan la corriente de carga.

El tiempo de conmutación es del orden de 20 ms, es decir de un ciclo completo de la tensión de entrada. Durante ese tiempo no hay corriente de salida. En general esto no genera ningún inconveniente, salvo en ciertas instalaciones como por ejemplo: redes de computadoras, surtidores de combustible, equipos de revelado de fotos, lámparas de mercurio de alta presión etc.

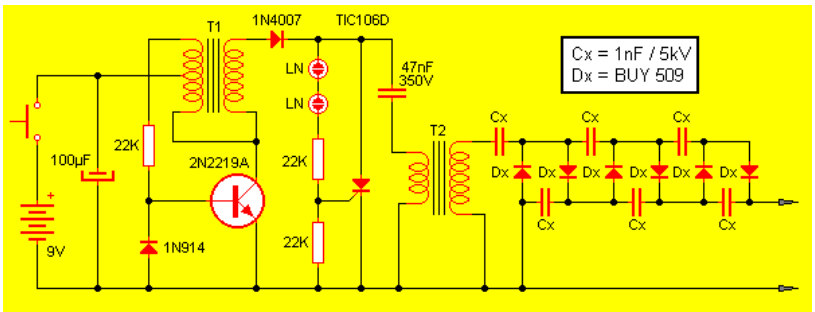


Figura 15: Elevador de voltaje o tensión circuito genérico

Fuente: (DSL, 2010)

Dicho elevador se puede emplear para todo tipo de carga, sólo se tiene en cuenta que un coseno de fi elevado dará como consecuencia un mal aprovechamiento del equipo. Siempre es conveniente compensar la carga para optimizar el funcionamiento y prolongar la vida útil de los contactos de los relays. Los circuitos electrónicos se encuentran temporizados cuando la tensión baja para que sumado a la histéresis de los mismos produzca una integración de la tensión evitando las conmutaciones intermitentes (DSL, 2010).

2.1.11 Conversor DC-DC MELCHER

La familia PSA de reguladores de giro positivo está diseñada como módulos de alimentación para sistemas electrónicos. Su principal ventaja es tener un alto nivel de eficiencia ya que prácticamente se mantiene constante en todo el rango de entrada, posee alta fiabilidad, ondulación baja y excelente respuesta dinámica. Los módulos con entrada de hasta 60 V están especialmente orientados a aplicaciones secundarias conmutadas y accionadas por baterías **figura 16** (Melcher AG, 1996).

Figura 16: Conversor DC-DC MELCHER

Fuente: (Melcher AG, 1996)

El circuito de salida. Sin embargo, es responsabilidad exclusiva del usuario para asegurar el cumplimiento de los requisitos y las normas de seguridad aplicables, la salida de cualquier conmutación regulada se considera un circuito SELV de tensión de salida nominal de hasta 36 V.

2.1.12 MAX31856

El MAX31856 **figura 17** realiza la compensación de la unión fría y digitaliza la señal de cualquier tipo de termopar. Los datos de salida se envian en grados Celsius. Este conversor posee una resolución en temperatura de 0.0078125 °C, permite lecturas tan altas como + 1800 °C y tan bajas como -210 °C (dependiendo del tipo del termopar), y tiene una exactitud de la medida de voltaje del termopar de ± 0.15%. Las entradas del termopar están protegidas contra condiciones de sobretensión de hasta ± 45V. Una tabla de consulta almacena datos de corrección de linealidad para varios tipos de termopares (K, J, N, R, S, T, E y B). El filtro de frecuencia de 50 Hz y 60 Hz está incluido, al igual que la detección de fallo del termopar. Una interfaz compatible con SPI permite la selección del tipo de termopar y la configuración de los procesos de conversión y detección de fallos. (alldatasheet, 2015).

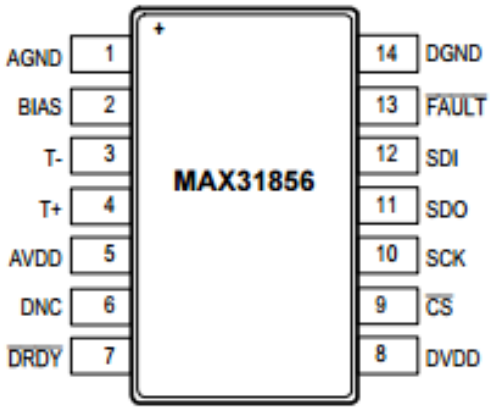


Figura 17: MAX31856

Fuente: (alldatasheet, 2015).

Características

- Proporciona una temperatura de termopar de alta precisión
- Incluye corrección automática de linealización para 8 tipos de termopares
- 19-Bits de resolución o 0.0078125 °C temperatura del termopar
- ● Realiza la compensación interna de la unión fría con una exactitud de ± 0,7 ° C (máximo, -20 °C a + 85 °C).
- ● Posee un sistema robusto ya que brinda una protección de entrada de ± 45 V.
- ● Simplifica la gestión de fallos del sistema y detecta termopares abiertos
- Detección de fallas de sobre temperatura y sub temperatura
- ● Filtración de ruido 50 Hz / 60 Hz.

Aplicaciones

- Controladores de Temperatura
- Hornos Industriales, Hornos y Ambientales
- Cámaras
- Equipamiento Industrial

2.1.13 AD5293

El AD5293 **figura 18** es un circuito de un solo canal con un potenciómetro digital de 1.024 posiciones, tiene un error de tolerancia de resistencia de 1% de extremo a extremo. Este circuito realiza la misma función de ajuste electrónico que un potenciómetro mecánico con una resolución mejorada, tiene un rendimiento superior de bajo coeficiente de temperatura además es capaz de funcionar a altas tensiones y apoyando tanto la operación de doble suministro de ± 10,5 V a ± 15 V y la operación de un solo suministro a 21 V a 30 V.

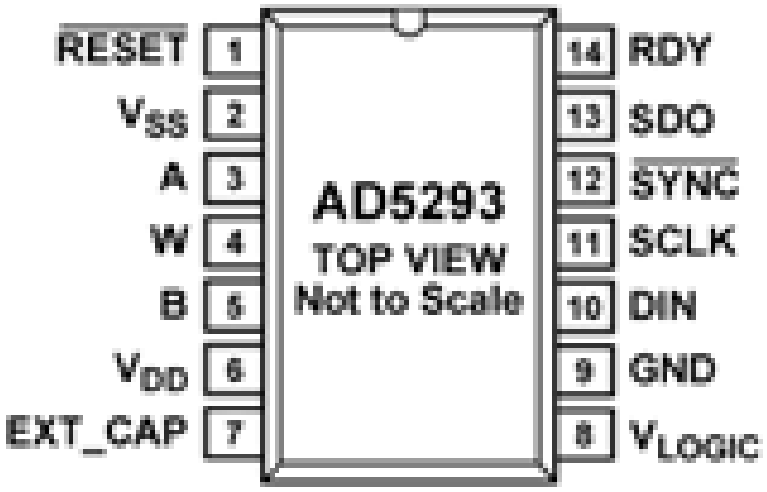


Figura 18: Circuito integrado Ad 5293

Fuente: (Analog Devices, 1995)

Características

- Monocanal, con una resolución de 1024 posiciones.
- 1% de tolerancia nominal de la resistencia.
- Coeficiente de temperatura modo de reostato: 35 ppm / °C
- Coeficiente de temperatura divisor de tensión: 5 ppm / °C
- Operación de alimentación individual: 9 V a 33 V
- Operación de alimentación dual: ± 9 V a ± 16,5 V
- interfaz en serie compatible con SPI
- Ajuste de relectura.

2.1.14 AD5420

El AD5410/AD5420 **figura 19** es un integrado de bajo costo, alta precisión, ofreciendo una salida de fuente de corriente programable para satisfacer los requisitos de las aplicaciones de control de procesos industriales. El rango de corriente de salida es programable de 4 mA a 20 mA, 0 mA a 20 mA, 0 mA a 24 mA. La salida es de circuito abierto protegido.

El dispositivo funciona con una fuente de alimentación de 10,8 V a 60 V, posee un bucle de salida de 0 V a VDD -2,5 V. (Analog Devices, Inc., 1998)

La interfaz serie es flexible con las comunicaciones SPI, MICROWIRE, QSPI, y DSP y puede ser operado en el modo de 3 hilos para reducir al mínimo el aislamiento digital requerida en aplicaciones aisladas.

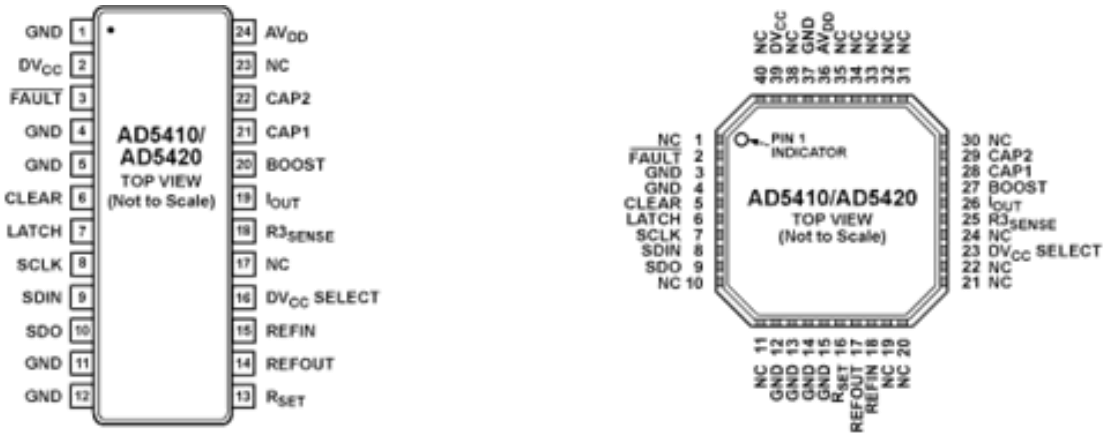


Figura 19: Circuito integrado Ad 5420

Fuente: (Analog Devices, 1995)

Características

- 12/16-bit de resolución
- Rangos de salida de corriente: 4 mA a 20 mA, 0 mA a 20 mA o 0 mA a 24 mA ± 0,01% de error no ajustado.
- Interfaz digital en serie flexible
- Solución industrial 4-20 mA DAC
- Convertidores D / A

CAPÍTULO III

3. DISEÑO E IMPLEMENTACIÓN

En este capítulo se explicará de forma detallada el hardware y software que posee el equipo implementado, en la **figura 20** se muestra el diagrama de bloques del calibrador documentador.

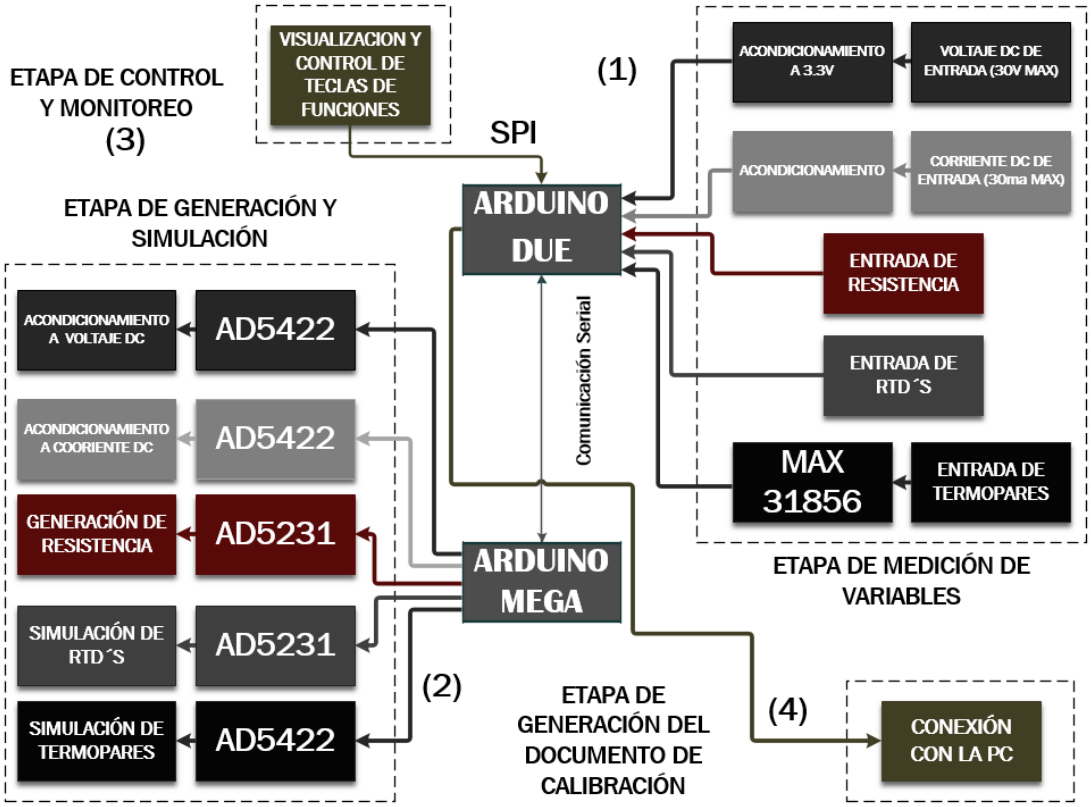


Figura 20: Diagrama de bloques del Calibrador Documentador

El proyecto consta de cuatro etapas: la primera es una etapa de medición de variables eléctricas, la segunda genera y simula las mismas variables eléctricas, la tercera se encarga del monitoreo y control y la cuarta etapa es la de generación del documento con los datos de calibración, a continuación se describe cada una de las etapas:

1. **Medición:** Una tarjeta Arduino Due es la que se encarga de realizar la lectura de las señales acondicionadas de voltaje DC, corriente DC, resistencia eléctrica y rtds, para ello se realiza una conversión análoga-digital mediante un ADC de 12 bits, la lectura de termocuplas se realiza utilizando un MAX31856, mismo que envía el valor de la medición al Arduino Due de forma digital.
2. **Generación o simulación:** Se realiza en un Arduino MEGA el mismo que genera señales de voltaje DC (10V máx.), corriente DC (24 mA máx.), resistencia eléctrica, además simula termocuplas y rtds, mediante el control de varios circuitos integrados especiales y envía los valores de generación-simulación al Arduino DUE que actúa como el maestro de todo el hardware.
3. **Control y monitoreo:** Una pantalla touch de 7 pulgadas es la encargada de mostrar los valores de las señales que mide, genera y simula el prototipo, además controla mediante varias teclas las distintas opciones (medición generación y simulación) que puede realizar el equipo.
4. **Generación del documento:** En esta etapa la PC se comunica de forma serial con el arduino Mega y de esta manera generar el documento de calibración con los datos que se envían desde el Arduino Due.

3.1 Diseño del Hardware

Para explicar de una mejor manera el diseño y construcción del hardware del prototipo implementado, se dividió en dos etapas: la primera es la medición de variables eléctricas y la segunda es la generación – simulación de variables eléctricas.

3.1.1 Etapa de Medición de Variables Eléctricas

En esta etapa se explica cada uno de los acondicionamientos realizados para la medición de variables eléctricas, así como también se detalla el funcionamiento de los integrados utilizados.

a. Adquisición de datos

La adquisición de datos consiste en la medición de una o varias magnitudes físicas (análogas) para después convertirlas en datos digitales los cuales puedan ser gestionados y procesados para su presentación.

La mayor parte de sistemas de adquisición de datos son basados en computadoras, microprocesadores o tarjetas y todo sistema de adquisición de datos posee un conversor Análogo – Digital. La **figura 21** muestra un diagrama de bloques en el cual se representa un sistema de adquisición de datos.

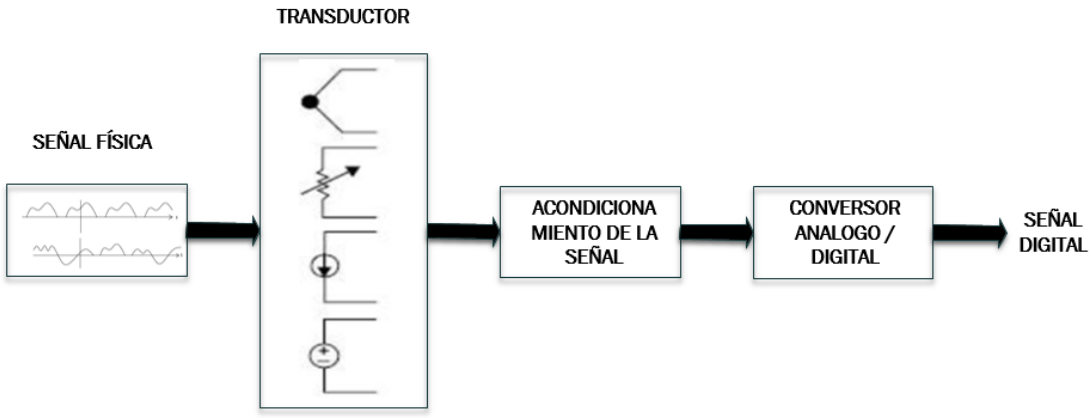


Figura 21: Diagrama de bloques de un sistema de adquisición de datos

b. Acondicionamiento de señales

Acondicionar una señal es manipular dicha señal, con los dispositivos correctos para obtener valores de voltaje o corriente adecuados para la siguiente etapa de operación. Es decir una señal podría ser demasiado pequeña por lo que sería necesario amplificarla; podría ser no lineal y requerir su linealización; ser análoga y ser convertida en digital, etc.

El acondicionamiento de señales es de mucha utilidad para realizar la instrumentación del proyecto, ya que al trabajar con señales de rangos de voltaje amplios, se puede obtener una mejor resolución, además de mayor inmunidad al ruido.

Este sistema de instrumentación tiene dos clases principales: analógicos y digitales.

- **Sistema Analógico:** Es un conjunto de dispositivos interconectados que tienen la capacidad de generar, procesar, transmitir o almacenar señales análogas. En este sistema, las variables físicas varían sobre un intervalo continuo de tiempo.
- **Sistema Digital:** Es cualquier dispositivo electrónico destinado a la generación, transmisión, procesamiento o almacenamiento de señales digitales, este tipo de sistemas solo pueden tomar valores discretos.

b.1 Acondicionamiento de voltaje

Se realizó el acondicionamiento de una señal de voltaje continuo, en la **figura 22** se muestra el circuito de acondicionamiento para que esta señal

pueda ingresar al conversor análogo digital del arduino DUE, usando divisores de tensión, seguidores de tensión, filtros pasivos, etc.

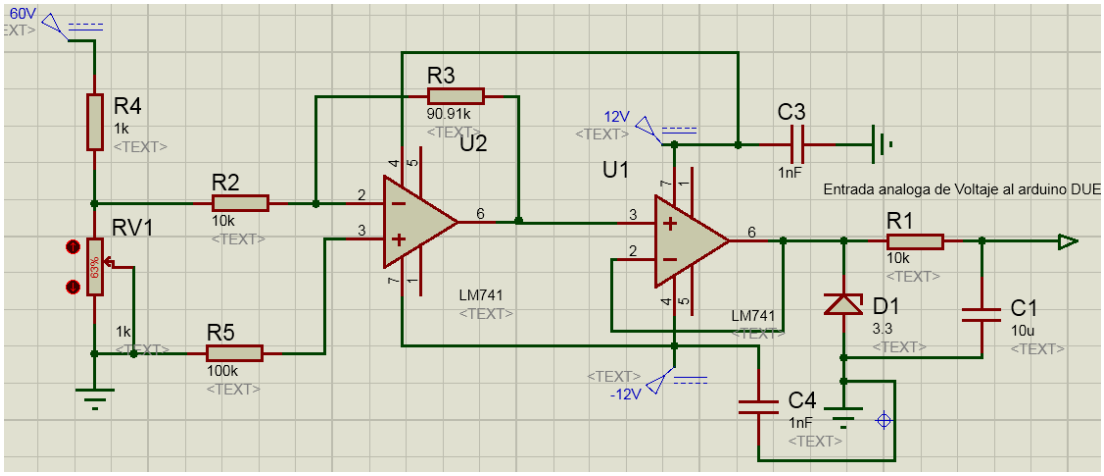


Figura 22: Diagrama esquemático del acondicionamiento de voltaje

La señal de entrada de voltaje tendrá el rango de 0 a 30 Vdc misma que se obtendrá de un divisor de tensión, después será acondicionada a una señal de 0 a 3.3 Vdc mediante un conversor de niveles de voltaje, para evitar caídas de tensión se coloca un seguidor de tensión, con el fin de evitar el daño de la entrada analógica de la tarjeta es necesario un diodo zener de 3.3 V, antes de que la señal ingrese a la tarjeta pasa por un filtro pasa bajo pasivo RC para disminuir el ruido de dicha señal.

La ecuación para obtener los valores necesarios para el conversor de niveles de voltaje es la siguiente:

$$A = \frac{Vo2 - Vo1}{Vin1 - Vin2} = \frac{30 - 0}{3.3 - 0} = 9.0909$$

$$B = Vo1 - A * Vin1 = 0$$

$$R1 = 10k$$

$$R2 = A * R1 = 9.0909 * 10k$$

$$R2 = 90.91k$$

Ecuación para la obtención de los valores del filtro RC pasivo.

$$\omega C = \frac{1}{R * C}$$

$$R = 1k$$

$$\omega C = \frac{1}{R * C}$$

$$C = \frac{1}{1000 * 100Hz}$$

$$C = 10uf$$

Cabe recalcar que estos mismos cálculos de este filtro se utilizaron para la medición de voltaje, corriente, resistencia y RTDS.

b.2 Acondicionamiento de corriente

Para realizar la medición de corriente se implementó el circuito de la **figura 23** al cual ingresa una fuente de corriente de 0 a 30 mA y mediante una resistencia de un ohmio se transforma en voltaje continuo que es directamente proporcional a la corriente que ingresa al circuito es decir al variar 1mA el cambio de voltaje será de 1mv, en la siguiente etapa este voltaje ingresa a un amplificador de instrumentación ad620 al cual se le asigna la ganancia necesaria para que envíe una señal de voltaje en el rango de 0 a 3.3 Vdc, ya que es el valor idóneo para la entrada analógica del arduino due, para evitar caídas de tensión la señal pasa por un seguidor de tensión, sin embargo es necesario colocar un diodo zener de 3.3 voltios para evitar que la entrada análoga de la tarjeta sufra algún daño, antes de que la señal ingrese al arduino due pasa por un filtro pasa bajo pasivo para disminuir el ruido que generan los mismos elementos electrónicos, mediante la aplicación de la ley de ohm en el software se obtiene el valor

correspondiente de corriente el cual se lo visualiza en la pantalla del prototipo.

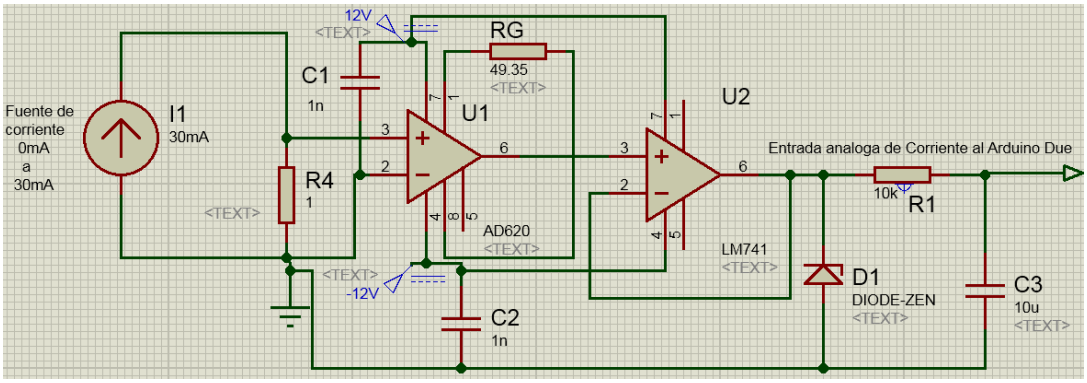


Figura 23: Diagrama esquemático del acondicionamiento de corriente

Ecuación para obtener el valor de resistencia para el AD620.

$$G = \frac{49.4k}{Rg} + 1$$

$$Rg = \frac{49.4k}{G + 1} = \frac{49.4k}{1001 + 1}$$

$$Rg = 49.35\Omega$$

b.3Medición de resistencia

El circuito de medición de resistencia se muestra en la **figura 24** en el cual ingresa una fuente de corriente constante de 350 uA misma que circula por una resistencia de un ohmio y la resistencia que se espera obtener el valor, cabe recalcar que el valor de la fuente de corriente no debe exceder los 0.2mA ya que es la corriente máxima que puede soportar el generador de resistencia del calibrador Fluke 724 el mismo que se emplea como instrumento patrón, el voltaje que se genera es enviado hacia un AD620 para ser amplificado ya que es un valor muy pequeño, una vez obtenido este nuevo voltaje la señal pasa por un seguidor de tensión con el fin de evitar

caídas de tensión, además se coloca un diodo zener de 3.3 voltios para evitar que la entrada análoga de la tarjeta sufra algún daño, antes de que la señal ingrese al arduino due pasa por un filtro pasa bajo pasivo para disminuir el ruido que generan los mismos elementos electrónicos, mediante la aplicación de la ley de ohm en el software se obtiene el valor correspondiente de resistencia el cual se lo visualiza en la pantalla del prototipo.

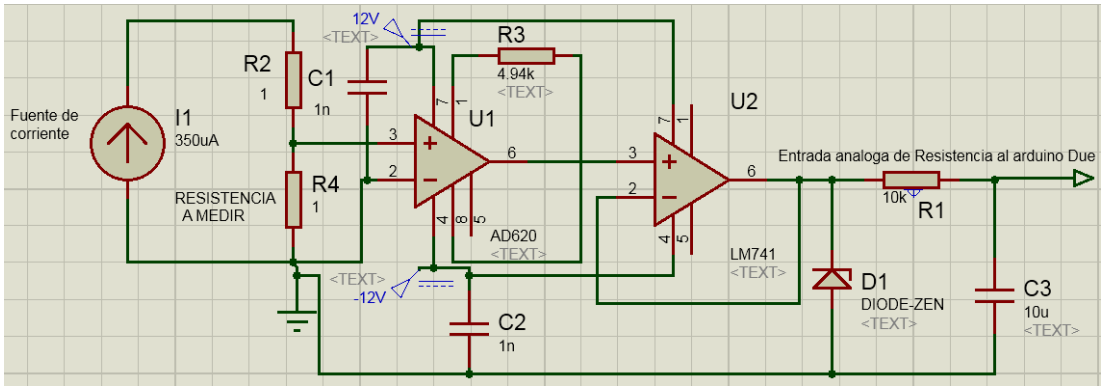


Figura 24: Diagrama esquemático del acondicionamiento de resistencia

Ecuación para obtener el valor de resistencia para el AD620.

$$G = \frac{49.4k}{Rg} + 1$$

$$Rg = \frac{49.4k}{G + 1} = \frac{49.4k}{9 + 1}$$

$$Rg = 4.94k\Omega$$

Esta ganancia obtenida se la utiliza también para medir las RTDS de 2, 3 y 4 hilos

b.4Medición de RTD

A continuación se muestran los circuitos implementados para realizar la medición de RTDS PT100 de 2, 3 y 4 hilos, ya que esta medición se basa en la variación de resistencia el circuito implementado es el que se utiliza en la medición de resistencia, con ciertas variaciones que se muestran en las siguientes figuras.

RTD de 2 hilos

En la **figura 25** se puede apreciar el circuito necesario para realizar la medición de una RTD de 2 hilos.

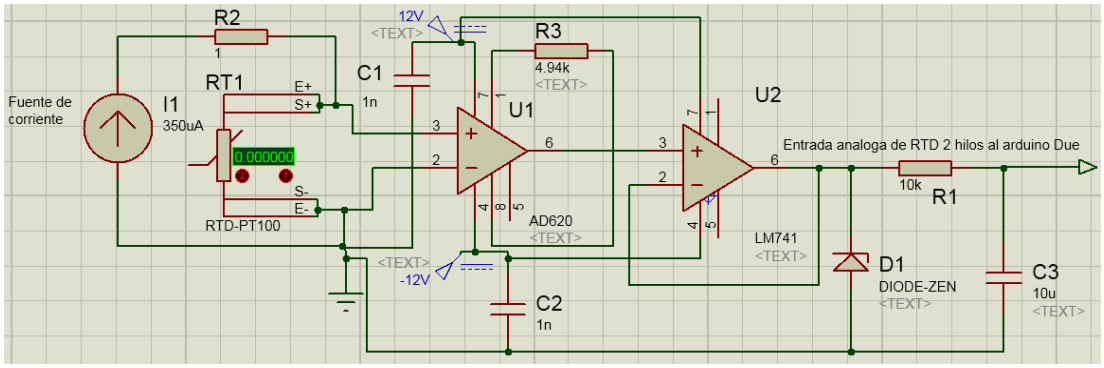


Figura 25: Diagrama esquemático del acondicionamiento de RTD de 2 hilos

RTD de 3 hilos

En la **figura 26** se muestra el circuito que se requiere para realizar la medición de una RTD de 3 hilos.

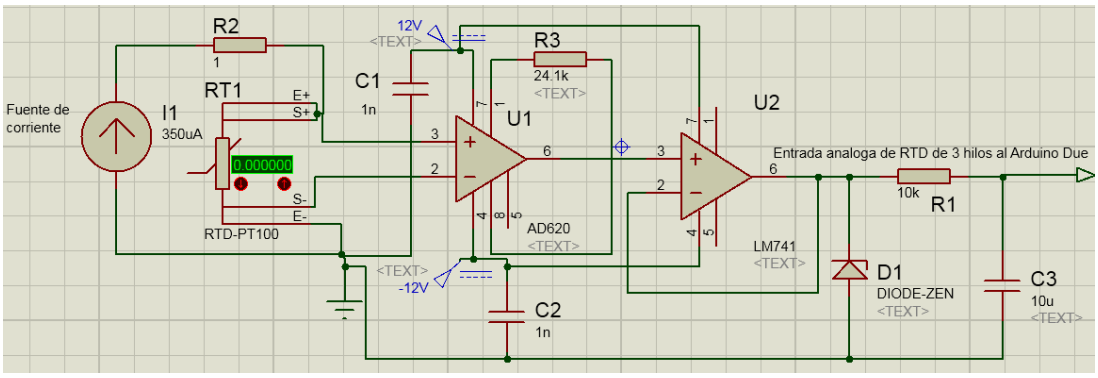


Figura 26: Diagrama esquemático del acondicionamiento de RTD de 3 hilos

RTD de 4 hilos

En la **figura 27** se puede apreciar la conexión necesaria para realizar la medición de una RTD de 4 hilos.

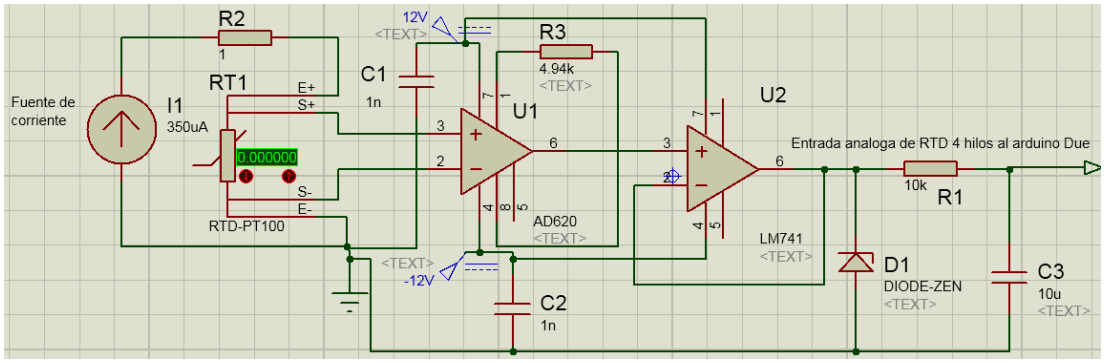


Figura 27: Diagrama esquemático del acondicionamiento de RTD de 4 hilos

b.1Medición de Termocupla

Para realizar la medición de las termocuplas se utilizó el circuito integrado MAX31856 que se comunica por SPI con el Arduino Due, el voltaje que genera la termocupla ingresa al integrado y se lo amplifica, ya que produce valores en el orden de los milivoltios, la señal se la digitaliza con un ADC de

19 bits, al valor obtenido se lo compensa con un sensor de temperatura que posee el integrado ya que realiza la compensación de la unión fría, en la siguiente etapa se selecciona el tipo de termocupla que está ingresando al integrado, dicho valor de medición se lo envía de manera digital al arduino due, esta explicación se muestra en la **figura 28.**

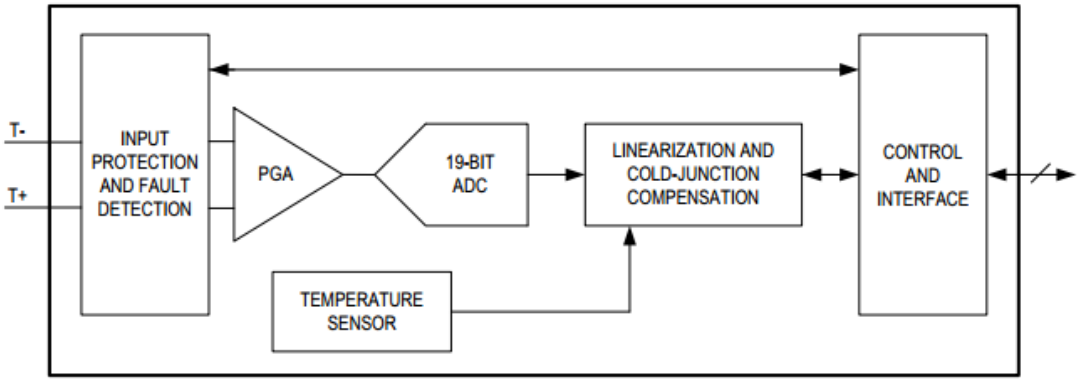


Figura 28: Diagrama de medición de termocupla del integrado MAX31856.

En la **figura 29** se muestra la conexión del integrado MAX31856 con el arduino para el envío de los datos de forma digital para la medición de varias termocuplas.

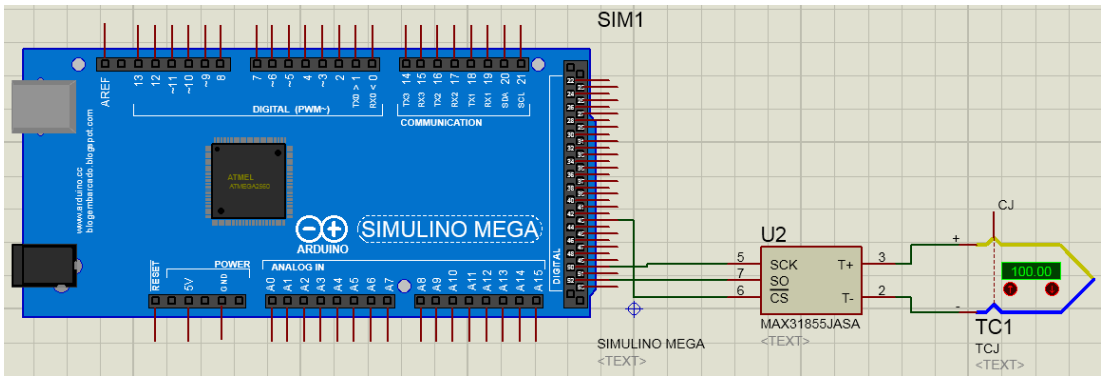


Figura 29: Diagrama esquemático del circuito de medición de termocupla.

c. Conversión Análoga Digital

También llamada digitalización, consiste en realizar medidas de amplitud de la señal de entrada, siempre de forma periódica y traducirlas a un lenguaje binario, en la **figura 30** se puede apreciar el proceso para convertir una señal análoga en señal digital.

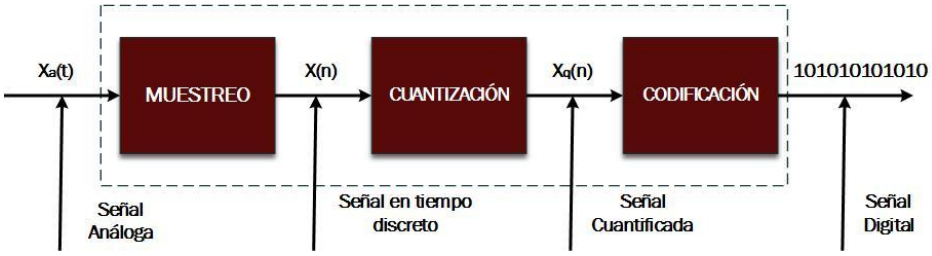


Figura 30: Proceso de conversión Análogo Digital

La tarjeta Arduino Due posee una gran capacidad para realizar la conversión Análogo/Digital, ya que la capacidad de muestreo esta entre los 15 y 100 ksps (kilo muestras por segundo), además tiene 12 entradas análogas conectadas a un conversor análogo digital de 12 bits. Esto significa que convierte tensiones de 0 a 3.3V que soportan los pines de entrada a un número entero 0 y 4095.

El conversor análogo digital recibe señales eléctricas con las que se trabaja en este proyecto y después las convierte en señales digitales, para ejecutar esta tarea tiene que realizar los procesos de: muestreo, cuantización, codificación.

c.1 Muestreo

Para convertir una señal analógica en digital el primer paso consiste en tomar diferentes muestras de tensiones o voltajes en diferentes puntos de la

señal analógica. La frecuencia con la cual se realiza el muestreo se denomina razón, tasa o frecuencia de muestreo y se la mide en kilohertz (Khz).

Para garantizar la toma de muestras y la conversión de forma correcta se debe considerar la velocidad de muestreo, el teorema de Nyquist establece que la frecuencia de muestreo fs, debe ser como mínimo el doble que el ancho de banda de la señal muestreada. Si esto no se cumple, aparece el efecto aliasing que es la interposición de muestras.

$$fs > 2 * fm$$

c.2 Cuantización

La cuantización se encarga de convertir una sucesión de muestras de amplitud continua en una sucesión de valores discretos preestablecidos, es decir, representa el componente de muestreo de los diferentes niveles de amplitud que contiene la señal analógica original, que permite medirlos y asignarles sus correspondientes valores en el sistema decimal antes de convertir dichos valores al sistema binario, en consecuencia se puede decir que el muestreo representa el tiempo de captura de una señal y la cuantización es el componente de la amplitud del muestreo.

c.3 Codificación

Los valores muestreados se representan numéricamente por medio de códigos establecidos, la codificación consiste en representar los diferentes niveles de amplitud cuantificada en un código numérico binario, la exactitud de la medida digitalizada depende de la capacidad de bits que posee el conversor ADC.

3.1.2 Etapa de generación y simulación de variables físicas

En esta etapa se explica detalladamente el funcionamiento de cada uno de los integrados que son utilizados en la generación – simulación de las variables eléctricas, también se explica los circuitos necesarios para realizar estas funciones.

a. Generación de voltaje

Para efectuar la generación de voltaje se utilizó el circuito integrado AD5422 el cual se comunica mediante SPI con un arduino mega, este integrado posee un DAC de 12 a 16 bits el valor de dicha DAC se almacena en un buffer y se escala para producir una gama de escalas de voltajes tales como 0-5 V, 0-10 V, los cuales son seleccionables mediante software, por último el amplificador de salida de tensión es capaz de generar voltajes de salida unipolares y bipolares, este circuito es capaz de soportar una carga de hasta 1 kilo-ohmio (con compensación externa), lo anteriormente explicado se muestra en la **figura 31**.

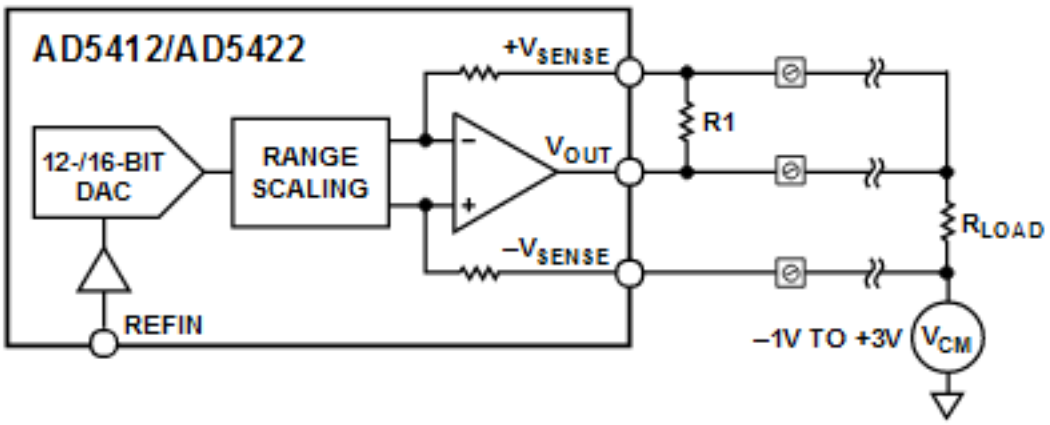


Figura 31: Diagrama de generación de voltaje del AD5422.

En la **figura 32** se muestra el circuito de conexión externa para garantizar el correcto funcionamiento del integrado para la generación de voltaje.

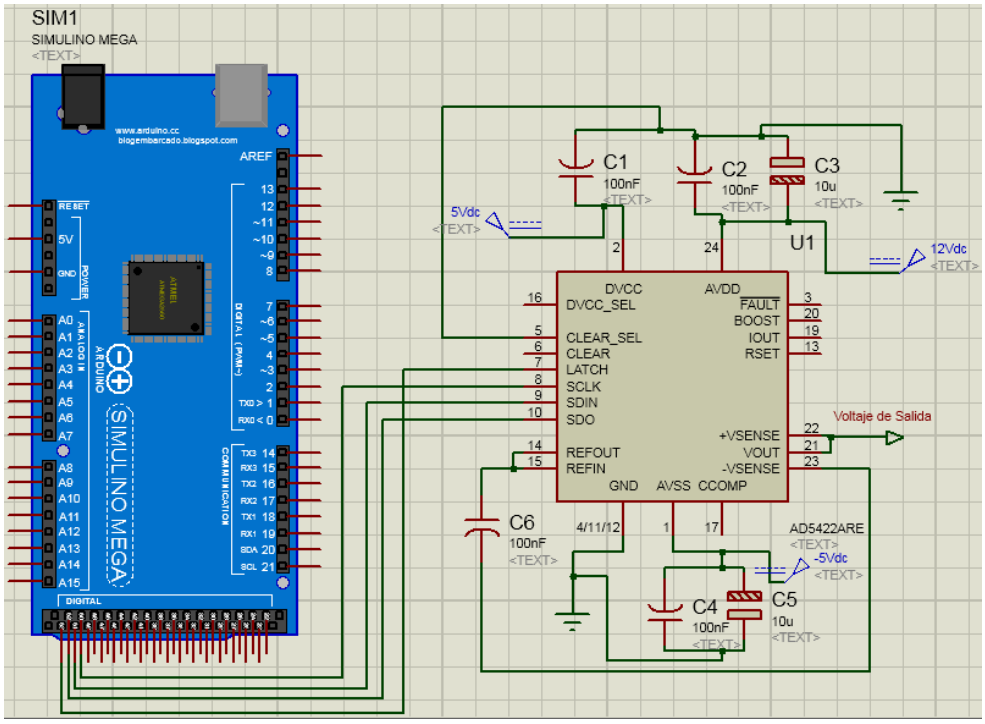


Figura 32: Diagrama esquemático de generación de voltaje.

b. Generación de corriente

El mismo integrado AD5422 se utilizó para realizar la generación de corriente ya que este posee otro circuito con 2 amplificadores, el primero es un seguidor de tensión, en el segundo amplificador ya que tiene retroalimentación negativa, se produce un cortocircuito virtual, generando el voltaje requerido el cual pasa por una resistencia de valor pequeño, cabe recalcar que existe varios rangos de corriente como: 0-20 mA, 4-20 mA, etc, los cuales pueden ser seleccionados mediante software, en la **figura 33** se observa el circuito encargado de generar los rangos de corriente del integrado antes mencionado.

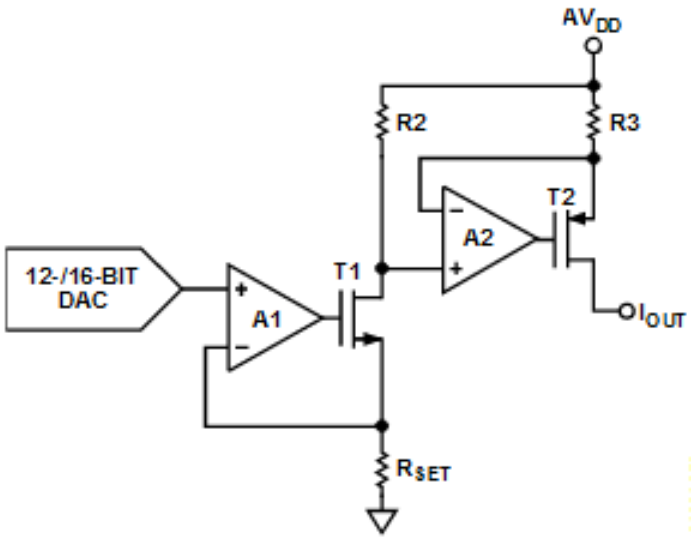


Figura 33: Diagrama de generación de corriente del circuito integrado AD5422.

En la **figura 34** se muestra el circuito de conexión externa para garantizar el correcto funcionamiento del integrado para la generación de corriente.

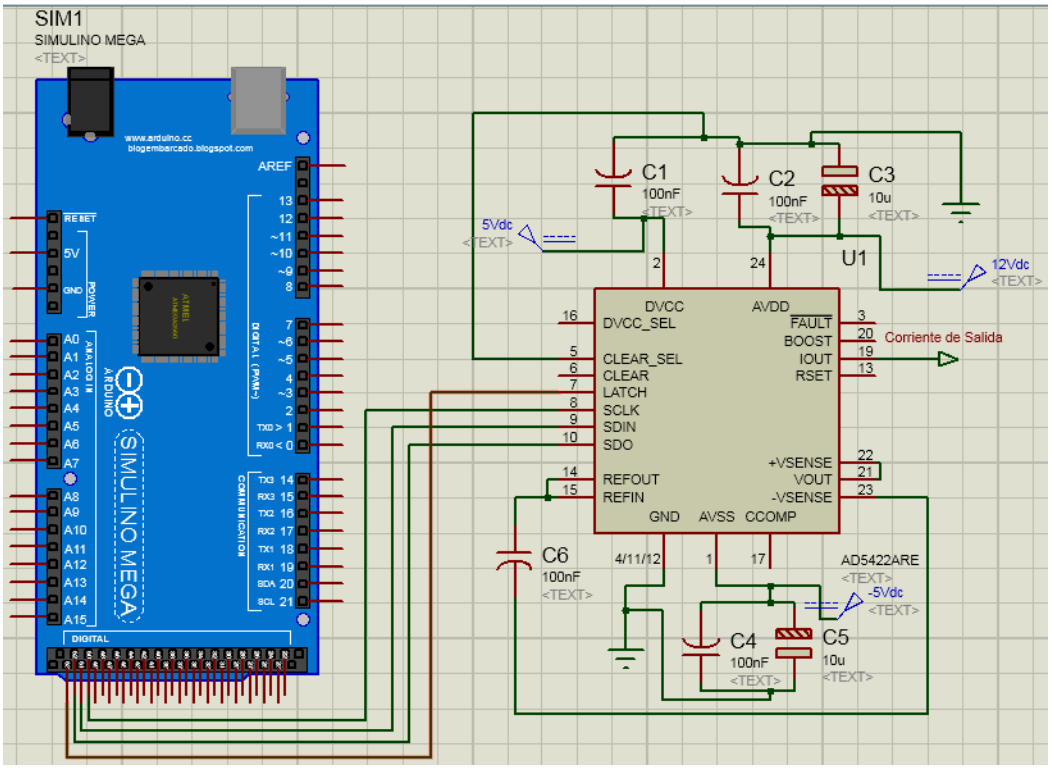


Figura 34: Diagrama esquemático del circuito de generación de corriente.

c. Generación de resistencia

Para realizar la generación de resistencia se utilizó el AD5321 que es un integrado el cual tiene una resistencia inicial de 15 ohmios, además posee varias cadenas de segmentos de resistencia conectadas a una matriz de conmutadores conectados al resistor variable. El integrado posee una resolución de 1024 posiciones, es decir 1024 conexiones, posee una resistencia inicial de 15 ohmios el cual aparecerá al escribir el dato 1023, con la variación de un bit el valor de resistencia del integrado sube 9,7 ohmios, en la **figura 35** se muestra el diagrama de conexión de conmutadores del integrado.

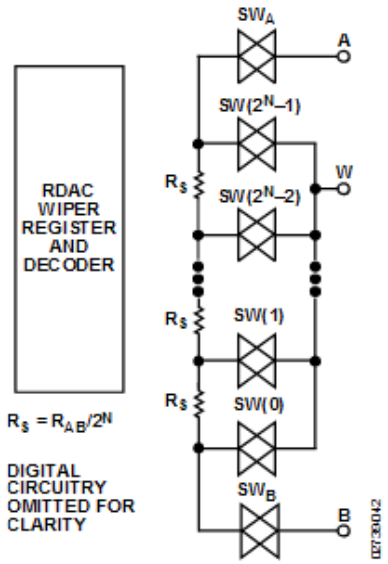


Figura 35: Diagrama de conexión de conmutadores del AD5321.

En la **figura 36** se muestra el circuito de conexión externa de 3 integrados AD5321 ya que al ser la variación de 9,7 ohmios por cada bit los integrados están conectados en paralelo por lo cual mediante el envío de distintos datos a los 3 integrados se obtendrá una resistencia resultante del paralelo de los 3 valores.

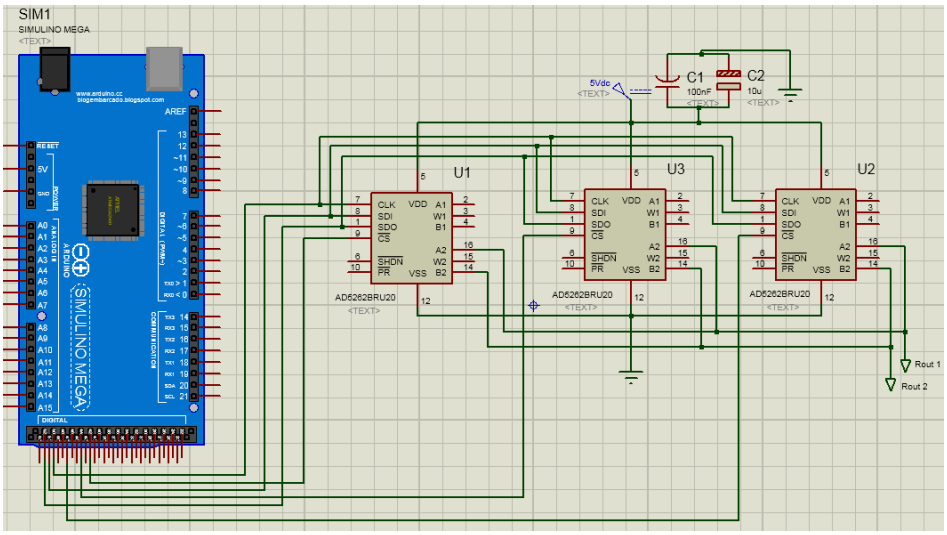


Figura 36: Diagrama esquemático del circuito de generación de resistencia.

d. Simulación de RTD de 2 hilos

Para realizar la generación de RTD de 2 hilos se utilizó los mismos integrados y la misma conexión del circuito ya que al estar conectados en paralelo existe 1070599167 posibles valores de resistencia entre los cuales estarán los valores de RTD de una PT100, en la **figura 37** se muestra el circuito de conexión externa de los integrados para la generación de RTD.

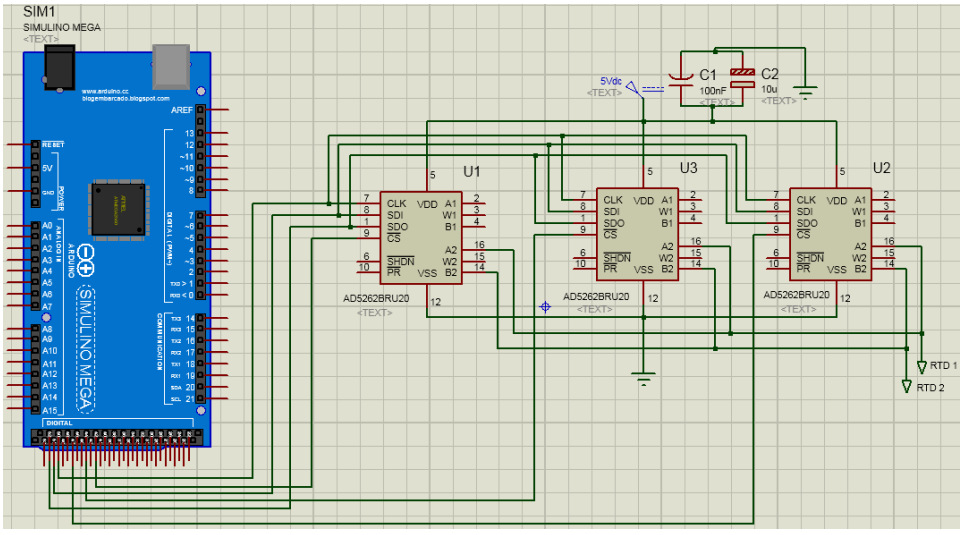


Figura 37: Diagrama esquemático del circuito de generación de RTD de 2 hilos.

e. Simulación de termocupla

Para desarrollar la simulación de las termocuplas se utilizó el mismo integrado y la misma conexión externa que la generación de corriente, en la **figura 38** se muestra el diagrama de conexión externo para el correcto funcionamiento de la simulación de valores de termocupla.

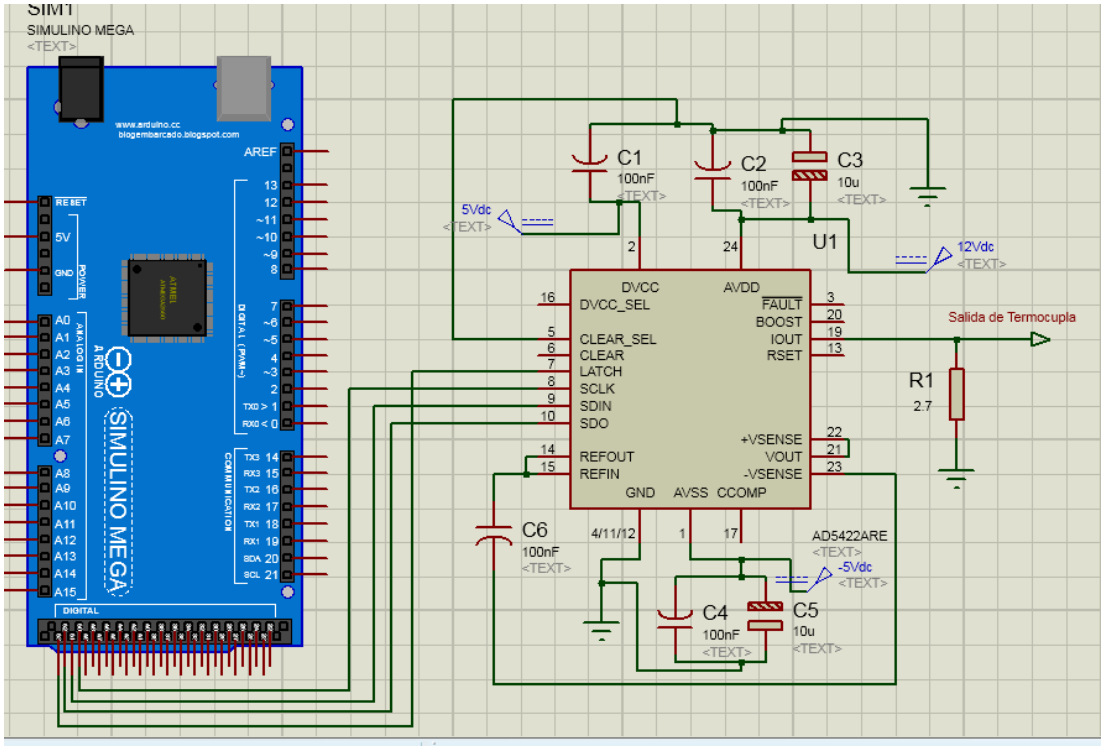


Figura 38: Diagrama esquemático del circuito de generación de termocupla.

Ya que el AD5422 tiene una DAQ de 16 bits varía 0,0763 mV por cada variación de un bit, se conecta una resistencia de un valor de 2.7 ohmios, a partir de ahí al variar los valores de la fuente de corriente el voltaje que existe en la resistencia también varía en el rango de los milivoltios que es el orden en el cual varían los valores de la termocupla.

f. Comunicación I2C entre arduino DUE y MEGA

La etapa de generación y simulación de variables físicas se la realiza en el Arduino Mega, pero todo el envío y recepción de información la gestiona el Arduino Due.

Los arduinos DUE y MEGA envían y transmiten los datos necesarios mediante el protocol de comunicación I2C, pero ya que operan a distintos valores de voltaje (3.3 V y 5 V respectivamente) es necesario realizar un circuito que convierta los valores de 3.3 V a 5 V y viceversa, los MOSFET utilizados son de canal N de altas frecuencias y baja corriente condiciones que son necesarias para que funcione la comunicación, en la **figura 39** se muestra el circuito de conversión de niveles para el correcto funcionamiento de la comunicación I2C,

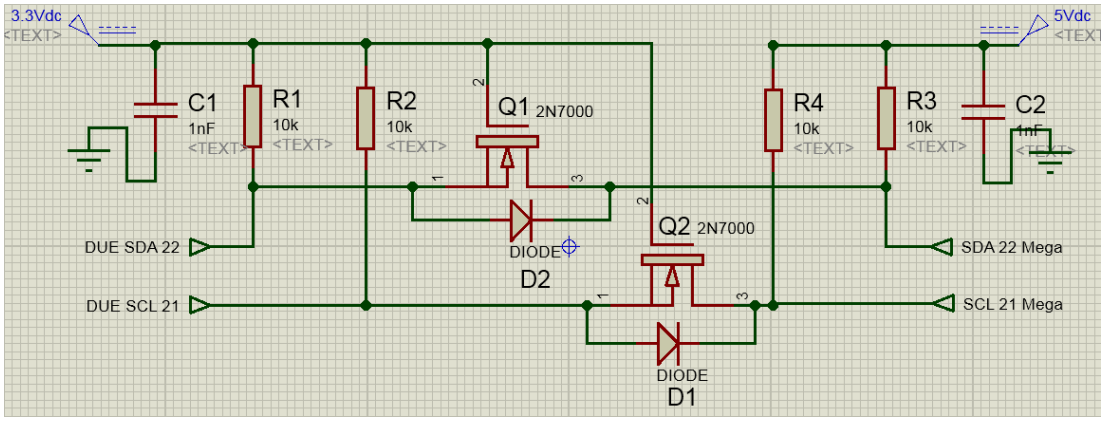


Figura 39: Circuito de conversión de niveles de 3.3V a 5V y viceversa.

3.2 Diseño del Software

Mediante diagramas de flujo se explica la programación realizada para medir generar y simular las variables físicas del prototipo implementado.

3.2.1 Entorno de desarrollo arduino

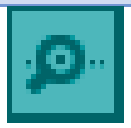
El entorno de desarrollo Arduino **figura 40**, está compuesto por un editor de texto en el cual se puede escribir el código de programación, una barra de herramientas con varios botones de las funciones más comunes, un área de mensajes, una consola de texto, además de una barra de menús los cuales tienen entre sus opciones: guardar un programa, escoger la tarjeta con la cual trabajar, etc.

Figura 40: Entorno de desarrollo de Arduino.

Para escribir el código de programación arduino utiliza un software denominado “sketch”. Dichos programas son escritos en el editor de texto,

existe la posibilidad de cortar/pegar y buscar/reemplazar texto. En el área de mensajes se muestra información mientras los programas se cargan en la placa y también muestra los errores de haberlos. La consola muestra el texto de salida para el entorno de Arduino incluyendo los mensajes de error completos y ciertas informaciones más. La barra de herramientas permite verificar el proceso de carga, creación, apertura y guardado de programas, también inicia la monitorización serial, en la **tabla 6** se observa cada una de las funciones que tiene la barra de herramientas.

Tabla 6:
Funciones del entorno de desarrollo de Arduino.

HERRAMIENTAS	FUNCIÓN
	**Verificar:** Chequea el código en busca de errores
	**Cargar:** Compila el código del programa y lo graba en la placa E/S de Arduino.
	**Nuevo:** Genera un nuevo sketch
	**Abrir:** Presenta un menú de todos los sketch que se alojan en su “sketchbook”, el que se escoja se abrirá en una nueva ventana.
	**Guardar:** Guarda el Sketch con la programación realizada.
	**Monitor Serial:** Abre una ventana en la cual se puede realizar la monitorización serie del arduino.

3.2.2 Librerías

a. AnalogReadResolution( )

Esta librería es una extensión de la API analógica para la tarjeta Arduino Due. Define el tamaño (en bits) del valor devuelto por analogRead(). Por defecto es 10 bits, es decir devuelve valores entre 0 y 1023, esto es necesario para la compatibilidad con placas basadas en AVR.

Sin embargo la tarjeta Arduino Due tiene una capacidad ADC de 12 bits, por lo tanto esta tarjeta tiene la capacidad de cambiar la resolución a 12. En consecuencia devuelve valores de analogRead( ) entre 0 y 4095.

b. AnalogWriteResolution( )

Esta librería es una extensión de la API analógica para la tarjeta Arduino Due. Se encarga de establecer la resolución de la función analogWrite(). La cual por defecto es de 8 bits, es decir valores entre 0 y 255.

La tarjeta posee 12 pines para PWM que vienen configurados por defecto de 8 bits, estos pueden ser cambiados a una resolución de 12 bits, además cuenta con 2 pines de conversión digital a análoga también de 12 bits. Al establecer la resolución de escritura en 12 bits, la función analogWrite() puede ser usada con valores entre 0 y 4095 para aprovechar la máxima resolución del conversor.

c. WIRE

Esta librería permite que el arduino pueda realizar la comunicación I2C, mediante los pines SDA (línea de datos) y SLC (línea de reloj) que corresponden a los pines 20 y 21 respectivamente del arduino DUE y MEGA, la librería wire utiliza direcciones de 7 bits mientras que el octavo bit determina si el dispositivo está siendo escrito o leído.

d. SPI

Esta librería permite la comunicación SPI entre varios dispositivos periféricos de forma rápida y a cortas distancias con el arduino como maestro, en esta comunicación serial siempre existe un maestro que generalmente es un microcontrolador y tiene 3 líneas las cuales realizan la comunicación:

- **MISO (Master In Slave Out)** Esta línea sirve para enviar datos al maestro desde los periféricos.
- **MOSI (Master Out Slave In)** Esta línea sirve para enviar datos a los periféricos desde el maestro.
- **SCK (Serial Clock)** Esta línea emite los pulsos de reloj para que sirve para sincronizar la comunicación de datos.

3.2.3 Programación Principal

El diagrama de flujo de la **figura 41** explica de forma detallada la programación realizada para la medición, generación de las variables y de la comunicación con la PC para el certificado de calibración, la programación del arduino se encuentra en el ANEXO A.

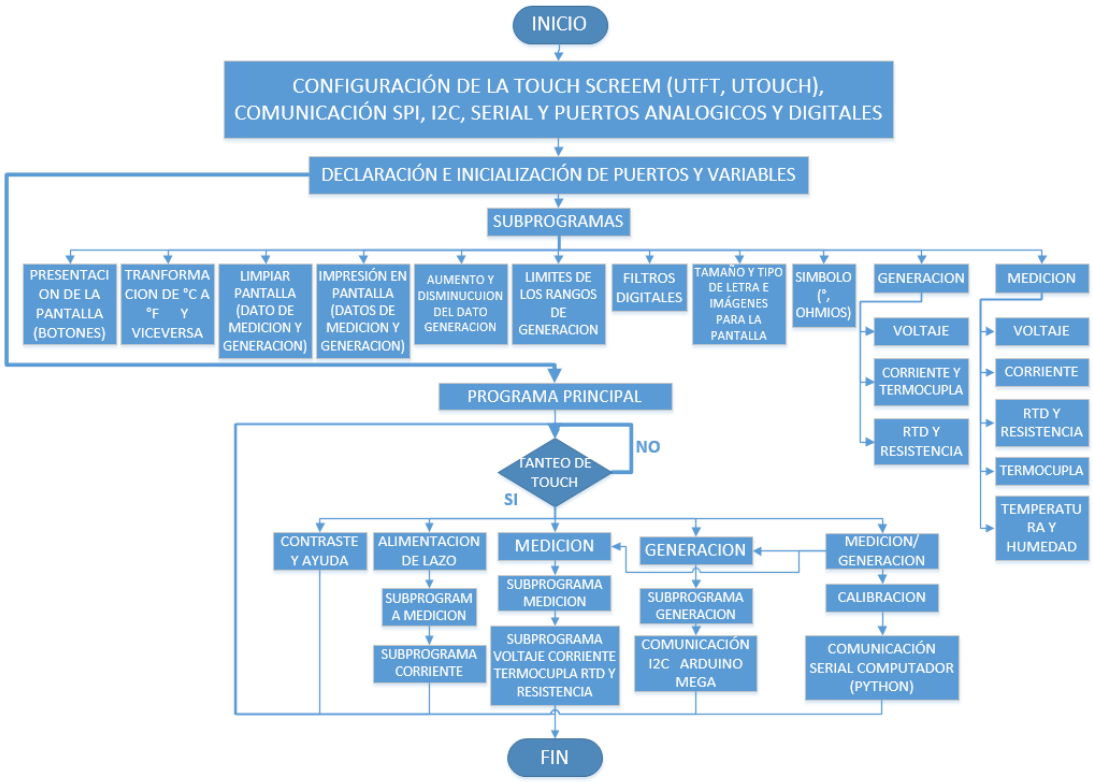


Figura 41: Diagrama de flujo del programa principal del prototipo.

3.2.4 Etapa de Medición

En la etapa de medición de las variables eléctricas la programación es parecida ya que para todas las mediciones lo que ingresa al arduino es voltaje, pero hacia un canal análogo distinto de la tarjeta.

a. Medición de Voltaje

Para realizar la medición de voltaje se siguió el diagrama de flujo mostrado en la **figura 42**, por consiguiente la programación realizada en el arduino se puede revisar en el ANEXO A.

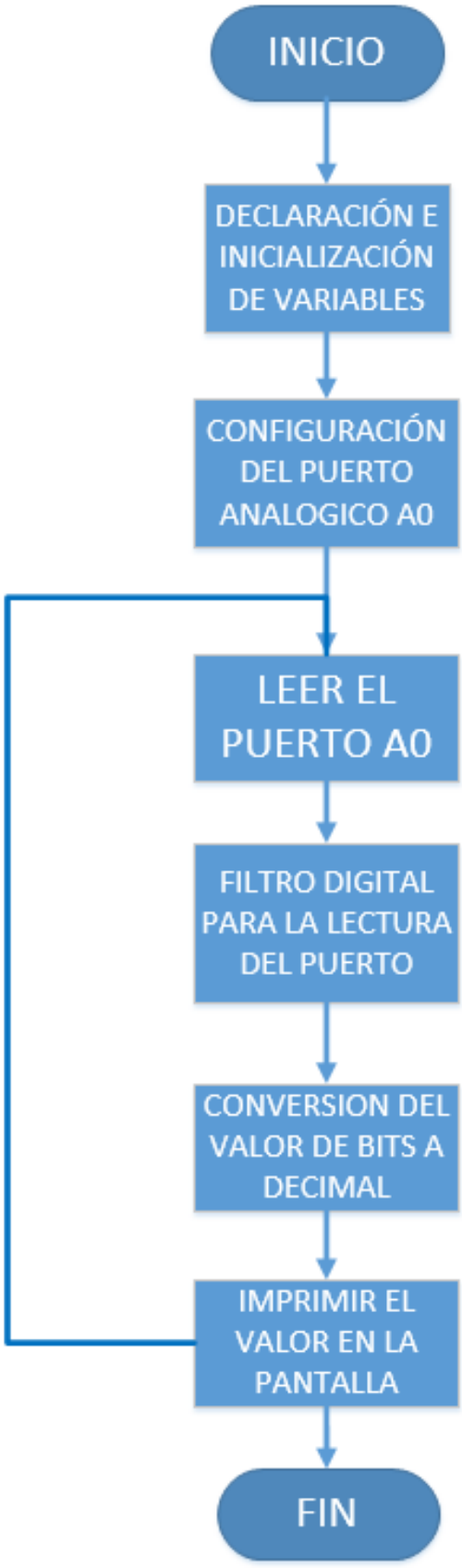


Figura 42: Diagrama de flujo para la medición de voltaje.

b. Medición de Corriente

El diagrama de flujo de la **figura 43**, explica los pasos a seguir para realizar la programación necesaria de la medición de corriente, dicha programación se la puede observar en el ANEXO A.

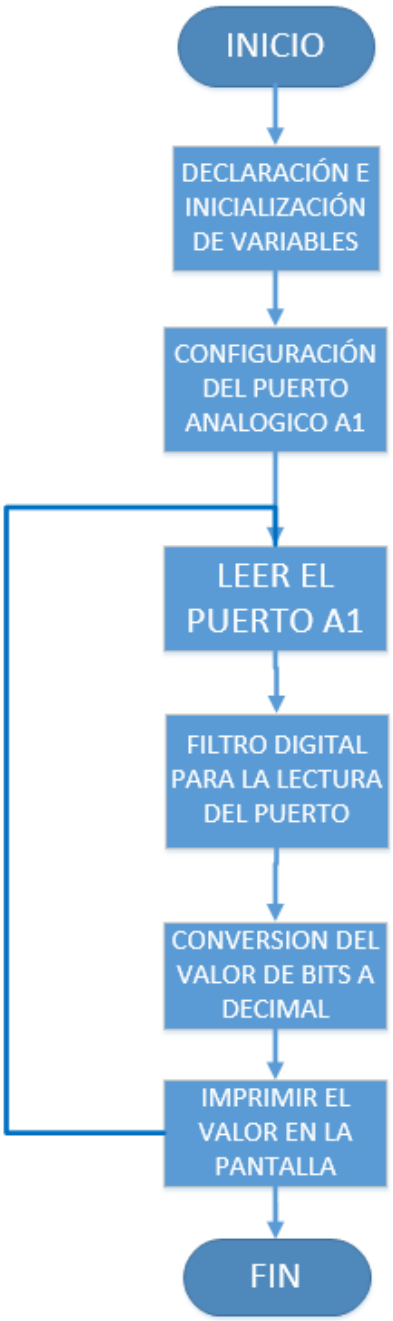


Figura 43: Diagrama de flujo para la medición de corriente.

c. Medición de Resistencia

El diagrama de flujo mostrado en la **figura 44**, indica la programación de la medición de resistencia, la entrada analógica es tratada en el arduino y se obtiene una medida y unidad real con un porcentaje de error el cual será detallado en el capítulo 4, la programación para la medición de resistencia se puede revisar en el ANEXO A.

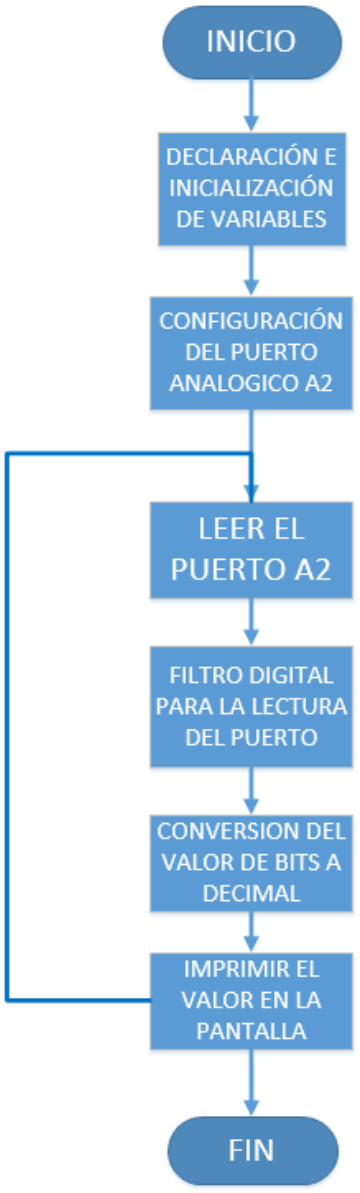


Figura 44: Diagrama de flujo para la medición de resistencia.

d. Medición de RTD de 2, 3 y 4 hilos

Para realizar la medición de RTDS de 2, 3 y 4 hilos se siguió el diagrama de flujo mostrado en la **figura 45**, la entrada analógica es tratada en el arduino y se obtiene una medida y unidad real con un porcentaje de error el cual será detallado en el capítulo 4, la programación para la medición de resistencia se puede revisar en el ANEXO A.

Figura 45: Diagrama de flujo para la medición de RTDS de 2, 3 y 4 hilos.

e. Medición de Termocupla

El diagrama de flujo (**figura 46)** indica los pasos a seguir en el software para realizar la medición de termocuplas tipo J y K, la programación se realiza en el arduino se puede revisar en el ANEXO A

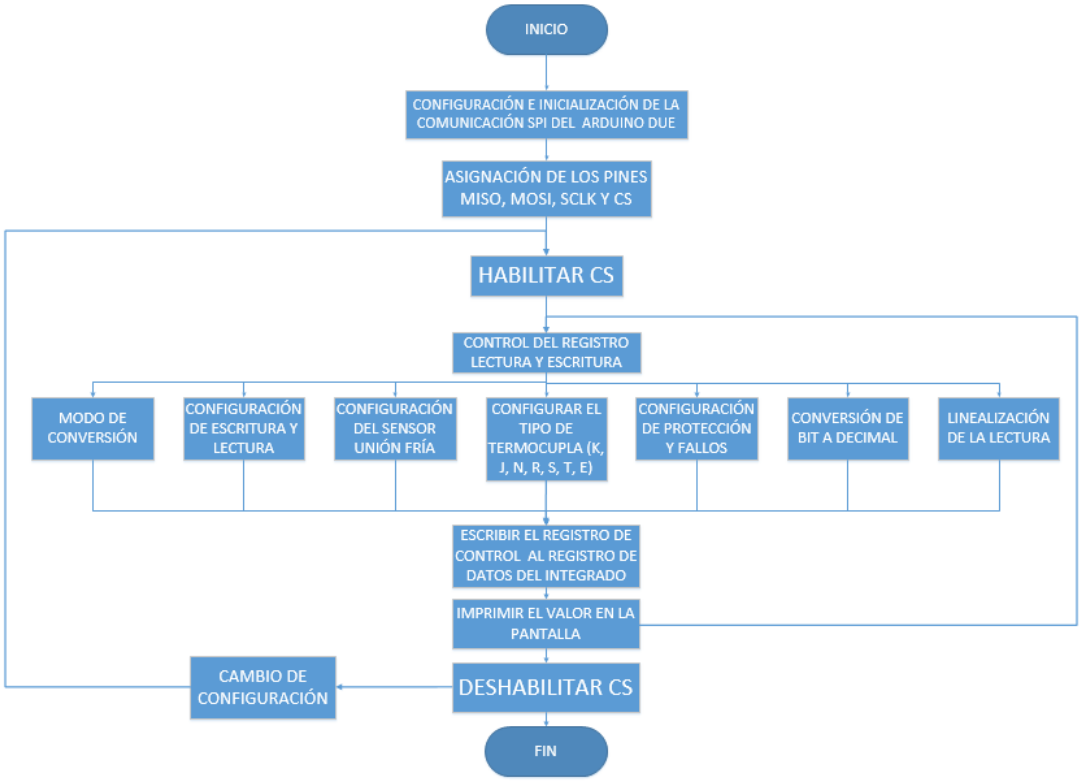


Figura 46: Diagrama de flujo para la medición de varios tipos de termocuplas.

3.2.5 Etapa de Generación - Simulación

En la etapa de generación - simulación de variables eléctricas la programación es distinta ya que para simular RTDS el programa maneja siempre tres integrados, mientras que para simular termocuplas únicamente maneja uno solo.

a. Generación de Voltaje

El diagrama de flujo de la **figura 47**, muestra la programación realizada para la generación de voltaje, configurando el software se puede decidir entre varios rangos de voltaje, la programación diseñada en el arduino se puede revisar en el ANEXO B.

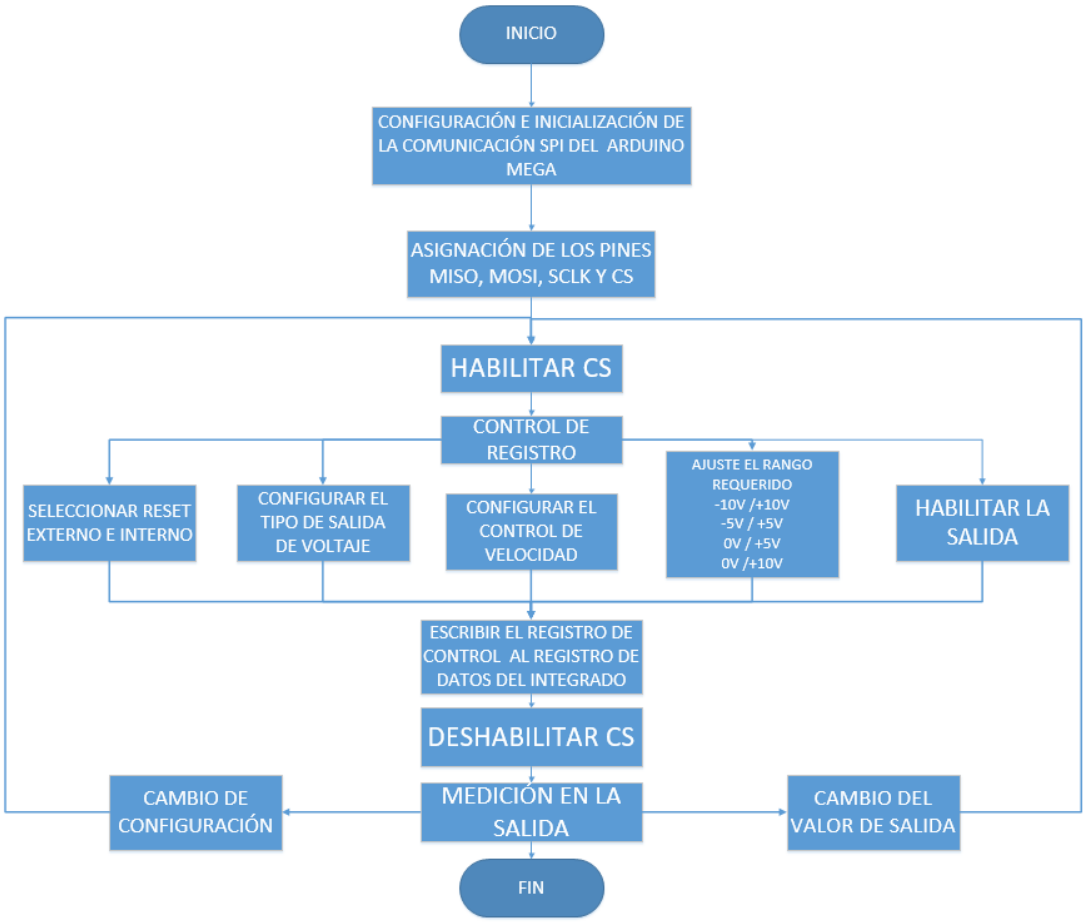


Figura 47: Diagrama de flujo para la generación de voltaje.

b. Generación de Corriente

El diagrama de flujo de la **figura 48**, muestra la programación realizada para la generación de corriente, configurando el software se puede decidir

entre varios rangos de corriente, la programación diseñada en el arduino se puede revisar en el ANEXO B.

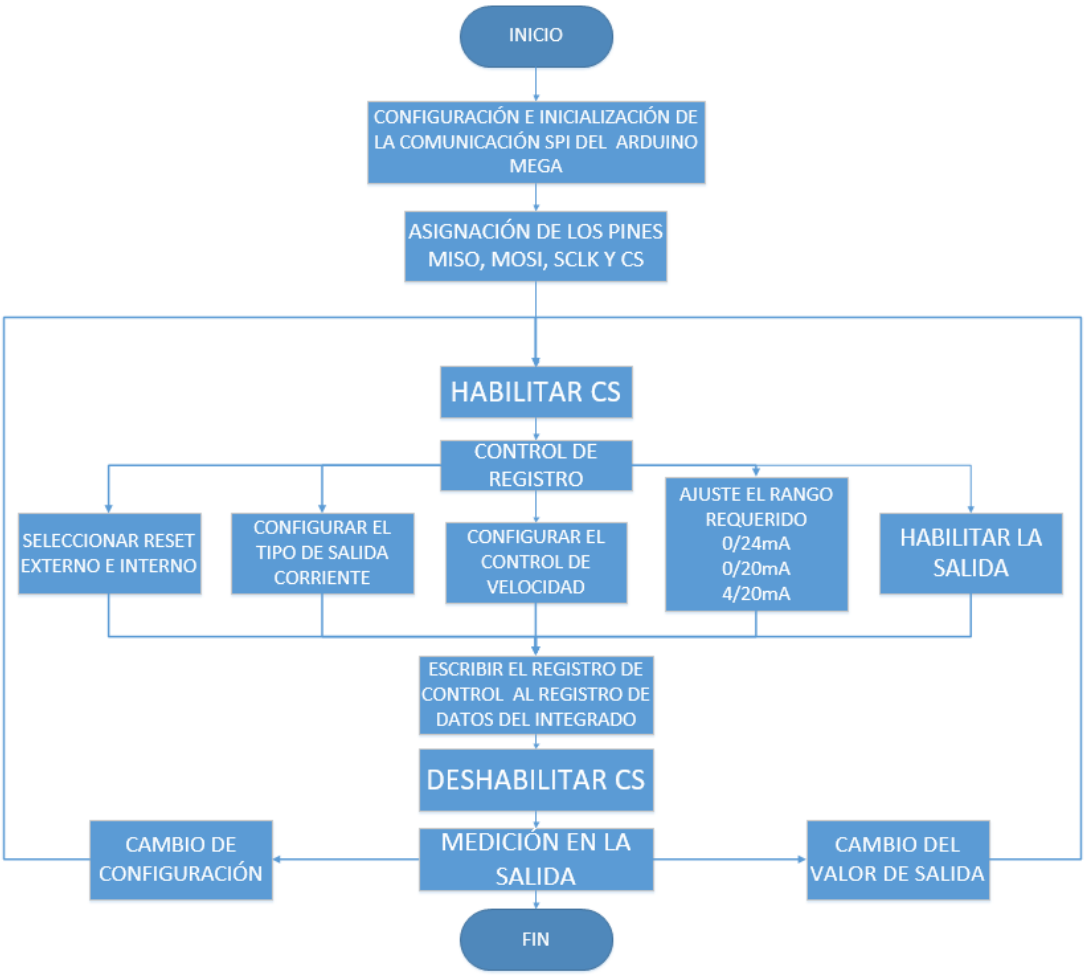


Figura 48: Diagrama de flujo para la generación de corriente.

c. Generación de Resistencia

El diagrama de flujo de la **figura 49**, muestra la programación realizada para la generación de resistencia, mediante la programación en software se puede decidir el dato en bits entre 0 y 1023, ya que los integrados poseen una variación de 9,8 ohmios es necesario enviar distintos datos hacia los integrados y de esta manera realizar el paralelo entre tres circuitos

integrados, la programación diseñada en el arduino se puede revisar en el ANEXO B.

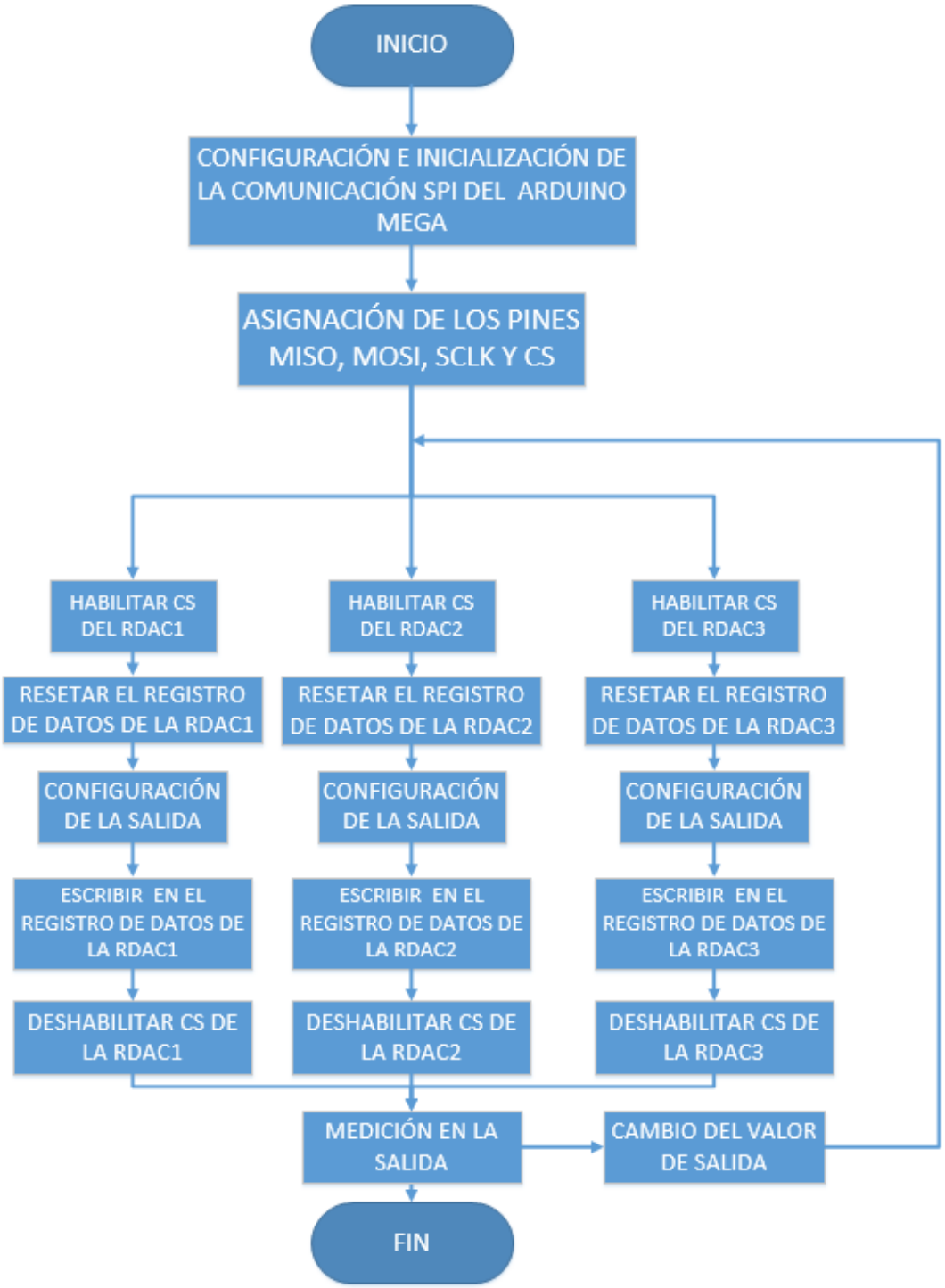


Figura 49: Diagrama de flujo para la generación de resistencia.

d. Simulación de RTD de 2 hilos

Para realizar la generación de RTDS de 2 hilos se utilizó el mismo diagrama de flujo de la generación de resistencia como se muestra en la **figura 50**, con la diferencia de que la variación de bits para cada circuito integrado es menor ya que la variación de resistencia también es menor, la programación diseñada en el arduino se puede revisar en el ANEXO B.

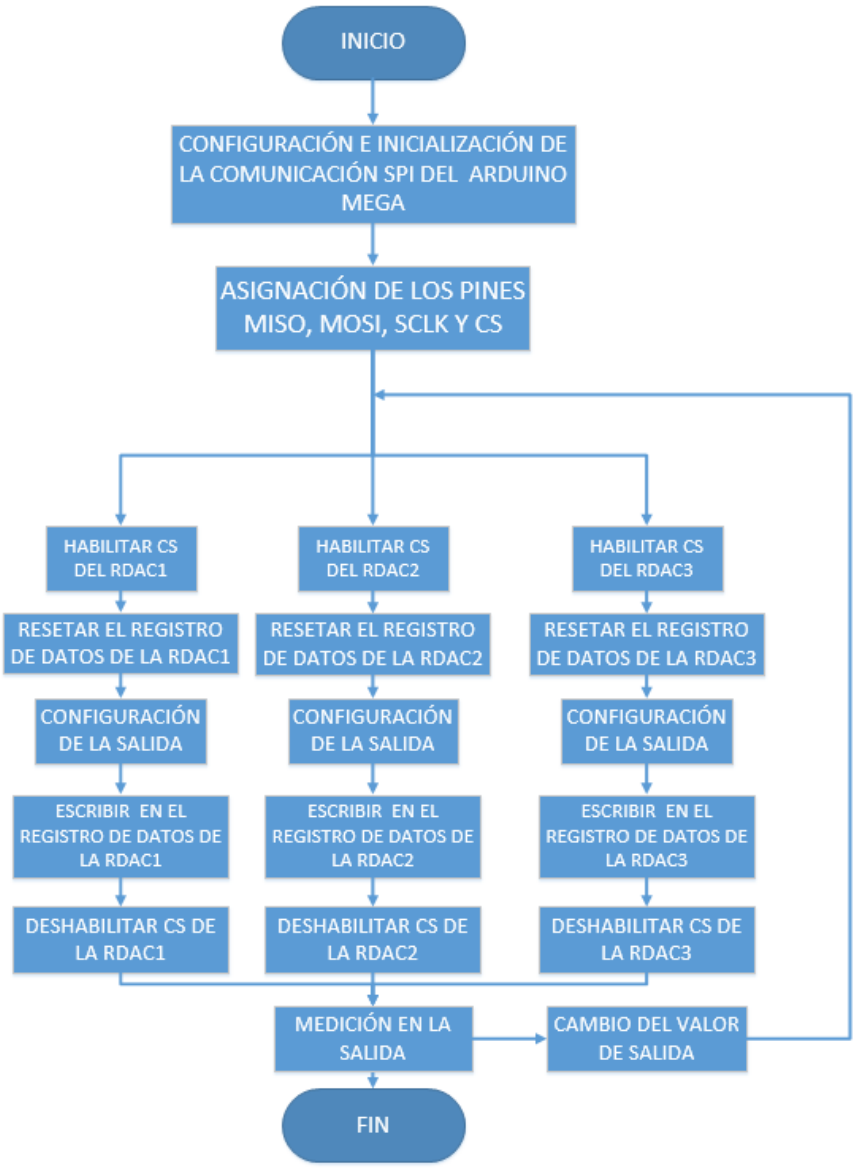


Figura 50: Diagrama de flujo para la generación de RTD de 2 hilos.

e. Simulación de Termocupla

Para realizar la generación de termocupla se siguió el mismo diagrama de flujo de la generación de corriente como se muestra en la **figura 51**, ya que la variación de corriente que se envía por una resistencia de valor pequeño generará una variación de voltaje el cual se utilizará para realizar la simulación de varias termocuplas, la programación diseñada en el arduino se puede revisar en el ANEXO A.

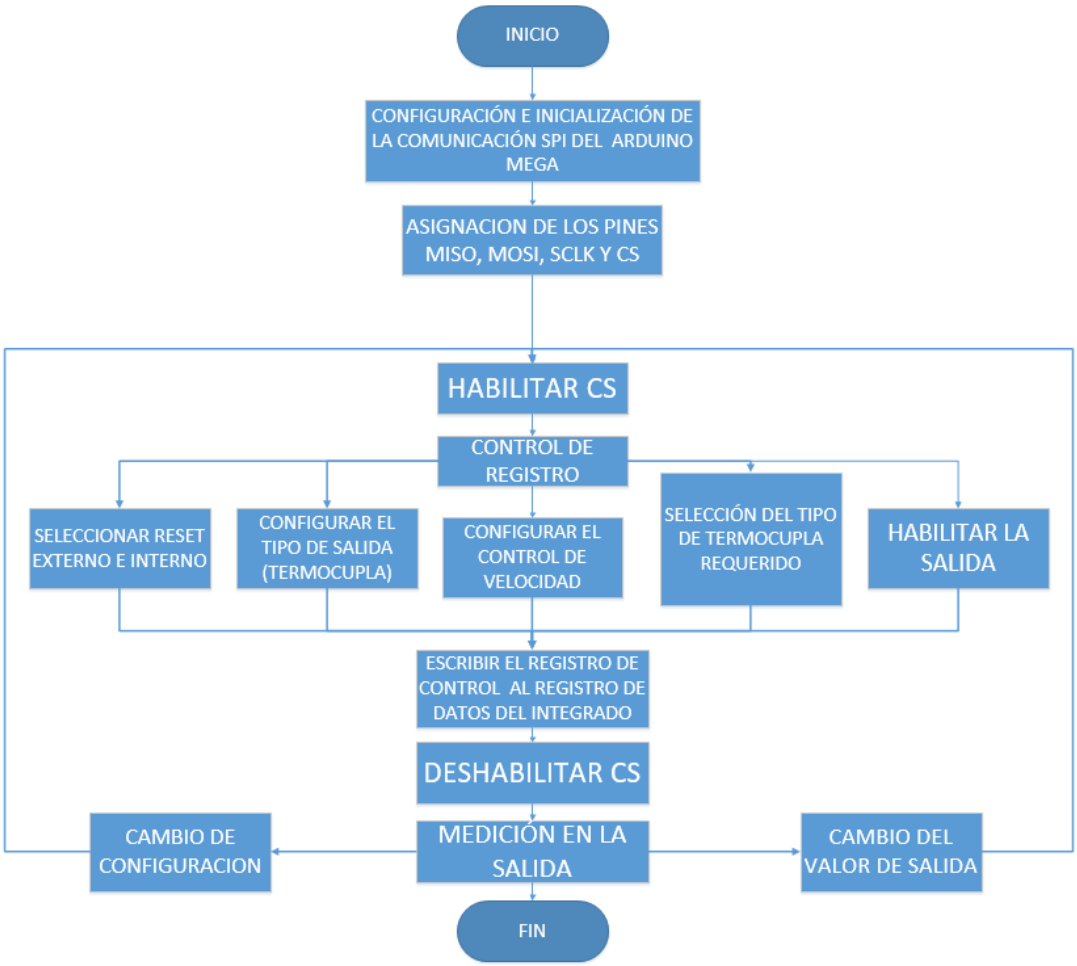


Figura 51: Diagrama de flujo para la generación de termocupla.

3.2.6 Etapa de comunicación con la PC.

En esta etapa el prototipo se conecta con la PC mediante comunicación serial, enviando de esta forma los datos de calibración para realizar el documento de calibración.

a. Programación para la Generación del Documento de Calibración.

Para realizar la generación del documento de calibración se siguió el diagrama de flujo que se muestra en la **figura 52**, en el cual se importa varias librerías necesarias para que el software funcione de manera correcta, después se declara el puerto COM por el cual el arduino está conectado con la computadora, se obtiene los datos enviados desde el arduino, el programa realizado mediante códigos determina la fecha y hora con la que se realizó la calibración y genera un documento con extensión .pdf, la programación realizada en phyton se la puede revisar en el ANEXO C.

Figura 52: Diagrama de flujo de la generación del documento de calibración.

3.2.7 Etapa de monitoreo y control de variables

Esta etapa es la encargada de monitorear, generar y simular las variables eléctricas del equipo, mediante la pantalla táctil.

a. Programación de la pantalla táctil TFT

El diagrama de flujo de la **figura 53** explica de manera detallada el procedimiento para poder manejar correctamente todos los recursos de la pantalla táctil del equipo, dicha programación se encuentra en el ANEXO A.

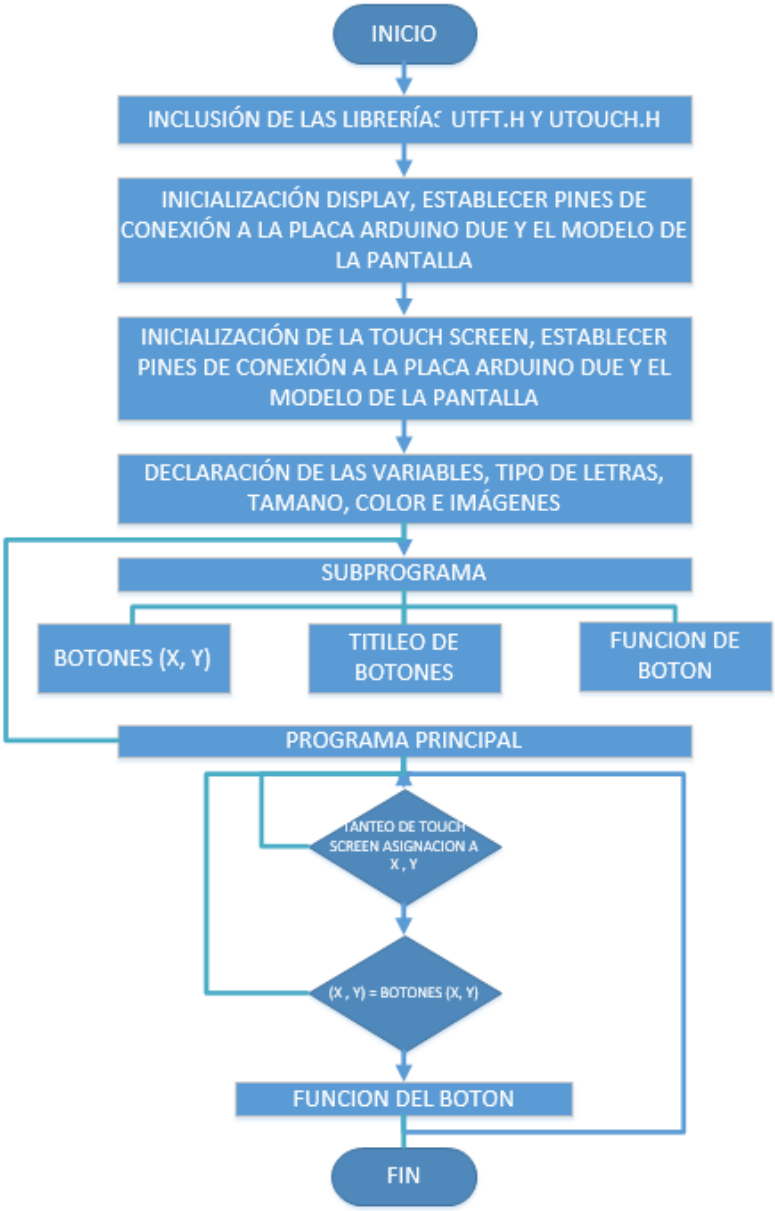


Figura 53: Diagrama de flujo del manejo de la pantalla táctil.

CAPÍTULO IV

4. PRUEBAS Y ANÁLISIS DE RESULTADOS

En este capítulo se describen las pruebas realizadas al prototipo construido para, dichas pruebas serán realizadas en el Laboratorio de Redes Industriales y Control de Procesos de la Universidad de las Fuerzas Armadas ESPE extensión Latacunga de acuerdo a los valores de la **Tabla 7**, para lo cual se ha definido los rangos de generación y medición de cada una de las variables.

Tabla 7:
Rangos de medición y generación de las Variables

VARIABLE	TIPO DE PRUEBA			
	TIPO	**RANGO**	**TIPO**	**RANGO**
VOLTAJE	MEDICIÓN	0 - 30Vdc	GENERACIÓN	0 – 10 Vdc
CORRIENTE	MEDICIÓN	0 - 24mA	GENERACIÓN	0 – 30 mA
RESISTENCIA	MEDICIÓN	0 – 3.2KΩ	GENERACIÓN	0 – 1 KΩ
RTDS PT100	MEDICIÓN	-30° - 200°C	GENERACIÓN	-30° - 200 °C
TERMOCUPLA	MEDICIÓN	-50° - 500°C	GENERACIÓN	-50° - 500 °C

4.1 Pruebas de Mediciones de Variables

Para realizar las pruebas de medición, generación y simulación de las variables eléctricas del equipo implementado se usó como instrumento patrón únicamente el calibrador FLUKE 724, ya que posee las funciones muy parecidas a las del prototipo.

4.1.1 Pruebas de medición de Voltaje

En la prueba de medición de voltaje se realiza la conexión mostrada en la **figura 54**, para lo cual un valor de voltaje deseado generado por una fuente externa ingresa al prototipo y también es medido por el calibrador Fluke 724, después se obtiene el porcentaje de error con respecto a dicho calibrador ya que este será utilizado como instrumento patrón para realizar las pruebas de voltaje del prototipo construido.

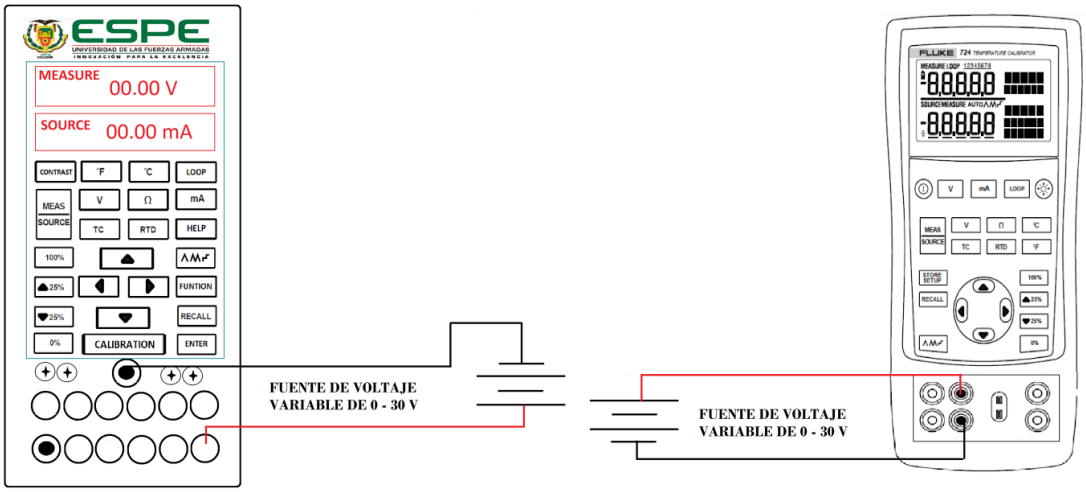


Figura 54: Conexión para la medición de voltaje.

Para determinar el error de medida de las variables físicas del prototipo se aplicó la siguiente fórmula:

$$Error = \frac{V.medido - V.patrón}{Rango\ de\ operación} x100$$

El instrumento patrón es el calibrador Fluke 724 ya que tiene varias certificaciones que aseguran un valor de medida muy confiable y garantiza las pruebas del prototipo implementado.

Tabla 8:
Prueba de medición de voltaje del prototipo

MEDICION DE VOLTAJE DEL CALIBRADOR VS PROTOTIPO				
N. DE PRUEBA	**Valor Deseado**	**CALIBRADOR**	**PROTOTIPO**	**ERROR [%]**
1	0	0,002	0,06	0,19
2	1	1,003	1,04	0,12
3	2	2,002	2,07	0,23
4	3	3,002	3,05	0,16
5	4	4	4,06	0,20
6	6	6	6,08	0,27
7	8	8	8,08	0,27
8	10	10	10,05	0,17
9	12	12	12,06	0,20
10	14	14	14,07	0,23
11	16	16	16,08	0,27
12	18	18	18,07	0,23
13	20	20	20,07	0,23
14	22	22	22,06	0,20
15	24	24	24,08	0,27
16	26	26	26,06	0,20
17	30	30	30,02	0,07
18	26	26	26,06	0,20
19	24	24	24,08	0,27
20	22	22	22,06	0,20
21	20	20	20,07	0,23
22	18	18	18,07	0,23
23	16	16	16,08	0,27
24	14	14	14,07	0,23
25	12	12	12,06	0,20
26	10	10	10,05	0,17
27	8	8	8,08	0,27
28	6	6	6,08	0,27
29	4	4	4,06	0,20
30	3	3	3,05	0,17
31	2	2	2,07	0,23
32	0	0	0,06	0,20
		PROMEDIO		**0,21**
		VARIANZA		**0,00215**
		DESVIACIÓN ESTÁNDAR		**0,046**

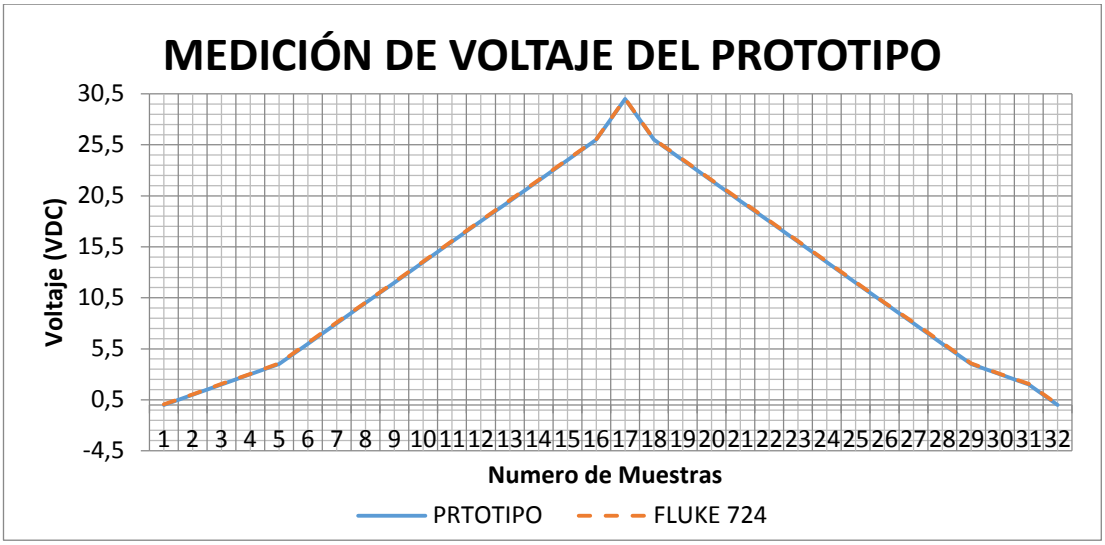


Figura 55: Curva de respuesta de medición de voltaje del prototipo.

4.1.2 Pruebas de medición de corriente

La **figura 56** muestra la conexión necesaria del prototipo implementado para realizar la prueba de medición de corriente, el valor que genere la fuente de corriente deseado será ingresado al prototipo y también al calibrador Fluke 724 y se obtendrá el porcentaje de error de acuerdo a la diferencia entre los valores medidos por el prototipo y el calibrador Fluke, mismo que es utilizado como instrumento patrón para realizar las pruebas de corriente del prototipo.

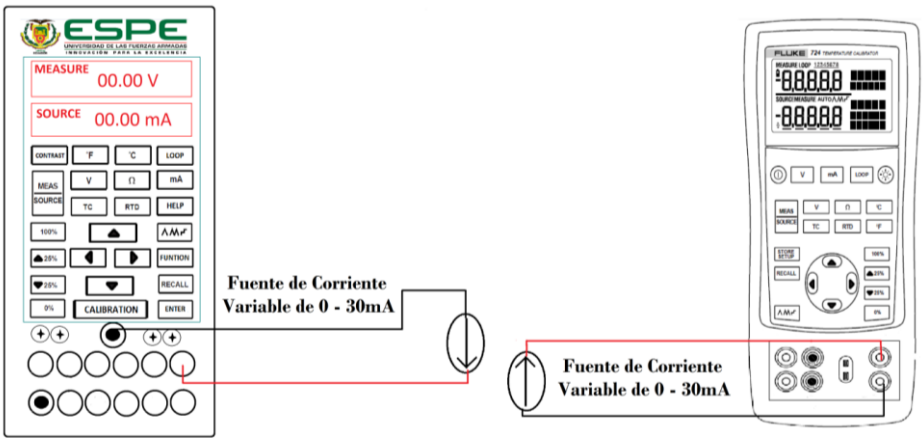


Figura 56: Conexión para la medición de corriente.

Tabla 9:
Prueba de medición de corriente del prototipo

MEDICIÓN DE CORRIENTE DEL CALIBRADOR VS PROTOTIPO				
N. DE PRUEBA	**Valor Deseado**	**CALIBRADOR**	**PROTOTIPO**	**ERROR [%]**
1	0	0,002	0,05	0,20
2	1	1,003	1,04	0,15
3	2	2,002	2,06	0,24
4	3	3,002	3,05	0,20
5	4	4	4,06	0,25
6	5	5	5,06	0,25
7	6	6	6,07	0,29
8	7	7	7,08	0,33
9	8	8	8,05	0,21
10	9	9	9,07	0,29
11	10	10	10,06	0,25
12	12	12	12,04	0,17
13	14	14	14,05	0,21
14	16	16	16,07	0,29
15	18	18	18,08	0,33
16	20	20	20,06	0,25
17	24	24	24,08	0,33
18	20	20	20,08	0,33
19	18	18	18,07	0,29
20	16	16	16,08	0,33
21	14	14	14,07	0,29
22	12	12	12,06	0,25
23	10	10	10,05	0,21
24	9	9	9,07	0,29
25	8	8	8,08	0,33
26	7	7	7,07	0,29
27	6	6	6,08	0,33
28	5	5	5,06	0,25
29	4	4	4,06	0,25
30	3	3,002	3,05	0,20
31	2	2,002	2,07	0,28
32	0	0,002	0,06	0,24
			PROMEDIO	**0,26**
			VARIANZA	**0,0027**
			DESVIACIÓN ESTÁNDAR	**0,0523**

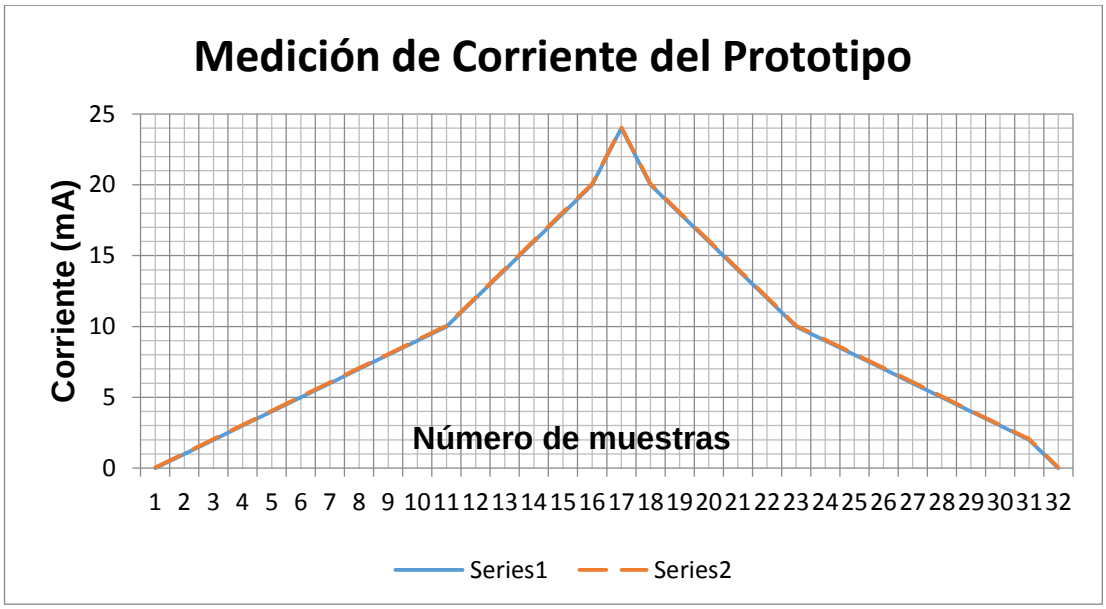


Figura 57: Curva de respuesta de medición de corriente del prototipo.

4.1.3 Pruebas de medición de resistencia

En la prueba de medición de resistencia el prototipo se conecta de acuerdo al circuito mostrado en la **figura 58**, en el cual la misma resistencia se ingresa al prototipo y al calibrador Fluke 724 y se obtendrá el porcentaje de error con respecto al mismo calibrador ya que este será utilizado como instrumento patrón para realizar las pruebas de resistencia del prototipo.

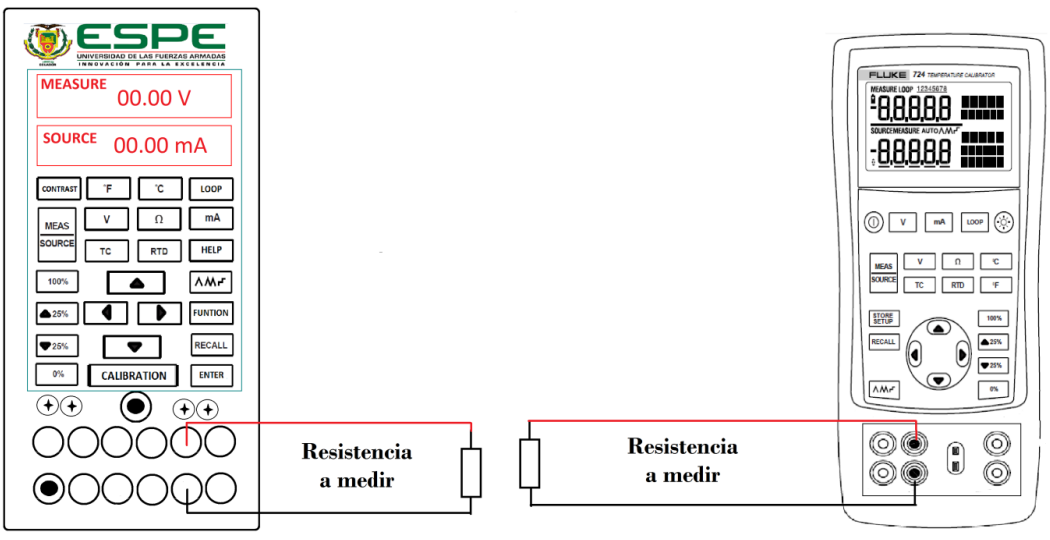


Figura 58: Conexión para la medición de resistencia.

Tabla 10:
Prueba de medición de resistencia del prototipo

MEDICIÓN DE RESISTENCIA DEL CALIBRADOR VS PROTOTIPO				
N. DE PRUEBA	**Valor Deseado**	**CALIBRADOR**	**PROTOTIPO**	**ERROR [%]**
1	0	0	0,2	0,02
2	100	100	99,9	-0,01
3	200	200	201,1	0,11
4	300	300	300,7	0,07
5	400	400	400,6	0,06
6	500	500	501	0,10
7	600	600	598,8	-0,12
8	700	700	700,3	0,03
9	800	800	801,2	0,12
10	900	900	900,3	0,03
11	1000	1000	999,7	-0,03
12	1100	1100	1101	0,10
13	1200	1200	1198,2	-0,18
14	1100	1100	1098,9	-0,11
15	1000	1000	999	-0,10
16	900	900	898,8	-0,12
17	800	800	801	0,10
18	700	700	701,5	0,15
19	600	600	601,3	0,13
20	500	500	501	0,10
21	400	400	398,8	-0,12
22	300	300	301,2	0,12
23	200	200	201,1	0,11
24	100	100	99,9	-0,01
25	0	0	0,3	0,03
			PROMEDIO	**0,02**
			VARIANZA	0,009556
			DESVIACIÓN ESTÁNDAR	0,0977548

Nota: El calibrador Fluke 724 posee una fuente de resistencia a la cual se le debe enviar una corriente de excitación que no debe ser mayor a 0,2 mA o 0,8 mA dependiendo de los rangos de generación, de otra forma dicha fuente sufriría daños, al ser un valor bajo de corriente el acondicionamiento de voltaje también entregará un valor pequeño, para leer dichos valores se

necesita un ADC con una alta resolución, debido a que la tarjeta arduino due posee un ADC de 12 bits no tiene una resolución suficiente para detectar una variación mínima de resistencia, por ende la medición posee una histéresis de 2 ohmios.

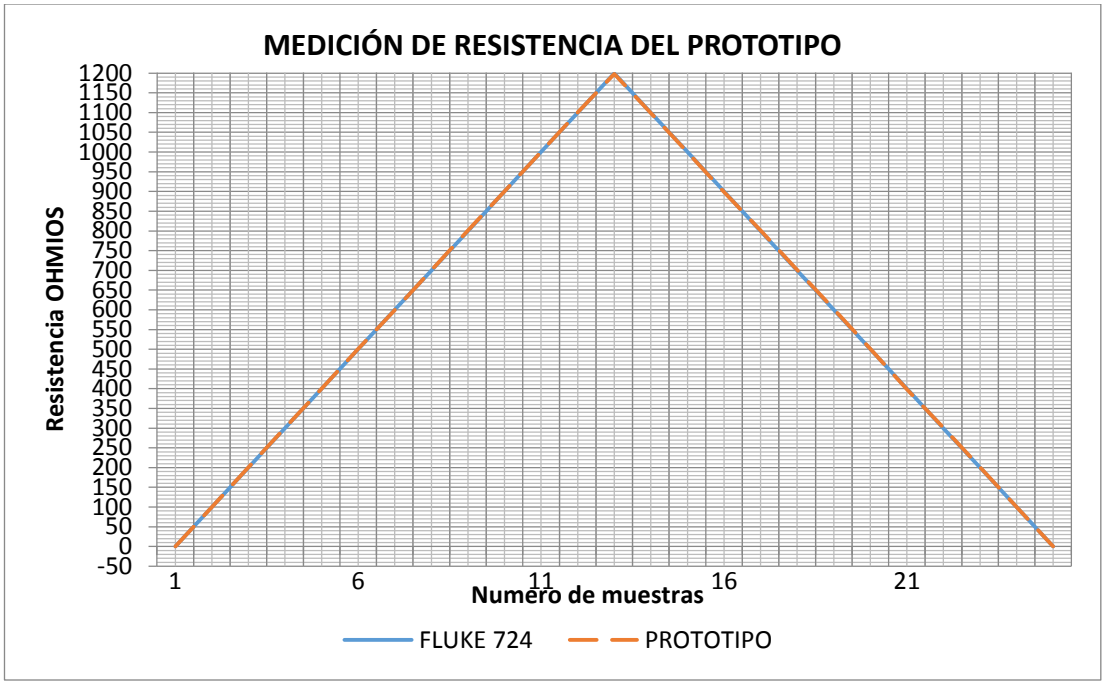


Figura 59: Curva de respuesta de medición de resistencia del prototipo.

4.1.4 Pruebas de medición de RTD de 2, 3 y 4 hilos

En la prueba de medición de RTDS de 2, 3 y 4 hilos se conecta el circuito mostrado en la **figura 60**, tanto para el calibrador Fluke 724 como en el prototipo construido, el valor del error se obtendrá de los valores medidos del prototipo y el calibrador, dicho calibrador se utiliza como instrumento patrón para realizar las pruebas de RTDS de 2, 3 y 4 hilos del prototipo construido.

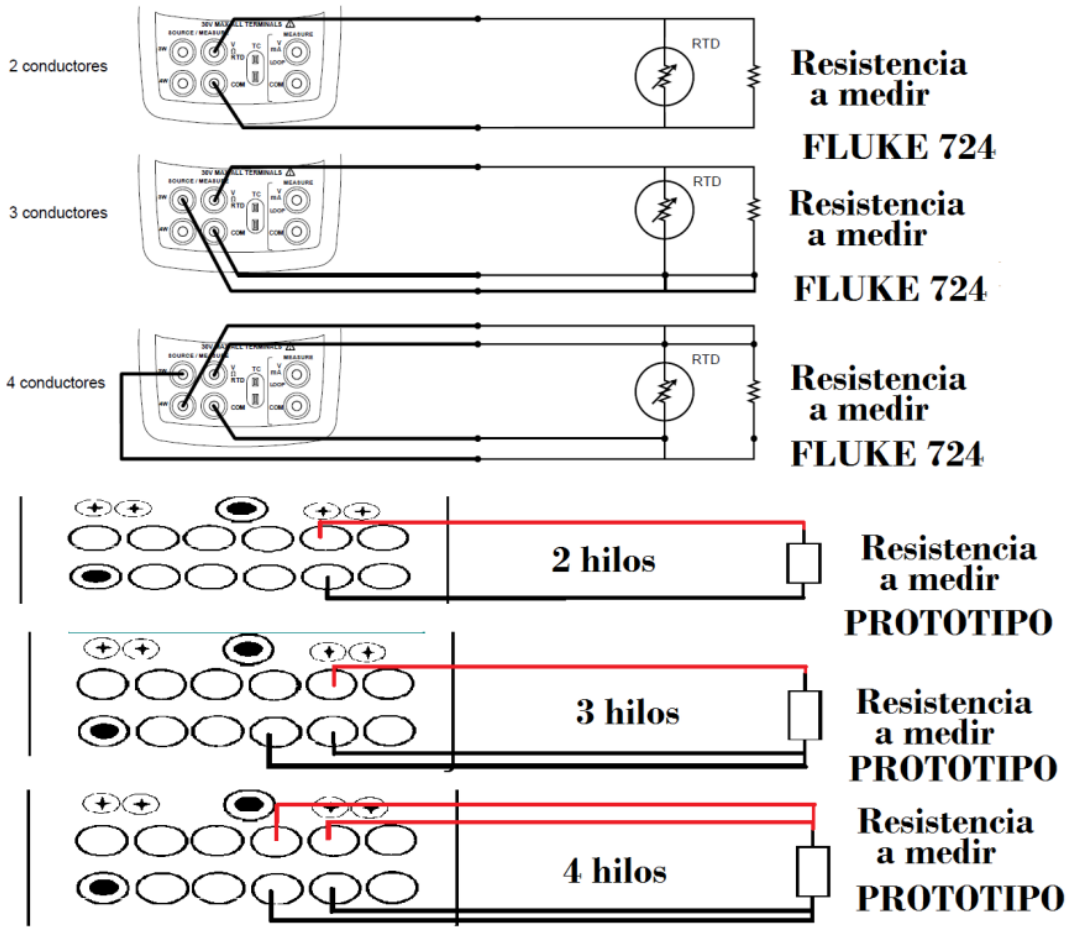


Figura 60: Conexión para la medición de RTDS de 2, 3 y 4 hilos.

Tabla 11:
Prueba de medición de RTD PT100 de 2,3 y 4 hilos del prototipo

PRUEBA DE MEDICIÓN DE UNA RTD PT100 DE 2, 3 Y 4 HILOS										
N. DE PRUEBA	Valor Deseado Grados	RTD DE 2 HILOS			RTD DE 3 HILOS			RTD DE 4 HILOS		
		CALIBRADOR	PROTOTIPO	ERROR [%]	CALIBRADOR	PROTOTIPO	ERROR [%]	CALIBRADOR	PROTOTIPO	ERROR [%]
1	100	100	101,1	0,19	100	100,9	0,15	100	100,3	0,05
2	150	150	150,9	0,15	150	149,2	0,14	150	150,9	0,15
3	200	200	201,2	0,21	200	201,3	0,22	200	201	0,17
4	250	250	249,1	0,15	250	249,8	0,03	250	251,4	0,24
5	300	300	301,3	0,22	300	301,2	0,21	300	301,3	0,22
6	350	350	351	0,17	350	351,4	0,24	350	351,2	0,21
7	400	400	401,5	0,26	400	401	0,17	400	401,1	0,19
8	450	450	448,9	0,19	450	449,6	0,07	450	450,6	0,10
9	500	500	502	0,34	500	502,1	0,36	500	501,1	0,19
10	550	550	551,1	0,19	550	551	0,17	550	550,8	0,14
11	600	600	601,2	0,21	600	600,8	0,14	600	600,9	0,15
12	650	650	651,1	0,19	650	650,8	0,14	650	651	0,17
13	683	683	682,1	0,15	683	682	0,17	683	682,3	0,12
14	650	650	650,9	0,15	650	650,3	0,05	650	650,4	0,07
15	600	600	601,2	0,21	600	600,4	0,07	600	600,6	0,10
16	550	550	551,2	0,21	550	549,3	0,12	550	549,3	0,12
17	500	500	500,9	0,15	500	501,4	0,24	500	501,2	0,21
18	450	450	451	0,17	450	452	0,34	450	451,1	0,19
19	400	400	401,2	0,21	400	401	0,17	400	399,5	0,09
20	350	350	351,2	0,21	350	349,6	0,07	350	349,7	0,05
21	300	300	300,8	0,14	300	299,8	0,03	300	301,1	0,19
22	250	250	251,1	0,19	250	251	0,17	250	250,7	0,12
23	200	200	201,1	0,19	200	201,3	0,22	200	201,6	0,27
24	150	150	148,8	0,21	150	149,6	0,07	150	149,7	0,05
25	100	100	99,6	0,07	100	101	0,17	100	100,9	0,15
		PROMEDIO		0,13	PROMEDIO		0,10	PROMEDIO		0,11
		VARIANZA		0,02	VARIANZA		0,02	VARIANZA		0,01
		DESV.EST		0,15	DESV.EST		0,15	DESV.EST		0,11

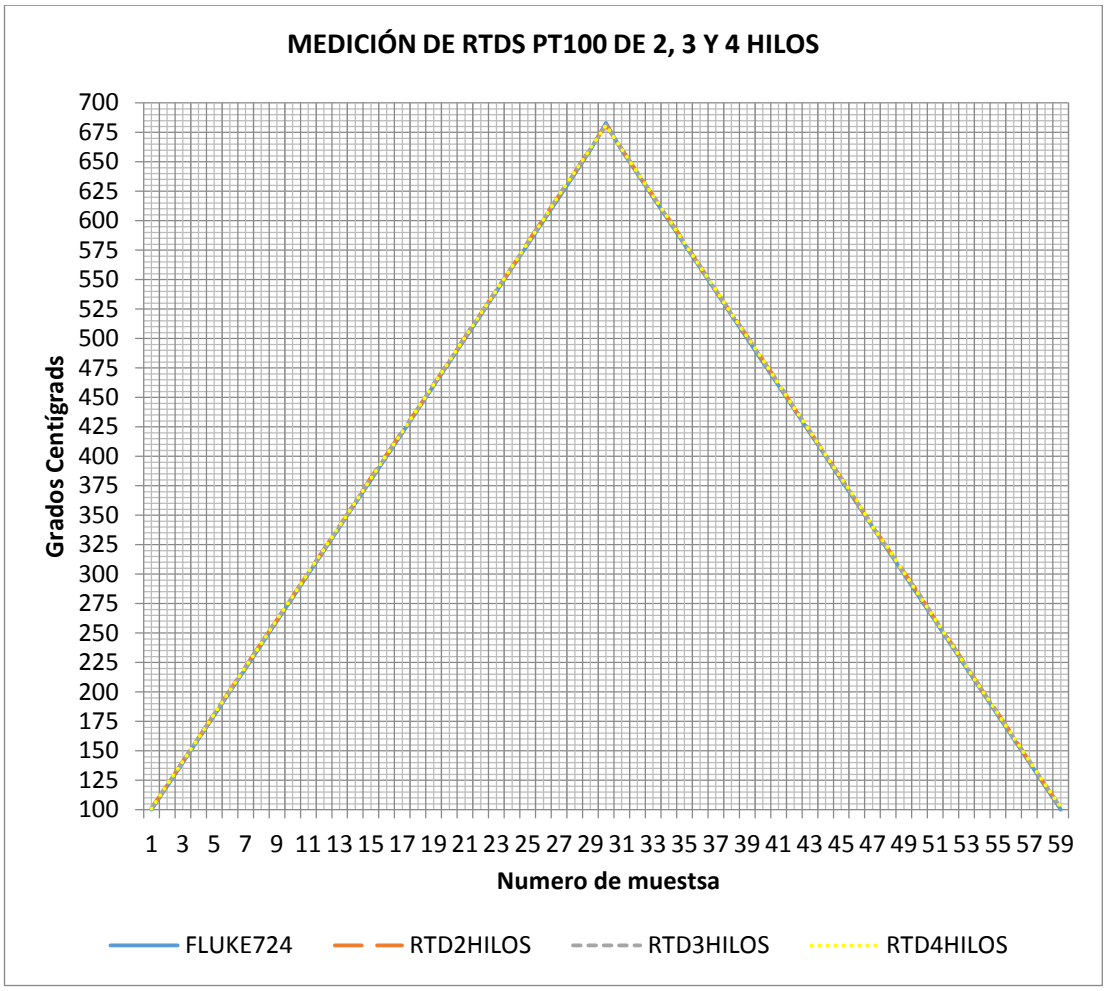


Figura 61: Curva de respuesta de medición de RTDS PT100 de 2, 3 y 4 hilos.

4.1.5 Pruebas de medición de Termocuplas tipo J y K

Para realizar la prueba de medición de las termocuplas se conecta el circuito mostrado en la **figura 62**, donde la misma termocupla ingresa al prototipo y al calibrador Fluke 724 y se obtendrá el porcentaje de error con respecto al mismo calibrador ya que ambos compensan el valor de la unión fría, sin embargo de una manera distinta y de acuerdo al tipo de termocupla que se desee medir.

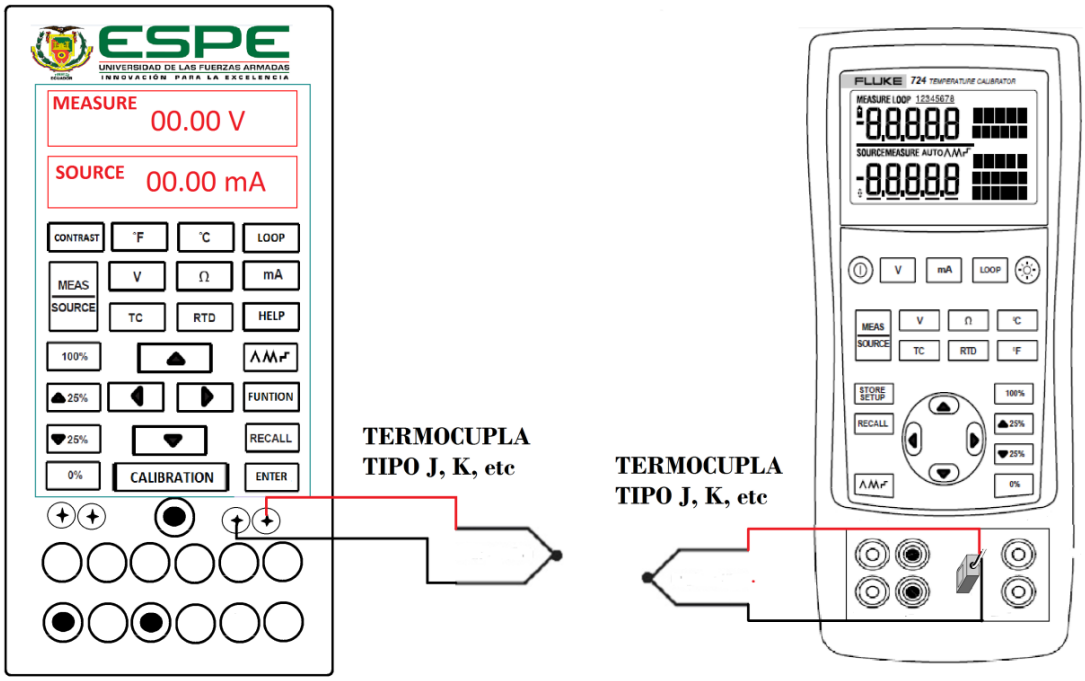


Figura 62: Conexión para la medición de termocuplas tipo j, k, etc.

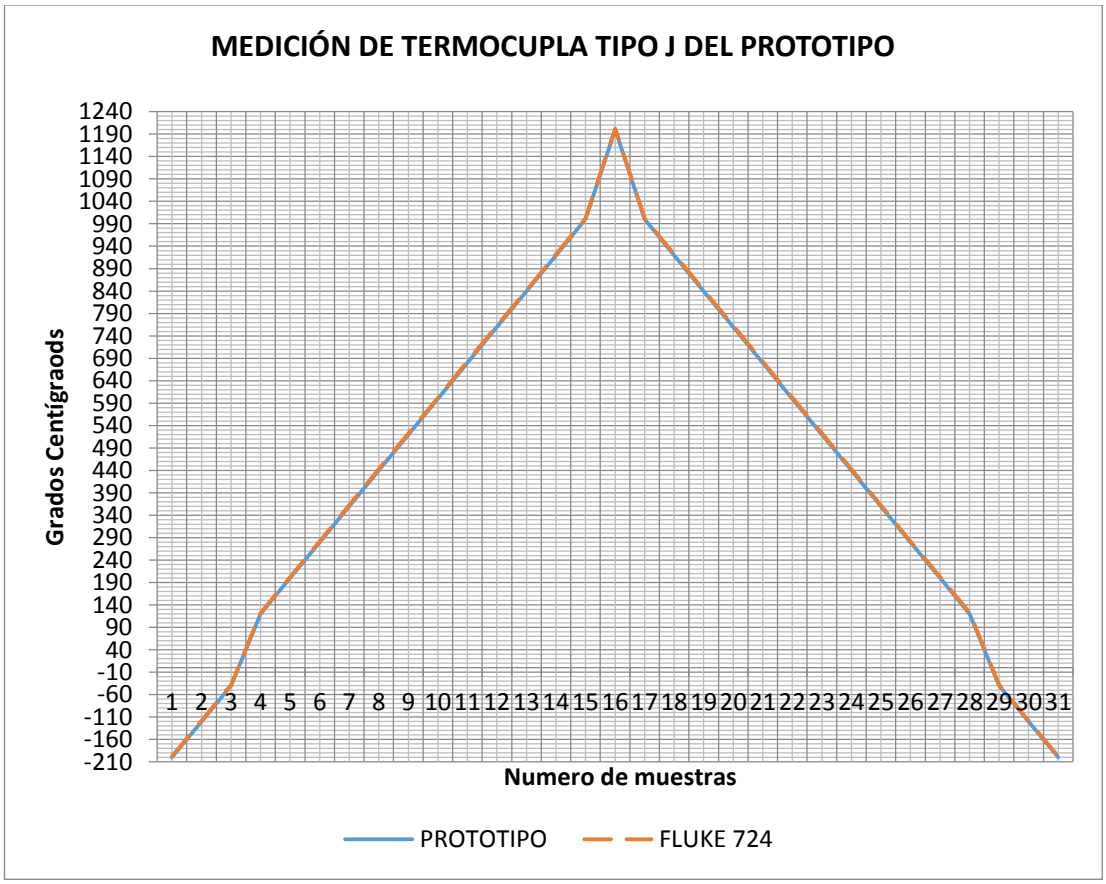


Figura 63: Curva de respuesta de medición de terocupla tipo J del prototipo.

Tabla 12:
Prueba de medición de termocupla tipo J del prototipo

MEDICIÓN DE TERMOCUPLA TIPO J DEL CALIBRADOR VS PROTOTIPO				
N. DE PRUEBA	**Valor Deseado**	**CALIBRADOR**	**PROTOTIPO**	**ERROR [%]**
1	-200	-200	-197,6	0,17
2	-120	-120	-117,9	0,15
3	-40	-40	-38,2	0,13
4	120	120	122,5	0,18
5	200	200	202,3	0,16
6	280	280	281,9	0,14
7	360	360	362,5	0,18
8	440	440	442,6	0,19
9	520	520	522,4	0,17
10	600	600	602,1	0,15
11	680	680	683	0,21
12	760	760	762,6	0,19
13	840	840	842	0,14
14	920	920	922,4	0,17
15	1000	1000	1002	0,14
16	1200	1200	1202,6	0,19
17	1000	1000	1002	0,14
18	920	920	922,3	0,16
19	840	840	842,8	0,20
20	760	760	762,7	0,19
21	680	680	682,4	0,17
22	600	600	602,3	0,16
23	520	520	523	0,21
24	440	440	443	0,21
25	360	360	362,4	0,17
26	280	280	281,8	0,13
27	200	200	202,8	0,20
28	120	120	122,7	0,19
29	-40	-40	-37,6	0,17
30	-120	-120	-117,4	0,19
31	-200	-200	-197,8	0,16
		PROMEDIO		**0,17**
		VARIANZA		**0,00058822**
		DESVIACIÓN ESTÁNDAR		**0,02425316**

Tabla 13:
Prueba de medición de termocupla tipo K del prototipo

MEDICIÓN DE TERMOCUPLA TIPO K DEL CALIBRADOR VS PROTOTIPO				
N. DE PRUEBA	**Valor Deseado**	**CALIBRADOR**	**PROTOTIPO**	**ERROR [%]**
1	-200	-200	-197,4	0,19
2	-120	-120	-117,7	0,16
3	-40	-40	-37,9	0,15
4	120	120	122,6	0,19
5	200	200	202,6	0,19
6	280	280	282,3	0,16
7	360	360	362,3	0,16
8	440	440	442,8	0,20
9	520	520	522,3	0,16
10	600	600	602,3	0,16
11	680	680	682,4	0,17
12	760	760	762,2	0,16
13	840	840	842,5	0,18
14	920	920	921,9	0,14
15	1000	1000	1001,9	0,14
16	1200	1200	1202,4	0,17
17	1000	1000	1002,3	0,16
18	920	920	923	0,21
19	840	840	842,5	0,18
20	760	760	762,2	0,16
21	680	680	681,2	0,09
22	600	600	602,3	0,16
23	520	520	522,6	0,19
24	440	440	442,2	0,16
25	360	360	362,2	0,16
26	280	280	282,2	0,16
27	200	200	202,5	0,18
28	120	120	122,3	0,16
29	-40	-40	-38,1	0,14
30	-120	-120	-117,7	0,16
31	-200	-200	-198,2	0,13
			PROMEDIO	**0,16**
			VARIANZA	**0,00055563**
			DESVIACIÓN ESTÁNDAR	**0,02357178**

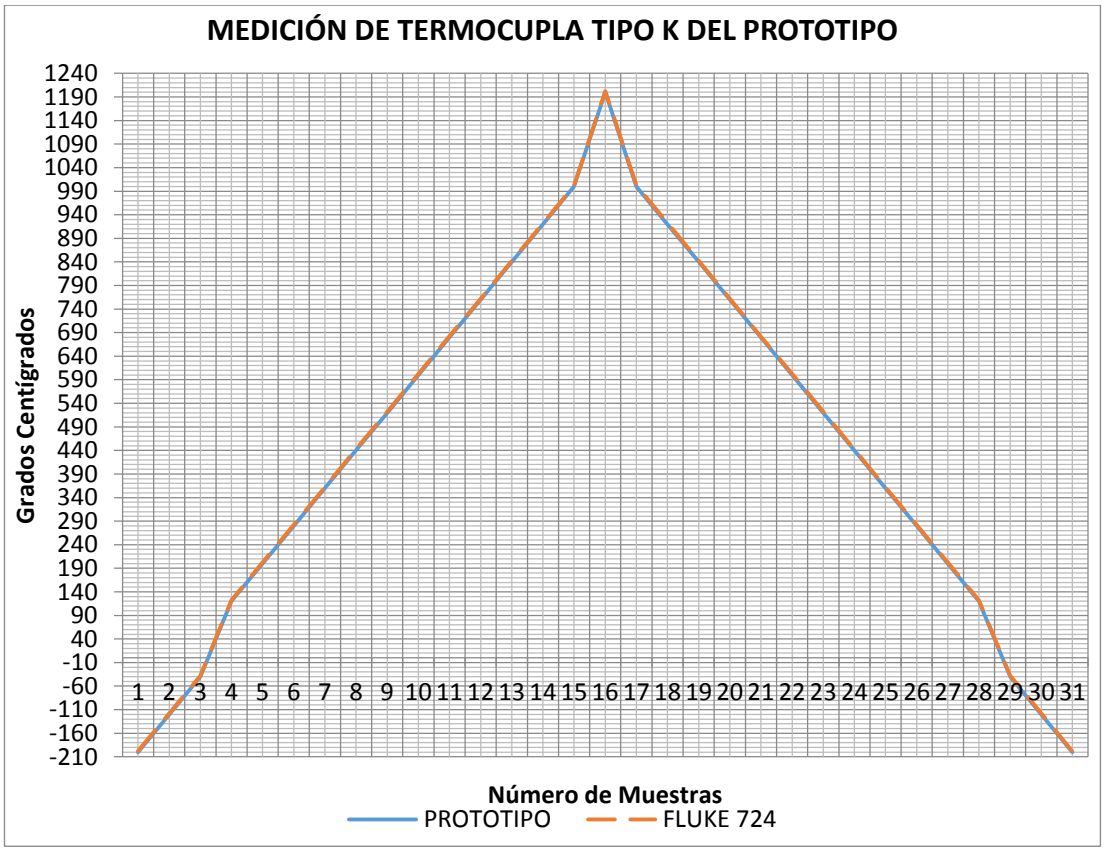


Figura 64: Curva de respuesta de medición de terocupla tipo K del prototipo.

4.2 Pruebas de Generación y Simulación de Variables

Una vez realizado el respectivo acondicionamiento de las variables eléctricas, las pruebas de generación de variables del prototipo construido serán realizadas en el Laboratorio de Redes Industriales y Control de Procesos de la Universidad de las Fuerzas Armadas ESPE extensión Latacunga y conforme a los rangos que se establecieron en la **Tabla 7.**

4.2.1 Pruebas de generación de Voltaje

En la prueba de generación de voltaje se conecta el circuito mostrado en la **figura 65**, para lo cual se enviará un valor deseado de voltaje desde el

prototipo, mismo que será medido por el calibrador Fluke 724, y se obtendrá el error del valor medido por el calibrador y el valor deseado enviado desde el prototipo.

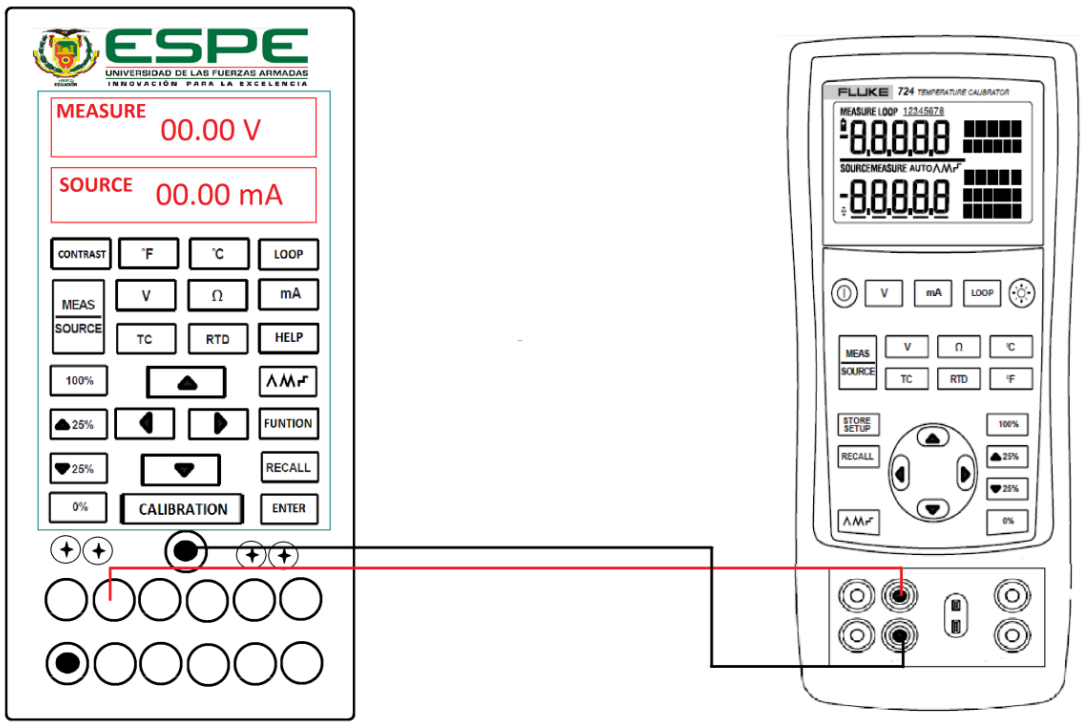


Figura 65: Conexión para la generación de voltaje.

Para determinar el error de generación y simulación de las variables físicas del prototipo se aplicó la siguiente fórmula:

$$Error = \frac{V.medido\ calibrador - V.generado\ prototipo}{Rango\ de\ operación} x100$$

El instrumento patrón para medir el valor generado por el prototipo construido es el calibrador Fluke 724 ya que tiene varias certificaciones que aseguran un valor de medida muy confiable y garantiza las pruebas de generación del prototipo.

Tabla 14:
Prueba de generación de Voltaje.

GENERACIÓN DE VOLTAJE DEL PROTOTIPO				
N. DE PRUEBA	**VALOR [Vdc]**	**V.Generado PROTOTIPO**	**V.Medido CALIBRADOR**	**ERROR [%]**
1	0	0	0,004	0,04
2	0,02	0,02	0,019	-0,01
3	0,04	0,04	0,039	-0,01
4	0,06	0,06	0,059	-0,01
5	0,08	0,08	0,079	-0,01
6	0,1	0,1	0,099	-0,01
7	0,2	0,2	0,199	-0,01
8	0,6	0,6	0,599	-0,01
9	1	1	0,998	-0,02
10	3	3	2,998	-0,02
11	4	4	3,998	-0,02
12	5	5	4,998	-0,02
13	6	6	5,998	-0,02
14	7	7	6,998	-0,02
15	8	8	7,998	-0,02
16	9	9	8,998	-0,02
17	10	10	9,998	-0,02
18	9	9	8,998	-0,02
19	8	8	7,998	-0,02
20	7	7	6,998	-0,02
21	6	6	5,999	-0,01
22	5	5	4,998	-0,02
23	4	4	3,998	-0,02
24	3	3	2,998	-0,02
25	1	1	0,999	-0,01
26	0,6	0,6	0,599	-0,01
27	0,2	0,2	0,199	-0,01
28	0,1	0,1	0,099	-0,01
29	0,08	0,08	0,079	-0,01
30	0,06	0,06	0,069	0,09
31	0,04	0,04	0,049	0,09
32	0	0	0,004	0,04
			PROMEDIO	**-0,01**
			VARIANZA	**0,00083216**
			DESVIACIÓN ESTÁNDAR	**0,02884714**

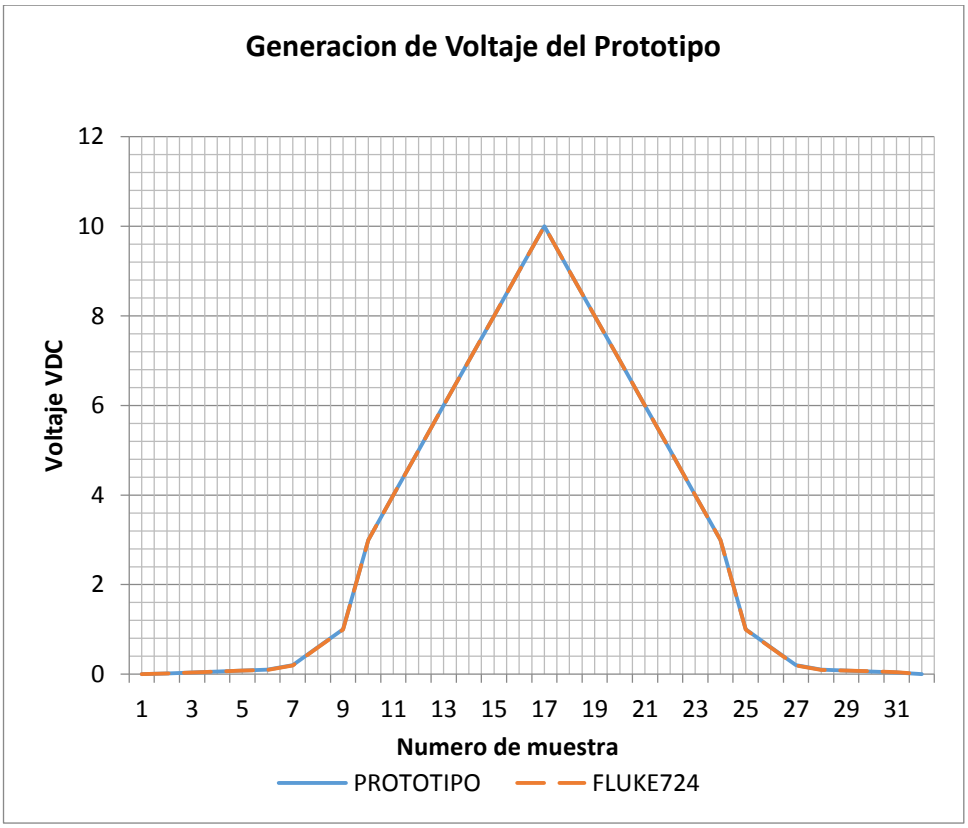


Figura 66: Curva de respuesta de la generación de voltaje.

4.2.2 Pruebas de generación de Corriente

La **figura 67** muestra la conexión necesaria del prototipo implementado con el calibrador para realizar la prueba de generación de corriente, el valor que genere la fuente de corriente la medirá el calibrador Fluke 724 y se obtendrá el porcentaje de error de acuerdo a la diferencia entre los valores generados por el prototipo y los medidos por el calibrador, mismo que es utilizado como instrumento patrón para realizar las pruebas de generación de corriente del prototipo.

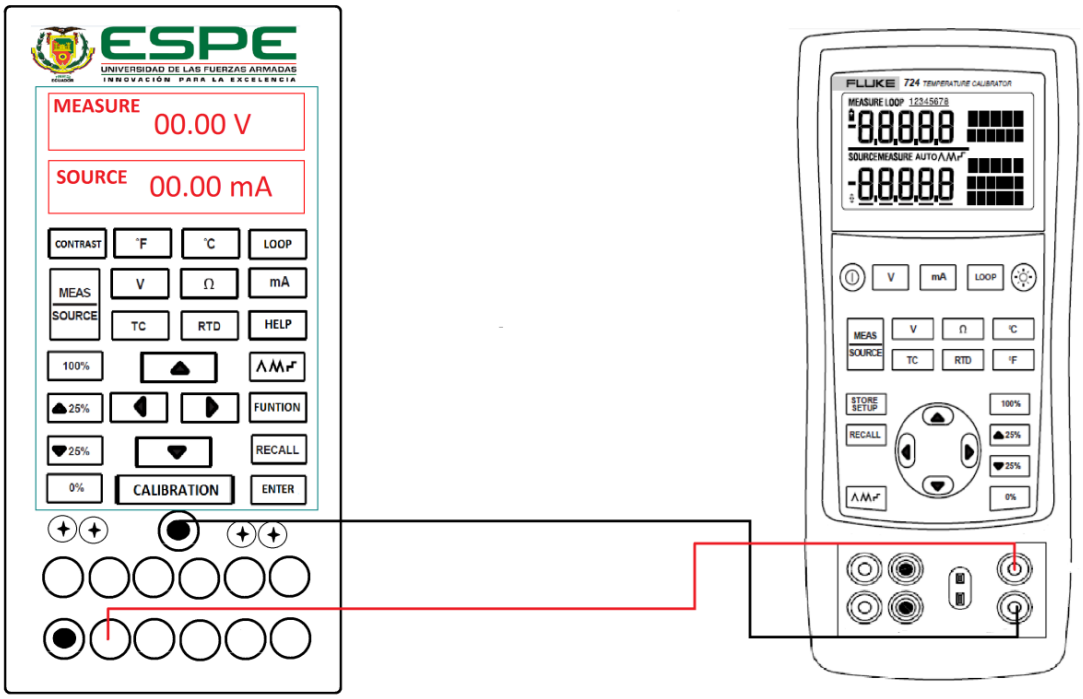


Figura 67: Conexión para la generación de corriente.

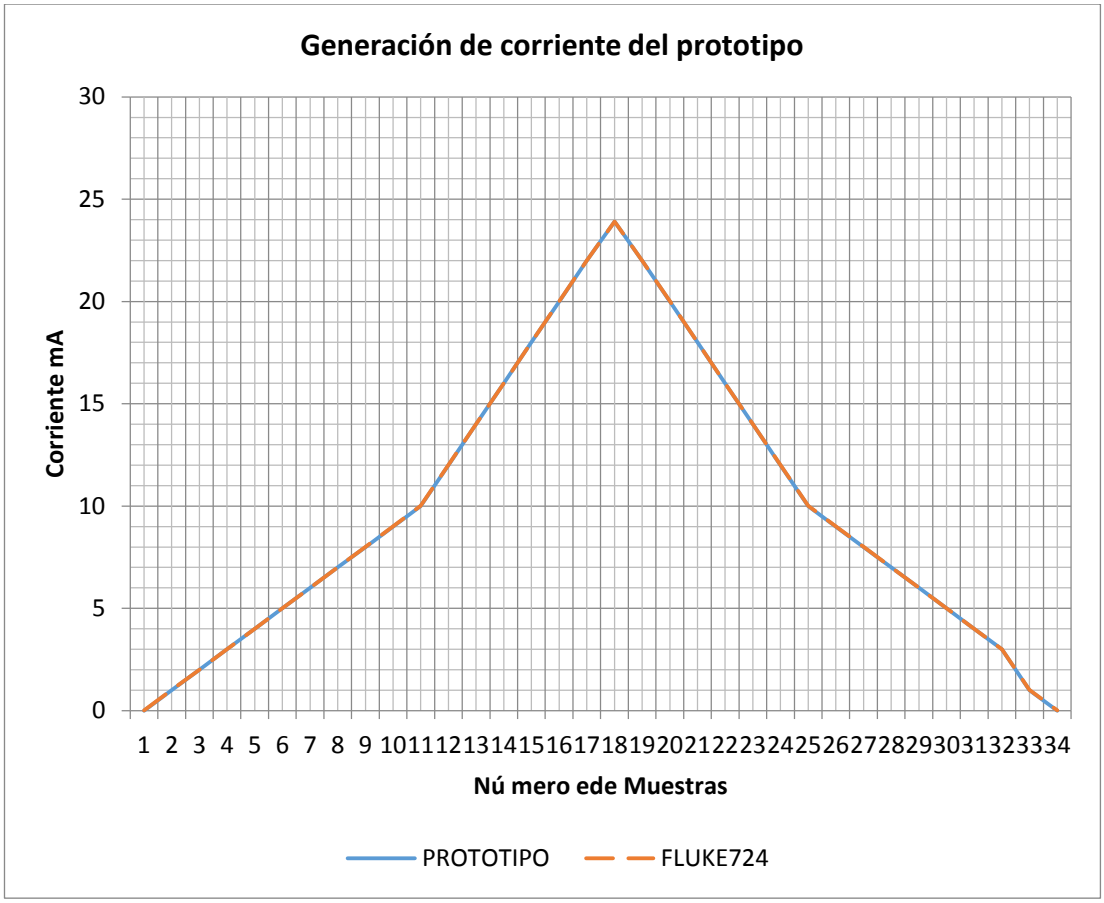


Figura 68: Curva de respuesta de la generación de corriente

Tabla 15:
Prueba de generación de Corriente.

GENERACIÓN DE CORRIENTE DEL PROTOTIPO				
N. DE PRUEBA	**VALOR EN [mA]**	**V. Generado PROTOTIPO**	**V. Medido CALIBRADOR**	**ERROR [%]**
0	0	0	0,004	0,02
1	1	1	1,011	0,05
2	2	2	2,005	0,02
3	3	3	3,005	0,02
4	4	4	4,005	0,02
5	5	5	5,003	0,01
6	6	6	6,003	0,01
7	7	7	7,003	0,01
8	8	8	8,005	0,02
9	9	9	9,002	0,01
10	10	10	10,006	0,03
11	12	12	12,007	0,03
12	14	14	14,007	0,03
13	16	16	16,005	0,02
14	18	18	18,006	0,03
15	20	20	20,007	0,03
16	22	22	22,004	0,02
17	24	23,9	23,904	0,02
18	22	22	22,004	0,02
19	20	20	20,008	0,03
20	18	18	18,006	0,03
21	16	16	16,004	0,02
22	14	14	14,007	0,03
23	12	12	12,007	0,03
25	9	9	9,003	0,01
26	8	8	8,005	0,02
27	7	7	7,003	0,01
28	6	6	6,003	0,01
29	5	5	5,004	0,02
30	4	4	4,005	0,02
31	3	3	3,004	0,02
32	1	1	1,011	0,05
33	0	0	0,004	0,02
			PROMEDIO	**0,02**
			VARIANZA	**7,63E-05**
			DESVIACIÓN ESTÁNDAR	**0,00873228**

4.2.3 Pruebas de generación de Resistencia

En la prueba de generación de resistencia el prototipo se conecta con el calibrador Fluke 724 de acuerdo al circuito mostrado en la **figura 69**, en el cual el valor de resistencia generado por el prototipo ingresa al calibrador Fluke 724 y se obtendrá el porcentaje de error de acuerdo a la fórmula mencionada en la generación de voltaje, una vez más el instrumento patrón para realizar las pruebas de generación de resistencia del prototipo es el calibrador Fluke 724.

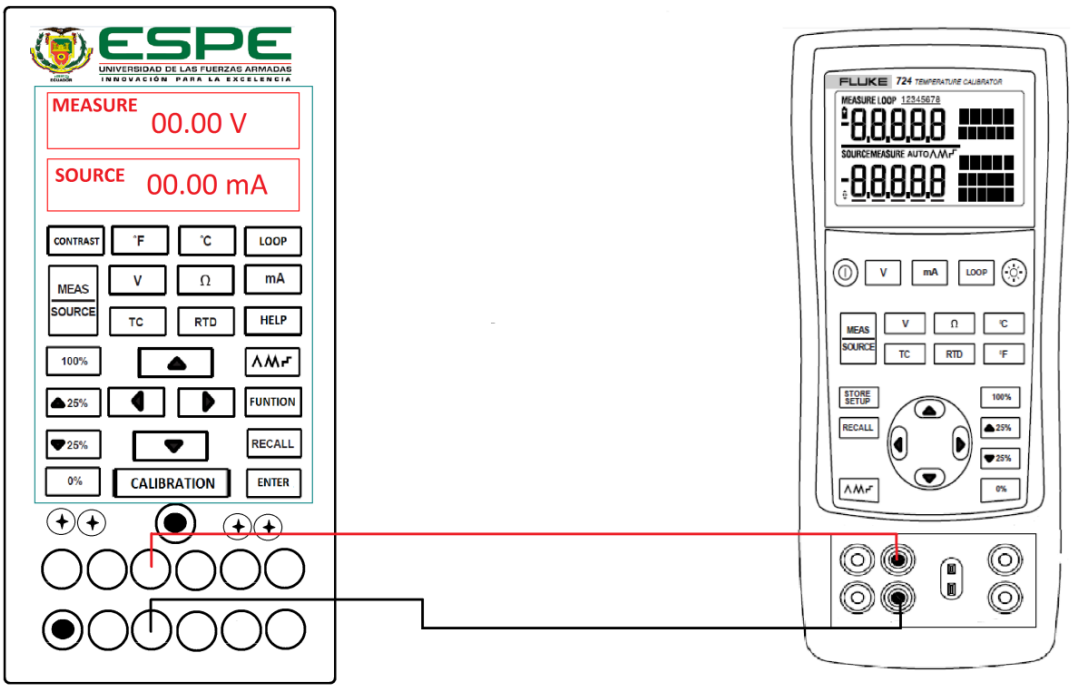


Figura 69: Conexión para la generación de resistencia.

Tabla 16:
Prueba de generación de resistencia del prototipo

GENERACIÓN DE RESISTENCIA DEL PROTOTIPO				
N. DE PRUEBA	**VALOR [Ohmios]**	**V.Generado PROTOTIPO**	**V.Medido CALIBRADOR**	**ERROR [%]**
1	30	30	31,6	0,28
2	70	70	71,44	0,25
3	110	110	111,68	0,29
4	150	150	151,66	0,29
5	190	190	191,34	0,24
6	230	230	231,38	0,24
7	270	270	271,76	0,31
8	310	310	311,24	0,22
9	350	350	351,86	0,33
10	390	390	391,34	0,24
11	430	430	431,42	0,25
12	470	470	471,66	0,29
13	510	510	511,34	0,24
14	550	550	551,46	0,26
15	600	600	601,4	0,25
16	550	550	551,56	0,27
17	510	510	511,42	0,25
18	470	470	471,42	0,25
19	430	430	431,22	0,21
20	390	390	391,44	0,25
21	350	350	351,66	0,29
22	310	310	311,16	0,20
23	270	270	271,61	0,28
24	230	230	231,22	0,21
25	190	190	191,44	0,25
26	150	150	151,46	0,26
27	110	110	111,24	0,22
28	70	70	71,62	0,28
29	30	30	31,44	0,25
		PROMEDIO		**0,26**
		VARIANZA		**0,0009**
		DESVIACIÓN ESTÁNDAR		**0,0307**

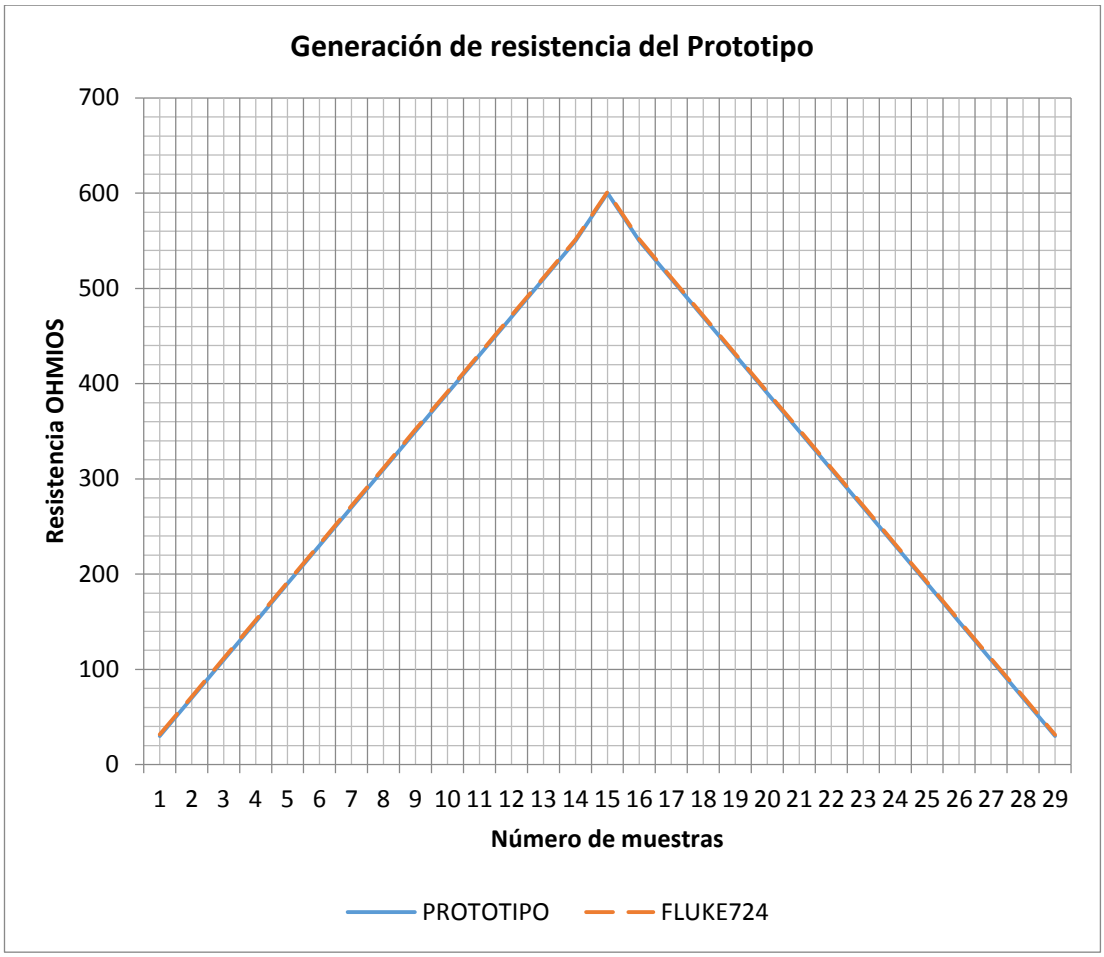


Figura 70: Curva de respuesta de la generación de resistencia del prototipo.

Nota: debido a las características propias del integrado AD5422, la generación de resistencia varía alrededor de 3 ohmios ya que la resistencia inicial de dichos integrados cambian la resistencia inicial fluctúan una vez que se enciende el equipo, es decir ninguno de los tres integrados tiene la misma resistencia inicial, además por la variación de un bit la variación de resistencia de cada uno de los dispositivos es distinta en un rango de 0,3 ohmios después de cierto tiempo de uso del equipo la resistencia inicial de los integrados, varía nuevamente produciendo así un desplazamiento en los valores de resistencia por lo que los valores de resistencia generados tendrán una histéresis de 2 grados, debido a que estas características intrínsecas del integrado no son posibles de eliminar.

4.2.4 Pruebas de simulación de RTD

A continuación se muestran los diagramas de conexión requeridos para realizar las pruebas de generación de RTDS de 2, 3 y 4 hilos del prototipo construido, ya que esta prueba se basa en la variación de resistencia el circuito implementado es el que se utiliza en la medición de resistencia, con ciertas variaciones que se muestran en las siguientes figuras.

a. Pruebas de simulación de RTD de 2 hilos

En la **figura 71** se puede apreciar la conexión para realizar la simulación de una RTD de 2 hilos.

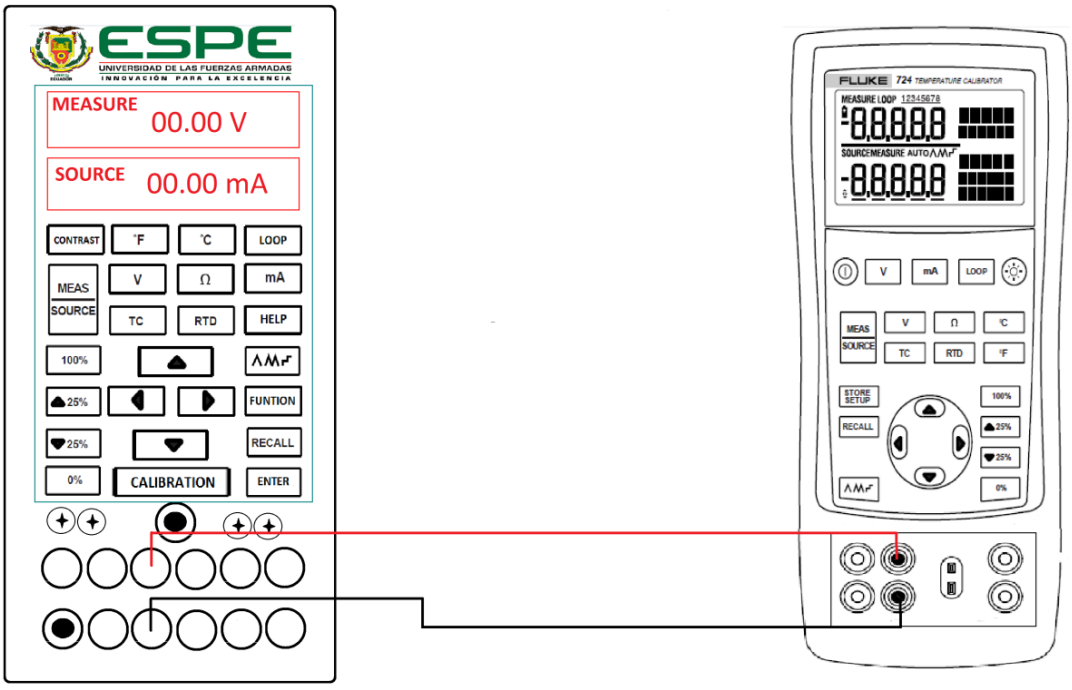


Figura 71: Conexión para la simulación de RTD de 2 hilos.

Tabla 17:
Prueba de generación de RTD PT100 de 2 hilos del prototipo

GENERACIÓN DE RTD DE 2 HILOS DEL PROTOTIPO				
N. DE PRUEBA	**VALOR [Grados]**	**V.Generado PROTOTIPO**	**V.Medido CALIBRADOR**	**ERROR [%]**
1	-150	-150	-149,2	0,18
2	-120	-120	-118,6	0,31
3	-90	-90	-89,1	0,20
4	-60	-60	-58,2	0,40
5	-30	-30	-28,2	0,40
6	0	0	1,8	0,40
7	30	30	32,1	0,47
8	60	60	61,7	0,38
9	90	90	91,4	0,31
10	120	120	122,1	0,47
11	150	150	151,4	0,31
12	180	180	182,4	0,53
13	210	210	212,2	0,49
14	240	240	242,1	0,47
15	270	270	272,1	0,47
16	300	300	301,2	0,27
17	270	270	272,1	0,47
18	240	240	242,4	0,53
19	210	210	212,2	0,49
20	180	180	181,8	0,40
21	150	150	153,1	0,69
22	120	120	121,3	0,29
23	90	90	91,7	0,38
24	60	60	62,4	0,53
25	30	30	31,5	0,33
26	0	0	1,7	0,38
27	-30	-30	-28,6	0,31
28	-60	-60	-57,9	0,47
29	-90	-90	-89,1	0,20
30	-120	-120	-119,2	0,18
31	-150	-150	-152,8	-0,62
			PROMEDIO	**0,36**
			VARIANZA	**0,04710899**
			DESVIACIÓN ESTÁNDAR	**0,21704605**

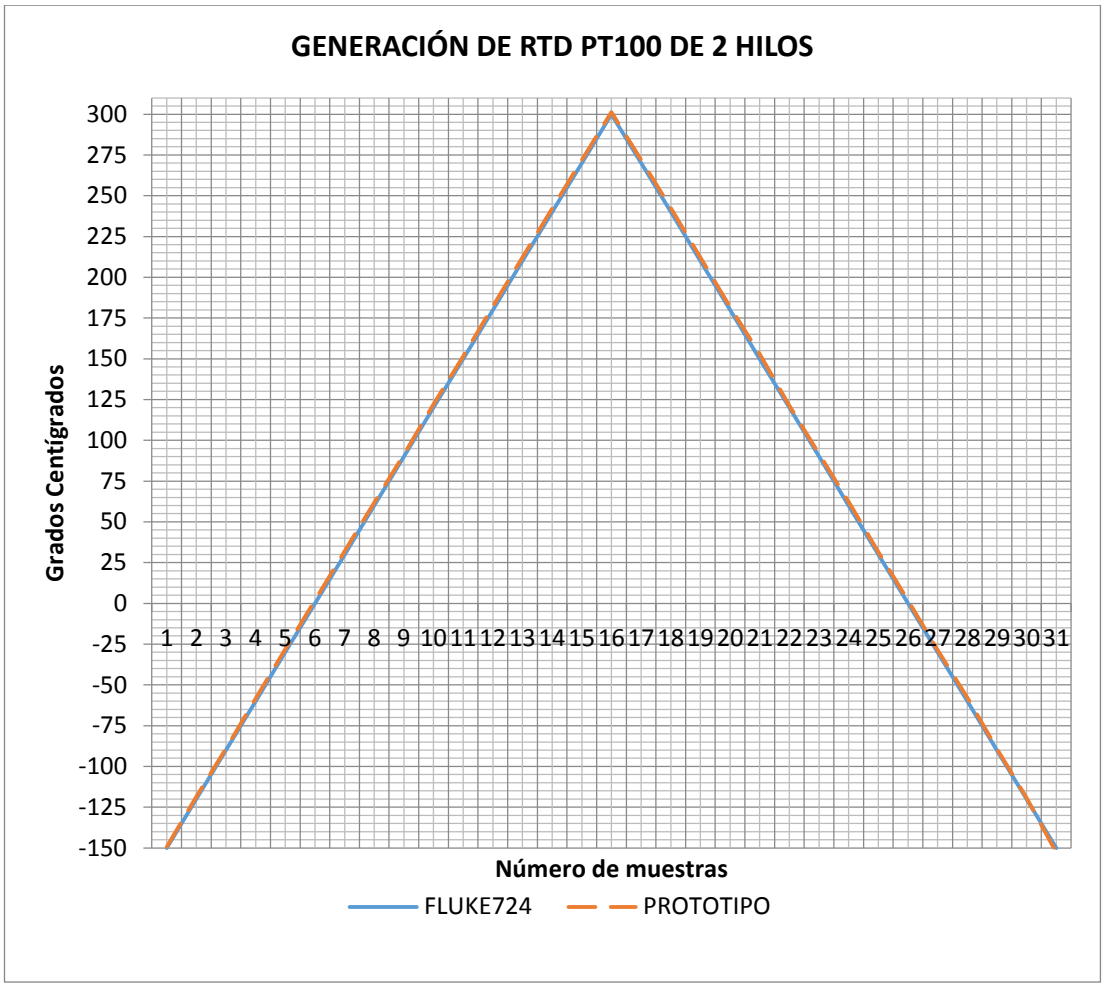


Figura 72: Curva de respuesta de la generación de RTD PT100 de 2 hilos.

4.2.5 Pruebas de simulación de Termocuplas tipo J y K

Para realizar la prueba de simulación de las termocuplas se conecta el circuito mostrado en la **figura 73**, donde la termocupla que genera el prototipo ingresa al calibrador Fluke 724 y se obtendrá el porcentaje de error con respecto al mismo calibrador ya que ambos compensan el valor de la unión fría, sin embargo de una manera distinta y de acuerdo al tipo de termocupla que se desee generar.

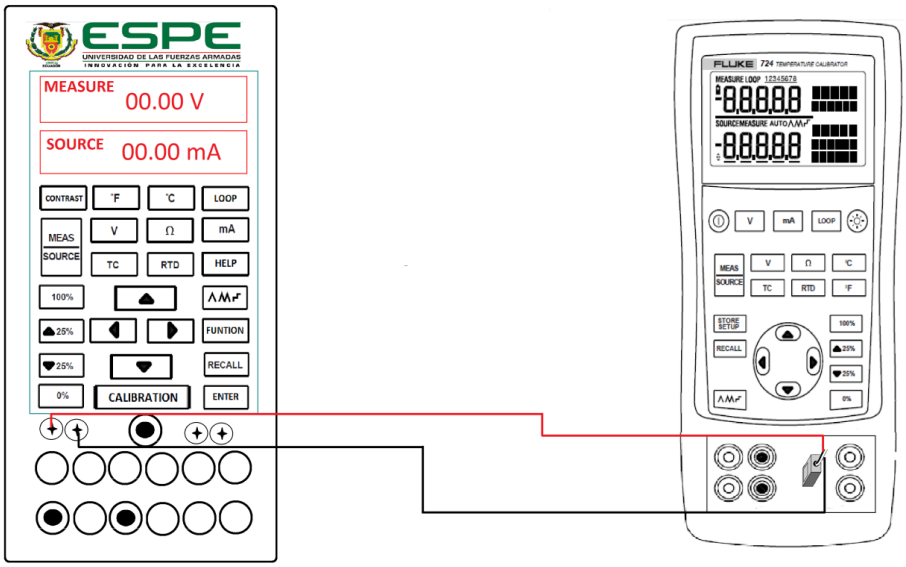


Figura 73: Conexión para la simulación de termocuplas.

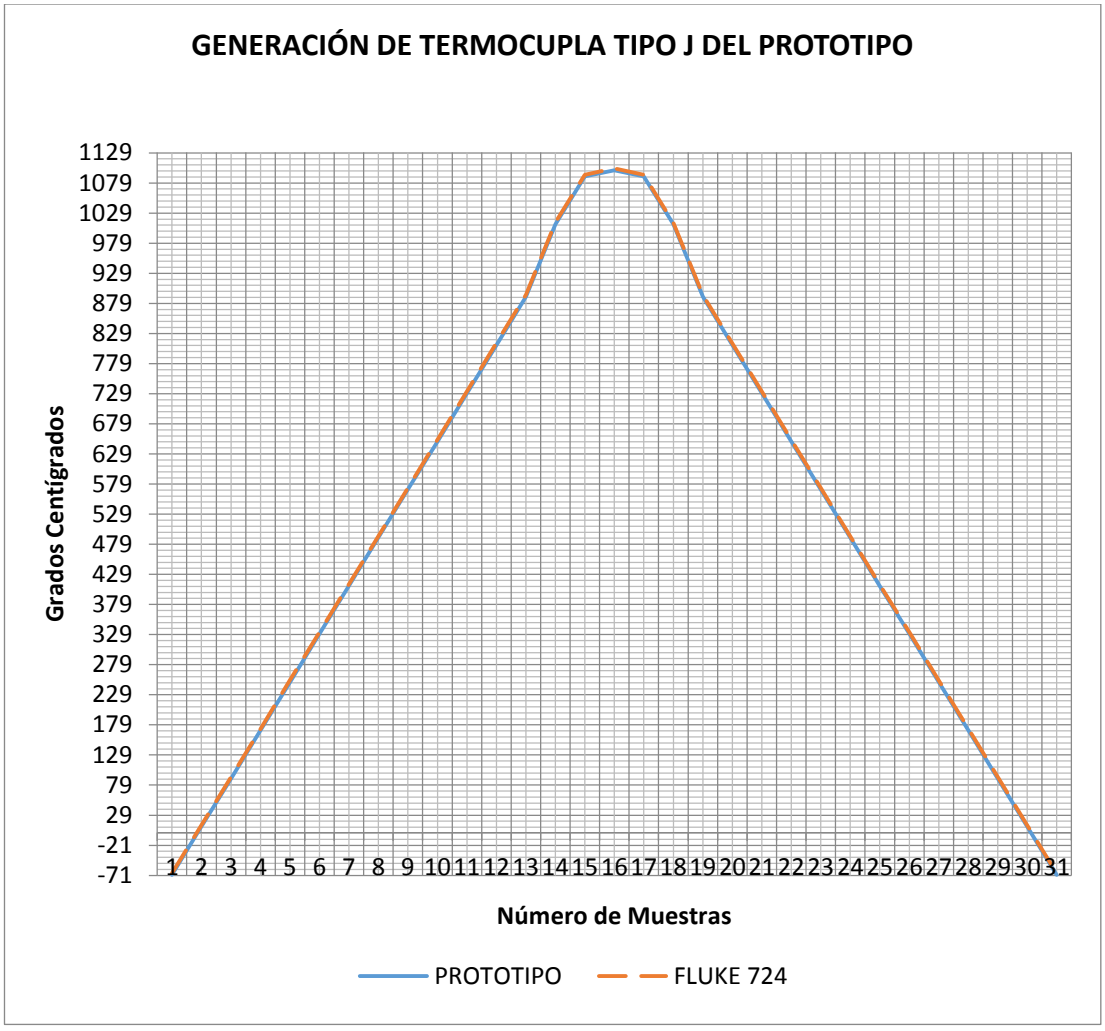


Figura 74: Curva de respuesta de la generación de termocupla tipo J.

Tabla 18:
Prueba de generación de termocupla tipo J del prototipo.

GENERACIÓN DE TERMOCUPLA TIPO J DEL PROTOTIPO				
N. DE PRUEBA	**VALOR [°C]**	**Valor Generad PROTOTIPO**	**Valor Medido CALIBRADOR**	**ERROR [%]**
1	-70	-70	-67,3	0,23
2	10	10	12,4	0,21
3	90	90	92,6	0,22
4	170	170	172,5	0,21
5	250	250	253,6	0,31
6	330	330	332,6	0,22
7	410	410	412,7	0,23
8	490	490	492,5	0,21
9	570	570	572,3	0,20
10	650	650	652,7	0,23
11	730	730	733,2	0,27
12	810	810	813,4	0,29
13	890	890	893,4	0,29
14	1010	1010	1013,3	0,28
15	1090	1090	1092,6	0,22
16	1100	1100	1103,2	0,27
17	1090	1090	1092,5	0,21
18	1010	1010	1012,6	0,22
19	890	890	892,5	0,21
20	810	810	812,3	0,20
21	730	730	732,8	0,24
22	650	650	652,6	0,22
23	570	570	572,6	0,22
24	490	490	492,5	0,21
25	410	410	412	0,17
26	330	330	333	0,26
27	250	250	252,4	0,21
28	170	170	172,6	0,22
29	90	90	92,5	0,21
30	10	10	12,3	0,20
31	-70	-70	-67,3	0,23
		PROMEDIO		**0,23**
		VARIANZA		**0,001**
		DESVIACIÓN ESTÁNDAR		**0,032**

Tabla 19:
Prueba de generación de termocupla tipo K del prototipo.

GENERACIÓN DE TERMOCUPLA TIPO K DEL PROTOTIPO				
N. DE PRUEBA	**VALOR EN GRADOS**	**Valor Generado PROTOTIPO**	**Valor Medido CALIBRADOR**	**ERROR [%]**
1	-110	-110	-107,6	0,17
2	-10	-10	-8,2	0,12
3	70	70	72,6	0,18
4	150	150	152,6	0,18
5	230	230	232,7	0,19
6	310	310	312,6	0,18
7	390	390	392,5	0,17
8	470	470	472,4	0,17
9	550	550	552,4	0,17
10	630	630	633	0,21
11	710	710	712,7	0,19
12	790	790	792,6	0,18
13	870	870	872,7	0,19
14	950	950	952,6	0,18
15	1030	1030	1032,5	0,17
16	1190	1190	1192,3	0,16
17	1300	1300	1303	0,21
18	1190	1190	1192,5	0,17
19	1030	1030	1032,6	0,18
20	950	950	952,6	0,18
22	790	790	792,6	0,18
23	710	710	712,4	0,17
24	630	630	632,2	0,15
25	550	550	552,3	0,16
26	470	470	472,6	0,18
27	390	390	393	0,21
28	310	310	312,4	0,17
29	230	230	232,6	0,18
30	150	150	152,8	0,19
31	70	70	72,4	0,17
32	-10	-10	-8,9	0,08
33	-110	-110	-107,1	0,20
			PROMEDIO	**0,17**
			VARIANZA	**0,00058399**
			DESVIACIÓN ESTÁNDAR	**0,0241659**

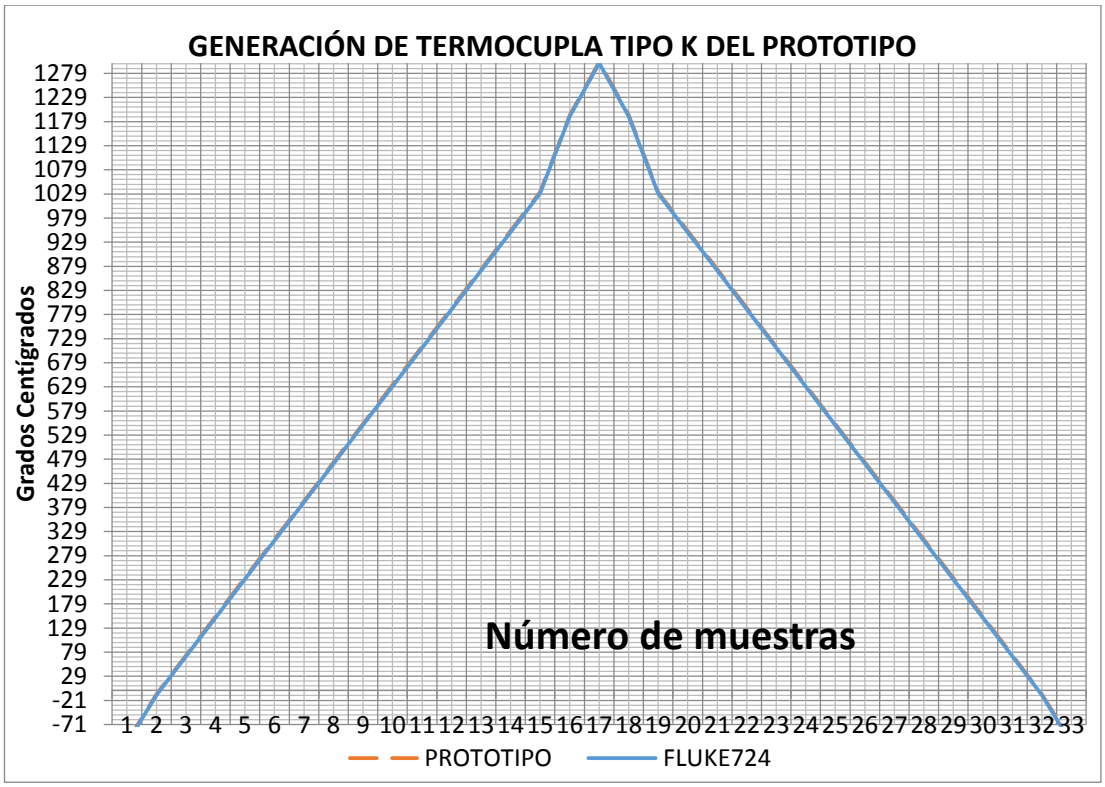


Figura 75: Curva de respuesta de la generación de termocupla tipo K.

4.3 Prueba de Calibración de un Transmisor de Temperatura

La **figura 76** muestra la conexión necesaria del prototipo implementado con el transmisor de temperatura ROSEMOUNT, este procedimiento se lo realizará mediante la generación de una RTD PT100 de 2 hilos y el transmisor será alimentado por la fuente de 24 voltios que posee el propio equipo, los valores obtenidos de esta calibración serán comparados con una calibración realizada por el calibrador Fluke 724 y se obtendrá el porcentaje de error de acuerdo a la diferencia entre las dos calibraciones realizadas.

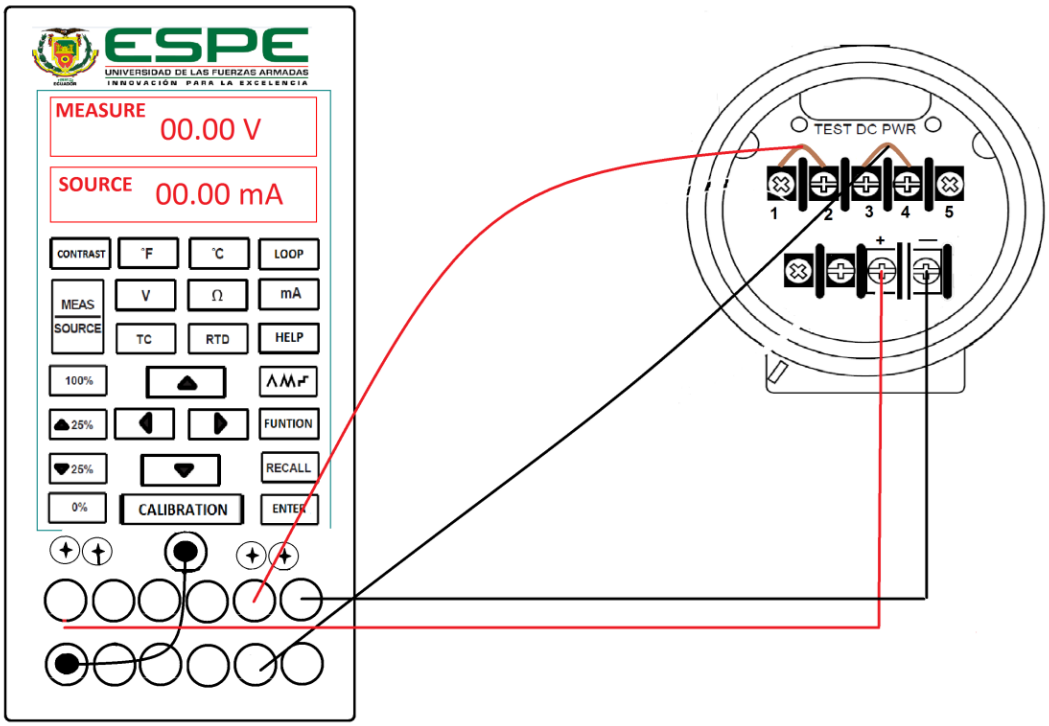


Figura 76: Conexión para la calibración del transmisor de temperatura.

Tabla 20:
Prueba de calibración con el prototipo.

PRUEBA DE CALIBRACIÓN					
No. De Prueba	**Porcentaje**	**Valor Deseado**	**Valor medido Calibrador**	**Valor medido Prototipo**	**ERROR**
1	0	4,00	4,00	3,86	-0,88
2	25	8,00	8,00	7,87	-0,81
3	50	12,00	12,00	11,86	-0,88
4	75	16,00	16,00	16,18	1,13
5	100	20,00	20,00	20,13	0,81
6	75	16,00	16,00	16,15	0,94
7	50	12,00	12,00	11,81	-1,19
8	25	8,00	8,00	7,85	-0,94
9	0	4,00	4,00	3,91	-0,56
			PROMEDIO		**-0,26**
			VARIANZA		**0,8718533**
			DESVIACIÓN ESTÁNDAR		**0,93373085**

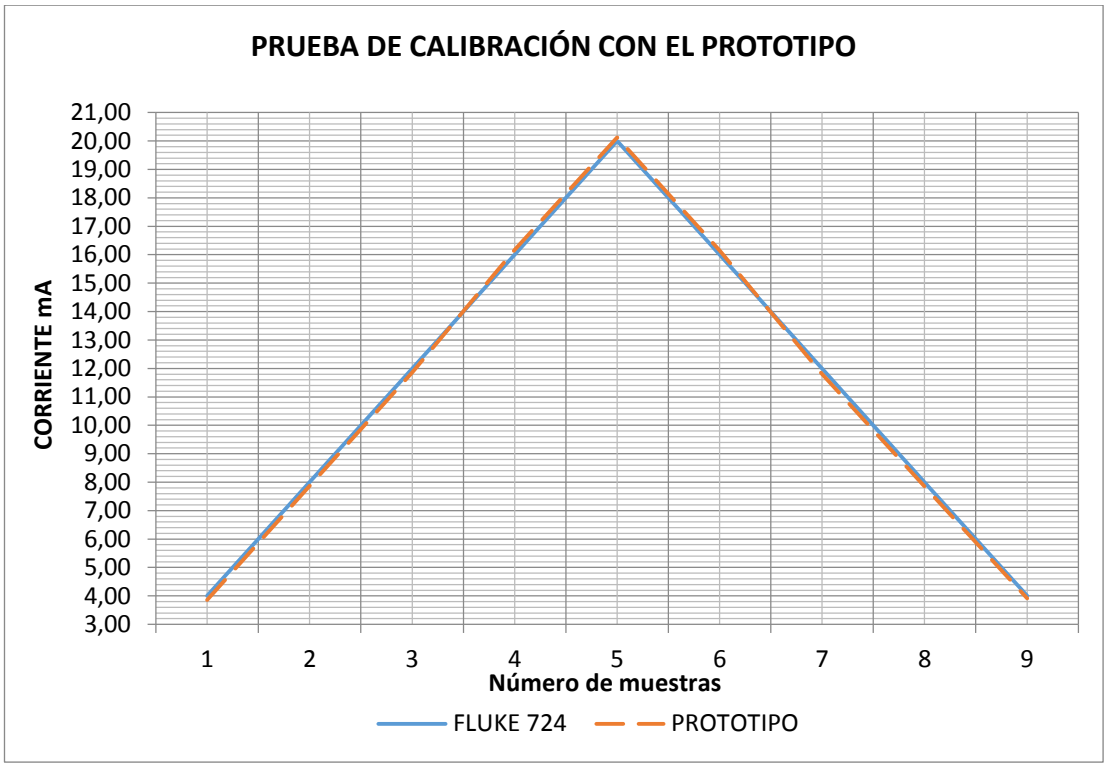


Figura 77: Curva de respuesta de calibración con el prototipo.

4.4 Prueba de Generación de Certificado de Calibracción

Para generar el documento con los datos de temperatura, medición de corriente y el porcentaje de error que se generan en la calibración de un transmisor de temperatura, se debe abrir el símbolo del sistema **figura 78** y escribir la sentencia **python** junto al nombre y extensión del archivo en el cual se toman los datos y se genera un archivo de extensión .pdf, en este caso documento.py después ingresamos el número del puerto serial al cual se conecta el arduino, a continuación se presentan los datos de calibración y el nombre del archivo generado **“Documento de Calibración”**.

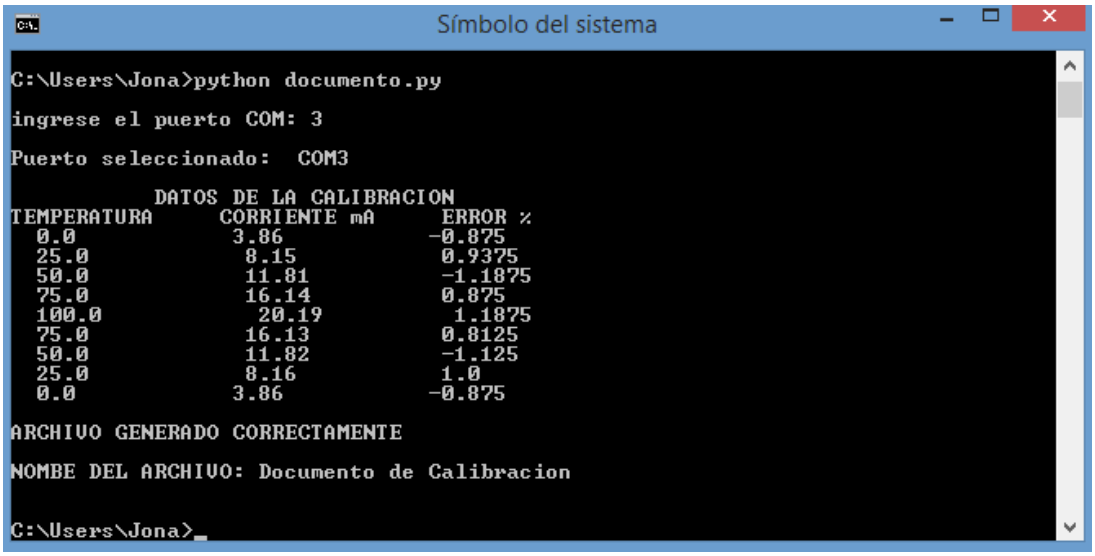
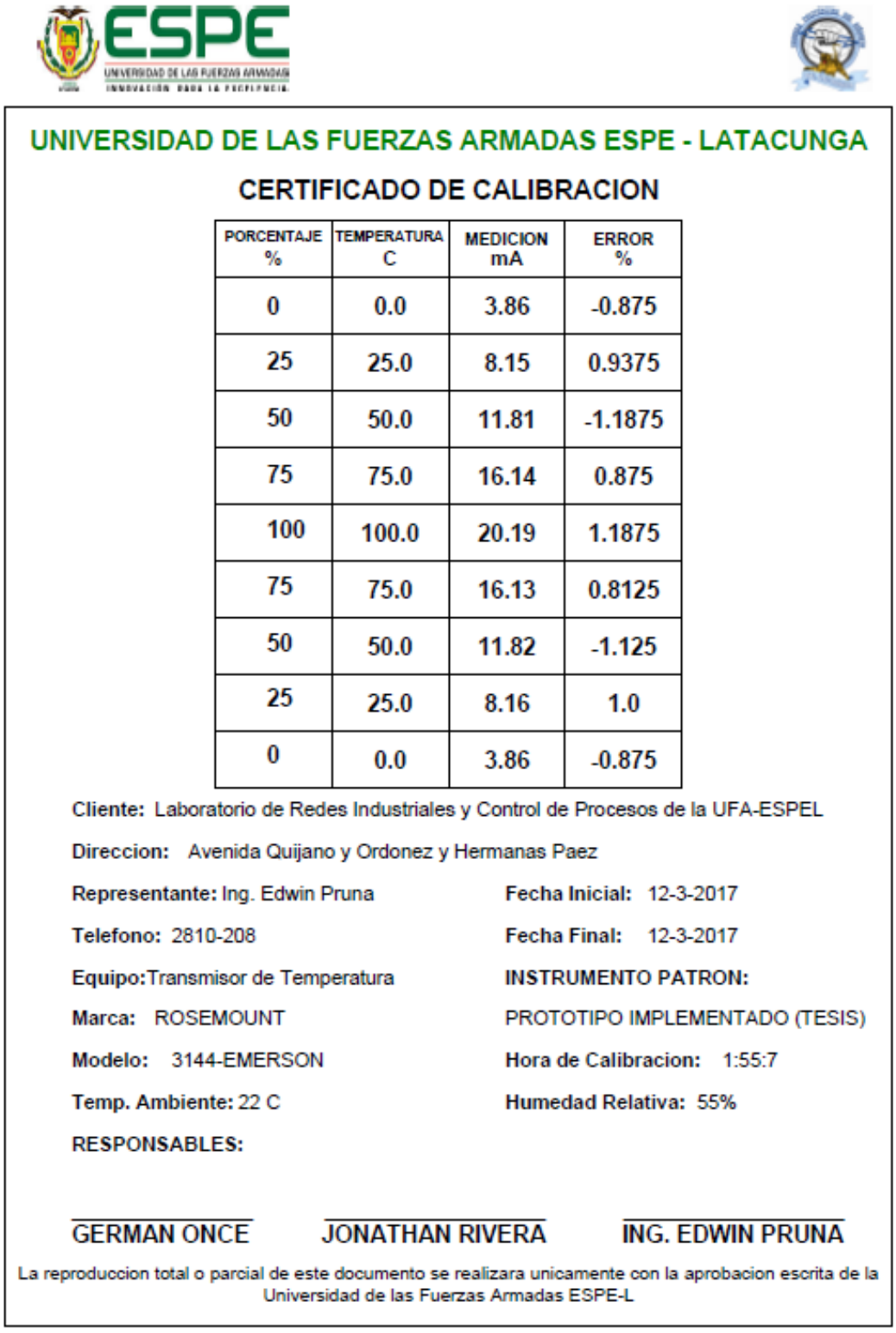


Figura 78: Ejecución de la generación del documento de calibración.

Luego de los pasos realizados en el índice 4.4 se genera un documento mostrado en la figura 79.



UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE - LATACUNGA

CERTIFICADO DE CALIBRACION

PORCENTAJE %	TEMPERATURA C	MEDICION mA	ERROR %
0	0.0	3.86	-0.875
25	25.0	8.15	0.9375
50	50.0	11.81	-1.1875
75	75.0	16.14	0.875
100	100.0	20.19	1.1875
75	75.0	16.13	0.8125
50	50.0	11.82	-1.125
25	25.0	8.16	1.0
0	0.0	3.86	-0.875

Cliente: Laboratorio de Redes Industriales y Control de Procesos de la UFA-ESPEL

Direccion: Avenida Quijano y Ordonez y Hermanas Paez

Representante: Ing. Edwin Pruna

Telefono: 2810-208

Equipo: Transmisor de Temperatura

Marca: ROSEMOUNT

Modelo: 3144-EMERSON

Temp. Ambiente: 22 C

Fecha Inicial: 12-3-2017

Fecha Final: 12-3-2017

INSTRUMENTO PATRON:

PROTOTIPO IMPLEMENTADO (TESIS)

Hora de Calibracion: 1:55:7

Humedad Relativa: 55%

RESPONSABLES:

GERMAN ONCE JONATHAN RIVERA ING. EDWIN PRUNA

Figura 79: Documento de Calibración.

4.5 Alacances y Limitaciones

4.5.1 Alcances

En el diseño y construcción del calibrador – documentador de procesos se determinaron los siguientes alcances:

- El calibrador brinda un entorno amigable e intuitivo al usuario mediante la pantalla táctil de siete pulgadas.
- Permite desarrollar la destreza de calibración de transmisores de temperatura mediante la simulación de RTD PT100 de 2 hilos.
- El calibrador posee la función de generar corriente DC en el rango de 0 a 24 mA,que es muy útil en los procesos industriales.
- La función de calibración del equipo implementado es automática, el usuario únicamente ingresa los valores requeridos para dicha operación y los resultados se presentan en la pantalla.
- El calibrador – documentador de procesos genera un documento con la extensión .pdf que presenta los datos obtenidos de la calibración

4.5.2 Limitaciones

De igual manera se pudo observar limitaciones del calibrador – documentador de Procesos, mismas que se detallan a continuación:

- El calibrador – documentador de procesos únicamente es capaz de simular la RTD PT100 de 2 hilos y las termocuplas tipo J y K que son las más comunes.
- El calibrador de procesos únicamente puede realizar el proceso de calibración con la simulación de RTD PT100 de 2 hilos.

- El calibrador – documentador de procesos, debido al alto consumo de corriente siempre debe estar conectado a la red eléctrica, con lo cual se asegura que todos los acondicionamientos permanecen estables.
- La generación de resistencia y RTD del equipo, en ocaciones difiere de su valor calibrado por las propias características de los integrados.

CAPÍTULO V

5 CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Luego de culminar el proyecto se obtuvo las siguientes conclusiones:

- El prototipo permite realizar las funciones de medición, simulación y generación de variables eléctricas, con una función adicional que es la generación de un documento digital con los datos de calibración, el cual permitirá una mejor manera de análisis de los mismos.

- El prototipo implementado necesita de una fuente de voltaje DC constante y estable, ya que los acondicionamientos empleados en la medición no deben tener variaciones de voltaje para que la lectura sea correcta, el equipo consume una corriente alrededor de 1 amperio por lo que no es posible que funcione con baterías recargables ya que las mismas se descargan rápidamente.

- Los errores se obtuvieron en base al Calibrador de Temperatura FLUKE724 como instrumento patrón, las mediciones realizadas con el instrumento son menores al 0.3% siendo un rango valido para el desarrollo de prácticas de laboratorio en la temática de calibración de instrumentos de temperatura.

- Para la comprobación de la etapa de generación y simulación de variables físicas del prototipo, se utilizó como instrumento patrón el Calibrador de Temperatura FLUKE724, obteniendo un valor menor al 0.4%, siendo un rango valido para el desarrollo de prácticas de laboratorio en la temática de calibración de instrumentos de temperatura.

- Para la comunicación de las tarjetas utilizadas en el proyecto se realiza mediante protocolo I2C, dado la diferencia de tensión de operación Arduino Mega 5 voltios y Arduino Due 3.3 se utiliza un conversor de niveles para evitar que la tarjeta Arduino Due sufra daños.

- Se realizó el procedimiento de la función calibración del prototipo, el cual se implementó la simulación de una RTD PT100 de 2 hilos en un transmisor de temperatura ROSEMOUNT 3144P con un rango de operación de 0 °C a 100 °C, el error obtenido menor al 1.3% en comparación calibrador documentador FLUKE 744, dicho calibrador fue utilizado para este proceso debido a que el FLUKE724 no posee la función de calibración automática.

5.2 RECOMENDACIONES

- Debido a que la tarjeta arduino due posee un ADC de 12 bits, se recomienda utilizar un ADC de al menos de 24 bits para realizar una medición más estable y con mayor resolución de las variables físicas.

- Para generar el certificado de calibración, se recomienda conectar a la PC después del proceso de calibración ya que de otra manera se produce una caída de tensión y por ende un diferente valor de medición de corriente del transmisor.

- Se recomienda esperar 2 segundos para observar los valores de medición de variables físicas para que el equipo muestre la medida real de la variable.

- Para obtener el certificado de calibración se recomienda instalar python con todas las librerías necesarias en la PC a la cual se le conecte el prototipo.

- Se recomienda utilizar el prototipo implementado como un apoyo académico en clase del calibrador FLUKE 724.

REFERENCIAS BIBLIOGRÁFICAS

Antioquia, L. M. (2010). *Laboratorio Metrológico de Antioquia*. Recuperado el 01 de 03 de 2016, de http://www.laboratoriometrologico.com/sitio/contenidos_mo.php?it=184

Arduino. (2016). *Arduino-usa*. Recuperado el 18 de Marzo de 2016, de Arduino-usa: http://firmata.sourceforge.net/Design_Issues

Artero, Ó. T. (2013). Arduino : curso práctico de formación. En Ó. T. Artero, *Arduino : curso práctico de formación* (págs. 80-115). Madrid: RClibros.

Barriga. (21 de Enero de 2013). *WIRING.* Recuperado el 17 de Marzo de 2016, de WIRING.: http://www.wiring.org.co/

Ben Fry, C. R. (2004). *PROCESSING2:*. Recuperado el 17 de Marzo de 2016, de PROCESSING2:: http://www.processing.org/

cecuamaq. (2016). *cecuamaq*. Recuperado el 15 de Marzo de 2016, de cecuamaq: http://www.cecuamaq.com/pdf/Fluke/Calibradores%20de%20Campo.pdf

clerus. (2010). Recuperado el 19 de Marzo de 2016, de clerus: http://www.clerus.org/

CV, I. C. (2016). *INGENIERIA CONTROL Y TEMPERATURA SA DE CV*. (icytsa) Recuperado el 15 de 02 de 2016, de http://icytsa.com.mx/1976378_FABRICACION-DE-RTDS--PT-100.html

Danica Schwarzkopf, P. C. (19 de Diciembre de 2012). *bloginstrumentacion*. Recuperado el 15 de Marzo de 2016, de http://www.bloginstrumentacion.com/blog/2012/12/19/3058/

FLUKE. (1995-2016). Recuperado el 10 de 03 de 2016, de FLUKE: http://la.flukecal.com/products/process-calibration-tools/temperature-calibrators/handheld-temperature-calibrators/calibr-0

Galindo, M. J. (2010). Escaneando la informática. En M. J. Galindo, *Escaneando la informática* (págs. 130-150). Barcelona: Editorial UOC.

Guerrero, I. S. (Julio de 2002). *Termopares*. (Metas) Recuperado el 12 de 02 de 2016, de ww.mw¿etas.com.mx

hispalinux. (2010). *hispalinux*. Recuperado el 18 de Marzo de 2016, de hispalinux: http://hispalinux.es/

Juan Carlos Álvarez Antón, L. M. (2007). Introduccion al analisis de circuitos electricos. En *Introduccion al analisis de circuitos electricos* (págs. 5-9). Oviedo: eduino.

MARTINEZ, E. R. (07 de Febrero de 2015). *electrolabmedic*. Recuperado el 16 de Marzo de 2016, de http://www.electrolabmedic.com/joomla/index.php/fluke/fluke-cal-proc

Ojeda, L. T. (2016). *Arduino*. Recuperado el 17 de Marzo de 2016, de Arduino: http://arduino.cl/arduino-due/

Ramos, S. R. (2014). Instrumentación y control en instalaciones de proceso, energía y servicios auxiliares. En *Instrumentación y control en instalaciones de proceso, energía y servicios auxiliares.* (págs. 70-80). ic editorial.

Rosa, J. M. (octubre de 2014). *Temariosformativos*. Recuperado el 15 de 02 de 2016, de http://jesuitaseduca.com/pluginfile.php/3499/mod_resource/content/1/Termopares.pdf

Ruben, S. (2011). *ingeniatic*. Recuperado el 25 de Febrero de 2016, de ingeniatic: http://ingeniatic.euitt.upm.es/

srl, A. (2016). *Arduino srl_itlaia*. Recuperado el 17 de Marzo de 2016, de Arduino srl_itlaia: http://www.arduino.org/products/boards/4-arduino-boards/arduino-due

Villajulca, J. C. (09 de Febreo de 2011). *intrumentacion y control.net*. Recuperado el 15 de Marzo de 2016, de intrumentacion y control.net: http://www.instrumentacionycontrol.net/cursos-libres/instrumentacion/curso-practico-de-instrumentacion/item/352-usando-calibradores-de-lazo-medici%C3%B3n-generaci%C3%B3n-y-simulaci%C3%B3n.html

wordpress. (2016). *wordpress*. Recuperado el 18 de Marzo de 2016, de wordpress: https://aprendiendoarduino.wordpress.com

ANEXOS

ANEXO A
PROGRAMACIÓN ARDUINO DUE

```
/*************************************************************************
 * LIBRERIAS PARA EL MANEJO DE LA PANTALLA TACTIL
*************************************************************************/
#include <Wire.h>
#include <UTFT.h>
#include <UTouch.h>
#include <Adafruit_MAX31856.h>
#include <ResponsiveAnalogRead.h>
#include <Filters.h>
/*************************************************************************
 * LIBRERIAS PARA EL MANEJO DE IMAGENES
*************************************************************************/
#if defined(__AVR__)
#define imagedatatype  unsigned int
#elif defined(__PIC32MX__)
#define imagedatatype  unsigned short
#elif defined(__arm__)
#define imagedatatype  unsigned short
#endif
/*************************************************************************
 * PINES DE CONFIGURACION DE LA PANTALLA TACTIL
*************************************************************************/
UTFT myGLCD(TFT01_70, 38, 39, 40, 41);
UTouch  myTouch( 6, 5, 4, 3, 2);
Adafruit_MAX31856 max = Adafruit_MAX31856(10, 75, 74, 76);
/*************************************************************************
 * CLASES DE FORMAS DE LETRA
*************************************************************************/
extern uint8_t SmallFont[];
extern uint8_t BigFont[];
extern uint8_t various_symbols[];
extern uint8_t Grotesk32x64[];
/*************************************************************************
 * CLASES DE IMAGENES
*************************************************************************/
extern imagedatatype espe1[];
extern imagedatatype Electronica[];

/*************************************************************************
 * RTD PT1000
*************************************************************************/
float Pt100_tabla [] = {
22.83,  22.4,  21.97,  21.54,  21.11,  20.68,  20.25,  19.82,  19.38,  18.95,
27.1,  26.67,  26.24,  25.82,  25.39,  24.97,  24.54,  24.11,  23.68,  23.25,
31.34,  30.91,  30.49,  30.07,  29.64,  29.22,  28.8,  28.37,  27.95,  27.52,
35.54,  35.12,  34.7,  34.28,  33.86,  33.44,  33.02,  32.6,  32.18,  31.76,
39.72,  39.31,  38.89,  38.47,  38.05,  37.64,  37.22,  36.8,  36.38,  35.96,
43.88,  43.46,  43.05,  42.63,  42.22,  41.8,  41.39,  40.97,  40.56,  40.14,
48.00,  47.59,  47.18,  46.77,  46.36,  45.94,  45.53,  45.12,  44.7,  44.29,
52.11,  51.7,  51.29,  50.88,  50.47,  50.06,  49.65,  49.24,  48.83,  48.42,
```

56.19, 55.79, 55.38, 54.97, 54.56, 54.15, 53.75, 53.34, 52.93, 52.52,
60.26, 59.85, 59.44, 59.04, 58.63, 58.23, 57.82, 57.41, 57.01, 56.6,
64.3, 63.9, 63.49, 63.09, 62.68, 62.28, 61.88, 61.47, 61.07, 60.66,
68.33, 67.92, 67.52, 67.12, 66.72, 66.31, 65.91, 65.51, 65.11, 64.7,
72.33, 71.93, 71.53, 71.13, 70.73, 70.33, 69.93, 69.53, 69.13, 68.73,
76.33, 75.93, 75.53, 75.13, 74.73, 74.33, 73.93, 73.53, 73.13, 72.73,
80.31, 79.91, 79.51, 79.11, 78.72, 78.32, 77.92, 77.52, 77.12, 76.73,
84.27, 83.87, 83.48, 83.08, 82.69, 82.29, 81.89, 81.5, 81.1, 80.7,
88.22, 87.83, 87.43, 87.04, 86.64, 86.25, 85.85, 85.46, 85.06, 84.67,
92.16, 91.77, 91.37, 90.98, 90.59, 90.19, 89.8, 89.4, 89.01, 88.62,
96.09, 95.69, 95.3, 94.91, 94.52, 94.12, 93.73, 93.34, 92.95, 92.55,
100.00, 99.61, 99.22, 98.83, 98.44, 98.04, 97.65, 97.26, 96.87, 96.48,
100.00, 100.39, 100.78, 101.17, 101.56, 101.95, 102.34, 102.73, 103.12, 103.51,
103.9, 104.29, 104.68, 105.07, 105.46, 105.85, 106.24, 106.63, 107.02, 107.4,
107.79, 108.18, 108.57, 108.96, 109.35, 109.73, 110.12, 110.51, 110.9, 111.29,
111.67, 112.06, 112.45, 112.83, 113.22, 113.61, 114.00, 114.38, 114.77, 115.15,
115.54, 115.93, 116.31, 116.7, 117.08, 117.47, 117.86, 118.24, 118.63, 119.01,
119.4, 119.78, 120.17, 120.55, 120.94, 121.32, 121.71, 122.09, 122.47, 122.86,
123.24, 123.63, 124.01, 124.39, 124.78, 125.16, 125.54, 125.93, 126.31, 126.69,
127.08, 127.46, 127.84, 128.22, 128.61, 128.99, 129.37, 129.75, 130.13, 130.52,
130.9, 131.28, 131.66, 132.04, 132.42, 132.8, 133.18, 133.57, 133.95, 134.33,
134.71, 135.09, 135.47, 135.85, 136.23, 136.61, 136.99, 137.37, 137.75, 138.13,
138.51, 138.88, 139.26, 139.64, 140.02, 140.4, 140.78, 141.16, 141.54, 141.91,
142.29, 142.67, 143.05, 143.43, 143.8, 144.18, 144.56, 144.94, 145.31, 145.69,
146.07, 146.44, 146.82, 147.2, 147.57, 147.95, 148.33, 148.7, 149.08, 149.46,
149.83, 150.21, 150.58, 150.96, 151.33, 151.71, 152.08, 152.46, 152.83, 153.21,
153.58, 153.96, 154.33, 154.71, 155.08, 155.46, 155.83, 156.2, 156.58, 156.95,
157.33, 157.7, 158.07, 158.45, 158.82, 159.19, 159.56, 159.94, 160.31, 160.68,
161.05, 161.43, 161.8, 162.17, 162.54, 162.91, 163.29, 163.66, 164.03, 164.4,
164.77, 165.14, 165.51, 165.89, 166.26, 166.63, 167.00, 167.37, 167.74, 168.11,
168.48, 168.85, 169.22, 169.59, 169.96, 170.33, 170.7, 171.07, 171.43, 171.8,
172.17, 172.54, 172.91, 173.28, 173.65, 174.02, 174.38, 174.75, 175.12, 175.49,
175.86, 176.22, 176.59, 176.96, 177.33, 177.69, 178.06, 178.43, 178.79, 179.16,
179.53, 179.89, 180.26, 180.63, 180.99, 181.36, 181.72, 182.09, 182.46, 182.82,
183.19, 183.55, 183.92, 184.28, 184.65, 185.01, 185.38, 185.74, 186.11, 186.47,
186.84, 187.2, 187.56, 187.93, 188.29, 188.66, 189.02, 189.38, 189.75, 190.11,
190.47, 190.84, 191.2, 191.56, 191.92, 192.29, 192.65, 193.01, 193.37, 193.74,
194.1, 194.46, 194.82, 195.18, 195.55, 195.91, 196.27, 196.63, 196.99, 197.35,
197.71, 198.07, 198.43, 198.79, 199.15, 199.51, 199.87, 200.23, 200.59, 200.95,
201.31, 201.67, 202.03, 202.39, 202.75, 203.11, 203.47, 203.83, 204.19, 204.55,
204.9, 205.26, 205.62, 205.98, 206.34, 206.7, 207.05, 207.41, 207.77, 208.13,
208.48, 208.84, 209.2, 209.56, 209.91, 210.27, 210.63, 210.98, 211.34, 211.7,
212.05, 212.41, 212.76, 213.12, 213.48, 213.83, 214.19, 214.54, 214.9, 215.25,
215.61, 215.96, 216.32, 216.67, 217.03, 217.38, 217.74, 218.09, 218.44, 218.8,
219.15, 219.51, 219.86, 220.21, 220.57, 220.92, 221.27, 221.63, 221.98, 222.33,
222.68, 223.04, 223.39, 223.74, 224.09, 224.45, 224.8, 225.15, 225.5, 225.85,
226.21, 226.56, 226.91, 227.26, 227.61, 227.96, 228.31, 228.66, 229.02, 229.37,
229.72, 230.07, 230.42, 230.77, 231.12, 231.47, 231.82, 232.17, 232.52, 232.87,
233.21, 233.56, 233.91, 234.26, 234.61, 234.96, 235.31, 235.66, 236.00, 236.35,
236.7, 237.05, 237.4, 237.74, 238.09, 238.44, 238.79, 239.13, 239.48, 239.83,
240.18, 240.52, 240.87, 241.22, 241.56, 241.91, 242.26, 242.6, 242.95, 243.29,
243.64, 243.99, 244.33, 244.68, 245.02, 245.37, 245.71, 246.06, 246.4, 246.75,

```
247.09, 247.44, 247.78, 248.13, 248.47, 248.81, 249.16, 249.5, 249.85, 250.19,
250.53, 250.88, 251.22, 251.56, 251.91, 252.25, 252.59, 252.93, 253.28, 253.62,
253.96, 254.3, 254.65, 254.99, 255.33, 255.67, 256.01, 256.35, 256.7, 257.04,
257.38, 257.72, 258.06, 258.4, 258.74, 259.08, 259.42, 259.76, 260.1, 260.44,
260.78, 261.12, 261.46, 261.8, 262.14, 262.48, 262.82, 263.16, 263.5, 263.84,
264.18, 264.52, 264.86, 265.2, 265.53, 265.87, 266.21, 266.55, 266.89, 267.22,
267.56, 267.9, 268.24, 268.57, 268.91, 269.25, 269.59, 269.92, 270.26, 270.6,
270.93, 271.27, 271.61, 271.94, 272.28, 272.61, 272.95, 273.29, 273.62, 273.96,
274.29, 274.63, 274.96, 275.3, 275.63, 275.97, 276.3, 276.64, 276.97, 277.31,
277.64, 277.98, 278.31, 278.64, 278.98, 279.31, 279.64, 279.98, 280.31, 280.64,
280.98, 281.31, 281.64, 281.98, 282.31, 282.64, 282.97, 283.31, 283.64, 283.97,
284.3, 284.63, 284.97, 285.3, 285.63, 285.96, 286.29, 286.62, 286.95, 287.29,
287.62, 287.95, 288.28, 288.61, 288.94, 289.27, 289.6, 289.93, 290.26, 290.59,
290.92, 291.25, 291.58, 291.91, 292.24, 292.56, 292.89, 293.22, 293.55, 293.88,
294.21, 294.54, 294.86, 295.19, 295.52, 295.85, 296.18, 296.5, 296.83, 297.16,
297.49, 297.81, 298.14, 298.47, 298.8, 299.12, 299.45, 299.78, 300.1, 300.43,
300.75, 301.08, 301.41, 301.73, 302.06, 302.38, 302.71, 303.03, 303.36, 303.69,
304.01, 304.34, 304.66, 304.98, 305.31, 305.63, 305.96, 306.28, 306.61, 306.93,
307.25, 307.58, 307.9, 308.23, 308.55, 308.87, 309.2, 309.52, 309.84, 310.16,
310.49, 310.81, 311.13, 311.45, 311.78, 312.1, 312.42, 312.74, 313.06, 313.39,
313.71, 314.03, 314.35, 314.67, 314.99, 315.31, 315.64, 315.96, 316.28, 316.6,
316.92, 317.24, 317.56, 317.88, 318.2, 318.52, 318.84, 319.16, 319.48, 319.8,
320.12, 320.43, 320.75, 321.07, 321.39, 321.71, 322.03, 322.35, 322.67, 322.98,
323.3, 323.62, 323.94, 324.26, 324.57, 324.89, 325.21, 325.53, 325.84, 326.16,
326.48, 326.79, 327.11, 327.43, 327.74, 328.06, 328.38, 328.69, 329.01, 329.32,
329.64, 329.96, 330.27, 330.59, 330.9, 331.22, 331.53, 331.85, 332.16, 332.48,
332.79, 333.11, 333.42, 333.74, 334.05, 334.36, 334.68, 334.99, 335.31, 335.62,
335.93, 336.25, 336.56, 336.87, 337.18, 337.5, 337.81, 338.12, 338.44, 338.75,
339.06, 339.37, 339.69, 340.00, 340.31, 340.62, 340.93, 341.24, 341.56, 341.87,
342.18, 342.49, 342.8, 343.11, 343.42, 343.73, 344.04, 344.35, 344.66, 344.97,
345.28, 345.59, 345.9, 346.21, 346.52, 346.83, 347.14, 347.45, 347.76, 348.07,
348.38, 348.69, 348.99, 349.3, 349.61, 349.92, 350.23, 350.54, 350.84, 351.15,
351.46, 351.77, 352.08, 352.38, 352.69, 353.00, 353.3, 353.61, 353.92, 354.22,
354.53, 354.84, 355.14, 355.45, 355.76, 356.06, 356.37, 356.67, 356.98, 357.28,
357.59, 357.9, 358.2, 358.51, 358.81, 359.12, 359.42, 359.72, 360.03, 360.33,
360.64, 360.94, 361.25, 361.55, 361.85, 362.16, 362.46, 362.76, 363.07, 363.37,
363.67, 363.98, 364.28, 364.58, 364.89, 365.19, 365.49, 365.79, 366.1, 366.4,
366.7, 367.00, 367.3, 367.6, 367.91, 368.21, 368.51, 368.81, 369.11, 369.41,
369.71, 370.01, 370.31, 370.61, 370.91, 371.21, 371.51, 371.81, 372.11, 372.41,
372.71, 373.01, 373.31, 373.61, 373.91, 374.21, 374.51, 374.81, 375.11, 375.41,
375.7, 376.00, 376.3, 376.6, 376.9, 377.19, 377.49, 377.79, 378.09, 378.39,
378.68, 378.98, 379.28, 379.57, 379.87, 380.17, 380.46, 380.76, 381.06, 381.35,
381.65, 381.95, 382.24, 382.54, 382.83, 383.13, 383.42, 383.72, 384.01, 384.31,
384.6, 384.9, 385.19, 385.49, 385.78, 386.08, 386.37, 386.67, 386.96, 387.25,
387.55, 387.84, 388.14, 388.43, 388.72, 389.02, 389.31, 389.6, 389.9, 390.19,
};
/*****************************************************************
 * TERMOCUPLA K
*****************************************************************/
float TermocuplaK [] = {
-6.441, -6.444, -6.446, -6.448, -6.450, -6.452, -6.453, -6.455, -6.456, -6.457,
-6.404, -6.408, -6.413, -6.417, -6.421, -6.425, -6.429, -6.432, -6.435, -6.438,
```

-6.344, -6.351, -6.358, -6.364, -6.371, -6.377, -6.382, -6.388, -6.394, -6.399,
-6.262, -6.271, -6.280, -6.289, -6.297, -6.306, -6.314, -6.322, -6.329, -6.337,
-6.158, -6.170, -6.181, -6.192, -6.202, -6.213, -6.223, -6.233, -6.243, -6.253,
-6.035, -6.048, -6.061, -6.074, -6.087, -6.099, -6.111, -6.123, -6.135, -6.147,
-5.891, -5.907, -5.922, -5.936, -5.951, -5.965, -5.980, -5.994, -6.007, -6.021,
-5.730, -5.747, -5.763, -5.780, -5.796, -5.813, -5.829, -5.845, -5.860, -5.876,
-5.550, -5.569, -5.587, -5.606, -5.624, -5.642, -5.660, -5.678, -5.695, -5.712,
-5.354, -5.374, -5.394, -5.414, -5.434, -5.454, -5.474, -5.493, -5.512, -5.531,
-5.141, -5.163, -5.185, -5.207, -5.228, -5.249, -5.271, -5.292, -5.313, -5.333,
-4.912, -4.936, -4.959, -4.983, -5.006, -5.029, -5.051, -5.074, -5.097, -5.119,
-4.669, -4.694, -4.719, -4.743, -4.768, -4.792, -4.817, -4.841, -4.865, -4.889,
-4.410, -4.437, -4.463, -4.489, -4.515, -4.541, -4.567, -4.593, -4.618, -4.644,
-4.138, -4.166, -4.193, -4.221, -4.248, -4.276, -4.303, -4.330, -4.357, -4.384,
-3.852, -3.881, -3.910, -3.939, -3.968, -3.997, -4.025, -4.053, -4.082, -4.110,
-3.553, -3.584, -3.614, -3.644, -3.674, -3.704, -3.734, -3.764, -3.793, -3.823,
-3.242, -3.274, -3.305, -3.337, -3.368, -3.399, -3.430, -3.461, -3.492, -3.523,
-2.920, -2.953, -2.985, -3.018, -3.050, -3.082, -3.115, -3.147, -3.179, -3.211,
-2.586, -2.620, -2.654, -2.687, -2.721, -2.754, -2.788, -2.821, -2.854, -2.887,
-2.243, -2.277, -2.312, -2.347, -2.381, -2.416, -2.450, -2.484, -2.518, -2.552,
-1.889, -1.925, -1.961, -1.996, -2.032, -2.067, -2.102, -2.137, -2.173, -2.208,
-1.527, -1.563, -1.600, -1.636, -1.673, -1.709, -1.745, -1.781, -1.817, -1.853,
-1.156, -1.193, -1.231, -1.268, -1.305, -1.342, -1.379, -1.416, -1.453, -1.490,
-0.777, -0.816, -0.854, -0.892, -0.930, -0.968, -1.005, -1.043, -1.081, -1.118,
-0.392, -0.431, -0.469, -0.508, -0.547, -0.585, -0.624, -0.662, -0.701, -0.739,
0, -0.039, -0.079, -0.118, -0.157, -0.197, -0.236, -0.275, -0.314, -0.353,
0.397, 0.437, 0.477, 0.517, 0.557, 0.597, 0.637, 0.677, 0.718, 0.758,
0.798, 0.838, 0.879, 0.919, 0.960, 1.000, 1.041, 1.081, 1.122, 1.162,
1.203, 1.244, 1.285, 1.325, 1.366, 1.407, 1.448, 1.489, 1.529, 1.570,
1.611, 1.652, 1.693, 1.734, 1.776, 1.817, 1.858, 1.899, 1.940, 1.981,
2.022, 2.064, 2.105, 2.146, 2.188, 2.229, 2.270, 2.312, 2.353, 2.394,
2.436, 2.477, 2.519, 2.560, 2.601, 2.643, 2.684, 2.726, 2.767, 2.809,
2.850, 2.892, 2.933, 2.975, 3.016, 3.058, 3.100, 3.141, 3.183, 3.224,
3.266, 3.307, 3.349, 3.390, 3.432, 3.473, 3.515, 3.556, 3.598, 3.639,
3.681, 3.722, 3.764, 3.805, 3.847, 3.888, 3.930, 3.971, 4.012, 4.054,
4.095, 4.137, 4.178, 4.219, 4.261, 4.302, 4.343, 4.384, 4.426, 4.467,
4.508, 4.549, 4.590, 4.632, 4.673, 4.714, 4.755, 4.796, 4.837, 4.878,
4.919, 4.960, 5.001, 5.042, 5.083, 5.124, 5.164, 5.205, 5.246, 5.287,
5.327, 5.368, 5.409, 5.450, 5.490, 5.531, 5.571, 5.612, 5.652, 5.693,
5.733, 5.774, 5.814, 5.855, 5.895, 5.936, 5.976, 6.016, 6.057, 6.097,
6.137, 6.177, 6.218, 6.258, 6.298, 6.338, 6.378, 6.419, 6.459, 6.499,
6.539, 6.579, 6.619, 6.659, 6.699, 6.739, 6.779, 6.819, 6.859, 6.899,
6.939, 6.979, 7.019, 7.059, 7.099, 7.139, 7.179, 7.219, 7.259, 7.299,
7.338, 7.378, 7.418, 7.458, 7.498, 7.538, 7.578, 7.618, 7.658, 7.697,
7.737, 7.777, 7.817, 7.857, 7.897, 7.937, 7.977, 8.017, 8.057, 8.097,
8.137, 8.177, 8.216, 8.256, 8.296, 8.336, 8.376, 8.416, 8.456, 8.497,
8.537, 8.577, 8.617, 8.657, 8.697, 8.737, 8.777, 8.817, 8.857, 8.898,
8.938, 8.978, 9.018, 9.058, 9.099, 9.139, 9.179, 9.220, 9.260, 9.300,
9.341, 9.381, 9.421, 9.462, 9.502, 9.543, 9.583, 9.624, 9.664, 9.705,
9.745, 9.786, 9.826, 9.867, 9.907, 9.948, 9.989, 10.029, 10.070, 10.111,
10.151, 10.192, 10.233, 10.274, 10.315, 10.355, 10.396, 10.437, 10.478, 10.519,
10.560, 10.600, 10.641, 10.682, 10.723, 10.764, 10.805, 10.846, 10.887, 10.928,
10.969, 11.010, 11.051, 11.093, 11.134, 11.175, 11.216, 11.257, 11.298, 11.339,

11.381, 11.422, 11.463, 11.504, 11.546, 11.587, 11.628, 11.669, 11.711, 11.752,
11.793, 11.835, 11.876, 11.918, 11.959, 12.000, 12.042, 12.083, 12.125, 12.166,
12.207, 12.249, 12.290, 12.332, 12.373, 12.415, 12.456, 12.498, 12.539, 12.581,
12.623, 12.664, 12.706, 12.747, 12.789, 12.831, 12.872, 12.914, 12.955, 12.997,
13.039, 13.080, 13.122, 13.164, 13.205, 13.247, 13.289, 13.331, 13.372, 13.414,
13.456, 13.497, 13.539, 13.581, 13.623, 13.665, 13.706, 13.748, 13.790, 13.832,
13.874, 13.915, 13.957, 13.999, 14.041, 14.083, 14.125, 14.167, 14.208, 14.250,
14.292, 14.334, 14.376, 14.418, 14.460, 14.502, 14.544, 14.586, 14.628, 14.670,
14.712, 14.754, 14.796, 14.838, 14.880, 14.922, 14.964, 15.006, 15.048, 15.090,
15.132, 15.174, 15.216, 15.258, 15.300, 15.342, 15.384, 15.426, 15.468, 15.510,
15.552, 15.594, 15.636, 15.679, 15.721, 15.763 , 15.805, 15.847, 15.889, 15.931,
15.974, 16.016, 16.058, 16.100, 16.142, 16.184 , 16.227, 16.269 ,16.311, 16.353,
16.395, 16.438, 16.480, 16.522, 16.564, 16.607 , 16.649, 16.691 ,16.733, 16.776,
16.818, 16.860, 16.902, 16.945, 16.987, 17.029 , 17.072, 17.114 ,17.156, 17.199,
17.241, 17.283, 17.326, 17.368, 17.410, 17.453 , 17.495, 17.537 ,17.580, 17.622,
17.664, 17.707, 17.749, 17.792, 17.834, 17.876 , 17.919, 17.961 ,18.004, 18.046,
18.088, 18.131, 18.173, 18.216, 18.258, 18.301 , 18.343, 18.385 ,18.428 , 18.470,
18.513, 18.555, 18.598, 18.640, 18.683, 18.725 , 18.768, 18.810 ,18.853, 18.895,
18.938, 18.980, 19.023, 19.065, 19.108, 19.150 , 19.193, 19.235 ,19.278, 19.320,
19.363, 19.405, 19.448, 19.490, 19.533, 19.576 , 19.618, 19.661 ,19.703, 19.746,
19.788, 19.831, 19.873, 19.916, 19.959, 20.001 , 20.044, 20.086 ,20.129, 20.172,
20.214, 20.257, 20.299, 20.342, 20.385, 20.427, 20.470, 20.512 , 20.555, 20.598,
20.640, 20.683, 20.725, 20.768, 20.811, 20.853, 20.896, 20.938 ,20.981, 21.024,
21.066, 21.109, 21.152, 21.194, 21.237, 21.280 , 21.322, 21.365 ,21.407, 21.450,
21.493, 21.535, 21.578, 21.621, 21.663, 21.706 ,21.749, 21.791 ,21.834, 21.876,
21.919, 21.962, 22.004, 22.047, 22.090, 22.132, 22.175, 22.218 , 22.260, 22.303,
22.346, 22.388, 22.431, 22.473, 22.516, 22.559, 22.601, 22.644 , 22.687, 22.729,
22.772, 22.815, 22.857, 22.900, 22.942, 22.985, 23.028, 23.070 , 23.113, 23.156,
23.198, 23.241, 23.284, 23.326, 23.369, 23.411, 23.454, 23.497 ,23.539, 23.582,
23.624, 23.667, 23.710, 23.752, 23.795, 23.837, 23.880, 23.923 , 23.965, 24.008,
24.050, 24.093, 24.136, 24.178, 24.221, 24.263, 24.306, 24.348 , 24.391, 24.434,
24.476, 24.519, 24.561, 24.604, 24.646, 24.689, 24.731, 24.774 ,24.817 , 24.859,
24.902, 24.944, 24.987, 25.029, 25.072, 25.114, 25.157, 25.199 , 25.242, 25.284,
25.327, 25.369, 25.412, 25.454, 25.497, 25.539, 25.582, 25.624 , 25.666, 25.709,
25.751, 25.794, 25.836, 25.879, 25.921, 25.964, 26.006, 26.048, 26.091, 26.133,
26.176, 26.218, 26.260, 26.303, 26.345, 26.387, 26.430, 26.472 , 26.515, 26.557,
26.599, 26.642, 26.684, 26.726, 26.769, 26.811, 26.853 ,26.896 ,26.938 ,26.980,
27.022, 27.065, 27.107, 27.149, 27.192, 27.234, 27.276 , 27.318 , 27.361, 27.403,
27.445, 27.487, 27.529, 27.572, 27.614, 27.656 , 27.698 , 27.740 ,27.783 , 27.825,
27.867, 27.909, 27.951, 27.993, 28.035, 28.078, 28.120, 28.162 , 28.204, 28.246,
28.288, 28.330, 28.372, 28.414, 28.456, 28.498, 28.540, 28.583 , 28.625, 28.667,
28.709, 28.751, 28.793, 28.835, 28.877, 28.919, 28.961, 29.002 , 29.044, 29.086,
29.128, 29.170, 29.212, 29.254, 29.296, 29.338, 29.380, 29.422 , 29.464, 29.505,
29.547, 29.589, 29.631, 29.673, 29.715, 29.756, 29.798, 29.840, 29.882, 29.924,
29.965, 30.007, 30.049, 30.091, 30.132, 30.174, 30.216, 30.257, 30.299, 30.341,
30.383, 30.424, 30.466, 30.508, 30.549, 30.591, 30.632, 30.674, 30.716, 30.757,
30.799, 30.840, 30.882, 30.924, 30.965, 31.007, 31.048, 31.090, 31.131, 31.173,
31.214, 31.256, 31.297, 31.339, 31.380, 31.422, 31.463, 31.504, 31.546, 31.587,
31.629, 31.670, 31.712, 31.753, 31.794, 31.836, 31.877, 31.918, 31.960, 32.001,
32.042, 32.084, 32.125, 32.166, 32.207, 32.249, 32.290, 32.331, 32.372, 32.414,
32.455, 32.496, 32.537, 32.578, 32.619, 32.661, 32.702, 32.743, 32.784, 32.825,
32.866, 32.907, 32.948, 32.990, 33.031, 33.072, 33.113, 33.154, 33.195, 33.236,

```
33.277, 33.318, 33.359, 33.400, 33.441, 33.482, 33.523, 33.564, 33.604, 33.645,
33.686, 33.727, 33.768, 33.809, 33.850, 33.891, 33.931, 33.972, 34.013, 34.054,
34.095, 34.136, 34.176, 34.217, 34.258, 34.299, 34.339, 34.380, 34.421, 34.461,
34.502, 34.543, 34.583, 34.624, 34.665, 34.705, 34.746, 34.787, 34.827, 34.868,
34.909, 34.949, 34.990, 35.030, 35.071, 35.111, 35.152, 35.192, 35.233, 35.273,
35.314, 35.354, 35.395, 35.435, 35.476, 35.516, 35.557, 35.597, 35.637, 35.678,
35.718, 35.758, 35.799, 35.839, 35.880, 35.920, 35.960, 36.000, 36.041, 36.081,
36.121, 36.162, 36.202, 36.242, 36.282, 36.323, 36.363, 36.403, 36.443, 36.483,
36.524, 36.564, 36.604, 36.644, 36.684, 36.724, 36.764, 36.804, 36.844, 36.885,
36.925, 36.965, 37.005, 37.045, 37.085, 37.125, 37.165, 37.205, 37.245, 37.285,
37.325, 37.365, 37.405, 37.445, 37.484, 37.524, 37.564, 37.604, 37.644, 37.684,
37.724, 37.764, 37.803, 37.843, 37.883, 37.923, 37.963, 38.002, 38.042, 38.082,
38.122, 38.162, 38.201, 38.241, 38.281, 38.320, 38.360, 38.400, 38.439, 38.479,
38.519, 38.558, 38.598, 38.638, 38.677, 38.717, 38.756, 38.796, 38.836, 38.875,
38.915, 38.954, 38.994, 39.033, 39.073, 39.112, 39.152, 39.191, 39.231, 39.270,
39.310, 39.349, 39.388, 39.428, 39.467, 39.507, 39.546, 39.585, 39.625, 39.664,
39.703, 39.743, 39.782, 39.821, 39.861, 39.900, 39.939, 39.979, 40.018, 40.057,
40.096, 40.136, 40.175, 40.214, 40.253, 40.292, 40.332, 40.371, 40.410, 40.449,
40.488, 40.527, 40.566, 40.605, 40.645, 40.684, 40.723, 40.762, 40.801, 40.840,
40.879, 40.918, 40.957, 40.996, 41.035, 41.074, 41.113, 41.152, 41.191, 41.230,
41.269, 41.308, 41.347, 41.385, 41.424, 41.463, 41.502, 41.541, 41.580, 41.619,
41.657, 41.696, 41.735, 41.774, 41.813, 41.851, 41.890, 41.929, 41.968, 42.006,
42.045, 42.084, 42.123, 42.161, 42.200, 42.239, 42.277, 42.316, 42.355, 42.393,
42.432, 42.470, 42.509, 42.548, 42.586, 42.625, 42.663, 42.702, 42.740, 42.779,
42.817, 42.856, 42.894, 42.933, 42.971, 43.010, 43.048, 43.087, 43.125, 43.164,
43.202, 43.240, 43.279, 43.317, 43.356, 43.394, 43.432, 43.471, 43.509, 43.547,
43.585, 43.624, 43.662, 43.700, 43.739, 43.777, 43.815, 43.853, 43.891, 43.930,
43.968, 44.006, 44.044, 44.082, 44.121, 44.159, 44.197, 44.235, 44.273, 44.311,
};
/*****************************************************************
 * TERMOCUPLA J
*****************************************************************/
const float TermocuplaJ [] = {
-7.89,  -7.912, -7.934, -7.955, -7.976, -7.996, -8.017, -8.037, -8.057, -8.076,
-7.659, -7.683, -7.707, -7.731, -7.755, -7.778, -7.801, -7.824, -7.846, -7.868,
-7.402, -7.429, -7.455, -7.482, -7.508, -7.533, -7.559, -7.584, -7.609, -7.634,
-7.122, -7.151, -7.18 , -7.209, -7.237, -7.265, -7.293, -7.321, -7.348, -7.375,
-6.821, -6.852, -6.883, -6.914, -6.944, -6.974, -7.004, -7.034, -7.064, -7.093,
-6.499, -6.532, -6.565, -6.598, -6.63 , -6.663, -6.695, -6.727, -6.758, -6.79,
-6.159, -6.194, -6.228, -6.263, -6.297, -6.331, -6.365, -6.399, -6.433, -6.466,
-5.801, -5.837, -5.874, -5.91 , -5.946, -5.982, -6.018, -6.053, -6.089, -6.124,
-5.426, -5.464, -5.502, -5.54 , -5.578, -5.615, -5.653, -5.69 , -5.727, -5.764,
-5.036, -5.076, -5.115, -5.155, -5.194, -5.233, -5.272, -5.311, -5.349, -5.388,
-4.632, -4.673, -4.714, -4.755, -4.795, -4.836, -4.876, -4.916, -4.956, -4.996,
-4.215, -4.257, -4.299, -4.341, -4.383, -4.425, -4.467, -4.508, -4.55 , -4.591,
-3.785, -3.829, -3.872, -3.915, -3.958, -4.001, -4.044, -4.087, -4.13 , -4.172,
-3.344, -3.389, -3.433, -3.478, -3.522, -3.566, -3.61 , -3.654, -3.698, -3.742,
-2.892, -2.938, -2.984, -3.029, -3.074, -3.12 , -3.165, -3.21 , -3.255, -3.299,
-2.431, -2.478, -2.524, -2.57 , -2.617, -2.663, -2.709, -2.755, -2.801, -2.847,
-1.96 , -2.008, -2.055, -2.102, -2.15 , -2.197, -2.244, -2.291, -2.338, -2.384,
-1.481, -1.53 , -1.578, -1.626, -1.674, -1.722, -1.77 , -1.818, -1.865, -1.913,
-0.995, -1.044, -1.093, -1.141, -1.19 , -1.239, -1.288, -1.336, -1.385, -1.433,
```

-0.501, -0.55 , -0.6 , -0.65 , -0.699, -0.748, -0.798, -0.847, -0.896, -0.945,
0 , -0.05 , -0.101, -0.151, -0.201, -0.251, -0.301, -0.351, -0.401, -0.451,
0 , 0.05 , 0.101, 0.151, 0.202, 0.253, 0.303, 0.354, 0.405, 0.456,
0.507, 0.558, 0.609, 0.66 , 0.711, 0.762, 0.813, 0.865, 0.916, 0.967,
1.019, 1.07 , 1.122, 1.174, 1.225, 1.277, 1.329, 1.381, 1.432, 1.484,
1.536, 1.588, 1.64 , 1.693, 1.745, 1.797, 1.849, 1.901, 1.954, 2.006,
2.058, 2.111, 2.163, 2.216, 2.268, 2.321, 2.374, 2.426, 2.479, 2.532,
2.585, 2.638, 2.691, 2.743, 2.796, 2.849, 2.902, 2.956, 3.009, 3.062,
3.115, 3.168, 3.221, 3.275, 3.328, 3.381, 3.435, 3.488, 3.542, 3.595,
3.649, 3.702, 3.756, 3.809, 3.863, 3.917, 3.971, 4.024, 4.078, 4.132,
4.186, 4.239, 4.293, 4.347, 4.401, 4.455, 4.509, 4.563, 4.617, 4.671,
4.725, 4.78 , 4.834, 4.888, 4.942, 4.996, 5.05 , 5.105, 5.159, 5.213,
5.268, 5.322, 5.376, 5.431, 5.485, 5.54 , 5.594, 5.649, 5.703, 5.758,
5.812, 5.867, 5.921, 5.976, 6.031, 6.085, 6.14 , 6.195, 6.249, 6.304,
6.359, 6.414, 6.468, 6.523, 6.578, 6.633, 6.688, 6.742, 6.797, 6.852,
6.907, 6.962, 7.017, 7.072, 7.127, 7.182, 7.237, 7.292, 7.347, 7.402,
7.457, 7.512, 7.567, 7.622, 7.677, 7.732, 7.787, 7.843, 7.898, 7.953,
8.008, 8.063, 8.118, 8.174, 8.229, 8.284, 8.339, 8.394, 8.45 , 8.505,
8.56 , 8.616, 8.671, 8.726, 8.781, 8.837, 8.892, 8.947, 9.003, 9.058,
9.113, 9.169, 9.224, 9.279, 9.335, 9.39 , 9.446, 9.501, 9.556, 9.612,
9.667, 9.723, 9.778, 9.834, 9.889, 9.944, 10 , 10.055, 10.111, 10.166,
10.222, 10.277, 10.333, 10.388, 10.444, 10.499, 10.555, 10.61 , 10.666, 10.721,
10.777, 10.832, 10.888, 10.943, 10.999, 11.054, 11.11 , 11.165, 11.221, 11.276,
11.332, 11.387, 11.443, 11.498, 11.554, 11.609, 11.665, 11.72 , 11.776, 11.831,
11.887, 11.943, 11.998, 12.054, 12.109, 12.165, 12.22 , 12.276, 12.331, 12.387,
12.442, 12.498, 12.553, 12.609, 12.664, 12.72 , 12.776, 12.831, 12.887, 12.942,
12.998, 13.053, 13.109, 13.164, 13.22 , 13.275, 13.331, 13.386, 13.442, 13.497,
13.553, 13.608, 13.664, 13.719, 13.775, 13.83 , 13.886, 13.941, 13.997, 14.052,
14.108, 14.163, 14.219, 14.274, 14.33 , 14.385, 14.441, 14.496, 14.552, 14.607,
14.663, 14.718, 14.774, 14.829, 14.885, 14.94 , 14.995, 15.051, 15.106, 15.162,
15.217, 15.273, 15.328, 15.383, 15.439, 15.494, 15.55 , 15.605, 15.661, 15.716,
15.771, 15.827, 15.882, 15.938, 15.993, 16.048, 16.104, 16.159, 16.214, 16.27,
16.325, 16.38 , 16.436, 16.491, 16.547, 16.602, 16.657, 16.713, 16.768, 16.823,
16.879, 16.934, 16.989, 17.044, 17.1 , 17.155, 17.21 , 17.266, 17.321, 17.376,
17.432, 17.487, 17.542, 17.597, 17.653, 17.708, 17.763, 17.818, 17.874, 17.929,
17.984, 18.039, 18.095, 18.15 , 18.205, 18.26 , 18.316, 18.371, 18.426, 18.481,
18.537, 18.592, 18.647, 18.702, 18.757, 18.813, 18.868, 18.923, 18.978, 19.033,
19.089, 19.144, 19.199, 19.254, 19.309, 19.364, 19.42 , 19.475, 19.53 , 19.585,
19.64 , 19.695, 19.751, 19.806, 19.861, 19.916, 19.971, 20.026, 20.081, 20.137,
20.192, 20.247, 20.302, 20.357, 20.412, 20.467, 20.523, 20.578, 20.633, 20.688,
20.743, 20.798, 20.853, 20.909, 20.964, 21.019, 21.074, 21.129, 21.184, 21.239,
21.295, 21.35 , 21.405, 21.46 , 21.515, 21.57 , 21.625, 21.68 , 21.736, 21.791,
21.846, 21.901, 21.956, 22.011, 22.066, 22.122, 22.177, 22.232, 22.287, 22.342,
22.397, 22.453, 22.508, 22.563, 22.618, 22.673, 22.728, 22.784, 22.839, 22.894,
22.949, 23.004, 23.06 , 23.115, 23.17 , 23.225, 23.28 , 23.336, 23.391, 23.446,
23.501, 23.556, 23.612, 23.667, 23.722, 23.777, 23.833, 23.888, 23.943, 23.999,
24.054, 24.109, 24.164, 24.22 , 24.275, 24.33 , 24.386, 24.441, 24.496, 24.552,
24.607, 24.662, 24.718, 24.773, 24.829, 24.884, 24.939, 24.995, 25.05 , 25.106,
25.161, 25.217, 25.272, 25.327, 25.383, 25.438, 25.494, 25.549, 25.605, 25.661,
25.716, 25.772, 25.827, 25.883, 25.938, 25.994, 26.05 , 26.105, 26.161, 26.216,
26.272, 26.328, 26.383, 26.439, 26.495, 26.551, 26.606, 26.662, 26.718, 26.774,
26.829, 26.885, 26.941, 26.997, 27.053, 27.109, 27.165, 27.22 , 27.276, 27.332,

```
27.388, 27.444, 27.5 , 27.556, 27.612, 27.668, 27.724, 27.78 , 27.836, 27.893,
27.949, 28.005, 28.061, 28.117, 28.173, 28.23 , 28.286, 28.342, 28.398, 28.455,
28.511, 28.567, 28.624, 28.68 , 28.736, 28.793, 28.849, 28.906, 28.962, 29.019,
29.075, 29.132, 29.188, 29.245, 29.301, 29.358, 29.415, 29.471, 29.528, 29.585,
29.642, 29.698, 29.755, 29.812, 29.869, 29.926, 29.983, 30.039, 30.096, 30.153,
30.21 , 30.267, 30.324, 30.381, 30.439, 30.496, 30.553, 30.61 , 30.667, 30.724,
30.782, 30.839, 30.896, 30.954, 31.011, 31.068, 31.126, 31.183, 31.241, 31.298,
31.356, 31.413, 31.471, 31.528, 31.586, 31.644, 31.702, 31.759, 31.817, 31.875,
31.933, 31.991, 32.048, 32.106, 32.164, 32.222, 32.28 , 32.338, 32.396, 32.455,
32.513, 32.571, 32.629, 32.687, 32.746, 32.804, 32.862, 32.921, 32.979, 33.038,
33.096, 33.155, 33.213, 33.272, 33.33 , 33.389, 33.448, 33.506, 33.565, 33.624,
33.683, 33.742, 33.8 , 33.859, 33.918, 33.977, 34.036, 34.095, 34.155, 34.214,
34.273, 34.332, 34.391, 34.451, 34.51 , 34.569, 34.629, 34.688, 34.748, 34.807,
34.867, 34.926, 34.986, 35.046, 35.105, 35.165, 35.225, 35.285, 35.344, 35.404,
35.464, 35.524, 35.584 , 35.644, 35.704, 35.764, 35.825, 35.885, 35.945, 36.005,
36.066, 36.126, 36.186, 36.247, 36.307, 36.368, 36.428, 36.489, 36.549, 36.61,
36.671, 36.732, 36.792, 36.853, 36.914, 36.975, 37.036, 37.097, 37.158, 37.219,
37.28 , 37.341, 37.402, 37.463, 37.525, 37.586, 37.647, 37.709, 37.77 , 37.831,
37.893, 37.954, 38.016, 38.078, 38.139, 38.201, 38.262, 38.324, 38.386, 38.448,
38.51 , 38.572, 38.633, 38.695, 38.757, 38.819, 38.882, 38.944, 39.006, 39.068,
39.13 , 39.192, 39.255, 39.317, 39.379, 39.442, 39.504, 39.567, 39.629, 39.692,
39.754, 39.817, 39.88 , 39.942, 40.005, 40.068, 40.131, 40.193, 40.256, 40.319,
40.382, 40.445, 40.508, 40.571, 40.634, 40.697, 40.76 , 40.823, 40.886, 40.95,
41.013, 41.076, 41.139, 41.203, 41.266, 41.329, 41.393, 41.456, 41.52 , 41.583,
41.647, 41.71 , 41.774, 41.837, 41.901, 41.965, 42.028, 42.092, 42.156, 42.219,
42.283, 42.347, 42.411, 42.475, 42.538, 42.602, 42.666, 42.73 , 42.794, 42.858,
42.922,
};

/**********************************************************************
 * VARIABLES GLOBALES
**********************************************************************/
//VARIABLES PRINCIPALES
int x, y;           // Asiganacion de coordenadas en la touch
int BoPr = 0;       // Asignacion para cada boton
int MeSu = 5;       // Asignacion para el boton means/source
int aux_MeSu=0;     // Asignacion para el comienzo del boton means/source
int aux_TC=0;       // Asignacion para el comienzo del boton TC medicion
int aux_TCg=0;      // Asignacion para el comienzo del boton TC generacion
int aux_Voltimetro=0;    // Asignacion para el comienzo del boton V medicion
int aux_Voltimetrog=0;   // Asignacion para el comienzo del boton V generacion

char dato_impresion[50];

//ASIGNACION DE PINES DIGITALES ARDUINO DUE
const int Res_prin = 14;
const int Res_secu = 15;
const int TC_prin = 16;
const int TC_secu = 17;
const int E_corriente = 18;
const float temperatuta_UF_J = 40.5;
```

```
//ASIGNACION DE PINES ANALOGICO ARDUINO DUE
const int Ohmetro        = A0;
const int Milivoltimetro  = A1;
const int Amperimetro     = A2;
const int Voltimetro      = A3;

//VARIABLES PARA FILTROS DIGITALES
#define filterSamples   3
float filterFrequency = 1;

float valor_adc_Vol, filtro_adc_Vol = 0;
int valor_Vol, valor2_Vol;
int sensSmoothArray1_Vol [filterSamples];
int rawData1_Vol, smoothData1_Vol;
float acond;
int res_ini_med=0;

float valor_adc_mVol, filtro_adc_mVol = 0;
int valor_mVol, valor2_mVol;
int sensSmoothArray1_mVol [filterSamples];
int rawData1_mVol, smoothData1_mVol;

float valor_adc_Amp, filtro_adc_Amp = 0;
int valor_Amp, valor2_Amp;
int sensSmoothArray1_Amp [filterSamples];
int rawData1_Amp, smoothData1_Amp;

float valor_adc_Ohm, filtro_adc_Ohm = 0;
int valor_Ohm, valor2_Ohm;
int sensSmoothArray1_Ohm [filterSamples];
int rawData1_Ohm, smoothData1_Ohm;
int ohmetro_ohmetro_dato;

ResponsiveAnalogRead analog_dato_Vol(Voltimetro, true, 0.01);
ResponsiveAnalogRead analog_dato_mVol(Milivoltimetro, true, 0.01);
ResponsiveAnalogRead analog_dato_Amp(Amperimetro, true, 0.01);
ResponsiveAnalogRead analog_dato_Ohm(Ohmetro, true, 0.01);

// VARIABLES PARA MODIFICAR EL DATO DE GENERACION
int num_uni = -1;     // Unidades
int aux_num_uni1 = 0;
int aux_num_uni2 = 0;
int tiempo_medicion = 500;

int num_dec = -1;     // Unidades
int aux_num_dec1 = 0;
int aux_num_dec2 = 0;

int num_cen = -1;     // Unidades
int aux_num_cen1 = 0;
int aux_num_cen2 = 0;
```

```
int num_mil = -1;     // Unidades
int aux_num_mil1 = 0;
int aux_num_mil2 = 0;

int dato_act=0;
int aux_UP_DOWN = 0;
int aux_conteo_dato_uni = 0;
int aux_conteo_dato_dec = 0;
int aux_conteo_dato_cen = 0;

// VARIABLES PARA CALIBRACION
int potenciometro1 = 1023;
int potenciometro2 = 1023;
int potenciometro3 = 1023;
float voltaje_calibracion = 0;
int multiplicador=0;

//VARIABLES PARA RANGOS DE GENERACION
int RangoInfVolt=0;
int RangoSupVolt=10;
int RangoInfCorr=0;
int RangoSupCorr=24;
int RangoInfRes=30;
int RangoSupRes=1200;
int RangoInfRTD=-150;
int RangoSupRTD=850;
int RangoInfTCJ=-80;
int RangoSupTCJ=760;
int RangoInfTCK=-100;
int RangoSupTCK=1500;

//VARIABLES PARA GENERACION
float generacion_volt = RangoInfVolt;
float generacion_corr = RangoInfCorr;
float generacion_TCJ = RangoInfTCJ;
float generacion_TCK = RangoInfTCK;
float generacion_RTD = RangoInfRTD;
float generacion_res = RangoInfRes;

int factor_x, factor_inical;
float TCJrango, TCKrango;
float generarTCJ, generarTCK;
int TCJinicial = 1357 , TCKinicial = 1800;
int jsjs =0;
int TCg;
int sdsd =0;
float asasas=0;
float coriente_res;
float valor_adc_Ohm2,valor_adc_Ohm3;

// VARIABLES PARA DOCUMENTO DE CALIBRACION
int corriente_calibracion0=4;
```

```
int corriente_calibracion100=20;
float tolerancia_calibracion=0.5;
float demora=5;
int tipo_sensor=0;
float sensor_calibracion0=4;
float sensor_calibracion100=20;
int estrategia=0;
int posicion_calibracion = 0;
float valor_adc_Ohm1;

//VARIABLES AUXILIARES
int aux_medi_gene = 0;
int boton_especial = 0;
int BoCal = 0;

int aux_izde = 48, aux7 = 0, aux8 = 0; // Movimiento izquierda derecha generador
int aux_izde_ca = 339, aux7_ca = 0, aux8_ca = 0; // Movimiento izquierda derecha
calibracion

int num1 = 0, num2 = 0, num3 = 0, num4 = 0;

int valor_generado1, valor_generado2;
float valor_generado3 = 0;

int contraste = -1;
int aux_contraste1 = 0;
int aux_contraste2 = 0;
int r = 255;
int g = 255;
int h = 255;
float contador = 0;
float dato = 0;
float dato_100 = 0;
float dato_0 = 0;
float aux_dato = 0;
int aux_cambio = 0;

int aux_volt = 0;
int aux_fre = 0;
int aux_temp = 0;
int aux_medicion = 0;
int aux_loop = 0;

int limite_uni = 0;
int limite_dec = 0;
int limite_cen = 0;
int limite_mil = 0;

float v[]={0, 0, 0, 0, 0};
char cadena_com[6];
char cadena_com1[5];
char cadena_com2[5];
```

```
char cadena_com3[5];
String aux_cadena_com;
String aux_cadena_com1;
String aux_cadena_com2;
String aux_cadena_com3;
byte aux_cad;

int salir_voltaje=0;
int salir_corriente=0;
int salir_resistencia=0;
int aux_termocupla=0;

int I2C_res1, I2C_res2, I2C_res3;
char kkk[6];
int ang;

/*********************************************************************
 * FUNCION: BOTONES DE LA PANTALLA
*********************************************************************/
void drawButtons()
{
 myGLCD.setColor(VGA_RED);
 myGLCD.setBackColor(r, g, h);
 myGLCD.setFont(BigFont);
 myGLCD.print("CALIBRADOR DE TEMPERATURA", 35, 118);
 myGLCD.print("MEASURE", 7, 182);
 myGLCD.drawRoundRect (5, 180, 355, 280);
 myGLCD.print("SOURCE", 7, 302);
 myGLCD.drawRoundRect (5, 300, 355, 400);
 //---------------Botones 1ra fila----------------------
 myGLCD.drawRoundRect(5, 420, 115, 461.43);
 myGLCD.setFont(SmallFont);
 myGLCD.print("CONTRAST", 30, 433.22);
 myGLCD.drawRoundRect(125, 420, 235, 461.43);
 myGLCD.setFont(various_symbols);
 myGLCD.print("`", 165, 433.22);
 myGLCD.setFont(BigFont);
 myGLCD.print("F", 183, 433.22);
 myGLCD.drawRoundRect(245, 420, 355, 461.43);
 myGLCD.setFont(various_symbols);
 myGLCD.print("`", 285, 433.22);
 myGLCD.setFont(BigFont);
 myGLCD.print("C", 303, 433.22);
 myGLCD.drawRoundRect(365, 420, 475, 461.43);
 myGLCD.print("LOOP", 387, 433.22);
 //---------------Botones 2da fila----------------------
 myGLCD.drawRoundRect(5, 471.43, 115, 564.29);
 myGLCD.print("MEAS", 27, 484.65);
 myGLCD.drawLine(10, 517.86, 110, 517.86);
 myGLCD.print("SOURCE", 9, 536.08);
 myGLCD.setFont(BigFont);
 myGLCD.drawRoundRect(125, 471.43, 235, 512.86);
```

```
myGLCD.print("V", 174, 484.65);
myGLCD.drawRoundRect(245, 471.43, 355, 512.86);
myGLCD.setFont(various_symbols);
myGLCD.print("Z", 294, 484.65);
myGLCD.setFont(BigFont);
myGLCD.drawRoundRect(365, 471.43, 475, 512.86);
myGLCD.print("mA", 405, 484.65);
//---------------Botones 3ra fila----------------------
myGLCD.drawRoundRect(125, 522.86, 235, 564.29);
myGLCD.print("TC", 165, 536.08);
myGLCD.drawRoundRect(245, 522.86, 355, 564.29);
myGLCD.print("RTD", 276, 536.08);
myGLCD.drawRoundRect(365, 522.86, 475, 564.29);
myGLCD.print("HELP", 387, 536.08);
//---------------Botones 4ta fila----------------------
myGLCD.drawRoundRect(5, 574.29, 115, 615.72);
myGLCD.print("100%", 27, 587.51);
myGLCD.drawRoundRect(185, 574.29, 295, 615.72);
myGLCD.setFont(various_symbols);
myGLCD.print("=", 234, 587.51);
myGLCD.drawRoundRect(365, 574.29, 475, 615.72);
myGLCD.setFont(BigFont);
//-------------SIMBOLO DE ICONO RAMPAS--------------
//--------------------------------------
myGLCD.drawLine (380, 607.305, 390, 583.005);
myGLCD.drawLine (390, 583.005, 400, 607.305);
myGLCD.drawLine (410, 607.305, 415, 583.005);
myGLCD.drawLine (415, 583.005, 420, 607.305);
myGLCD.drawLine (420, 607.305, 425, 583.005);
myGLCD.drawLine (425, 583.005, 430, 607.305);
myGLCD.drawLine (440, 607.305, 440, 595.155);
myGLCD.drawLine (440, 595.155, 450, 595.155);
myGLCD.drawLine (450, 595.155, 450, 583.005);
myGLCD.drawLine (450, 583.005, 460, 583.005);
//---------------Botones 5ta fila----------------------
myGLCD.drawRoundRect(5, 625.72, 115, 667.15);
myGLCD.print("25%", 27, 638.94);
myGLCD.setFont(various_symbols);
myGLCD.print("=", 81, 638.94);
myGLCD.drawRoundRect(125, 625.72, 235, 667.15);
myGLCD.print("?", 174, 638.94);
myGLCD.drawRoundRect(245, 625.72, 355, 667.15);
myGLCD.print("T", 294, 638.94);
myGLCD.drawRoundRect(365, 625.72, 475, 667.15);
myGLCD.setFont(SmallFont);
myGLCD.print("FUNTION", 390, 638.94);
myGLCD.setFont(BigFont);
//---------------Botones 6ta fila----------------------
myGLCD.drawRoundRect(5, 677.15, 115, 718.58);
myGLCD.print("25%", 27, 690.37);
myGLCD.setFont(various_symbols);
myGLCD.print(">", 81, 690.37);
```

```
 myGLCD.drawRoundRect(185, 677.15, 295, 718.58);
 myGLCD.print(">", 234, 690.37);
 myGLCD.setFont(BigFont);
 myGLCD.drawRoundRect(365, 677.15, 475, 718.58);
 myGLCD.print("RECALL", 372, 690.37);
 //---------------Botones 7ma fila----------------------
 myGLCD.drawRoundRect(5, 728.58, 115, 770.01);
 myGLCD.print("0%", 45, 741.8);
 myGLCD.drawRoundRect(125, 728.58, 355, 770.01);
 myGLCD.print("CALIBRATION", 145, 741.8);
 myGLCD.drawRoundRect(365, 728.58, 475, 770.01);
 myGLCD.print("ENTER", 380, 741.8);
}
/*************************************************************************
 * FUNCION: SIMBOLO DE OHMIOS
*************************************************************************/
void signo_ohm(int h, int k, int radio)
{
 int jj = sqrt (pow(radio, 2) - pow((radio / 2), 2)) + k;
 for (int ii = 0; ii <= 3; ii++)
 {
  for (int i = h - radio; i <= (h + radio); i = i + 1)     {
   int jj = - sqrt (pow(radio, 2) - pow((i - h), 2)) + k;
   myGLCD.drawPixel(i, jj);                          }
  for (int i = h - radio; i <= (h - (radio / 2)); i = i + 1) {
   int j = sqrt (pow(radio, 2) - pow((i - h), 2)) + k;
   myGLCD.drawPixel(i, j);                           }
  for (int i = (h + (radio / 2)); i <= (h + radio); i = i + 1)  {
   int j = sqrt (pow(radio, 2) - pow((i - h), 2)) + k;
   myGLCD.drawPixel(i, j);                             }
  myGLCD.drawLine((h + (radio / 2)), jj, h + radio + 4, jj);
  myGLCD.drawLine((h - (radio / 2)), jj, h - radio - 4, jj);
  radio = radio + 1;
  jj++;
 }
}
/*************************************************************************
 * FUNCION: SIMBOLO DE GRADO
*************************************************************************/
void signo_grado(int h1, int k1)
{
 myGLCD.drawCircle(h1, k1, 5);
 myGLCD.drawCircle(h1, k1, 4);
 myGLCD.drawCircle(h1, k1, 3);
}

/*************************************************************************
 * FUNCION: SIMBOLO FLECHA ARRIBA
*************************************************************************/
void flecha_arriba(int h2, int k2)
{
 myGLCD.drawLine(h2, k2,h2, k2 );
```

```
 myGLCD.drawLine(h2, k2, h2, k2);
 myGLCD.drawLine(h2, k2, h2, k2);
}

/*********************************************************************
 * FUNCION: SIMBOLO FLECHA ABAJO
*********************************************************************/
void flecha_abajo(int h3, int k3)
{
 myGLCD.drawLine(h3, k3, h3, k3 );
 myGLCD.drawLine(h3, k3, h3, k3);
 myGLCD.drawLine(h3, k3, h3, k3);
}
/*********************************************************************
 * FUNCION:
*********************************************************************/
void transformacion_F_C(int uno, int dos)
{
 if (aux_temp == 0)             {
   myGLCD.print("C", uno, dos);
   signo_grado(uno - 5, dos + 5);  }
 else if (aux_temp == 1)         {
   myGLCD.print("F", uno, dos);
   signo_grado(uno - 5, dos + 5);  }
}
/*********************************************************************
 * FUNCION: LIMPIAR LA PANTALLA EL DATO DE MEDICION
*********************************************************************/
void borrar_medicion()
{
 if (MeSu == 1 || MeSu == 0)          {
   myGLCD.setFont(Grotesk32x64);
   myGLCD.print("        ", 50, 210);
   if (BoPr != 16){
     myGLCD.setFont(BigFont);
     myGLCD.print("      ",240, 182);
     myGLCD.print("      ",240, 302);
     myGLCD.setFont(SmallFont);
     myGLCD.print("                    ",7,166);
   }
   myGLCD.setFont(SmallFont);          }
}
/*********************************************************************
 * FUNCION: LIMPIAR LA PANTALLA EL DATO DE GENERACION
*********************************************************************/
void borrar_generacion()
{
 if (MeSu == 2 || MeSu == 0)         {
   myGLCD.setFont(Grotesk32x64);
   myGLCD.print("       ", 80, 330);
   if (BoPr != 16){
     myGLCD.setFont(BigFont);
```

```
    myGLCD.print("      ",240, 182);
    myGLCD.print("      ",240, 302);
    myGLCD.setFont(SmallFont);
    myGLCD.print("                    ",7,166);
   }
   myGLCD.setFont(SmallFont);        }
}
/*********************************************************************
 * FUNCIONES: IMPRESION DE DATO GENERADOR Y MEDICION
*********************************************************************/
void imprimir_generacion(float dato_generacion){
 int parte_entera = dato_generacion;
 float resta = (dato_generacion-parte_entera)*100;
 int parte_decimal = resta;
 if (BoPr == 8){
  sprintf(dato_impresion,"%2d.%02d mA",parte_entera,parte_decimal);
 }

 if (BoPr == 15){
  sprintf(dato_impresion,"%2d.%02d V",parte_entera,parte_decimal);
 }

 if (BoPr == 19){
  sprintf(dato_impresion,"%4d",parte_entera);
  myGLCD.setFont(Grotesk32x64);
  if (MeSu == 2 || MeSu == 0){
   signo_ohm(251, 371, 16);
  }
  if (MeSu == 1){
   signo_ohm(251, 251, 16);
  }
 }

 if (BoPr == 20 || BoPr == 16){
  sprintf(dato_impresion,"%4d",parte_entera);
  myGLCD.setFont(Grotesk32x64);
  if (MeSu == 2 || MeSu == 0){
   transformacion_F_C(240, 330);
  }
  if (MeSu == 1){
   transformacion_F_C(240, 210);
  }
 }

 if ((MeSu == 2 || MeSu == 0)){
  myGLCD.setFont(Grotesk32x64);
  myGLCD.print(dato_impresion, 80, 330);
 }

  if ((MeSu == 1)){
  myGLCD.setFont(Grotesk32x64);
  myGLCD.print(dato_impresion, 80, 210);
```

```
}

if (BoCal == 1){
  myGLCD.setFont(BigFont);
  if (posicion_calibracion == 1){
    sprintf(dato_impresion,"%2d.%02dmA",parte_entera,parte_decimal);
    myGLCD.print(dato_impresion, 355, 200);
  }
  if (posicion_calibracion == 2){
    sprintf(dato_impresion,"%2d.%02dmA",parte_entera,parte_decimal);
    myGLCD.print(dato_impresion, 355, 220);
  }
  if (posicion_calibracion == 3){
    sprintf(dato_impresion,"%2d.%02d %",parte_entera,parte_decimal);
    myGLCD.print(dato_impresion, 355, 240);
  }
  if (posicion_calibracion == 4){
    sprintf(dato_impresion,"%2d.%02d s",parte_entera,parte_decimal);
    myGLCD.print(dato_impresion, 355, 260);
  }
  if (posicion_calibracion == 5){
    if (tipo_sensor = 1){
      sprintf(dato_impresion,"TC TIPO J");
      myGLCD.print(dato_impresion, 355, 320);
    }
    if (tipo_sensor = 2){
      sprintf(dato_impresion,"TC TIPO K");
      myGLCD.print(dato_impresion, 355, 320);
    }
    if (tipo_sensor = 3){
      sprintf(dato_impresion,"RTD PT100");
      myGLCD.print(dato_impresion, 355, 320);
    }
  }
  if (posicion_calibracion == 6){
    sprintf(dato_impresion,"%4d",parte_entera);
    myGLCD.print(dato_impresion, 355, 340);
  }
  if (posicion_calibracion == 7){
    sprintf(dato_impresion,"%4d",parte_entera);
    myGLCD.print(dato_impresion, 355, 360);
  }
  if (posicion_calibracion == 8){
    if (estrategia == 1){
      sprintf(dato_impresion,"%1d",parte_entera);
      myGLCD.print(dato_impresion, 355, 380);
    }

  }

}
```

```
}

/*********************************************************************
 * FUNCIONES: DESCOMPOSICION DEL DATO DE GENERACION Y CONTRASTE
*********************************************************************/
void descomposicion (float aux_desc)
{
 int dividendo, centenas, decenas, unidades;
  if (BoPr == 19 || BoPr == 16 || BoPr == 20 || BoPr == 9 || BoPr == 15 || BoPr == 8)  {
  dividendo = aux_desc;
  num4 = dividendo / 1000;
  centenas = dividendo % 1000;
  num3 = centenas / 100;
  decenas = centenas % 100;
  num2 = decenas / 10;
  unidades = decenas % 10;
  num1 = unidades;                              }
 if (BoPr == 9)              {
  aux_desc = aux_desc * 10;
  dividendo = aux_desc;
  num4 = dividendo / 1000;
  centenas = dividendo % 1000;
  num3 = centenas / 100;
  decenas = centenas % 100;
  num2 = decenas / 10;
  unidades = decenas % 10;
  num1 = unidades;          }

}
/*********************************************************************
 * FUNCIONES: TITILEO DEL ICONO SELECCIONADO
*********************************************************************/
void waitForIt(int x1, int y1, int x2, int y2)
{
 myGLCD.setColor(VGA_BLACK);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
 while (myTouch.dataAvailable())
 {
  if (boton_especial == 3 || boton_especial == 6) {
   contador++;                          }
 }
 myTouch.read();
 myGLCD.setColor(VGA_RED);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
}
/*********************************************************************
 * FUNCIONES: BORRAR DE LA PANTALLA LOS DATOS DE CALIBRACION
*********************************************************************/
void waitForIt1(int x1, int y1, int x2, int y2, int x11, int y11, int x12, int y12, int color)
{
 if (color == 1){
  myGLCD.setColor(VGA_WHITE);
```

```
  myGLCD.drawRoundRect (5, 300, 355, 400);
  myGLCD.drawRoundRect (5, 180, 355, 280);
 }

 if (color == 0){
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawRoundRect (5, 300, 475, 400);
  myGLCD.drawRoundRect (5, 180, 475, 280);
  myGLCD.setFont(BigFont);
  myGLCD.print("                    ", 15, 200);
  myGLCD.print("                    ", 15, 220);
  myGLCD.print("                    ", 15, 240);
  myGLCD.print("                    ", 15, 260);
  myGLCD.print("                    ", 15, 320);
  myGLCD.print("                    ", 15, 340);
  myGLCD.print("                    ", 15, 360);
  myGLCD.print("                    ", 15, 380);
 }

 if ((MeSu == 0 || MeSu == 4) && color == 0){
  myGLCD.setColor(VGA_RED);
  myGLCD.drawRoundRect (5, 300, 355, 400);
 }

 if (MeSu == 1 && color == 0){
  myGLCD.setColor(VGA_RED);
  myGLCD.drawRoundRect (5, 180, 355, 280);
 }

 myGLCD.setColor(VGA_BLACK);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
 myGLCD.drawRoundRect (x11, y11, x12, y12);
 delay(250);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
 myGLCD.drawRoundRect (x11, y11, x12, y12);
 delay(250);
 myGLCD.setColor(VGA_BLACK);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
 myGLCD.drawRoundRect (x11, y11, x12, y12);
 delay(250);
 while (myTouch.dataAvailable())
  myTouch.read();
 myGLCD.setColor(VGA_RED);
 myGLCD.drawRoundRect (x1, y1, x2, y2);
 myGLCD.drawRoundRect (x11, y11, x12, y12);
}

/*********************************************************************
 * FUNCIONES: FILTRO DIGITAL
*********************************************************************/
int digitalSmooth(int rawIn, int *sensSmoothArray){
```

```
 int j, k, temp, top, bottom;
 long total;
 static int i;
 static int sorted[filterSamples];
 boolean done;

 i = (i + 1) % filterSamples;
 sensSmoothArray[i] = rawIn;
 for (j=0; j<filterSamples; j++){
  sorted[j] = sensSmoothArray[j];
 }

 done = 0;
 while(done != 1){
  done = 1;
  for (j = 0; j < (filterSamples - 1); j++){
   if (sorted[j] > sorted[j + 1]){
    temp = sorted[j + 1];
    sorted [j+1] =  sorted[j] ;
    sorted [j] = temp;
    done = 0;
   }
  }
 }

 bottom = max(((filterSamples * 15)  / 100), 1);
 top = min((((filterSamples * 85) / 100) + 1  ), (filterSamples - 1));
 k = 0;
 total = 0;
 for ( j = bottom; j< top; j++){
  total += sorted[j];
  k++;
 }

 return total / k;
}

/*********************************************************************
 * FUNCION PRINCIPAL (EJECUCION POR UNA VEZ)
*********************************************************************/
void setup() {
 myGLCD.InitLCD(PORTRAIT);
 myGLCD.clrScr();
 myTouch.InitTouch(PORTRAIT);
 myTouch.setPrecision(PREC_EXTREME);
 analogReadResolution(12);
 analogWriteResolution(12);
 analog_dato_Vol.setAnalogResolution(4096);
 analog_dato_mVol.setAnalogResolution(4096);
 analog_dato_Amp.setAnalogResolution(4096);
 analog_dato_Ohm.setAnalogResolution(4096);
 Wire.begin();
```

```
 max.begin();
 pinMode(DAC1,OUTPUT);
 analogWrite(DAC1,0);
 pinMode(Res_prin, OUTPUT);
 digitalWrite(Res_prin, LOW);
 pinMode(Res_secu, OUTPUT);
 digitalWrite(Res_secu, LOW);
 pinMode(TC_prin, OUTPUT);
 digitalWrite(TC_prin, LOW);
 pinMode(TC_secu, OUTPUT);
 digitalWrite(TC_secu, LOW);
 pinMode(E_corriente, OUTPUT);
 digitalWrite(E_corriente, LOW);
 myGLCD.fillScr(r, g, h);
 myGLCD.drawBitmap (15, 15, 300, 75, espe1, 0, 0, 0);
 myGLCD.drawBitmap (355, 0, 95, 100, Electronica, 0, 0, 0);
 myGLCD.setFont(BigFont);
 drawButtons();
}
/************************************************************************
 * FUNCION PRINCIPAL (EJECUCION REPETITIVA)
************************************************************************/

void loop()
{
 while (true)
 {

   if (aux_cambio != BoPr) {
    dato = 0;
    descomposicion(dato); }
   num4=num_mil;
   num3=num_cen;
   num2=num_dec;
   num1=num_uni;
   aux_dato = (dato_100 - dato_0) / 4;

/************************************************************************
 * MEDIDORES
************************************************************************/

  if ((MeSu == 1 || MeSu == 0)  && aux_medicion == 15 && aux_loop == 0)
   {
    // MEDICION DE VOLTAJE
    myGLCD.setFont(Grotesk32x64);
    salir_voltaje=1;
    jsjs=0;
    FilterOnePole lowpassFilter( LOWPASS, filterFrequency );
     while (salir_voltaje == 1)
    {
     analogRead(Voltimetro); // dummy read: enable channel 0
     delay(10);
```

```
      int voltaje_voltaje []={analogRead(Voltimetro), analogRead(Voltimetro),
analogRead(Voltimetro), analogRead(Voltimetro),};

      for (int x = 0; x < 200; x++){
        for (int y = 0; y < (200 - 1); y++){
          if (voltaje_voltaje[y] > voltaje_voltaje[y + 1]){
            int temp = voltaje_voltaje[y + 1];
            voltaje_voltaje[y + 1] = voltaje_voltaje[y];
            voltaje_voltaje[y] = temp;
          }
        }
      }

      valor2_Vol = lowpassFilter.input (voltaje_voltaje[100]);
      rawData1_Vol= valor2_Vol;
      valor_adc_Vol = (float)rawData1_Vol * 30.0/4000.0;
     if (valor_adc_Vol > 10){
       acond = (0.023*valor_adc_Vol) - 0.675;
      } else if (valor_adc_Vol <= 10 && valor_adc_Vol > 1){
       acond = (0.0351*valor_adc_Vol) - 0.8507;
      }


      valor_adc_Vol=valor_adc_Vol-acond;  //valor_adc_Vol-acond
      imprimir_generacion(valor_adc_Vol);

      //rawData1_Vol=sdsd/jsjs;
      if (myTouch.dataAvailable())
      {
        salir_voltaje=0;
      }
      delay(1);
    }

   }
   if ((MeSu == 1 || MeSu == 0) && aux_medicion == 8 || aux_loop == 1)
   {
    // MEDICION DE CORRIENTE
    myGLCD.setFont(Grotesk32x64);
    salir_corriente=1;
    digitalWrite(E_corriente,HIGH);
    FilterOnePole lowpassFilter( LOWPASS, filterFrequency );
    while (salir_corriente==1)
    {
      analogRead(Amperimetro); // dummy read: enable channel 0
      delay(10);
      //analog_dato_Vol.update();
      //valor_Vol = analog_dato_Vol.getValue();
      int coriente_corriente []={analogRead(Amperimetro), analogRead(Amperimetro),
analogRead(Amperimetro), analogRead(Amperimetro),};

      for (int x = 0; x < 200; x++){
```

```
    for (int y = 0; y < (200 - 1); y++){
      if (coriente_corriente[y] > coriente_corriente[y + 1]){
        int temp = coriente_corriente[y + 1];
        coriente_corriente[y + 1] = coriente_corriente[y];
        coriente_corriente[y] = temp;
      }
    }
   }

   valor2_Amp = lowpassFilter.input (coriente_corriente[100]);
   rawData1_Amp= valor2_Amp;
   valor_adc_Amp = (float)rawData1_Amp * 24.0/4000.0;
   if (valor_adc_Amp > 14.5){
    acond = (0.0273*valor_adc_Amp) - 0.8848;
   } else if (valor_adc_Amp <= 14.5 && valor_adc_Amp > 8){
    acond = (0.0195*valor_adc_Amp) - 0.7785;
   }else if (valor_adc_Amp <= 8){
    acond = (0.0198*valor_adc_Amp) - 0.774;
   }

   valor_adc_Amp=valor_adc_Amp-acond;
   imprimir_generacion(valor_adc_Amp);
   imprimir_generacion(valor_adc_Amp);
   if (myTouch.dataAvailable())
   {
    digitalWrite(E_corriente,LOW);
    salir_corriente=0;
   }
   delay(1);
  }
}
if ((MeSu == 1 || MeSu == 0) && aux_medicion == 19 && aux_loop == 0)
{
 // MEDICION DE RESISTENCIA
 myGLCD.setFont(Grotesk32x64);
 salir_resistencia=1;
 FilterOnePole lowpassFilter( LOWPASS, filterFrequency );
 digitalWrite(Res_prin, LOW);
 digitalWrite(Res_secu, LOW);
 delay(1);
 digitalWrite(TC_prin, HIGH);
 digitalWrite(TC_secu, HIGH);

 aux_cad=2;
 aux_cadena_com = String (1.6);
 aux_cadena_com.toCharArray(cadena_com,6);
 Wire.beginTransmission(8); // transmit to device #8
 Wire.write(cadena_com);
 Wire.write(aux_cad);
 Wire.endTransmission();
 while (salir_resistencia==1)
 {
```

```
      analogRead(Ohmetro); // dummy read: enable channel 0
      delay(10);
      //analog_dato_Vol.update();
      //valor_Vol = analog_dato_Vol.getValue();
      int ohmetro_ohmetro []={analogRead(Ohmetro), analogRead(Ohmetro),
analogRead(Ohmetro), analogRead(Ohmetro), analogRead(Ohmetro),};

      for (int x = 0; x < 200; x++){
        for (int y = 0; y < (200 - 1); y++){
          if (ohmetro_ohmetro[y] > ohmetro_ohmetro[y + 1]){
            int temp = ohmetro_ohmetro[y + 1];
            ohmetro_ohmetro[y + 1] = ohmetro_ohmetro[y];
            ohmetro_ohmetro[y] = temp;
          }
        }
      }

      valor2_Ohm = lowpassFilter.input (ohmetro_ohmetro[100]);
      rawData1_Ohm= valor2_Ohm;
      valor_adc_Ohm = (float)rawData1_Ohm * 280.0/3988.0;

      int parte_entera = valor_adc_Ohm;
      float resta = (valor_adc_Ohm-parte_entera)*10;
      int parte_decimal = resta;
      sprintf(dato_impresion,"%4d.%1d",parte_entera,parte_decimal);
      myGLCD.setFont(Grotesk32x64);
      myGLCD.print(dato_impresion,80, 210);
      signo_ohm(251, 251, 16);


      if (myTouch.dataAvailable())
      {
        digitalWrite(TC_prin, LOW);
        digitalWrite(TC_secu, LOW);
        digitalWrite(Res_prin, LOW);
        digitalWrite(Res_secu, LOW);
        salir_resistencia=0;
      }
      delay(1);

    }
  }


    if ((MeSu == 1 || MeSu == 0) && aux_medicion == 16 && aux_loop == 0)
    {
      // MEDICION DE TERMOCUPLAS
      if (aux_temp == 0){
      imprimir_generacion(max.readThermocoupleTemperature()-3);
      myGLCD.setFont(SmallFont);
      myGLCD.print("Temp union fria:  ",7,166);
      myGLCD.printNumF(max.readCJTemperature(),2, 160, 166);
```

```
    }
    if (aux_temp == 1){
    imprimir_generacion((max.readThermocoupleTemperature()*9/5)+32+37.4);
    myGLCD.setFont(SmallFont);
    myGLCD.print("Temp union fria:  ",7,166);
    myGLCD.printNumF((max.readCJTemperature()*9/5)+32,2, 150, 166);
    }

    // Check and print any faults
    myGLCD.setFont(SmallFont);
    uint8_t fault = max.readFault();
    if (fault) {
    if (fault & MAX31856_FAULT_CJRANGE) myGLCD.print("Fallo de rango union
fria",7,166);
    if (fault & MAX31856_FAULT_TCRANGE) myGLCD.print("Fallo de rango
Termocupla",7,166);
    if (fault & MAX31856_FAULT_CJHIGH)  myGLCD.print("Fallo union fria alta    ",7,166);
    if (fault & MAX31856_FAULT_CJLOW)   myGLCD.print("Fallo union fria baja    ",7,166);
    if (fault & MAX31856_FAULT_TCHIGH)  myGLCD.print("Fallo Termocupla alta    ",7,166);
    if (fault & MAX31856_FAULT_TCLOW)   myGLCD.print("Fallo Termocupla bajo    ",7,166);
    if (fault & MAX31856_FAULT_OVUV)    myGLCD.print("Fallo voltaje Over/Under ",7,166);
    if (fault & MAX31856_FAULT_OPEN)    myGLCD.print("Fallo sin termocupla     ",7,166);
 }

  }

  if ((MeSu == 1 || MeSu == 0) && aux_medicion == 20 && aux_loop == 0)
  {
    // MEDICION DE RTD's
    myGLCD.setFont(Grotesk32x64);
    digitalWrite(Res_prin, LOW);
    digitalWrite(Res_secu, LOW);
    delay(1);
    digitalWrite(TC_prin, HIGH);
    digitalWrite(TC_secu, HIGH);
    transformacion_F_C(260, 210);
    myGLCD.setFont(Grotesk32x64);
    salir_resistencia=1;

    aux_cad=2;
    aux_cadena_com = String (1.8);
    aux_cadena_com.toCharArray(cadena_com,6);
    Wire.beginTransmission(8); // transmit to device #8
    Wire.write(cadena_com);
    Wire.write(aux_cad);
    Wire.endTransmission();

    FilterOnePole lowpassFilter( LOWPASS, filterFrequency );
    while (salir_resistencia==1)
    {
     analogRead(Ohmetro); // dummy read: enable channel 0
     delay(10);
```

```
      //analog_dato_Vol.update();
      //valor_Vol = analog_dato_Vol.getValue();
      int rtd_rtd []={analogRead(Ohmetro), analogRead(Ohmetro), analogRead(Ohmetro),
analogRead(Ohmetro), analogRead(Ohmetro),};

      for (int x = 0; x < 200; x++){
       for (int y = 0; y < (200 - 1); y++){
        if (rtd_rtd[y] > rtd_rtd[y + 1]){
         int temp = rtd_rtd[y + 1];
         rtd_rtd[y + 1] = rtd_rtd[y];
         rtd_rtd[y] = temp;
        }
       }
      }

      valor2_Ohm = lowpassFilter.input (rtd_rtd[100]);
      rawData1_Ohm= valor2_Ohm;
      valor_adc_Ohm = (float)rawData1_Ohm * 245.0 / 3942.0;

      for (int ff = 0; ff <= 1050; ff++){
       if (valor_adc_Ohm >= 99.8){
        if (valor_adc_Ohm >= Pt100_tabla[ff] && valor_adc_Ohm <= Pt100_tabla[ff+1]){
         asasas = ff-200;
         ff=1080;
       }
       } else if (valor_adc_Ohm < 99.8){
          if (valor_adc_Ohm >= Pt100_tabla[ff] && valor_adc_Ohm <= Pt100_tabla[ff+1]){
         asasas = ff-200;
         ff=1080;
       }
        }
       }

      imprimir_generacion(asasas);

      if (myTouch.dataAvailable())
      {
       digitalWrite(TC_prin, LOW);
       digitalWrite(TC_secu, LOW);
       digitalWrite(Res_prin, LOW);
       digitalWrite(Res_secu, LOW);
       salir_resistencia=0;
      }
      delay(1);
     }
    }
/**************************************************************************
 * GENERADORES
**************************************************************************/
    //--------GENERACION VOLTAJE-------------
    if ((MeSu == 2 || MeSu == 0)  && (BoPr == 15))
    {
```

```
    imprimir_generacion(generacion_volt);
    myGLCD.setFont(SmallFont);
    myGLCD.printNumF(generacion_volt,2, 390, 258);
   }
   //--------GENERACION CORRIENTE---------
   if ((MeSu == 2 || MeSu == 0)  && (BoPr == 8))
   {
    imprimir_generacion(generacion_corr);
    digitalWrite(TC_prin, LOW);
    digitalWrite(TC_secu, LOW);
    digitalWrite(Res_prin, LOW);
    digitalWrite(Res_secu, LOW);
   }
   //--------GENERACION RESISTENCIA
   if ((MeSu == 2 || MeSu == 0)  && (BoPr == 19))
   {
    imprimir_generacion(generacion_res);
   }
   //--------GENERACION TERMOCUPLAS
   if ((MeSu == 2 || MeSu == 0)  && (BoPr == 16))
   {
    digitalWrite(TC_prin, LOW);
    digitalWrite(TC_secu, LOW);
    delay(1);
    digitalWrite(Res_prin, HIGH);
    digitalWrite(Res_secu, HIGH);

    if (aux_TCg == 1){
    imprimir_generacion(generacion_TCK);
    max.setThermocoupleType(MAX31856_TCTYPE_K);
    }
    if (aux_TCg == 0){
    imprimir_generacion(generacion_TCJ);
    max.setThermocoupleType(MAX31856_TCTYPE_J);
    }

    float compensacion_fria = max.readCJTemperature();
    myGLCD.setFont(SmallFont);
    myGLCD.printNumF(compensacion_fria,3, 300, 330);
   }
   //--------GENERACION RTD's
   if ((MeSu == 2 || MeSu == 0)  && (BoPr == 20))
   {
    imprimir_generacion(generacion_RTD);
   }
/*************************************************************************
 * RECONOCIMIENTO TACTIL
*************************************************************************/
   if (myTouch.dataAvailable())
   {
    myTouch.read();
    x = myTouch.getX();
```

```
y = myTouch.getY();
myGLCD.setFont(SmallFont);

if ((x >= 5) && (x <= 115)) //-------------------------1 COLUMNA
{
 if ((y >= 420) && (y <= 461.43)) //-----------------Button: CONTRAST
 {
  BoPr = 1;
  waitForIt(5, 420, 115, 461.43);
  myGLCD.print("          ", 390, 168);
  myGLCD.print("CONTRAST", 390, 168);
 }

 if ((y >= 471.43) && (y <= 564.29)) //-------------Button: Meas/Source
 {
  BoPr = 2;
  BoCal = 0;
  if (aux_MeSu == 0)  {
   MeSu=0;
   aux_MeSu=1;      }
  myGLCD.setFont(BigFont);
  waitForIt(5, 471.43, 115, 564.29);
  if ((MeSu == 0 || MeSu == 4) && aux_loop == 0)
  {
   aux_medicion = 0;
   waitForIt1(5, 180, 355, 280, 0, 0, 0, 0, 0);
   myGLCD.setFont(Grotesk32x64);
   myGLCD.print("       ", 80, 330);
   myGLCD.setFont(BigFont);
   myGLCD.setColor(VGA_BLACK);
   myGLCD.print("MEASURE", 7, 182);
   myGLCD.setColor(VGA_RED);
   myGLCD.print("SOURCE ", 7, 302);
   MeSu = 1;
  }
  else if (MeSu == 1 && aux_loop == 0)
  {
   aux_medi_gene = aux_medicion;
   myGLCD.setFont(Grotesk32x64);
   myGLCD.print("        ", 50, 210);
   myGLCD.setFont(BigFont);
   waitForIt1(5, 300, 355, 400, 0, 0, 0, 0, 0);
   myGLCD.setColor(VGA_BLACK);
   myGLCD.print("SOURCE ", 7, 302);
   myGLCD.setColor(VGA_RED);
   myGLCD.print("MEASURE", 7, 182);
   MeSu = 2;
  }
  else if (MeSu == 2 && aux_loop == 0)
  {
   aux_MeSu=2;
   aux_medicion = aux_medi_gene;
```

```
          myGLCD.setColor(VGA_RED);
          myGLCD.print("SOURCE ", 7, 302);
          myGLCD.print("MEASURE", 7, 182);
          waitForIt1(5, 300, 355, 400, 5, 180, 355, 280, 0);
          myGLCD.setColor(VGA_BLACK);
          myGLCD.print("SOURCE ", 7, 302);
          myGLCD.print("MEASURE", 7, 182);
          myGLCD.setColor(VGA_RED);
          MeSu = 0;
         }
         myGLCD.setFont(SmallFont);
         myGLCD.print("          ", 390, 168);
         myGLCD.print("Meas/source", 390, 168);
        }
        if (MeSu == 2 && (BoPr == 15 || BoPr == 8 || BoPr == 19 || BoPr == 9 || BoPr == 16 ||
BoPr == 20))
        {
         if ((y >= 574.29) && (y <= 615.72)) //---------------------------------------------Button: 100%
         {
          boton_especial = 3;
          contador = 0;
          waitForIt(5, 574.29, 115, 615.72);
          if (contador >= 100000)
          {
           dato = dato_100;
           aux_cambio = BoPr;
           if (BoPr == 8){     // CORRIENTE
            dato_100 = generacion_corr;
           }
           if (BoPr == 15){    // VOLTAJE
            dato_100 = generacion_volt;
           }
           if (BoPr == 16 && TCg == 0){    // TERMOCUPLA K
            dato_100 = generacion_TCK;
           }
           if (BoPr == 16 && TCg == 1){    // TERMOCUPLA J
            dato_100 = generacion_TCJ;
           }
           if (BoPr == 19){  // RESISTENCIA
            dato_100 = generacion_res;
           }
           if (BoPr == 20){  //RTD
            dato_100 = generacion_RTD;
           }

          }
          contador = 0;
          dato = dato_100;
          descomposicion (dato);

          myGLCD.print("          ", 390, 168);
          myGLCD.print("100%", 390, 168);
```

```
}

if ((y >= 625.72) && (y <= 667.15)) //-----------------------Button: 25% UP
{
  boton_especial = 4;
  waitForIt(5, 625.72, 115, 667.15);
  dato = dato + aux_dato;
  if (dato >= dato_100)    {
    dato = dato_100;
  }
  descomposicion (dato);
  myGLCD.print("          ", 390, 168);
  myGLCD.print("25% UP", 390, 168);
}

if ((y >= 677.15) && (y <= 718.58)) //---------------------------------Button: 25% DOWN
{
  boton_especial = 5;
  waitForIt(5, 677.15, 115, 718.58);
  dato = dato - aux_dato;
  if (dato <= dato_0)     {
    dato = dato_0;
  }
  descomposicion (dato);
  myGLCD.print("          ", 390, 168);
  myGLCD.print("25% DOWN", 390, 168);
}

if ((y >= 728.58) && (y <= 770.01)) //----------------------------------Button: 0%
{
  boton_especial = 6;
  contador = 0;
  waitForIt(5, 728.58, 115, 770.01);
  if (contador >= 100000)
  {
    dato = dato_0;
    if (BoPr == 8){    // CORRIENTE
      dato_0 = generacion_corr;
    }
    if (BoPr == 15){   // VOLTAJE
      dato_0 = generacion_volt;
    }
    if (BoPr == 16 && TCg == 0){   // TERMOCUPLA K
      dato_0 = generacion_TCK;
    }
    if (BoPr == 16 && TCg == 1){   // TERMOCUPLA J
      dato_0 = generacion_TCJ;
    }
    if (BoPr == 19){  // RESISTENCIA
      dato_0 = generacion_res;
    }
    if (BoPr == 20){  //RTD
```

```
        dato_0 = generacion_RTD;
       }
      }
      contador = 0;
      dato = dato_0;
      descomposicion (dato);
      myGLCD.print("          ", 390, 168);
      myGLCD.print("0%", 390, 168);
    }
  }
}
if (BoPr == 16)
{
if ((x >= 365) && (x <= 475))
{
  if ((y >= 358.57) && (y <= 400))
  {
    aux_termocupla=1;
    myGLCD.setFont(SmallFont);
    myGLCD.print("TC TIPO J", 270, 305);
   }
   if ((y >= 300) && (y <= 341.43))
   {
    aux_termocupla=2;
    myGLCD.setFont(SmallFont);
    myGLCD.print("TC TIPO K", 270, 305);
   }
}
}

//----------------------------------------------------------------------
//----------------------------------------------------------------------
if ((x >= 365) && (x <= 475)) //-------------------------------------4 COLUMNA
{
  if ((y >= 420) && (y <= 461.43)) //---------------------------------Button: LOOP
  {
    BoPr = 7;
    waitForIt(365, 420, 475, 461.43);
    if (aux_loop == 0)
    {
      BoPr = 0;
      MeSu = 3;
      aux_medicion = 0;
      myGLCD.print("  LOOP", 285, 182);
      myGLCD.print("ACTIVADO", 285, 192);
      myGLCD.setFont(Grotesk32x64);
      myGLCD.print("       ", 80, 330);
      myGLCD.setFont(SmallFont);
      aux_loop = 1;
    }
    else if (aux_loop == 1)
    {
```

```
   MeSu = 4;
   myGLCD.print("   LOOP    ", 260, 182);
   myGLCD.print("DESACTIVADO ", 260, 192);
   myGLCD.setFont(Grotesk32x64);
   myGLCD.print("        ", 50, 210);
   myGLCD.setFont(SmallFont);
   delay(1000);
   myGLCD.print("          ", 260, 182);
   myGLCD.print("          ", 260, 192);
   aux_loop = 0;
  }

  myGLCD.print("          ", 390, 168);
  myGLCD.print("LOOP", 390, 168);
}

if ((y >= 471.43) && (y <= 512.86)) //-------------------------------------Button: mA
{
  BoPr = 8;
  aux_medicion = 8;
  waitForIt(365, 471.43, 475, 512.86);
  borrar_medicion();
  borrar_generacion();
  potenciometro3=multiplicador+potenciometro3;
  myGLCD.print("     ", 425, 208);
  myGLCD.printNumI(potenciometro3, 425, 208);

  if (MeSu == 2)
  {

   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 240, 395);
   myGLCD.setColor(VGA_RED);
   myGLCD.drawLine(208, 395, 240, 395);
   aux_izde = 208;
   num_uni = num_dec = num_cen = num_mil = 0;
   num1 = num2 = num3 = num4 = 0;
  }
  myGLCD.print("          ", 390, 168);
  myGLCD.print("mA", 390, 168);
}

if ((y >= 522.86) && (y <= 564.29)) //-------------------------------------Button: Hz
{
  BoPr = 9;
  aux_medicion = 9;
  waitForIt(365, 522.86, 475, 564.29);
  borrar_medicion();
  borrar_generacion();
  potenciometro3=potenciometro3-multiplicador;
  myGLCD.print("     ", 425, 208);
  myGLCD.printNumI(potenciometro3, 425, 208);
```

```
 if (MeSu == 2)
 {
  if (aux_fre == 0)
  {
   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 206, 395);
   myGLCD.setColor(VGA_RED);
   myGLCD.drawLine(144, 395, 174, 395);
   aux_izde = 144;
   num_uni = num_dec = num_cen = num_mil = 0;
   num1 = num2 = num3 = num4 = 0;
   aux_fre = 1;
  }
  else if (aux_fre == 1)
  {
   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 206, 395);
   myGLCD.setColor(VGA_RED);
   myGLCD.drawLine(144, 395, 174, 395);
   aux_izde = 144;
   num_uni = num_dec = num_cen = num_mil = 0;
   num1 = num2 = num3 = num4 = 0;
   aux_fre = 0;
  }
 }
 myGLCD.print("          ", 390, 168);
 myGLCD.print("Hz", 390, 168);
}

if ((y >= 574.29) && (y <= 615.72)) //-------------------------------------Button: Rampas
{
 BoPr = 10;
 voltaje_calibracion++;
 myGLCD.print("     ", 425, 218);
 myGLCD.printNumF(voltaje_calibracion,1, 425, 218);
 waitForIt(365, 574.29, 475, 615.72);
 myGLCD.print("          ", 390, 168);
 myGLCD.print("FORMAS", 390, 168);
}

if ((y >= 625.72) && (y <= 667.15)) //-------------------------------Button: FUNTION
{
 BoPr = 11;
  multiplicador=100;
 voltaje_calibracion--;
 myGLCD.print("     ", 425, 218);
 myGLCD.printNumF(voltaje_calibracion,1, 425, 218);
 waitForIt(365, 625.72, 475, 667.15);
 myGLCD.print("          ", 390, 168);
 myGLCD.print("FUNTION", 390, 168);
}
```

```
if ((y >= 677.15) && (y <= 718.58)) //---------------------------------Button: RECALL
{
 BoPr = 12;
  multiplicador=10;
 waitForIt(365, 677.15, 475, 718.58);
 myGLCD.print("          ", 390, 168);
 myGLCD.print("RECALL", 390, 168);
}

if ((y >= 728.58) && (y <= 770.01)) //---------------------------------Button: ENTER
{
 waitForIt(365, 728.58, 475, 770.01);
 if (BoCal == 0) {

  if ((MeSu == 2 || MeSu == 0) && BoPr == 15) {        //Voltaje
   if (generacion_volt >= 0 && generacion_volt <= 5) {
    aux_cad=0;          }
   if (generacion_volt > 5 && generacion_volt <= 10) {
    aux_cad=1;                              }
  aux_cadena_com = String (generacion_volt);        }

  if ((MeSu == 2 || MeSu == 0) && BoPr == 8) {        //Corriente
  if (generacion_corr <= 24) {
    aux_cad=2;          }
  aux_cadena_com = String (generacion_corr);     }

  if ((MeSu == 2 || MeSu == 0) && BoPr == 16) {        //TC
  if (aux_TCg == 0){
   if (generacion_TCJ >= -80 &&  generacion_TCJ < -70){
    factor_x = generacion_TCJ + 80;
    factor_inical = TCJinicial;
    TCJrango = 40.5 * factor_x;    // 39.5 * 2 = 79
    generarTCJ = factor_inical + TCJrango; // envio
   } else if (generacion_TCJ >= -70 &&  generacion_TCJ < -60){
    factor_x = generacion_TCJ + 70;
    factor_inical = TCJinicial + 405;
    TCJrango = 41.5 * factor_x;
    generarTCJ = factor_inical + TCJrango;
   } else if (generacion_TCJ >= -60 &&  generacion_TCJ < -50){
    factor_x = generacion_TCJ + 60;
    factor_inical = TCJinicial + 820;
    TCJrango = 42.5 * factor_x;
    generarTCJ = factor_inical + TCJrango;
   } else if (generacion_TCJ >= -50 &&  generacion_TCJ < -40){
    factor_x = generacion_TCJ + 50;
    factor_inical = TCJinicial + 1245;
    TCJrango = 43.5 * factor_x;
    generarTCJ = factor_inical + TCJrango;
   } else if (generacion_TCJ >= -40 &&  generacion_TCJ < -30){
    factor_x = generacion_TCJ + 40;
    factor_inical = TCJinicial + 1680;
    TCJrango = 44 * factor_x;
```

```
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= -30 &&  generacion_TCJ < -20){
  factor_x = generacion_TCJ + 30;
  factor_inical = TCJinicial + 2120;
  TCJrango = 43.5 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= -20 &&  generacion_TCJ < -10){
  factor_x = generacion_TCJ + 20;
  factor_inical = TCJinicial + 2555;
  TCJrango = 45.5 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= -10 &&  generacion_TCJ < 0){
  factor_x = generacion_TCJ + 10;
  factor_inical = TCJinicial + 3010;
  TCJrango = 51.5 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 0 &&  generacion_TCJ < 10){
  factor_x = generacion_TCJ + 0;
  factor_inical = TCJinicial + 3525;
  TCJrango = 47 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 10 &&  generacion_TCJ < 20){
  factor_x = generacion_TCJ - 10;
  factor_inical = TCJinicial + 3995;
  TCJrango = 47 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 20 &&  generacion_TCJ < 30){
  factor_x = generacion_TCJ - 20;
  factor_inical = TCJinicial + 4465;
  TCJrango = 46 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 30 &&  generacion_TCJ < 40){
  factor_x = generacion_TCJ - 30;
  factor_inical = TCJinicial + 4925;
  TCJrango = 49 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 40 &&  generacion_TCJ < 50){
  factor_x = generacion_TCJ - 40;
  factor_inical = TCJinicial + 5415;
  TCJrango = 48 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 50 &&  generacion_TCJ < 60){
  factor_x = generacion_TCJ - 50;
  factor_inical = TCJinicial + 5895;
  TCJrango = 48.5 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 60 &&  generacion_TCJ < 70){
  factor_x = generacion_TCJ - 60;
  factor_inical = TCJinicial + 6380;
  TCJrango = 49 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 70 &&  generacion_TCJ < 80){
```

```
  factor_x = generacion_TCJ - 70;
  factor_inical = TCJinicial + 6870;
  TCJrango = 49.5 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 80 &&  generacion_TCJ < 90){
  factor_x = generacion_TCJ - 80;
  factor_inical = TCJinicial + 7365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 90 &&  generacion_TCJ < 100){
  factor_x = generacion_TCJ - 90;
  factor_inical = TCJinicial + 7865;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 100 &&  generacion_TCJ < 110){
  factor_x = generacion_TCJ - 100;
  factor_inical = TCJinicial + 8365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 110 &&  generacion_TCJ < 120){
  factor_x = generacion_TCJ - 110;
  factor_inical = TCJinicial + 8865;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 120 &&  generacion_TCJ < 130){
  factor_x = generacion_TCJ - 120;
  factor_inical = TCJinicial + 9365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 130 &&  generacion_TCJ < 140){
  factor_x = generacion_TCJ - 130;
  factor_inical = TCJinicial + 9865;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 140 &&  generacion_TCJ < 150){
  factor_x = generacion_TCJ - 140;
  factor_inical = TCJinicial + 10365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 150 &&  generacion_TCJ < 160){
  factor_x = generacion_TCJ - 150;
  factor_inical = TCJinicial + 10865;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 160 &&  generacion_TCJ < 170){
  factor_x = generacion_TCJ - 160;
  factor_inical = TCJinicial + 11365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 170 &&  generacion_TCJ < 180){
  factor_x = generacion_TCJ - 170;
  factor_inical = TCJinicial + 11865;
```

```
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 180 &&  generacion_TCJ < 190){
  factor_x = generacion_TCJ - 180;
  factor_inical = TCJinicial + 12365;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 190 &&  generacion_TCJ < 200){
  factor_x = generacion_TCJ - 190;
  factor_inical = TCJinicial + 12865;
  TCJrango = 51 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 200 &&  generacion_TCJ < 210){
  factor_x = generacion_TCJ - 200;
  factor_inical = TCJinicial + 13375;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 210 &&  generacion_TCJ < 220){
  factor_x = generacion_TCJ - 210;
  factor_inical = TCJinicial + 13875;
  TCJrango = 50 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 220 &&  generacion_TCJ < 230){
  factor_x = generacion_TCJ - 220;
  factor_inical = TCJinicial + 14375;
  TCJrango = 51 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 230 &&  generacion_TCJ < 240){
  factor_x = generacion_TCJ - 230;
  factor_inical = TCJinicial + 14885;
  TCJrango = 51 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >= 240 &&  generacion_TCJ < 250){
  factor_x = generacion_TCJ - 240;
  factor_inical = TCJinicial + 15395;
  TCJrango = 51 * factor_x;
  generarTCJ = factor_inical + TCJrango;
 } else if (generacion_TCJ >=250){
  factor_x = generacion_TCJ - 250;
  factor_inical = TCJinicial + 15905;
  TCJrango = 50.6 * factor_x;
  generarTCJ = factor_inical + TCJrango;
}}

if (aux_TCg==1){
if (generacion_TCK >= -100 &&  generacion_TCK < -90){
 factor_x = generacion_TCK + 100;
 factor_inical = TCKinicial;
 TCKrango = 27.5 * factor_x;    // 39.5 * 2 = 79
 generarTCK = factor_inical + TCKrango; // envio
} else if (generacion_TCK >= -90 &&  generacion_TCK < -80){
 factor_x = generacion_TCK + 90;
```

```
  factor_inical = TCKinicial + 275;
  TCKrango = 28.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -80 &&  generacion_TCK < -70){
  factor_x = generacion_TCK + 80;
  factor_inical = TCKinicial + 560;
  TCKrango = 29.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -70 &&  generacion_TCK < -60){
  factor_x = generacion_TCK + 70;
  factor_inical = TCKinicial + 855;
  TCKrango = 30.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -60 &&  generacion_TCK < -50){
  factor_x = generacion_TCK + 60;
  factor_inical = TCKinicial + 1160;
  TCKrango = 30.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -50 &&  generacion_TCK < -40){
  factor_x = generacion_TCK + 50;
  factor_inical = TCKinicial + 1465;
  TCKrango = 32.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -40 &&  generacion_TCK < -30){
  factor_x = generacion_TCK + 40;
  factor_inical = TCKinicial + 1785;
  TCKrango = 32.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -30 &&  generacion_TCK < -20){
  factor_x = generacion_TCK + 30;
  factor_inical = TCKinicial + 2110;
  TCKrango = 33.0 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -20 &&  generacion_TCK < -10){
  factor_x = generacion_TCK + 20;
  factor_inical = TCKinicial + 2440;
  TCKrango = 34 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= -10 &&  generacion_TCK < 0){
  factor_x = generacion_TCK + 10;
  factor_inical = TCKinicial + 2780;
  TCKrango = 34 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 0 &&  generacion_TCK < 10){
  factor_x = generacion_TCK ;
  factor_inical = TCKinicial + 3120;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 10 &&  generacion_TCK < 20){
  factor_x = generacion_TCK - 10;
  factor_inical = TCKinicial + 3470;
  TCKrango = 35.5 * factor_x;
```

```
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 20 &&  generacion_TCK < 30){
  factor_x = generacion_TCK - 20;
  factor_inical = TCKinicial + 3825;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 30 &&  generacion_TCK < 40){
  factor_x = generacion_TCK - 30;
  factor_inical = TCKinicial + 4180;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 40 &&  generacion_TCK < 50){
  factor_x = generacion_TCK - 40;
  factor_inical = TCKinicial + 4540;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 50 &&  generacion_TCK < 60){
  factor_x = generacion_TCK - 50;
  factor_inical = TCKinicial + 4900;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 60 &&  generacion_TCK < 70){
  factor_x = generacion_TCK - 60;
  factor_inical = TCKinicial + 5260;
  TCKrango = 37 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 70 &&  generacion_TCK < 80){
  factor_x = generacion_TCK - 70;
  factor_inical = TCKinicial + 5630;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 80 &&  generacion_TCK < 90){
  factor_x = generacion_TCK - 80;
  factor_inical = TCKinicial + 5995;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 90 &&  generacion_TCK < 100){
  factor_x = generacion_TCK - 90;
  factor_inical = TCKinicial + 6360;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 100 &&  generacion_TCK < 110){
  factor_x = generacion_TCK - 100;
  factor_inical = TCKinicial + 6725;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 110 &&  generacion_TCK < 120){
  factor_x = generacion_TCK - 110;
  factor_inical = TCKinicial + 7085;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 120 &&  generacion_TCK < 130){
```

```
  factor_x = generacion_TCK - 120;
  factor_inical = TCKinicial + 7445;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 130 &&  generacion_TCK < 140){
  factor_x = generacion_TCK - 130;
  factor_inical = TCKinicial + 7805;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 140 &&  generacion_TCK < 150){
  factor_x = generacion_TCK - 140;
  factor_inical = TCKinicial + 8160;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 150 &&  generacion_TCK < 160){
  factor_x = generacion_TCK - 150;
  factor_inical = TCKinicial + 8515;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 160 &&  generacion_TCK < 170){
  factor_x = generacion_TCK - 160;
  factor_inical = TCKinicial + 8865;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 170 &&  generacion_TCK < 180){
  factor_x = generacion_TCK - 170;
  factor_inical = TCKinicial + 9475;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 180 &&  generacion_TCK < 190){
  factor_x = generacion_TCK - 180;
  factor_inical = TCKinicial + 9825;
  TCKrango = 34.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 190 &&  generacion_TCK < 200){
  factor_x = generacion_TCK - 190;
  factor_inical = TCKinicial + 10170;
  TCKrango = 34.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 200 &&  generacion_TCK < 210){
  factor_x = generacion_TCK - 200;
  factor_inical = TCKinicial + 10515;
  TCKrango = 34.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 210 &&  generacion_TCK < 220){
  factor_x = generacion_TCK - 210;
  factor_inical = TCKinicial + 10860;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 220 &&  generacion_TCK < 230){
  factor_x = generacion_TCK - 220;
  factor_inical = TCKinicial + 11210;
```

```
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 230 &&  generacion_TCK < 240){
  factor_x = generacion_TCK - 230;
  factor_inical = TCKinicial + 11560;
  TCKrango = 35 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 240 &&  generacion_TCK < 250){
  factor_x = generacion_TCK - 240;
  factor_inical = TCKinicial + 11910;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 250 &&  generacion_TCK < 260){
  factor_x = generacion_TCK - 250;
  factor_inical = TCKinicial + 12265;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 260 &&  generacion_TCK < 270){
  factor_x = generacion_TCK - 260;
  factor_inical = TCKinicial + 12620;
  TCKrango = 35.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 270 &&  generacion_TCK < 280){
  factor_x = generacion_TCK - 270;
  factor_inical = TCKinicial + 12975;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 280 &&  generacion_TCK < 290){
  factor_x = generacion_TCK - 280;
  factor_inical = TCKinicial + 13335;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 290 &&  generacion_TCK < 300){
  factor_x = generacion_TCK - 290;
  factor_inical = TCKinicial + 13695;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 300 &&  generacion_TCK < 310){
  factor_x = generacion_TCK - 300;
  factor_inical = TCKinicial + 14055;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 310 &&  generacion_TCK < 320){
  factor_x = generacion_TCK - 310;
  factor_inical = TCKinicial + 14415;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
 } else if (generacion_TCK >= 320 &&  generacion_TCK < 330){
  factor_x = generacion_TCK - 320;
  factor_inical = TCKinicial + 14775;
  TCKrango = 36 * factor_x;
  generarTCK = factor_inical + TCKrango;
```

```
} else if (generacion_TCK >= 330 &&  generacion_TCK < 340){
  factor_x = generacion_TCK - 330;
  factor_inical = TCKinicial + 15135;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
} else if (generacion_TCK >= 340 &&  generacion_TCK < 350){
  factor_x = generacion_TCK - 340;
  factor_inical = TCKinicial + 1550;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
} else if (generacion_TCK >= 350 &&  generacion_TCK < 360){
  factor_x = generacion_TCK - 350;
  factor_inical = TCKinicial + 15865;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
} else if (generacion_TCK >= 360 &&  generacion_TCK < 370){
  factor_x = generacion_TCK - 360;
  factor_inical = TCKinicial + 16230;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
} else if (generacion_TCK >= 370 ){
  factor_x = generacion_TCK - 370;
  factor_inical = TCKinicial + 16595;
  TCKrango = 36.5 * factor_x;
  generarTCK = factor_inical + TCKrango;
}}



aux_cad=6;
if (aux_TCg == 0){
aux_cadena_com = String (generarTCJ/100);}

if (aux_TCg == 1){
aux_cadena_com = String (generarTCK/100);}

}
if ((MeSu == 2 || MeSu == 0) && BoPr == 19)    {       //Resistencia
  aux_cad=4;
  aux_cadena_com = String (generacion_res);     }

  // aux_cad='3'; 4-20mA

if ((MeSu == 2 || MeSu == 0) && BoPr == 20)  {       //RTD
  aux_cad=5;
  aux_cadena_com = String (generacion_RTD);     }

  if (MeSu == 1  && (BoPr == 15 || BoPr == 16))  {      // potenciometro1
  aux_cad=7;
  float pottt1 = potenciometro1/100.0;
  aux_cadena_com = String (pottt1);     }
```

```
    if (MeSu == 1  && (BoPr == 19 || BoPr == 20))  {      // potenciometro2
    aux_cad=8;
    float pottt2 = potenciometro2/100.0;
    aux_cadena_com = String (pottt2);     }

    if (MeSu == 1  && (BoPr == 8 || BoPr == 9))  {      // potenciometro3
    aux_cad=9;
    float pottt3 = potenciometro3/100.0;
    aux_cadena_com = String (pottt3);     }

    if (MeSu == 1  && (BoPr == 10 || BoPr == 11))  {      // voltaje
    aux_cad=6;
    float pottt4 = voltaje_calibracion/100.0;
    aux_cadena_com = String (pottt4);     }

    myGLCD.print("POT1: ", 390, 188);
    myGLCD.print("POT2: ", 390, 198);
    myGLCD.print("POT3: ", 390, 208);
    myGLCD.print("VOL1: ", 390, 218);
    myGLCD.print("     ", 425, 188);
    myGLCD.printNumI(potenciometro1, 425, 188);
    myGLCD.print("     ", 425, 198);
    myGLCD.printNumI(potenciometro2, 425, 198);
    myGLCD.print("     ", 425, 208);
    myGLCD.printNumI(potenciometro3, 425, 208);
    myGLCD.print("     ", 425, 218);
    myGLCD.printNumF(voltaje_calibracion,0, 425, 218);
    // aux_cad='6'; Termocupla

   myGLCD.setFont(SmallFont);
   myGLCD.printNumI(BoCal, 360, 178);

   aux_cadena_com.toCharArray(cadena_com,6);
   Wire.beginTransmission(8); // transmit to device #8
   Wire.write(cadena_com);
   Wire.write(aux_cad);
   Wire.endTransmission();    // stop transmitting
  waitForIt(365, 728.58, 475, 770.01);
  myGLCD.print("          ", 390, 168);
  myGLCD.print(cadena_com, 390, 168);
  for (int i1=0; i1 <= 3; i1++){
  cadena_com[i1]=' ';}

}

if (BoCal == 1 ){
  posicion_calibracion++;
  if (posicion_calibracion == 8){
    BoCal == 3;
  }
}
```

```
  }

 }
 if ((x >= 125) && (x <= 235)) //-------------------------------------2 COLUMNA
 {
  if ((y >= 420) && (y <= 461.43)) //-----------------------------Button: Grado F.
  {
   aux_temp = 1;
   waitForIt(125, 420, 235, 461.43);
   myGLCD.setFont(SmallFont);
   myGLCD.print("          ", 390, 168);
   myGLCD.print("GRADOS F.", 390, 168);
  }

  if ((y >= 471.43) && (y <= 512.86)) //----------------------------Button: V
  {
   BoPr = 15;
   aux_medicion = 15;
   waitForIt(125, 471.43, 235, 512.86);
   borrar_medicion();
   borrar_generacion();
   potenciometro1=multiplicador+potenciometro1;
   myGLCD.print("     ", 425, 188);
   myGLCD.printNumI(potenciometro1, 425, 188);

   if (MeSu == 2 && aux_TCg==0)  {
    myGLCD.setColor(VGA_WHITE);
    myGLCD.drawLine(80, 395, 206, 395);
    myGLCD.setColor(VGA_RED);
    myGLCD.drawLine(176, 395, 206, 395);
    aux_izde = 208;
    num_uni = num_dec = num_cen = num_mil = 0;
    num1 = num2 = num3 = num4 = 0;
    aux_TCg = 1;
   }else if (MeSu == 2 && aux_TCg == 1) {
    myGLCD.setColor(VGA_WHITE);
    myGLCD.drawLine(80, 395, 206, 395);
    myGLCD.setColor(VGA_RED);
    myGLCD.drawLine(176, 395, 206, 395);
    aux_izde = 208;
    num_uni = num_dec = num_cen = num_mil = 0;
    num1 = num2 = num3 = num4 = 0;
    aux_TCg = 0;
   }
   if (MeSu == 2)
   {
    myGLCD.setColor(VGA_WHITE);
    myGLCD.drawLine(80, 395, 240, 395);
    myGLCD.setColor(VGA_RED);
    myGLCD.drawLine(208, 395, 240, 395);
    aux_izde = 208;
    num_uni = num_dec = num_cen = num_mil = 0;
```

```
   num1 = num2 = num3 = num4 = 0;
  }

}

if ((y >= 522.86) && (y <= 564.29)) //---------------------------------------Button: TC
{
  BoPr = 16;
  aux_medicion = 16;
  waitForIt(125, 522.86, 235, 564.29);
  borrar_medicion();
  borrar_generacion();
  potenciometro1=potenciometro1-multiplicador;
  myGLCD.print("     ", 425, 188);
  myGLCD.printNumI(potenciometro1, 425, 188);
  if ( aux_TC == 0 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_B);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO B",240, 182);
    aux_TC = 1;
  } else if ( aux_TC == 1 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_E);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO E",240, 182);
    aux_TC = 2;
  } else if ( aux_TC == 2 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_J);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO J",240, 182);
    aux_TC = 3;
  } else if ( aux_TC == 3 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_K);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO K",240, 182);
    aux_TC = 4;
  } else if ( aux_TC == 4 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_N);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO N",240, 182);
    aux_TC = 5;
  } else if ( aux_TC == 5 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_R);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO R",240, 182);
    aux_TC = 6;
  } else if ( aux_TC == 6 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_S);
    myGLCD.setFont(BigFont);
    myGLCD.print("TIPO S",240, 182);
    aux_TC = 7;
  } else if ( aux_TC == 7 && (MeSu == 1 || MeSu == 0)){
    max.setThermocoupleType(MAX31856_TCTYPE_T);
```

```
   myGLCD.setFont(BigFont);
   myGLCD.print("TIPO T",240, 182);
   aux_TC = 0;
  }

  if (MeSu == 2 && aux_TCg==0)  {
   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 206, 395);
   myGLCD.setColor(VGA_RED);
   myGLCD.drawLine(176, 395, 206, 395);
   aux_izde = 176;
   num_uni = num_dec = num_cen = num_mil = 0;
   num1 = num2 = num3 = num4 = 0;
   myGLCD.setFont(BigFont);
   myGLCD.print("TIPO K",240, 302);
   aux_TCg = 1;
  }else if (MeSu == 2 && aux_TCg == 1) {
   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 206, 395);
   myGLCD.setColor(VGA_RED);
   myGLCD.drawLine(176, 395, 206, 395);
   aux_izde = 176;
   num_uni = num_dec = num_cen = num_mil = 0;
   num1 = num2 = num3 = num4 = 0;
   myGLCD.setFont(BigFont);
   myGLCD.print("TIPO J",240, 302);
   aux_TCg = 0;
  }
  myGLCD.print("          ", 390, 168);
  myGLCD.print("TC", 390, 168);

 }

 if ((y >= 625.72) && (y <= 667.15)) //-----------------------------------Button: LEFT
 {
  //BoPr=17;
  waitForIt(125, 625.72, 235, 667.15);
  myGLCD.print("          ", 390, 168);
  myGLCD.print("LEFT", 390, 168);

  if (BoPr == 8 || BoPr == 15){
  if (aux_izde <= 80)   {
   aux_izde = 240;
  }
  if (aux8 == 0)                                {
   aux_izde = aux_izde - 32;
   if ((BoPr == 8 || BoPr == 15)&&(aux_izde ==144)) {
   aux_izde = aux_izde - 32;
                          }
   myGLCD.setColor(VGA_WHITE);
   myGLCD.drawLine(80, 395, 240, 395);
   myGLCD.setColor(VGA_RED);
```

```
 myGLCD.drawLine(aux_izde, 395, aux_izde + 30, 395);
 aux8 = 1;
}
if (aux8 == 1)  {
 aux8 = 0;
}
}

if (BoPr == 19 || BoPr == 20 || BoPr == 16){
if (aux_izde <= 80)   {
 aux_izde = 208;
}
if (aux8 == 0)                              {
 aux_izde = aux_izde - 32;
 myGLCD.setColor(VGA_WHITE);
 myGLCD.drawLine(80, 395, 240, 395);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawLine(aux_izde, 395, aux_izde + 30, 395);
 aux8 = 1;
}
if (aux8 == 1)  {
 aux8 = 0;
}
}

if (BoCal == 1 && posicion_calibracion == 1){
if (aux_izde_ca <= 355)   {
 aux_izde_ca = 435;
}
if (aux8_ca == 0)                                 {
 aux_izde_ca = aux_izde_ca - 16;
 if (aux_izde_ca == 387) {
   aux_izde = aux_izde - 16;
 }
 myGLCD.setColor(VGA_WHITE);
 myGLCD.drawLine(355, 215, 435, 215);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawLine(aux_izde_ca, 215, aux_izde_ca + 15, 215);
 aux8_ca = 1;
}
if (aux8_ca == 1)  {
 aux8_ca = 0;
}
}

if (BoCal == 1 && posicion_calibracion == 2){
if (aux_izde_ca <= 355)   {
 aux_izde_ca = 435;
}
if (aux8_ca == 0)                                 {
 aux_izde_ca = aux_izde_ca - 16;
 if (aux_izde_ca == 387) {
```

```
   aux_izde = aux_izde - 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 235, 435, 235);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 235, aux_izde_ca + 15, 235);
  aux8_ca = 1;
 }
 if (aux8_ca == 1)  {
  aux8_ca = 0;
 }
 }

 if (BoCal == 1 && posicion_calibracion == 3){
 if (aux_izde_ca <= 355)   {
  aux_izde_ca = 435;
 }
 if (aux8_ca == 0)                              {
  aux_izde_ca = aux_izde_ca - 16;
  if (aux_izde_ca == 387) {
    aux_izde = aux_izde - 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 255, 435, 255);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 255, aux_izde_ca + 15, 255);
  aux8_ca = 1;
 }
 if (aux8_ca == 1)  {
  aux8_ca = 0;
 }
 }

 if (BoCal == 1 && posicion_calibracion == 4){
 if (aux_izde_ca <= 355)   {
  aux_izde_ca = 435;
 }
 if (aux8_ca == 0)                              {
  aux_izde_ca = aux_izde_ca - 16;
  if (aux_izde_ca == 387) {
    aux_izde = aux_izde - 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 275, 435, 275);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 275, aux_izde_ca + 15, 275);
  aux8_ca = 1;
 }
 if (aux8_ca == 1)  {
  aux8_ca = 0;
 }
 }
```

```
  if (BoCal == 1 && posicion_calibracion == 6){
   if (aux_izde_ca <= 355)   {
    aux_izde_ca = 435;
   }
   if (aux8_ca == 0)                              {
    aux_izde_ca = aux_izde_ca - 16;
    if (aux_izde_ca == 387) {
      aux_izde = aux_izde - 16;
    }
    myGLCD.setColor(VGA_WHITE);
    myGLCD.drawLine(355, 355, 435, 355);
    myGLCD.setColor(VGA_RED);
    myGLCD.drawLine(aux_izde_ca, 355, aux_izde_ca + 15, 355);
    aux8_ca = 1;
   }
   if (aux8_ca == 1)  {
    aux8_ca = 0;
   }
   }

   if (BoCal == 1 && posicion_calibracion == 7){
   if (aux_izde_ca <= 355)   {
    aux_izde_ca = 435;
   }
   if (aux8_ca == 0)                              {
    aux_izde_ca = aux_izde_ca - 16;
    if (aux_izde_ca == 387) {
      aux_izde = aux_izde - 16;
    }
    myGLCD.setColor(VGA_WHITE);
    myGLCD.drawLine(355, 375, 435, 375);
    myGLCD.setColor(VGA_RED);
    myGLCD.drawLine(aux_izde_ca, 375, aux_izde_ca + 15, 375);
    aux8_ca = 1;
   }
   if (aux8_ca == 1)  {
    aux8_ca = 0;
   }
   }
   /////

  }

}

if ((x >= 245) && (x <= 355)) //---------------------------------------3 COLUMNA
{
  if ((y >= 420) && (y <= 461.43)) //--------------------------------Button: Grados C.
  {
    aux_temp = 0;
    waitForIt(245, 420, 355, 461.43);
```

```
 myGLCD.setFont(SmallFont);
 myGLCD.print("          ", 390, 168);
 myGLCD.print("Grados C.", 390, 168);
}

if ((y >= 471.43) && (y <= 512.86)) //-------------------------------------Button: OHM
{
 BoPr = 19;
 aux_medicion = 19;
 waitForIt(245, 471.43, 355, 512.86);
 borrar_medicion();
 borrar_generacion();
 potenciometro2=potenciometro2+multiplicador;
 myGLCD.print("     ", 425, 198);
 myGLCD.printNumI(potenciometro2, 425, 198);
 if (MeSu == 2)
 {
  myGLCD.setFont(Grotesk32x64);
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(80, 395, 206, 395);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(176, 395, 206, 395);
  aux_izde = 176;
  num_uni = num_dec = num_cen = num_mil = 0;
  num1 = num2 = num3 = num4 = 0;
  myGLCD.setFont(SmallFont);
 }
 myGLCD.print("          ", 390, 168);
 myGLCD.print("OHM", 390, 168);
}

if ((y >= 522.86) && (y <= 564.29)) //-------------------------------------Button: RTD
{
 BoPr = 20;
 aux_medicion = 20;
 waitForIt(245, 522.86, 355, 564.29);
 borrar_medicion();
 borrar_generacion();
 potenciometro2=potenciometro2-multiplicador;
 myGLCD.print("     ", 425, 198);
 myGLCD.printNumI(potenciometro2, 425, 198);
 if (MeSu == 2 )
 {
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(80, 395, 206, 395);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(176, 395, 206, 395);
  aux_izde = 176;
  num_uni = num_dec = num_cen = num_mil = 0;
  num1 = num2 = num3 = num4 = 0;
 }
 myGLCD.print("          ", 390, 168);
```

```
 myGLCD.print("RTD", 390, 168);
}

if ((y >= 625.72) && (y <= 667.15)) //-----------------------------------Button: RIGHT
{

 //BoPr=21;
 waitForIt(245, 625.72, 355, 667.15);
 myGLCD.print("          ", 390, 168);
 myGLCD.print("RIGHT", 390, 168);

 if (BoPr == 8 || BoPr == 15){
 if (aux_izde >= 208) {
  aux_izde = 48;
 }
 if (aux7 == 0)                              {
  aux_izde = aux_izde + 32;
  if (aux_izde ==144) {
  aux_izde = aux_izde + 32;
                        }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(80, 395, 240, 395);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde, 395, aux_izde + 30, 395);
  aux7 = 1;
 }
 if (aux7 == 1) {
  aux7 = 0;
 } }

 if (BoPr == 19 || BoPr == 20 || BoPr == 16 ){
 if (aux_izde >= 176) {
  aux_izde = 48;
 }
 if (aux7 == 0)                              {
  aux_izde = aux_izde + 32;
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(80, 395, 240, 395);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde, 395, aux_izde + 30, 395);
  aux7 = 1;
 }
 if (aux7 == 1) {
  aux7 = 0;
 } }


 if (BoCal == 1 && posicion_calibracion == 1){
 if (aux_izde_ca >= 419) {
  aux_izde_ca = 339;
 }
 if (aux7_ca == 0)                                {
```

```
  aux_izde_ca = aux_izde_ca + 16;
  if (aux_izde_ca == 387) {
    aux_izde_ca = aux_izde_ca + 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 215, 435, 215);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 215, aux_izde_ca + 15, 215);
  aux7_ca = 1;
}
if (aux7_ca == 1) {
  aux7_ca = 0;
} }

if (BoCal == 1 && posicion_calibracion == 2){
if (aux_izde_ca >= 419) {
  aux_izde_ca = 339;
}
if (aux7_ca == 0)                              {
  aux_izde_ca = aux_izde_ca + 16;
  if (aux_izde_ca == 387) {
    aux_izde_ca = aux_izde_ca + 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 235, 435, 235);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 235, aux_izde_ca + 15, 235);
  aux7_ca = 1;
}
if (aux7_ca == 1) {
  aux7_ca = 0;
} }

if (BoCal == 1 && posicion_calibracion == 3){
if (aux_izde_ca >= 419) {
  aux_izde_ca = 339;
}
if (aux7_ca == 0)                              {
  aux_izde_ca = aux_izde_ca + 16;
  if (aux_izde_ca == 387) {
    aux_izde_ca = aux_izde_ca + 16;
  }
  myGLCD.setColor(VGA_WHITE);
  myGLCD.drawLine(355, 255, 435, 255);
  myGLCD.setColor(VGA_RED);
  myGLCD.drawLine(aux_izde_ca, 255, aux_izde_ca + 15, 255);
  aux7_ca = 1;
}
if (aux7_ca == 1) {
  aux7_ca = 0;
} }
```

```
if (BoCal == 1 && posicion_calibracion == 4){
if (aux_izde_ca >= 419) {
 aux_izde_ca = 339;
}
if (aux7_ca == 0)                    {
 aux_izde_ca = aux_izde_ca + 16;
 if (aux_izde_ca == 387) {
  aux_izde_ca = aux_izde_ca + 16;
 }
 myGLCD.setColor(VGA_WHITE);
 myGLCD.drawLine(355, 275, 435, 275);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawLine(aux_izde_ca, 275, aux_izde_ca + 15, 275);
 aux7_ca = 1;
}
if (aux7_ca == 1) {
 aux7_ca = 0;
} }

if (BoCal == 1 && posicion_calibracion == 6){
if (aux_izde_ca >= 419) {
 aux_izde_ca = 339;
}
if (aux7_ca == 0)                    {
 aux_izde_ca = aux_izde_ca + 16;
 if (aux_izde_ca == 387) {
  aux_izde_ca = aux_izde_ca + 16;
 }
 myGLCD.setColor(VGA_WHITE);
 myGLCD.drawLine(355, 355, 435, 355);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawLine(aux_izde_ca, 355, aux_izde_ca + 15, 355);
 aux7_ca = 1;
}
if (aux7_ca == 1) {
 aux7_ca = 0;
} }

if (BoCal == 1 && posicion_calibracion == 7){
if (aux_izde_ca >= 419) {
 aux_izde_ca = 339;
}
if (aux7_ca == 0)                    {
 aux_izde_ca = aux_izde_ca + 16;
 if (aux_izde_ca == 387) {
  aux_izde_ca = aux_izde_ca + 16;
 }
 myGLCD.setColor(VGA_WHITE);
 myGLCD.drawLine(355, 375, 435, 375);
 myGLCD.setColor(VGA_RED);
 myGLCD.drawLine(aux_izde_ca, 375, aux_izde_ca + 15, 375);
 aux7_ca = 1;
```

```
      }
      if (aux7_ca == 1) {
       aux7_ca = 0;
      } }

     }

    }

    if ((x >= 185) && (x <= 295)) //-----------------------------------Button: UP y DOWN
    {
     if ((y >= 574.29) && (y <= 615.72)) //-------------------------------Button: UP
     {
      waitForIt(185, 574.29, 295, 615.72);
      aux_UP_DOWN = 1;
      //-----------CONTRASTE----------
      //-----------------------------
      if (BoPr == 1)
      {
       if (contraste >= 9) {
        contraste = -1;
       }

       if (aux_contraste1 == 0)   {
        contraste++;
        r = 255 - (contraste * 25);
        g = 255 - (contraste * 25);
        h = 255 - (contraste * 25);
        aux_contraste1 = 1;
        setup();
       }
       if (aux_contraste1 == 1)  {
        aux_contraste1 = 0;
       }
      }
      //-----------CONTRASTE----------
      //-----------------------------

      if ((generacion_volt <= RangoSupVolt)&& (BoPr == 15) && (MeSu == 2 || MeSu == 0))
{
       if (aux_izde == 208 && (generacion_volt <= (RangoSupVolt)))  {
        generacion_volt=generacion_volt+0.01;
       }
       if (aux_izde == 176 && (generacion_volt < (RangoSupVolt-0.09)))  {
        generacion_volt=generacion_volt+0.1;
       }
       if (aux_izde == 112 && generacion_volt <= (RangoSupVolt-1))  {
        generacion_volt=generacion_volt+1;
       }
       if (aux_izde == 80 && generacion_volt <= 0)  {
        generacion_volt=generacion_volt+10;
       }
```

```
        }

        if ((generacion_corr <= RangoSupCorr)&& (BoPr == 8) && (MeSu == 2 || MeSu == 0)) {
          if (aux_izde == 208 && (generacion_corr <= (RangoSupCorr)))  {
           generacion_corr=generacion_corr+0.01;
          }
          if (aux_izde == 176 && (generacion_corr < (RangoSupCorr-0.09)))  {
           generacion_corr=generacion_corr+0.1;
          }
          if (aux_izde == 112 && generacion_corr <= (RangoSupCorr-1))  {
           generacion_corr=generacion_corr+1;
          }
          if (aux_izde == 80 && generacion_corr <= 13.999)  {
           generacion_corr=generacion_corr+10;
          }
        }

        if ((generacion_RTD <= RangoSupRTD)&& (BoPr == 20) && (MeSu == 2 || MeSu == 0))
{
          if (aux_izde == 176 && (generacion_RTD <= (RangoSupRTD+1)))  {
           generacion_RTD=generacion_RTD+1;
          }
          if (aux_izde == 144 && (generacion_RTD < (RangoSupRTD-9)))  {
           generacion_RTD=generacion_RTD+10;
          }
          if (aux_izde == 112 && generacion_RTD <= (RangoSupRTD-100))  {
           generacion_RTD=generacion_RTD+100;
          }
          if (aux_izde == 80 && generacion_RTD <= 999)  {
           generacion_RTD=generacion_RTD+1000;
          }
        }

        if ((generacion_TCJ <= RangoSupTCJ)&& (BoPr == 16) && (MeSu == 2 || MeSu ==
0)&& (aux_TCg==0)) {
          if (aux_izde == 176 )  { // && (generacion_RTD < (RangoSupRTD+1))
           generacion_TCJ=generacion_TCJ+1;
          }
          if (aux_izde == 144 )  { //&& (generacion_RTD < (RangoSupRTD-9))
           generacion_TCJ=generacion_TCJ+10;
          }
          if (aux_izde == 112 )  { //&& generacion_RTD <= (RangoSupRTD-100)
           generacion_TCJ=generacion_TCJ+100;
          }
          if (aux_izde == 80 )  {  //&& generacion_RTD <= 900
           generacion_TCJ=generacion_TCJ+1000;
          }
        }

        if ((generacion_TCK <= RangoSupTCK)&& (BoPr == 16) && (MeSu == 2 || MeSu == 0)
&& (aux_TCg==1)) {
          if (aux_izde == 176 )  { // && (generacion_RTD < (RangoSupRTD+1))
```

```
    generacion_TCK=generacion_TCK+1;
   }
   if (aux_izde == 144 )  { //&& (generacion_RTD < (RangoSupRTD-9))
    generacion_TCK=generacion_TCK+10;
   }
   if (aux_izde == 112 )  { //&& generacion_RTD <= (RangoSupRTD-100)
    generacion_TCK=generacion_TCK+100;
   }
   if (aux_izde == 80 )  {  //&& generacion_RTD <= 900
    generacion_TCK=generacion_TCK+1000;
   }
  }

  if ((generacion_res <= RangoSupRes)&& (BoPr == 19) && (MeSu == 2 || MeSu == 0)) {
   if (aux_izde == 176 && (generacion_res <= (RangoSupRes)))  { //
    generacion_res=generacion_res+1;
   }
   if (aux_izde == 144 && (generacion_res < (RangoSupRes-9)))  { //
    generacion_res=generacion_res+10;
   }
   if (aux_izde == 112 && generacion_res <= (RangoSupRes-100))  { //
    generacion_res=generacion_res+100;
   }
   if (aux_izde == 80 && generacion_res <= 200)  {  //
    generacion_res=generacion_res+1000;
   }
  }

  if (BoPr == 8 || BoPr == 15){
  //----PRIMER DIGITO-----
  if ((MeSu == 2 || MeSu == 0) && aux_izde == 208)
  {
   primer_digito_ascend(limite_uni);
   //aux_conteo_dato_uni++;
  }
  //----SEGUNDO DIGITO-----
  if (((MeSu == 2 || MeSu == 0) && aux_izde == 176) )// || (aux_conteo_dato_uni ==
10)
  {
   segundo_digito_ascend(limite_dec);
   //aux_conteo_dato_dec++;
  }
  }

  if (BoPr == 19 || BoPr == 20 || BoPr == 16){
  //----PRIMER DIGITO-----
  if ((MeSu == 2 || MeSu == 0) && aux_izde == 176)
  {
   primer_digito_ascend(limite_uni);
  }
  //----SEGUNDO DIGITO-----
  if (((MeSu == 2 || MeSu == 0) && aux_izde == 144) ) //|| (num_uni == 9)
```

```
        {
          segundo_digito_ascend(limite_dec);
        }
        }

        //----TERCER DIGITO-----
        if (((MeSu == 2 || MeSu == 0) && aux_izde == 112) )// || (num_dec == 9)
        {
          tercer_digito_ascend(limite_cen);
          //aux_conteo_dato_cen++;
        }
        //----CUARTO DIGITO-----
        if (((MeSu == 2 || MeSu == 0) && aux_izde == 80) )// || (num_cen == 9)
        {
          cuarto_digito_ascend(limite_mil);
        }


      }


      if ((y >= 677.15) && (y <= 718.58)) //------------------------------------------------------------
Button: DOWN
      {
        waitForIt(185, 677.15, 295, 718.58);
        aux_UP_DOWN = 2;
        //-----------CONTRASTE----------
        //------------------------------
        if (BoPr == 1)
        {
          if (contraste <= 0) {
            contraste = 10;
          }
          if (aux_contraste2 == 0)    {
            contraste--;
            r = 255 - (contraste * 25);
            g = 255 - (contraste * 25);
            h = 255 - (contraste * 25);
            aux_contraste2 = 1;
            setup();
          }
          if (aux_contraste2 == 1)    {
            aux_contraste2 = 0;
          }
        }
        //-----------CONTRASTE----------
        //------------------------------

        if ( (generacion_volt >= RangoInfVolt) && (BoPr == 15) && (MeSu == 2 || MeSu == 0))
{
          if (aux_izde == 208 && generacion_volt >= (RangoInfVolt+0.01))  {
            generacion_volt=generacion_volt-0.01;
```

```
        }
        if (aux_izde == 176 && generacion_volt >= (RangoInfVolt+0.1))  {
          generacion_volt=generacion_volt-0.1;
        }
        if (aux_izde == 112 && generacion_volt >= (RangoInfVolt+1))  {
          generacion_volt=generacion_volt-1;
        }
        if (aux_izde == 80 && generacion_volt >= 10)  {
          generacion_volt=generacion_volt-1;
        }
       }

       if ( (generacion_corr >= RangoInfCorr) && (BoPr == 8) && (MeSu == 2 || MeSu == 0)) {
        if (aux_izde == 208 && generacion_corr >= (RangoInfCorr+0.01))  {
          generacion_corr=generacion_corr-0.01;
        }
        if (aux_izde == 176 && generacion_corr >= (RangoInfCorr+0.1))  {
          generacion_corr=generacion_corr-0.1;
        }
        if (aux_izde == 112 && generacion_corr >= (RangoInfCorr+1))  {
          generacion_corr=generacion_corr-1;
        }
        if (aux_izde == 80 && generacion_corr >= 10)  {
          generacion_corr=generacion_corr-10;
        }
       }


       if ( (generacion_RTD >=RangoInfRTD) && (BoPr == 20) && (MeSu == 2 || MeSu == 0))
{
        if (aux_izde == 176 && generacion_RTD >= (RangoInfRTD+1))  {
          generacion_RTD=generacion_RTD-1;
        }
        if (aux_izde == 144 && generacion_RTD >= (RangoInfRTD+10))  {
          generacion_RTD=generacion_RTD-10;
        }
        if (aux_izde == 112 && generacion_corr >= (RangoInfRTD+100))  {
          generacion_RTD=generacion_RTD-100;
        }
        if (aux_izde == 80 && generacion_RTD >= 1000)  {
          generacion_RTD=generacion_RTD-1000;
        }
       }

       if ( (generacion_TCJ >=RangoInfTCJ) && (BoPr == 16) && (MeSu == 2 || MeSu == 0)&&
(aux_TCg==0)) {
        if (aux_izde == 176 )  { // && generacion_RTD > RangoInfRTD
          generacion_TCJ=generacion_TCJ-1;
        }
        if (aux_izde == 144 )  { //&& generacion_RTD >= (RangoInfRTD+9)
          generacion_TCJ=generacion_TCJ-10;
        }
```

```
  if (aux_izde == 112 )  { // && generacion_corr >= (RangoInfRTD+99)
    generacion_TCJ=generacion_TCJ-100;
  }
  if (aux_izde == 80 )  {  //&& generacion_RTD >= 900
    generacion_TCJ=generacion_TCJ-1000;
  }
 }

 if ( (generacion_TCK >=RangoInfTCK) && (BoPr == 16) && (MeSu == 2 || MeSu ==
0)&& (aux_TCg==1)) {
  if (aux_izde == 176 )  { // && generacion_RTD > RangoInfRTD
    generacion_TCK=generacion_TCK-1;
  }
  if (aux_izde == 144 )  { //&& generacion_RTD >= (RangoInfRTD+9)
    generacion_TCK=generacion_TCK-10;
  }
  if (aux_izde == 112 )  { // && generacion_corr >= (RangoInfRTD+99)
    generacion_TCK=generacion_TCK-100;
  }
  if (aux_izde == 80 )  {  //&& generacion_RTD >= 900
    generacion_TCK=generacion_TCK-1000;
  }
 }

 if ( (generacion_res >=RangoInfRes) && (BoPr == 19) && (MeSu == 2 || MeSu == 0)) {
  if (aux_izde == 176 && generacion_res >= (RangoInfRes+1))  { //
    generacion_res=generacion_res-1;
  }
  if (aux_izde == 144 && generacion_res >= (RangoInfRes+10))  { //
    generacion_res=generacion_res-10;
  }
  if (aux_izde == 112 && generacion_res >= (RangoInfRes+100))  { //
    generacion_res=generacion_res-100;
  }
  if (aux_izde == 80 && generacion_res >= 1000)  {  //
    generacion_res=generacion_res-1000;
  }
 }




 limites();
 if (BoPr == 8 || BoPr == 15){
 //----PRIMER DIGITO-----
 if ((MeSu == 2 || MeSu == 0) && aux_izde == 208)
 {
  primer_digito_descen(limite_uni + 1);
  //aux_conteo_dato_uni--;
 }
 //----SEGUNDO DIGITO-----
 if (((MeSu == 2 || MeSu == 0) && aux_izde == 176) )// || (num_uni == 0)
```

```
   {
    segundo_digito_descen(limite_dec + 1);
    //aux_conteo_dato_dec--;
   }
   }

   if (BoPr == 19 || BoPr == 20 || BoPr == 16){
   //----PRIMER DIGITO-----
   if ((MeSu == 2 || MeSu == 0) && aux_izde == 176)
   {
    primer_digito_descen(limite_uni + 1);
   }
   //----SEGUNDO DIGITO-----
   if ((MeSu == 2 || MeSu == 0) && aux_izde == 144)
   {
    segundo_digito_descen(limite_dec + 1);
   }
   }

   //----TERCER DIGITO-----
   if (((MeSu == 2 || MeSu == 0) && aux_izde == 112) )// || (num_dec == 0)
   {
    tercer_digito_descen(limite_cen + 1);
    //aux_conteo_dato_cen--;
   }
   //----CUARTO DIGITO-----
   if (((MeSu == 2 || MeSu == 0) && aux_izde == 80) )// || (num_cen == 0)
   {
    cuarto_digito_descen(limite_mil + 1);
   }

  }

}
myGLCD.setFont(SmallFont);
if ((x >= 125) && (x <= 355)) //---------------------------------------Button: CALIBRATION
{
  if ((y >= 728.58) && (y <= 770.01))
  {
    BoPr = 24;
    BoCal = 1;
     posicion_calibracion = 1;
     waitForIt(125, 728.58, 355, 770.01);
     aux_medicion = aux_medi_gene;
     myGLCD.setFont(BigFont);
     myGLCD.setColor(VGA_RED);
     myGLCD.print("SOURCE ", 7, 302);
     myGLCD.print("MEASURE", 7, 182);
     waitForIt1(5, 300, 475, 400, 5, 180, 475, 280, 1);
     myGLCD.setColor(VGA_BLACK);
     myGLCD.print("SOURCE ", 7, 302);
     myGLCD.print("MEASURE", 7, 182);
```

```
          myGLCD.setColor(VGA_RED);
          MeSu = 0;

        myGLCD.setFont(Grotesk32x64);
        myGLCD.print("        ", 50, 210);
        myGLCD.print("        ", 50, 330);
        myGLCD.setFont(BigFont);
        myGLCD.setColor(VGA_BLACK);
        myGLCD.print("      Valor de   0%:", 15, 200);

sprintf(dato_impresion,"%2d.%02dmA",corriente_calibracion100,corriente_calibracion0);
        myGLCD.print(dato_impresion, 355, 200);
        myGLCD.print("      Valor de 100%:", 15, 220);

sprintf(dato_impresion,"%2d.%02dmA",corriente_calibracion0,corriente_calibracion100);
        myGLCD.print(dato_impresion, 355, 220);
        myGLCD.print("         Tolerancia:", 15, 240);
        sprintf(dato_impresion,"%2d%",tolerancia_calibracion);
        myGLCD.print(dato_impresion, 355, 240);
        myGLCD.print("            Demora:", 15, 260);
        myGLCD.print("     Tipo de sensor:", 15, 320);
        myGLCD.print("      Valor de   0%:", 15, 340);
        myGLCD.print("      Valor de 100%:", 15, 360);
        myGLCD.print("Estrategia de prueba:", 15, 380);
        myGLCD.drawLine(371, 392, 371, 380);
        myGLCD.drawLine(363, 385, 371, 380);
        myGLCD.drawLine(379, 385, 371, 380);

        myGLCD.setColor(VGA_RED);

        myGLCD.setFont(SmallFont);
        myGLCD.print("          ", 390, 168);
        myGLCD.print("CALIBRATION", 390, 168);
      }
     }


    }

  }
}
```

ANEXO B

PROGRAMACIÓN ARDUINO MEGA

```
/*********************************************************************
 * LIBRERIAS PARA EL MANEJO DE LA COMUNICACION I2C, SPI Y SERIAL
*********************************************************************/
#include <Wire.h>
#include <SPI.h>

/*********************************************************************
 * VARIABLES GLOBALES
*********************************************************************/
//ASIGNACION DE PINES DIGITALES ARDUINO DUE
const int CS1 = 53;// ad5231 fijo
const int CS2 = 42;// ad5231 variable1
const int CS3 = 44;// ad5231 variable2
const int CS4 = 47;// ad5322 voltaje y corriente
const int CS5 = 46;// ad5231 variable3
//VARIABLES PRINCIPALES
int i,d,aux_d; // varaibles a escribir en cada integrado de resistencia
char a[6];
float w, z;
float aux_res1=0,aux_res2=0,aux_res3=0;
int b=1023;
int c=1023;
/*********************************************************************
 * FUNCION PRINCIPAL (EJECUCION POR UNA VEZ)
*********************************************************************/
void setup() {
  Wire.begin(8);
  Serial.begin(9600);
  SPI.begin();
  pinMode(CS1, OUTPUT);
  digitalWrite(CS1, HIGH);
  pinMode(CS2, OUTPUT);
  digitalWrite(CS2, HIGH);
  pinMode(CS3, OUTPUT);
  digitalWrite(CS3, HIGH);
  pinMode(CS4, OUTPUT);
  digitalWrite(CS4, HIGH);
  pinMode(CS5, OUTPUT);
  digitalWrite(CS5, HIGH);
  i = 0;
  Wire.onReceive(recibir);
  reset_CI1();
  delay(100);
  reset();
  delay(100);
}

/*********************************************************************
 * FUNCION PRINCIPAL (EJECUCION REPETITIVA)
*********************************************************************/
void loop() {
delay(100);
```

```
}
/*********************************************************************
 * FUNCION: PARA ESCRIBIR VALORES EN EL INTEGRADO1 AD5231
*********************************************************************/
void Potenciometro1(int direccion, int dato) {

  byte comando = 0xB0;
  comando += direccion;
  byte b1 = (dato >> 8);
  byte b0 = (dato & 0xFF);
  digitalWrite(CS2, HIGH);
  delay(10);
  digitalWrite(CS3, HIGH);
  delay(10);
  digitalWrite(CS5, HIGH);
  delay(10);
  digitalWrite(CS1, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS1, HIGH);
}
/*********************************************************************
 * FUNCION: PARA ESCRIBIR VALORES EN EL INTEGRADO2 AD5231
*********************************************************************/
void Potenciometro2(int direccion, int dato) {

  byte comando = 0xB0;
  comando += direccion;
  byte b1 = (dato >> 8);
  byte b0 = (dato & 0xFF);
  digitalWrite(CS3, HIGH);
  delay(10);
  digitalWrite(CS1, HIGH);
  delay(10);
  digitalWrite(CS5, HIGH);
  delay(10);
  digitalWrite(CS2, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS2, HIGH);
}
/*********************************************************************
 * FUNCION: PARA ESCRIBIR VALORES EN EL INTEGRADO3 AD5231
*********************************************************************/
void Potenciometro3(int direccion, int dato) {

  byte comando = 0xB0;
  comando += direccion;
  byte b1 = (dato >> 8);
  byte b0 = (dato & 0xFF);
```

```
 digitalWrite(CS1, HIGH);
 delay(10);
 digitalWrite(CS2, HIGH);
 delay(10);
 digitalWrite(CS5, HIGH);
 delay(10);
 digitalWrite(CS3, LOW);
 SPI.transfer(comando);
 SPI.transfer(b1);
 SPI.transfer(b0);
 digitalWrite(CS3, HIGH);
}
/*****************************************************************
 * FUNCION: PARA ESCRIBIR VALORES EN EL INTEGRADO4 AD5231
*****************************************************************/
void Potenciometro4(int direccion, int dato) {

 byte comando = 0xB0;
 comando += direccion;
 byte b1 = (dato >> 8);
 byte b0 = (dato & 0xFF);
 digitalWrite(CS1, HIGH);
 delay(10);
 digitalWrite(CS2, HIGH);
 delay(10);
 digitalWrite(CS3, HIGH);
 delay(10);
 digitalWrite(CS5, LOW);
 SPI.transfer(comando);
 SPI.transfer(b1);
 SPI.transfer(b0);
 digitalWrite(CS5, HIGH);
}
/*****************************************************************
 * FUNCION: PARA RESETEAR EL INTEGRADO AD5422
*****************************************************************/
void reset_CI1() {

 byte b1 = 0x00;
 byte b0 = 0x01;
 byte comando = 0x56;
 digitalWrite(CS4, LOW);
 SPI.transfer(comando);
 SPI.transfer(b1);
 SPI.transfer(b0);
 digitalWrite(CS4, HIGH);
}
/*****************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE VOLTAJE RANGO 0-5V INTEGRADO AD5422
*****************************************************************
*/
```

```
void configurarV1() {

  byte comando = 0x55;
  byte b1 = 0x98;
  byte b0 = 0xA0;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*****************************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE VOLTAJE RANGO 0-10V INTEGRADO
AD5422
*****************************************************************************
*/
void configurarV2() {

  byte comando = 0x55;
  byte b1 = 0x1A;
  byte b0 = 0xA1;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*****************************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE VOLTAJE RANGO -5 A 5V INTEGRADO
AD5422
*****************************************************************************
*/
void configurarV3() {

  byte comando = 0x55;
  byte b1 = 0x1A;
  byte b0 = 0xA2;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*****************************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE CORRIENTE RANGO 4-20mA INTEGRADO
AD5422
*****************************************************************************
*/
void configurarI1() {
```

```
  byte comando = 0x55;
  byte b1 = 0x1A;
  byte b0 = 0xAD;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*************************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE CORRIENTE RANGO 0-20mA INTEGRADO
AD5422
*************************************************************************
*/
void configurarI2() {

  byte comando = 0x55;
  byte b1 = 0x1A;
  byte b0 = 0xAE;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*************************************************************************
**
 * FUNCION: PARA CONFIGUARAR LA SALIDA DE CORRIENTE RANGO 0-24mA INTEGRADO
AD5422
*************************************************************************
*/
void configurarI3() {

  byte comando = 0x55;
  byte b1 = 0x1A;
  byte b0 = 0xAF;
  digitalWrite(CS4, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS4, HIGH);
}
/*************************************************************************
**
 * FUNCION: PARA LA SALIDA DEL INTEGRADO AD5422
*************************************************************************
*/
void Salida_CI1(int dato) {

  byte comando = 0x01;
```

```
 byte b1 = byte(highByte(dato));
 byte b0 = byte(lowByte(dato));
 digitalWrite(CS4, LOW);
 SPI.transfer(comando);
 SPI.transfer(b1);
 SPI.transfer(b0);
 digitalWrite(CS4, HIGH);
}
/*****************************************************************************
**
 * FUNCION: COMUNICACION I2C ARDUINO MEGA Y DUE
*****************************************************************************
*/
void recibir(int h) {
 while (1 < Wire.available()) { // loop through all but the last
  char c = Wire.read(); // receive byte as a character
  a[i]= c;
  i++;
 }
 i=0;
 String y = String(a);
 float w = y.toFloat();
 d = Wire.read();
 switch (d) {
  // GENERACION DE VOLTAJE 0-5V
  case 0:
   if (w <= 0){
    z=0;
   }
   else {
    z = ((w / 5.0) * 65535)-75;
   }

   reset_CI1();
   delay(500);
   configurarV1();
   Salida_CI1(round(z));
   Serial.println("v1 escrito");
   Serial.println(z);
   break;
   // GENERACION DE VOLTAJE 0-10V
  case 1:
   z = ((w / 10.0) * 65535)-33;
   reset_CI1();
   delay(500);
   configurarV2();
   Salida_CI1(round(z));
   Serial.println("v2 escrito");
   Serial.println(z);
   break;
 case 11:
 // GENERACION DE VOLTAJE -10 A 10V
```

```
  z = (w * (20 / 65535)) + (200 / 65535);
  reset_CI1();
  delay(500);
  configurarV3();
  Salida_CI1(round(z));
  Serial.println("v3 escrito");
  break;
// GENERACION DE CORRIENTE 4-20mA
case 2:
  //z = ((w*(65535/16)) - ((65535/16)*4))+45;
  if ( w > 23.9){
    z = ((23.9 / 24.0) * 65535)+260;
  }else if (w >= 23.001 && w <= 23.9){
    z = ((w / 24.0) * 65535)+260;
  } else if (w >= 20 && w <= 23) {
    z = ((w / 24.0) * 65535) + 252;
  }else if (w >= 18 && w < 20) {
    z = ((w / 24.0) * 65535) + 244;
  }else if (w >= 17 && w < 18) {
    z = ((w / 24.0) * 65535) + 240;
  }else if (w >= 16 && w < 17) {
    z = ((w / 24.0) * 65535) + 232;
  }else if (w >= 14 && w < 16) {
    z = ((w / 24.0) * 65535) + 228;
  }else if (w >= 13 && w < 14) {
    z = ((w / 24.0) * 65535) + 224;
  }else if (w >= 12 && w < 13) {
    z = ((w / 24.0) * 65535) + 220;
  }else if (w >= 10 && w < 12) {
    z = ((w / 24.0) * 65535) + 208;
  }else if (w >= 8 && w < 10) {
    z = ((w / 24.0) * 65535) + 192;
  }else if (w >= 7 && w < 8) {
    z = ((w / 24.0) * 65535) + 180;
  }else if (w >= 6 && w < 7) {
    z = ((w / 24.0) * 65535) + 172;
  }else if (w >= 5 && w < 6) {
    z = ((w / 24.0) * 65535) + 160;
  }else if (w >= 4 && w < 5) {
    z = ((w / 24.0) * 65535) + 152;
  }else if (w >= 3 && w < 4) {
    z = ((w / 24.0) * 65535) + 136;
  }else if (w >= 2 && w < 3) {
    z = ((w / 24.0) * 65535) + 116;
  }else if (w >= 1.17 && w < 2) {
    z = ((w / 24.0) * 65535) + 110;
  }else if (w >= 0.97 && w < 1.17) {
    z = ((w / 24.0) * 65535) + 104;
  }else if (w >= 0.8 && w < 1.17) {
    z = ((w / 24.0) * 65535) + 82;
  }else if (w >= 0.6 && w < 0.8) {
    z = ((w / 24.0) * 65535) + 70;
```

```
  }else if (w >= 0.4 && w < 0.6) {
    z = ((w / 24.0) * 65535) + 70;
  }else if (w >= 0.2 && w < 0.4) {
    z = ((w / 24.0) * 65535) + 70;
  }else if (w < 0.2) {
    z = ((w / 24.0) * 65535) + 48; //+36   0.02 0.018
  }
  reset_CI1();
  delay(500);
  configurarI3();
  Salida_CI1(round(z));
  Serial.println("i1 escrito");
  Serial.println(z);
  break;
  // GENERACION DE CORIENTE 0-24mA
 case 3:
  z = (w / 24.0) * 65535;
  reset_CI1();
  delay(500);
  configurarI2();
  Salida_CI1(round(z));
  Serial.println("i2 escrito");
  break;
 // GENERACION DE RESISTENCIA
 case 4:
  switch (round(w)) {
    case 30:
      Potenciometro1(0, 1023);
      Potenciometro2(0, 1023);
      Potenciometro3(0, 1009);
      break;
    case 31:
      Potenciometro1(0, 1023);
      Potenciometro2(0, 1023);
      Potenciometro3(0, 1003);
      break;
    case 32:
      Potenciometro1(0, 1023);
      Potenciometro2(0, 1023);
      Potenciometro3(0, 994);
      break;
    case 33:
      Potenciometro1(0, 1023);
      Potenciometro2(0, 1023);
      Potenciometro3(0, 978);
      break;
    case 34:
      Potenciometro1(0, 1023);
      Potenciometro2(0, 1023);
      Potenciometro3(0, 938);
    break;
    case 35:
```

```
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1023);
    Potenciometro3(0, 752);
   break;
   case 36:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1022);
    Potenciometro3(0, 931);
   break;
   case 37:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1022);
    Potenciometro3(0, 735);
   break;
   case 38:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1021);
    Potenciometro3(0, 902);
   break;
   case 39:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1021);
    Potenciometro3(0, 492);
  break;
   case 40:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 830);
break;
   case 41:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 887);
 break;
   case 42:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 466);
   break;
   case 43:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1018);
    Potenciometro3(0, 700);
   break;
   case 44:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1017);
    Potenciometro3(0, 780);
    break;
   case 45:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1016);
```

```
   Potenciometro3(0, 793);
  break;
   case 46:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1015);
   Potenciometro3(0, 776);
  break;
   case 47:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1014);
   Potenciometro3(0, 744);
  break;
   case 48:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1013);
   Potenciometro3(0, 642);
break;
   case 49:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1012);
   Potenciometro3(0, 474);
  break;
   case 50:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1010);
   Potenciometro3(0, 708);
  break;
   case 51:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1009);
   Potenciometro3(0, 435);
  break;
   case 52:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1007);
   Potenciometro3(0, 559);
break;
   case 53:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1005);
   Potenciometro3(0, 525);
  break;
   case 54:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1003);
   Potenciometro3(0, 422);
   break;
   case 55:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1000);
   Potenciometro3(0, 542);
   break;
```

```
 case 56:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 998);
 Potenciometro3(0, 138);
 break;
 case 57:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 994);
 Potenciometro3(0, 401);
 break;
 case 58:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 991);
 Potenciometro3(0, 142);
break;
 case 59:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 986);
 Potenciometro3(0, 282);
 break;
 case 60:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 61:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 973);
 Potenciometro3(0, 333);
 break;
 case 62:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 962);
 Potenciometro3(0, 475);
 break;
 case 63:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 952);
 Potenciometro3(0, 325);
 break;
 case 64:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 934);
 Potenciometro3(0, 303);
 break;
 case 65:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 907);
 Potenciometro3(0, 323);
 break;
 case 66:
 Potenciometro1(0, 1023);
```

```
Potenciometro2(0, 865);
Potenciometro3(0, 320);
break;
case 67:
Potenciometro1(0, 1023);
Potenciometro2(0, 777);
Potenciometro3(0, 360);
break;
case 68:
Potenciometro1(0, 1023);
Potenciometro2(0, 557);
Potenciometro3(0, 340);
break;
case 69:
Potenciometro1(0, 1022);
Potenciometro2(0, 957);
Potenciometro3(0, 315);
break;
case 70:
Potenciometro1(0, 1022);
Potenciometro2(0, 946);
Potenciometro3(0, 320);
break;
case 71:
Potenciometro1(0, 1022);
Potenciometro2(0, 931);
Potenciometro3(0, 342);
break;
case 72:
Potenciometro1(0, 1022);
Potenciometro2(0, 911);
Potenciometro3(0, 360);
break;
case 73:
Potenciometro1(0, 1022);
Potenciometro2(0, 881);
Potenciometro3(0, 350);
break;
case 74:
Potenciometro1(0, 1022);
Potenciometro2(0, 837);
Potenciometro3(0, 310);
break;
case 75:
Potenciometro1(0, 1022);
Potenciometro2(0, 749);
Potenciometro3(0, 310);
break;
case 76:
Potenciometro1(0, 1022);
Potenciometro2(0, 535);
Potenciometro3(0, 320);
```

```
break;
case 77:
Potenciometro1(0, 1021);
Potenciometro2(0, 929);
Potenciometro3(0, 683);
break;
case 78:
Potenciometro1(0, 1021);
Potenciometro2(0, 913);
Potenciometro3(0, 678);
break;
case 79:
Potenciometro1(0, 1021);
Potenciometro2(0, 890);
Potenciometro3(0, 675);
break;
case 80:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 678);
break;
case 81:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 679);
break;
case 82:
Potenciometro1(0, 1021);
Potenciometro2(0, 833);
Potenciometro3(0, 579);
break;
case 83:
Potenciometro1(0, 1021);
Potenciometro2(0, 793);
Potenciometro3(0, 479);
break;
case 84:
Potenciometro1(0, 1021);
Potenciometro2(0, 722);
Potenciometro3(0, 377);
break;
case 85:
Potenciometro1(0, 1021);
Potenciometro2(0, 575);
Potenciometro3(0, 307);
break;
case 86:
Potenciometro1(0, 1021);
Potenciometro2(0, 189);
Potenciometro3(0, 150);
break;
case 87:
```

```
Potenciometro1(0, 1020);
Potenciometro2(0, 821);
Potenciometro3(0, 836);
break;
case 88:
Potenciometro1(0, 1020);
Potenciometro2(0, 770);
Potenciometro3(0, 832);
break;
case 89:
Potenciometro1(0, 1020);
Potenciometro2(0, 721);
Potenciometro3(0, 804);
break;
case 90:
Potenciometro1(0, 1020);
Potenciometro2(0, 671);
Potenciometro3(0, 767);
break;
case 91:
Potenciometro1(0, 1020);
Potenciometro2(0, 609);
Potenciometro3(0, 712);
break;
case 92:
Potenciometro1(0, 1020);
Potenciometro2(0, 528);
Potenciometro3(0, 621);
break;
case 93:
Potenciometro1(0, 1020);
Potenciometro2(0, 408);
Potenciometro3(0, 471);
break;
case 94:
Potenciometro1(0, 1020);
Potenciometro2(0, 180);
Potenciometro3(0, 235);
break;
case 95:
Potenciometro1(0, 1019);
Potenciometro2(0, 805);
Potenciometro3(0, 795);
break;
case 96:
Potenciometro1(0, 1019);
Potenciometro2(0, 746);
Potenciometro3(0, 794);
break;
case 97:
Potenciometro1(0, 1019);
Potenciometro2(0, 690);
```

```
Potenciometro3(0, 772);
break;
case 98:
Potenciometro1(0, 1019);
Potenciometro2(0, 650);
Potenciometro3(0, 724);
break;
case 99:
Potenciometro1(0, 1019);
Potenciometro2(0, 590);
Potenciometro3(0, 666);
break;
/**************************1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*1*/
case 100:
Potenciometro1(0, 1019);
Potenciometro2(0, 503);
Potenciometro3(0, 581);
break;
case 101:
Potenciometro1(0, 1019);
Potenciometro2(0, 370);
Potenciometro3(0, 461);
break;
case 102:
Potenciometro1(0, 1019);
Potenciometro2(0, 154);
Potenciometro3(0, 223);
break;
case 103:
Potenciometro1(0, 1018);
Potenciometro2(0, 789);
Potenciometro3(0, 750);
break;
case 104:
Potenciometro1(0, 1018);
Potenciometro2(0, 730);
Potenciometro3(0, 749);
break;
case 105:
Potenciometro1(0, 1018);
Potenciometro2(0, 680);
Potenciometro3(0, 720);
break;
case 106:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 107:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
```

```
break;
case 108:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 109:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 110:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 111:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 112:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 113:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 114:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 115:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 116:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 117:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 118:
```

```
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 119:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 120:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 121:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 122:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 123:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 124:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 125:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
case 126:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 1022);
 Potenciometro3(0, 931);
break;
case 127:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 1022);
 Potenciometro3(0, 735);
break;
case 128:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 1021);
```

```
    Potenciometro3(0, 902);
   break;
   case 129:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1021);
    Potenciometro3(0, 492);
  break;
   case 130:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 830);
break;
   case 131:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 887);
 break;
   case 132:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 466);
    break;
   case 133:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1018);
    Potenciometro3(0, 700);
    break;
   case 134:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1017);
    Potenciometro3(0, 780);
    break;
   case 135:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1016);
    Potenciometro3(0, 793);
   break;
    case 136:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1015);
    Potenciometro3(0, 776);
    break;
    case 137:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1014);
    Potenciometro3(0, 744);
   break;
    case 138:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1013);
    Potenciometro3(0, 642);
 break;
```

```
  case 139:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1012);
  Potenciometro3(0, 474);
 break;
  case 140:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1010);
  Potenciometro3(0, 708);
 break;
  case 141:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1009);
  Potenciometro3(0, 435);
 break;
  case 142:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1007);
  Potenciometro3(0, 559);
break;
  case 143:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1005);
  Potenciometro3(0, 525);
 break;
  case 144:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1003);
  Potenciometro3(0, 422);
  break;
  case 145:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1000);
  Potenciometro3(0, 542);
  break;
  case 146:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 998);
  Potenciometro3(0, 138);
  break;
  case 147:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 994);
  Potenciometro3(0, 401);
  break;
  case 148:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 991);
  Potenciometro3(0, 142);
 break;
  case 149:
  Potenciometro1(0, 1023);
```

```
 Potenciometro2(0, 986);
 Potenciometro3(0, 282);
break;
 case 150:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 151:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 973);
 Potenciometro3(0, 333);
 break;
 case 152:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 962);
 Potenciometro3(0, 475);
 break;
 case 153:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 952);
 Potenciometro3(0, 325);
 break;
 case 154:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 934);
 Potenciometro3(0, 303);
 break;
 case 155:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 907);
 Potenciometro3(0, 323);
 break;
 case 156:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 865);
 Potenciometro3(0, 320);
 break;
 case 157:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 777);
 Potenciometro3(0, 360);
 break;
 case 158:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 557);
 Potenciometro3(0, 340);
 break;
 case 159:
 Potenciometro1(0, 1022);
 Potenciometro2(0, 957);
 Potenciometro3(0, 315);
```

```
break;
case 160:
Potenciometro1(0, 1022);
Potenciometro2(0, 946);
Potenciometro3(0, 320);
break;
case 161:
Potenciometro1(0, 1022);
Potenciometro2(0, 931);
Potenciometro3(0, 342);
break;
case 162:
Potenciometro1(0, 1022);
Potenciometro2(0, 911);
Potenciometro3(0, 360);
break;
case 163:
Potenciometro1(0, 1022);
Potenciometro2(0, 881);
Potenciometro3(0, 350);
break;
case 164:
Potenciometro1(0, 1022);
Potenciometro2(0, 837);
Potenciometro3(0, 310);
break;
case 165:
Potenciometro1(0, 1022);
Potenciometro2(0, 749);
Potenciometro3(0, 310);
break;
case 166:
Potenciometro1(0, 1022);
Potenciometro2(0, 535);
Potenciometro3(0, 320);
break;
case 167:
Potenciometro1(0, 1021);
Potenciometro2(0, 929);
Potenciometro3(0, 683);
break;
case 168:
Potenciometro1(0, 1021);
Potenciometro2(0, 913);
Potenciometro3(0, 678);
break;
case 169:
Potenciometro1(0, 1021);
Potenciometro2(0, 890);
Potenciometro3(0, 675);
break;
case 170:
```

```
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 678);
break;
case 171:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 172:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 173:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 174:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 175:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 176:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 177:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 178:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 179:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 180:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
```

```
Potenciometro3(0, 340);
break;
case 181:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 182:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 183:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 184:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 185:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 186:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 187:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 188:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 189:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 190:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
```

```
case 191:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 192:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 193:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 194:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 195:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 196:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 197:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 198:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 199:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
/*****************************2*2*2*2*2*2*2*2*2*2**2*2*2*2*2*2*/
case 200:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 201:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 202:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 203:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 204:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 205:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 206:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 207:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 208:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 209:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 210:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 211:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
```

```
Potenciometro3(0, 340);
break;
case 212:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 213:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 214:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 215:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 216:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 217:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 218:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 219:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 220:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 221:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
```

```
    case 222:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 980);
    Potenciometro3(0, 340);
    break;
    case 223:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 980);
    Potenciometro3(0, 340);
    break;
    case 224:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 980);
    Potenciometro3(0, 340);
    break;
    case 225:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 980);
    Potenciometro3(0, 340);
    break;
  case 226:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1022);
    Potenciometro3(0, 931);
  break;
  case 227:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1022);
    Potenciometro3(0, 735);
  break;
  case 228:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1021);
    Potenciometro3(0, 902);
  break;
  case 229:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1021);
    Potenciometro3(0, 492);
 break;
  case 230:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 830);
break;
  case 231:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 887);
 break;
  case 232:
    Potenciometro1(0, 1023);
```

```
  Potenciometro2(0, 1019);
  Potenciometro3(0, 466);
 break;
case 233:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1018);
  Potenciometro3(0, 700);
 break;
case 234:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1017);
  Potenciometro3(0, 780);
  break;
case 235:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1016);
  Potenciometro3(0, 793);
break;
  case 236:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1015);
  Potenciometro3(0, 776);
 break;
  case 237:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1014);
  Potenciometro3(0, 744);
break;
  case 238:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1013);
  Potenciometro3(0, 642);
break;
  case 239:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1012);
  Potenciometro3(0, 474);
break;
  case 240:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1010);
  Potenciometro3(0, 708);
break;
  case 241:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1009);
  Potenciometro3(0, 435);
break;
  case 242:
  Potenciometro1(0, 1023);
  Potenciometro2(0, 1007);
  Potenciometro3(0, 559);
```

```
break;
   case 243:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1005);
   Potenciometro3(0, 525);
  break;
   case 244:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1003);
   Potenciometro3(0, 422);
   break;
   case 245:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 1000);
   Potenciometro3(0, 542);
   break;
   case 246:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 998);
   Potenciometro3(0, 138);
   break;
   case 247:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 994);
   Potenciometro3(0, 401);
   break;
   case 248:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 991);
   Potenciometro3(0, 142);
  break;
   case 249:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 986);
   Potenciometro3(0, 282);
  break;
   case 250:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 980);
   Potenciometro3(0, 340);
   break;
   case 251:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 973);
   Potenciometro3(0, 333);
   break;
   case 252:
   Potenciometro1(0, 1023);
   Potenciometro2(0, 962);
   Potenciometro3(0, 475);
   break;
   case 253:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 952);
Potenciometro3(0, 325);
break;
case 254:
Potenciometro1(0, 1023);
Potenciometro2(0, 934);
Potenciometro3(0, 303);
break;
case 255:
Potenciometro1(0, 1023);
Potenciometro2(0, 907);
Potenciometro3(0, 323);
break;
case 256:
Potenciometro1(0, 1023);
Potenciometro2(0, 865);
Potenciometro3(0, 320);
break;
case 257:
Potenciometro1(0, 1023);
Potenciometro2(0, 777);
Potenciometro3(0, 360);
break;
case 258:
Potenciometro1(0, 1023);
Potenciometro2(0, 557);
Potenciometro3(0, 340);
break;
case 259:
Potenciometro1(0, 1022);
Potenciometro2(0, 957);
Potenciometro3(0, 315);
break;
case 260:
Potenciometro1(0, 1022);
Potenciometro2(0, 946);
Potenciometro3(0, 320);
break;
case 261:
Potenciometro1(0, 1022);
Potenciometro2(0, 931);
Potenciometro3(0, 342);
break;
case 262:
Potenciometro1(0, 1022);
Potenciometro2(0, 911);
Potenciometro3(0, 360);
break;
case 263:
Potenciometro1(0, 1022);
Potenciometro2(0, 881);
```

```
Potenciometro3(0, 350);
break;
case 264:
Potenciometro1(0, 1022);
Potenciometro2(0, 837);
Potenciometro3(0, 310);
break;
case 265:
Potenciometro1(0, 1022);
Potenciometro2(0, 749);
Potenciometro3(0, 310);
break;
case 266:
Potenciometro1(0, 1022);
Potenciometro2(0, 535);
Potenciometro3(0, 320);
break;
case 267:
Potenciometro1(0, 1021);
Potenciometro2(0, 929);
Potenciometro3(0, 683);
break;
case 268:
Potenciometro1(0, 1021);
Potenciometro2(0, 913);
Potenciometro3(0, 678);
break;
case 269:
Potenciometro1(0, 1021);
Potenciometro2(0, 890);
Potenciometro3(0, 675);
break;
case 270:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 678);
break;
case 271:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 272:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 273:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
```

```
case 274:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 275:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 276:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 277:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 278:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 279:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 280:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 281:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 282:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 283:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 284:
Potenciometro1(0, 1023);
```

```
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 285:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 286:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 287:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 288:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 289:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 290:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 291:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 292:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 293:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 294:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
```

```
 break;
 case 295:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 296:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 297:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 298:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 299:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
/**************************3*3*3*3*3*3*3*3*3*3*3*3*3*3*3*3*3*3*3*/
 case 300:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 301:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 302:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 303:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 304:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
```

```
case 305:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 306:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 307:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 308:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 309:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 310:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 311:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 312:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 313:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 314:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 315:
Potenciometro1(0, 1023);
```

```
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 316:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 317:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 318:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 319:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 320:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 321:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 322:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 323:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 324:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 325:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
```

```
break;
case 326:
Potenciometro1(0, 1023);
Potenciometro2(0, 1022);
Potenciometro3(0, 931);
break;
case 327:
Potenciometro1(0, 1023);
Potenciometro2(0, 1022);
Potenciometro3(0, 735);
break;
case 328:
Potenciometro1(0, 1023);
Potenciometro2(0, 1021);
Potenciometro3(0, 902);
break;
case 329:
Potenciometro1(0, 1023);
Potenciometro2(0, 1021);
Potenciometro3(0, 492);
break;
case 330:
Potenciometro1(0, 1023);
Potenciometro2(0, 1020);
Potenciometro3(0, 830);
break;
case 331:
Potenciometro1(0, 1023);
Potenciometro2(0, 1019);
Potenciometro3(0, 887);
break;
case 332:
Potenciometro1(0, 1023);
Potenciometro2(0, 1019);
Potenciometro3(0, 466);
break;
case 333:
Potenciometro1(0, 1023);
Potenciometro2(0, 1018);
Potenciometro3(0, 700);
break;
case 334:
Potenciometro1(0, 1023);
Potenciometro2(0, 1017);
Potenciometro3(0, 780);
break;
case 335:
Potenciometro1(0, 1023);
Potenciometro2(0, 1016);
Potenciometro3(0, 793);
break;
case 336:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 1015);
Potenciometro3(0, 776);
break;
case 337:
Potenciometro1(0, 1023);
Potenciometro2(0, 1014);
Potenciometro3(0, 744);
break;
case 338:
Potenciometro1(0, 1023);
Potenciometro2(0, 1013);
Potenciometro3(0, 642);
break;
case 339:
Potenciometro1(0, 1023);
Potenciometro2(0, 1012);
Potenciometro3(0, 474);
break;
case 340:
Potenciometro1(0, 1023);
Potenciometro2(0, 1010);
Potenciometro3(0, 708);
break;
case 341:
Potenciometro1(0, 1023);
Potenciometro2(0, 1009);
Potenciometro3(0, 435);
break;
case 342:
Potenciometro1(0, 1023);
Potenciometro2(0, 1007);
Potenciometro3(0, 559);
break;
case 343:
Potenciometro1(0, 1023);
Potenciometro2(0, 1005);
Potenciometro3(0, 525);
break;
case 344:
Potenciometro1(0, 1023);
Potenciometro2(0, 1003);
Potenciometro3(0, 422);
break;
case 345:
Potenciometro1(0, 1023);
Potenciometro2(0, 1000);
Potenciometro3(0, 542);
break;
case 346:
Potenciometro1(0, 1023);
Potenciometro2(0, 998);
```

```
 Potenciometro3(0, 138);
 break;
 case 347:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 994);
 Potenciometro3(0, 401);
 break;
 case 348:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 991);
 Potenciometro3(0, 142);
 break;
 case 349:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 986);
 Potenciometro3(0, 282);
break;
 case 350:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 980);
 Potenciometro3(0, 340);
 break;
 case 351:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 973);
 Potenciometro3(0, 333);
 break;
 case 352:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 962);
 Potenciometro3(0, 475);
 break;
 case 353:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 952);
 Potenciometro3(0, 325);
 break;
 case 354:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 934);
 Potenciometro3(0, 303);
 break;
 case 355:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 907);
 Potenciometro3(0, 323);
 break;
 case 356:
 Potenciometro1(0, 1023);
 Potenciometro2(0, 865);
 Potenciometro3(0, 320);
 break;
```

```
case 357:
Potenciometro1(0, 1023);
Potenciometro2(0, 777);
Potenciometro3(0, 360);
break;
case 358:
Potenciometro1(0, 1023);
Potenciometro2(0, 557);
Potenciometro3(0, 340);
break;
case 359:
Potenciometro1(0, 1022);
Potenciometro2(0, 957);
Potenciometro3(0, 315);
break;
case 360:
Potenciometro1(0, 1022);
Potenciometro2(0, 946);
Potenciometro3(0, 320);
break;
case 361:
Potenciometro1(0, 1022);
Potenciometro2(0, 931);
Potenciometro3(0, 342);
break;
case 362:
Potenciometro1(0, 1022);
Potenciometro2(0, 911);
Potenciometro3(0, 360);
break;
case 363:
Potenciometro1(0, 1022);
Potenciometro2(0, 881);
Potenciometro3(0, 350);
break;
case 364:
Potenciometro1(0, 1022);
Potenciometro2(0, 837);
Potenciometro3(0, 310);
break;
case 365:
Potenciometro1(0, 1022);
Potenciometro2(0, 749);
Potenciometro3(0, 310);
break;
case 366:
Potenciometro1(0, 1022);
Potenciometro2(0, 535);
Potenciometro3(0, 320);
break;
case 367:
Potenciometro1(0, 1021);
```

```
Potenciometro2(0, 929);
Potenciometro3(0, 683);
break;
case 368:
Potenciometro1(0, 1021);
Potenciometro2(0, 913);
Potenciometro3(0, 678);
break;
case 369:
Potenciometro1(0, 1021);
Potenciometro2(0, 890);
Potenciometro3(0, 675);
break;
case 370:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 678);
break;
case 371:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 372:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 373:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 374:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 375:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 376:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 377:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
```

```
break;
case 378:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 379:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 380:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 381:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 382:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 383:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 384:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 385:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 386:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 387:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 388:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 389:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 390:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 391:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 392:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 393:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 394:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 395:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 396:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 397:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
Potenciometro3(0, 340);
break;
case 398:
Potenciometro1(0, 1023);
Potenciometro2(0, 980);
```

```
    Potenciometro3(0, 340);
    break;
    case 399:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 980);
    Potenciometro3(0, 340);
    break;

   default:
    break;
  }
  break;
  // GENERACION DE RTD PT100
case 5:
  z = w;
  switch (round(z)) {
    case -150:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 888);
    break;
    case -149:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 818);
    break;
    case -148:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1020);
    Potenciometro3(0, 616);
    break;
    case -147:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 899);
    break;
    case -146:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 843);
    break;
    case -145:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 717);
    break;
    case -144:
    Potenciometro1(0, 1023);
    Potenciometro2(0, 1019);
    Potenciometro3(0, 166);
    break;
    case -143:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 1018);
Potenciometro3(0, 847);
break;
case -142:
Potenciometro1(0, 1023);
Potenciometro2(0, 1018);
Potenciometro3(0, 740);
break;
case -141:
Potenciometro1(0, 1023);
Potenciometro2(0, 1018);
Potenciometro3(0, 387);
break;
case -140:
Potenciometro1(0, 1023);
Potenciometro2(0, 1017);
Potenciometro3(0, 830);
break;
case -139:
Potenciometro1(0, 1023);
Potenciometro2(0, 1017);
Potenciometro3(0, 709);
break;
case -138:
Potenciometro1(0, 1023);
Potenciometro2(0, 1017);
Potenciometro3(0, 255);
break;
case -137:
Potenciometro1(0, 1023);
Potenciometro2(0, 1016);
Potenciometro3(0, 788);
break;
case -136:
Potenciometro1(0, 1023);
Potenciometro2(0, 1016);
Potenciometro3(0, 605);
break;
case -135:
Potenciometro1(0, 1023);
Potenciometro2(0, 1015);
Potenciometro3(0, 814);
break;
case -134:
Potenciometro1(0, 1023);
Potenciometro2(0, 1015);
Potenciometro3(0, 687);
break;
case -133:
Potenciometro1(0, 1023);
Potenciometro2(0, 1015);
```

```
Potenciometro3(0, 207);
break;
case -132:
Potenciometro1(0, 1023);
Potenciometro2(0, 1014);
Potenciometro3(0, 714);
break;
case -131:
Potenciometro1(0, 1023);
Potenciometro2(0, 1014);
Potenciometro3(0, 376);
break;
case -130:
Potenciometro1(0, 1023);
Potenciometro2(0, 1013);
Potenciometro3(0, 713);
break;
case -129:
Potenciometro1(0, 1023);
Potenciometro2(0, 1013);
Potenciometro3(0, 342);
break;
case -128:
Potenciometro1(0, 1023);
Potenciometro2(0, 1012);
Potenciometro3(0, 676);
break;
case -127:
Potenciometro1(0, 1023);
Potenciometro2(0, 1012);
Potenciometro3(0, 233);
break;
case -126:
Potenciometro1(0, 1023);
Potenciometro2(0, 1011);
Potenciometro3(0, 608);
break;
case -125:
Potenciometro1(0, 1023);
Potenciometro2(0, 1010);
Potenciometro3(0, 720);
break;
case -124:
Potenciometro1(0, 1023);
Potenciometro2(0, 1010);
Potenciometro3(0, 474);
break;
case -123:
Potenciometro1(0, 1023);
Potenciometro2(0, 1009);
Potenciometro3(0, 628);
break;
```

```
case -122:
Potenciometro1(0, 1023);
Potenciometro2(0, 1009);
Potenciometro3(0, 65);
break;
case -121:
Potenciometro1(0, 1023);
Potenciometro2(0, 1008);
Potenciometro3(0, 419);
break;
case -120:
Potenciometro1(0, 1023);
Potenciometro2(0, 1007);
Potenciometro3(0, 553);
break;
case -119:
Potenciometro1(0, 1023);
Potenciometro2(0, 1006);
Potenciometro3(0,598);
break;
case -118:
Potenciometro1(0, 1023);
Potenciometro2(0, 1005);
Potenciometro3(0, 635);
break;
case -117:
Potenciometro1(0, 1023);
Potenciometro2(0, 1005);
Potenciometro3(0, 202);
break;
case -116:
Potenciometro1(0, 1023);
Potenciometro2(0, 1004);
Potenciometro3(0, 302);
break;
case -115:
Potenciometro1(0, 1023);
Potenciometro2(0, 1003);
Potenciometro3(0, 332);
break;
case -114:
Potenciometro1(0, 1023);
Potenciometro2(0, 1002);
Potenciometro3(0, 322);
break;
case -113:
Potenciometro1(0, 1023);
Potenciometro2(0, 1001);
Potenciometro3(0, 292);
break;
case -112:
Potenciometro1(0, 1023);
```

```
Potenciometro2(0, 999);
Potenciometro3(0, 579);
break;
case -111:
Potenciometro1(0, 1023);
Potenciometro2(0, 999);
Potenciometro3(0, 48);
break;
case -110:
Potenciometro1(0, 1023);
Potenciometro2(0, 997);
Potenciometro3(0, 438);
break;
case -109:
Potenciometro1(0, 1023);
Potenciometro2(0, 996);
Potenciometro3(0, 321);
break;
case -108:
Potenciometro1(0, 1023);
Potenciometro2(0, 995);
Potenciometro3(0, 91);
break;
case -107:
Potenciometro1(0, 1023);
Potenciometro2(0, 993);
Potenciometro3(0, 311);
break;
case -106:
Potenciometro1(0, 1023);
Potenciometro2(0, 992);
Potenciometro3(0, 8);
break;
case -105:
Potenciometro1(0, 1023);
Potenciometro2(0, 990);
Potenciometro3(0, 281);
break;
case -104:
Potenciometro1(0, 1023);
Potenciometro2(0, 988);
Potenciometro3(0, 312);
break;
case -103:
Potenciometro1(0, 1023);
Potenciometro2(0, 986);
Potenciometro3(0, 312);
break;
case -102:
Potenciometro1(0, 1023);
Potenciometro2(0, 984);
Potenciometro3(0, 272);
```

```
break;
case -101:
Potenciometro1(0, 1023);
Potenciometro2(0, 982);
Potenciometro3(0, 180);
break;
case -100:
Potenciometro1(0, 1023);
Potenciometro2(0, 979);
Potenciometro3(0, 293);
break;
case -99:
Potenciometro1(0, 1023);
Potenciometro2(0, 976);
Potenciometro3(0, 333);
break;
case -98:
Potenciometro1(0, 1023);
Potenciometro2(0, 973);
Potenciometro3(0, 319);
break;
case -97:
Potenciometro1(0, 1023);
Potenciometro2(0, 969);
Potenciometro3(0, 390);
break;
case -96:
Potenciometro1(0, 1023);
Potenciometro2(0, 965);
Potenciometro3(0, 405);
break;
case -95:
Potenciometro1(0, 1023);
Potenciometro2(0, 960);
Potenciometro3(0, 455);
break;
case -94:
Potenciometro1(0, 1023);
Potenciometro2(0, 955);
Potenciometro3(0, 472);
break;
case -93:
Potenciometro1(0, 1023);
Potenciometro2(0, 952);
Potenciometro3(0, 312);
break;
case -92:
Potenciometro1(0, 1023);
Potenciometro2(0, 945);
Potenciometro3(0, 358);
break;
case -91:
```

```
Potenciometro1(0, 1023);
Potenciometro2(0, 934);
Potenciometro3(0, 452);
break;
case -90:
Potenciometro1(0, 1023);
Potenciometro2(0, 929);
Potenciometro3(0, 313);
break;
case -89:
Potenciometro1(0, 1023);
Potenciometro2(0, 918);
Potenciometro3(0, 323);
break;
case -88:
Potenciometro1(0, 1023);
Potenciometro2(0, 906);
Potenciometro3(0, 323);
break;
case -87:
Potenciometro1(0, 1023);
Potenciometro2(0, 892);
Potenciometro3(0, 316);
break;
case -86:
Potenciometro1(0, 1023);
Potenciometro2(0, 874);
Potenciometro3(0, 316);
break;
case -85:
Potenciometro1(0, 1023);
Potenciometro2(0, 850);
Potenciometro3(0, 324);
break;
case -84:
Potenciometro1(0, 1023);
Potenciometro2(0, 817);
Potenciometro3(0, 350);
break;
case -83:
Potenciometro1(0, 1023);
Potenciometro2(0, 777);
Potenciometro3(0, 328);
break;
case -82:
Potenciometro1(0, 1023);
Potenciometro2(0, 717);
Potenciometro3(0, 340);
break;
case -81:
Potenciometro1(0, 1023);
Potenciometro2(0, 617);
```

```
Potenciometro3(0, 340);
break;
case -80:
Potenciometro1(0, 1023);
Potenciometro2(0, 417);
Potenciometro3(0, 350);
break;
case -79:
Potenciometro1(0, 1022);
Potenciometro2(0, 960);
Potenciometro3(0, 315);
break;
case -78:
Potenciometro1(0, 1022);
Potenciometro2(0, 957);
Potenciometro3(0, 270);
break;
case -77:
Potenciometro1(0, 1022);
Potenciometro2(0, 952);
Potenciometro3(0, 343);
break;
case -76:
Potenciometro1(0, 1022);
Potenciometro2(0, 948);
Potenciometro3(0, 295);
break;
case -75:
Potenciometro1(0, 1022);
Potenciometro2(0, 942);
Potenciometro3(0, 360);
break;
case -74:
Potenciometro1(0, 1022);
Potenciometro2(0, 936);
Potenciometro3(0, 365);
break;
case -73:
Potenciometro1(0, 1022);
Potenciometro2(0, 930);
Potenciometro3(0, 350);
break;
case -72:
Potenciometro1(0, 1022);
Potenciometro2(0, 923);
Potenciometro3(0, 342);
break;
case -71:
Potenciometro1(0, 1022);
Potenciometro2(0, 914);
Potenciometro3(0, 362);
break;
```

```
case -70:
Potenciometro1(0, 1022);
Potenciometro2(0, 904);
Potenciometro3(0, 358);
break;
case -69:
Potenciometro1(0, 1022);
Potenciometro2(0, 892);
Potenciometro3(0, 355);
break;
case -68:
Potenciometro1(0, 1022);
Potenciometro2(0, 878);
Potenciometro3(0, 360);
break;
case -67:
Potenciometro1(0, 1022);
Potenciometro2(0, 863);
Potenciometro3(0, 330);
break;
case -66:
Potenciometro1(0, 1022);
Potenciometro2(0, 843);
Potenciometro3(0, 330);
break;
case -65:
Potenciometro1(0, 1022);
Potenciometro2(0, 819);
Potenciometro3(0, 315);
break;
case -64:
Potenciometro1(0, 1022);
Potenciometro2(0, 787);
Potenciometro3(0, 305);
break;
case -63:
Potenciometro1(0, 1022);
Potenciometro2(0, 741);
Potenciometro3(0, 310);
break;
case -62:
Potenciometro1(0, 1022);
Potenciometro2(0, 680);
Potenciometro3(0, 310);
break;
case -61:
Potenciometro1(0, 1022);
Potenciometro2(0, 583);
Potenciometro3(0, 320);
break;
case -60:
Potenciometro1(0, 1022);
```

```
Potenciometro2(0, 416);
Potenciometro3(0, 280);
break;
case -59:
Potenciometro1(0, 1021);
Potenciometro2(0, 934);
Potenciometro3(0, 680);
break;
case -58:
Potenciometro1(0, 1021);
Potenciometro2(0, 928);
Potenciometro3(0, 682);
break;
case -57:
Potenciometro1(0, 1021);
Potenciometro2(0, 923);
Potenciometro3(0, 682);
break;
case -56:
Potenciometro1(0, 1021);
Potenciometro2(0, 916);
Potenciometro3(0, 680);
break;
case -55:
Potenciometro1(0, 1021);
Potenciometro2(0, 908);
Potenciometro3(0, 674);
break;
case -54:
Potenciometro1(0, 1021);
Potenciometro2(0, 899);
Potenciometro3(0, 671);
break;
case -53:
Potenciometro1(0, 1021);
Potenciometro2(0, 888);
Potenciometro3(0, 679);
break;
case -52:
Potenciometro1(0, 1021);
Potenciometro2(0, 877);
Potenciometro3(0, 677);
break;
case -51:
Potenciometro1(0, 1021);
Potenciometro2(0, 863);
Potenciometro3(0, 678);
break;
case -50:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 712);
```

```
break;
case -49:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 645);
break;
case -48:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 538);
break;
case -47:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 393);
break;
case -46:
Potenciometro1(0, 1021);
Potenciometro2(0, 857);
Potenciometro3(0, 69);
break;
case -45:
Potenciometro1(0, 1021);
Potenciometro2(0, 753);
Potenciometro3(0, 659);
break;
case -44:
Potenciometro1(0, 1021);
Potenciometro2(0, 713);
Potenciometro3(0, 659);
break;
case -43:
Potenciometro1(0, 1021);
Potenciometro2(0, 661);
Potenciometro3(0, 653);
break;
case -42:
Potenciometro1(0, 1021);
Potenciometro2(0, 591);
Potenciometro3(0, 643);
break;
case -41:
Potenciometro1(0, 1021);
Potenciometro2(0, 566);
Potenciometro3(0, 586);
break;
case -40:
Potenciometro1(0, 1021);
Potenciometro2(0, 506);
Potenciometro3(0, 538);
break;
case -39:
```

```
Potenciometro1(0, 1021);
Potenciometro2(0, 443);
Potenciometro3(0, 475);
break;
case -38:
Potenciometro1(0, 1021);
Potenciometro2(0, 383);
Potenciometro3(0, 315);
break;
case -37:
Potenciometro1(0, 1021);
Potenciometro2(0, 212);
Potenciometro3(0, 205);
break;
case -36:
Potenciometro1(0, 1021);
Potenciometro2(0, 12);
Potenciometro3(0, 13);
break;
case -35:
Potenciometro1(0, 1020);
Potenciometro2(0, 802);
Potenciometro3(0, 872);
break;
case -34:
Potenciometro1(0, 1020);
Potenciometro2(0, 780);
Potenciometro3(0, 870);
break;
case -33:
Potenciometro1(0, 1020);
Potenciometro2(0, 748);
Potenciometro3(0, 870);
break;
case -32:
Potenciometro1(0, 1020);
Potenciometro2(0, 748);
Potenciometro3(0, 858);
break;
case -31:
Potenciometro1(0, 1020);
Potenciometro2(0, 735);
Potenciometro3(0, 846);
break;
case -30:
Potenciometro1(0, 1020);
Potenciometro2(0, 715);
Potenciometro3(0, 841);
break;
case -29:
Potenciometro1(0, 1020);
Potenciometro2(0, 695);
```

```
Potenciometro3(0, 831);
break;
case -28:
Potenciometro1(0, 1020);
Potenciometro2(0, 675);
Potenciometro3(0, 819);
break;
case -27:
Potenciometro1(0, 1020);
Potenciometro2(0, 655);
Potenciometro3(0, 806);
break;
case -26:
Potenciometro1(0, 1020);
Potenciometro2(0, 615);
Potenciometro3(0, 807);
break;
case -25:
Potenciometro1(0, 1020);
Potenciometro2(0, 595);
Potenciometro3(0, 792);
break;
case -24:
Potenciometro1(0, 1020);
Potenciometro2(0, 575);
Potenciometro3(0, 773);
break;
case -23:
Potenciometro1(0, 1020);
Potenciometro2(0, 555);
Potenciometro3(0, 747);
break;
case -22:
Potenciometro1(0, 1020);
Potenciometro2(0, 535);
Potenciometro3(0, 719);
break;
case -21:
Potenciometro1(0, 1020);
Potenciometro2(0, 515);
Potenciometro3(0, 685);
break;
case -20:
Potenciometro1(0, 1020);
Potenciometro2(0, 495);
Potenciometro3(0, 642);
break;
case -19:
Potenciometro1(0, 1020);
Potenciometro2(0, 465);
Potenciometro3(0, 592);
break;
```

```
case -18:
Potenciometro1(0, 1020);
Potenciometro2(0, 435);
Potenciometro3(0, 522);
break;
case -17:
Potenciometro1(0, 1020);
Potenciometro2(0, 375);
Potenciometro3(0, 462);
break;
case -16:
Potenciometro1(0, 1020);
Potenciometro2(0, 325);
Potenciometro3(0, 352);
break;
case -15:
Potenciometro1(0, 1020);
Potenciometro2(0, 225);
Potenciometro3(0, 252);
break;
case -14:
Potenciometro1(0, 1020);
Potenciometro2(0, 95);
Potenciometro3(0, 102);
break;
case -13:
Potenciometro1(0, 1019);
Potenciometro2(0, 812);
Potenciometro3(0, 802);
break;
case -12:
Potenciometro1(0, 1019);
Potenciometro2(0, 793);
Potenciometro3(0, 801);
break;
case -11:
Potenciometro1(0, 1019);
Potenciometro2(0, 770);
Potenciometro3(0, 801);
break;
case -10:
Potenciometro1(0, 1019);
Potenciometro2(0, 737);
Potenciometro3(0, 801);
break;

case -9:
Potenciometro1(0, 1019);
Potenciometro2(0, 717);
Potenciometro3(0, 793);
break;
case -8:
```

```
    Potenciometro1(0, 1019);
    Potenciometro2(0, 697);
    Potenciometro3(0, 783);
    break;
    case -7:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 677);
    Potenciometro3(0, 771);
    break;
    case -6:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 657);
    Potenciometro3(0, 757);
    break;
    case -5:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 627);
    Potenciometro3(0, 745);
    break;
    case -4:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 587);
    Potenciometro3(0, 738);
    break;
    case -3:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 567);
    Potenciometro3(0, 716);
    break;
    case -2:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 547);
    Potenciometro3(0, 686);
    break;
    case -1:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 527);
    Potenciometro3(0, 651);
    break;
    case 0:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 494);
    Potenciometro3(0, 614);
    break;
    case 1:
    Potenciometro1(0, 1019);
    Potenciometro2(0, 464);
    Potenciometro3(0, 567);
    break;
  default:
    break;
}
```

```
    break;
    // GENERACION DE CORRIENTE PARA LA SUIMULACION DE TERMOCUPLA
   case 6:
    reset_CI1();
    delay(500);
    configurarI3();
    Salida_CI1(round(w*100));
    Serial.print("Corrie_tc escrito: ");
    Serial.println(w*100);
   break;

  }
}
/*****************************************************************************
**
 * FUNCION: PARA EL RESET DEL INTEGRADO AD5231
*****************************************************************************
*/
void reset() {

  byte comando = 0x80;
  byte b1 = 0x00;
  byte b0 = 0x00;
  digitalWrite(CS2, LOW);
  delay(10);
  digitalWrite(CS3, LOW);
  delay(10);
  digitalWrite(CS1, LOW);
  SPI.transfer(comando);
  SPI.transfer(b1);
  SPI.transfer(b0);
  digitalWrite(CS1, HIGH);
  digitalWrite(CS2, HIGH);
  digitalWrite(CS3, HIGH);
}
```

ANEXO C
PROGRAMACIÓN DE PYTHON

```
# -*- coding: cp1252 -*-
from reportlab.pdfgen import canvas
from reportlab.lib.units import inch
from reportlab.lib.colors import pink, green, brown, white
from reportlab.platypus import Table
from reportlab.platypus import Spacer
from reportlab.lib import colors
from reportlab.platypus import SimpleDocTemplate
from reportlab.lib.styles import getSampleStyleSheet
from reportlab.platypus import SimpleDocTemplate, Table, TableStyle, Paragraph
from reportlab.lib.styles import getSampleStyleSheet
import serial, time, sys
from Tkinter import *
from time import sleep
import sys
print ("")
serialPort= input ("ingrese el puerto COM: ")
ser= int (serialPort)
cadena ="COM"+str(ser)
print ("")
print ("Puerto seleccionado:  "+cadena)
print ("")
Arduino = serial.Serial(cadena, 9600, timeout = 1)
flagCharacter = 'k'
flagCharacter1 = 'l'
flagCharacter2 = 'm'
time.sleep(2.2)
Arduino.write(flagCharacter)
e = Arduino.readline()
time.sleep(1.2)
Arduino.close()

cero= e[0:4]
mas25= e[4:8]
mas50= e[8:13]
mas75= e[13:18]
cien= e[18:23]
menos75= e[23:28]
menos50= e[28:33]
menos25= e[33:37]
menoscero= e[37:41]
grado0= e[41:45]
grado100= e[45:51]

c = canvas.Canvas("Documento de Calibracion.pdf")
c.setFont("Helvetica-Bold", 14)
c.drawCentredString(331,636,cero)
```

```
c.drawCentredString(331,602,mas25)
c.drawCentredString(331,568,mas50)
c.drawCentredString(331,534,mas75)
c.drawCentredString(331,500,cien)
c.drawCentredString(331,466,menos75)
c.drawCentredString(331,432,menos50)
c.drawCentredString(331,398,menos25)
c.drawCentredString(331,364,menoscero)

h1=float(unicode(cero))
h11=((h1-4)/16)*100
c.drawCentredString(401,636,str(h11))

h2=float(unicode(mas25))
h22=((h2-8)/16)*100
c.drawCentredString(401,602,str(h22))

h3=float(unicode(mas50))
h33=((h3-12)/16)*100
c.drawCentredString(401,568,str(h33))

h4=float(unicode(mas75))
h44=((h4-16)/16)*100
c.drawCentredString(401,534,str(h44))

h5=float(unicode(cien))
h55=((h5-20)/16)*100
c.drawCentredString(401,500,str(h55))

h6=float(unicode(menos75))
h66=((h6-16)/16)*100
c.drawCentredString(401,466,str(h66))

h7=float(unicode(menos50))
h77=((h7-12)/16)*100
c.drawCentredString(401,432,str(h77))

h8=float(unicode(menos25))
h88=((h8-8)/16)*100
c.drawCentredString(401,398,str(h88))

h9=float(unicode(menoscero))
h99=((h9-4)/16)*100
c.drawCentredString(401,364,str(h99))

g1=float(unicode(grado0))
g2=float(unicode(grado100))
g3=g2-g1
```

```
g4=g3/4
c.drawCentredString(261,636,str(g1))
g5=g1+g4
c.drawCentredString(261,602,str(g5))
g6=g5+g4
c.drawCentredString(261,568,str(g6))
g7=g6+g4
c.drawCentredString(261,534,str(g7))
c.drawCentredString(261,500,str(g2))
c.drawCentredString(261,466,str(g7))
c.drawCentredString(261,432,str(g6))
c.drawCentredString(261,398,str(g5))
c.drawCentredString(261,364,str(g1))



#restar 34

fila1 = ['0',' ','25',' ', '50',' ', '75',' ','100',' ','75',' ','50',' ','25',' ','0']
#TODO: Get this line right instead of just copying it from the docs
textobject = c.beginText()
textobject.setTextOrigin(187, 636)
textobject.setFont("Helvetica-Bold", 14)
for line in fila1:
   uniLine = unicode(line, 'latin-1')
   textobject.textLine(uniLine)
c.drawText(textobject)



c.setFont("Helvetica-Bold", 8.5)
c.drawString(162,680,"PORCENTAJE")
c.setFont("Helvetica-Bold", 12)
c.drawCentredString(191,665,"%")
c.setFont("Helvetica-Bold", 8.5)
c.drawString(228,680,"TEMPERATURA")
c.setFont("Helvetica-Bold", 12)
c.drawCentredString(261,665,"C")
c.setFont("Helvetica-Bold", 10)
c.drawCentredString(331,678,"MEDICION")
c.setFont("Helvetica-Bold", 12)
c.drawCentredString(331,665,"mA")
c.setFont("Helvetica-Bold", 10)
c.drawCentredString(401,678,"ERROR")
c.setFont("Helvetica-Bold", 12)
c.drawCentredString(401,665,"%")



c.setFont("Helvetica-Bold", 12)
```

```
c.drawString(70,335,"Cliente:")
c.setFont("Helvetica", 12)
c.drawString(120,335,"Laboratorio de Redes Industriales y Control de Procesos de la UFA-
ESPEL")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,310,"Direccion:")
c.setFont("Helvetica", 12)
c.drawString(140,310,"Avenida Quijano y Ordonez y Hermanas Paez")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,285,"Representante:")
c.setFont("Helvetica", 12)
c.drawString(160,285,"Ing. Edwin Pruna")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,260,"Telefono:")
c.setFont("Helvetica", 12)
c.drawString(130,260,"2810-208")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,235,"Equipo:")
c.setFont("Helvetica", 12)
c.drawString(115,235,"Transmisor de Temperatura")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,210,"Marca:")
c.setFont("Helvetica", 12)
c.drawString(120,210,"ROSEMOUNT")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,185,"Modelo:")
c.setFont("Helvetica", 12)
c.drawString(130,185,"3144-EMERSON")

c.setFont("Helvetica-Bold", 12)
c.drawString(330,285,"Fecha Inicial:")
c.setFont("Helvetica", 12)
c.drawString(415,285,str (time.localtime()[2])+"-"+str(time.localtime()[1])+"-
"+str(time.localtime()[0]))

c.setFont("Helvetica-Bold", 12)
c.drawString(330,260,"Fecha Final:")
c.setFont("Helvetica", 12)
c.drawString(415,260,str (time.localtime()[2])+"-"+str(time.localtime()[1])+"-
"+str(time.localtime()[0]))

c.setFont("Helvetica-Bold", 12)
```

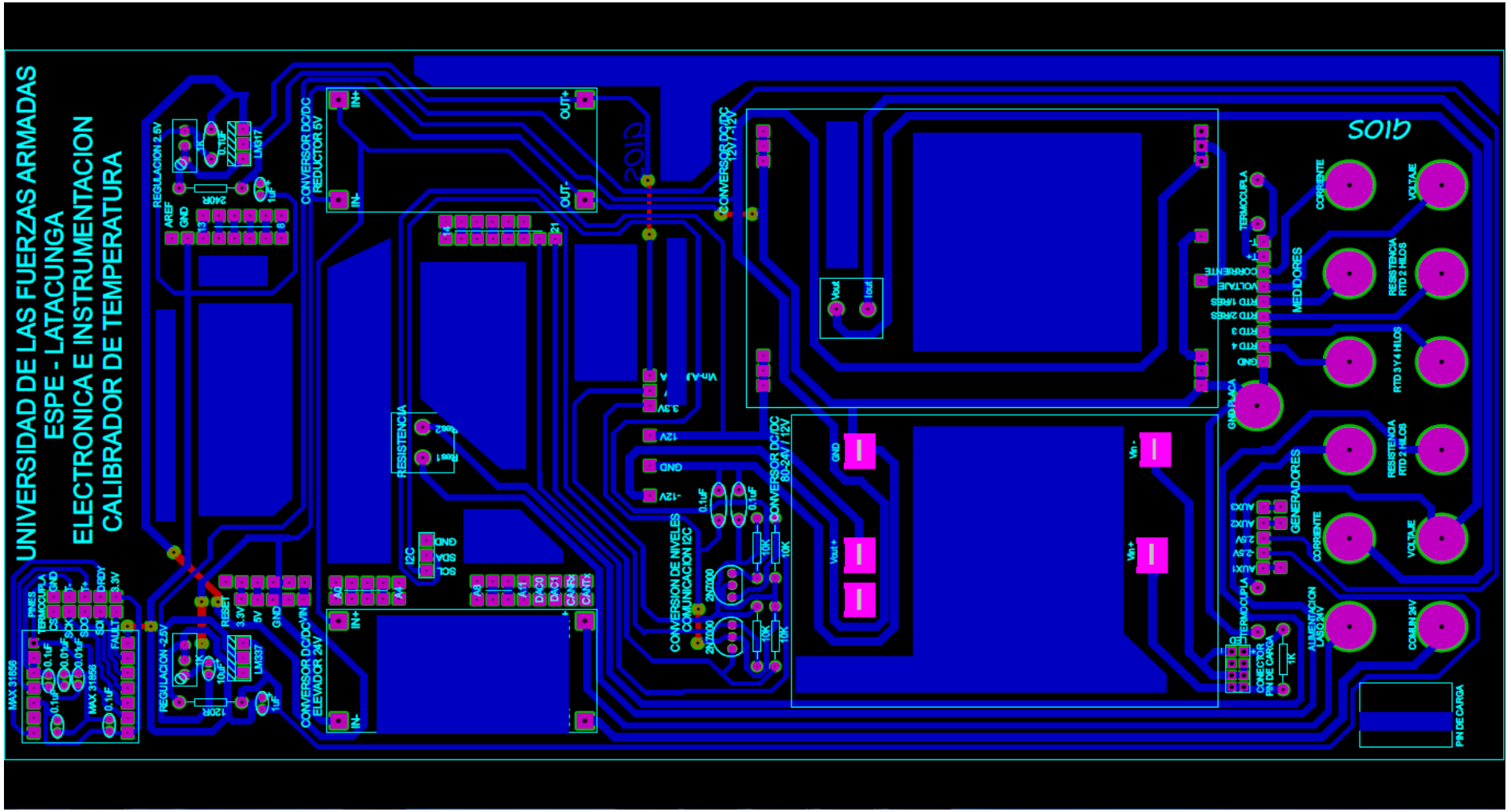
```
c.drawString(330,235,"INSTRUMENTO PATRON:")

c.setFont("Helvetica", 12)
c.drawString(330,210,"PROTOTIPO IMPLEMENTADO (TESIS)")

c.setFont("Helvetica-Bold", 12)
c.drawString(330,185,"Hora de Calibracion:")
c.setFont("Helvetica", 12)
c.drawString(460,185,str(time.localtime()[3])+":"+str(time.localtime()[4])+":"+str(time.localt
ime()[5]))

c.setFont("Helvetica-Bold", 12)
c.drawString(70,160,"Temp. Ambiente:")
c.setFont("Helvetica", 12)
c.drawString(170,160,"22 C")

c.setFont("Helvetica-Bold", 12)
c.drawString(330,160,"Humedad Relativa:")
c.setFont("Helvetica", 12)
c.drawString(445,160,"55%")

c.setFont("Helvetica-Bold", 12)
c.drawString(70,135,"RESPONSABLES:")

c.setFont("Helvetica-Bold", 14)
c.drawString(70,80,"GERMAN ONCE")
c.setFont("Helvetica-Bold", 14)
c.drawString(220,80,"JONATHAN RIVERA")
c.setFont("Helvetica-Bold", 14)
c.drawString(400,80,"ING. EDWIN PRUNA")

c.setFont("Helvetica", 11)
c.drawCentredString(296,58,"La reproduccion total o parcial de este documento se
realizara unicamente con la aprobacion escrita de la")
c.drawCentredString(296,45,"Universidad de las Fuerzas Armadas ESPE-L")
c.setFont("Helvetica-Bold", 14)
c.drawString(70,97,"_____ _____
_____")

c.setFont("Helvetica-Bold", 16)
c.drawCentredString(296, 705, "CERTIFICADO DE CALIBRACION")
c.setFillColor(green)
c.drawCentredString(296, 735, "UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE -
LATACUNGA")
c.drawImage("C:\\Users\\Jona\\espe.jpg",50,770,150,50)
c.drawImage("C:\\Users\\Jona\\Electronica.jpg",500,770,50,50)

c.rect(30, 30, 532, 730, stroke=1, fill=0)
```

```
c.translate(0, 352)
c.grid([156, 226, 296, 366, 436], [0, 34, 68, 102, 136, 170, 204, 238, 272, 306, 340])
c.setLineWidth(5)
c.showPage()
c.save()
print "         ","DAOS DE LA CALIBRACION"
print "TEMPERATURA","   ","CORRIENTE mA","   ","ERROR %"
print " ",  g1, "         ",cero,"        ",h11
print " ",  g5, "         ",mas25,"        ",h22
print " ",  g6, "         ",mas50,"       ",h33
print " ",  g7, "         ",mas75,"       ",h44
print " ",  g2, "         ",cien,"       ",h55
print " ",  g7, "         ",menos75,"       ",h66
print " ",  g6, "         ",menos50,"       ",h77
print " ",  g5, "         ",menos25,"        ",h88
print " ",  g1, "         ",menoscero,"        ",h99
print""
print ("ARCHIVO GENERADO CORRECTAMENTE")
print ("")
print ("NOMBE DEL ARCHIVO: Documento de Calibracion")
print ("")
```

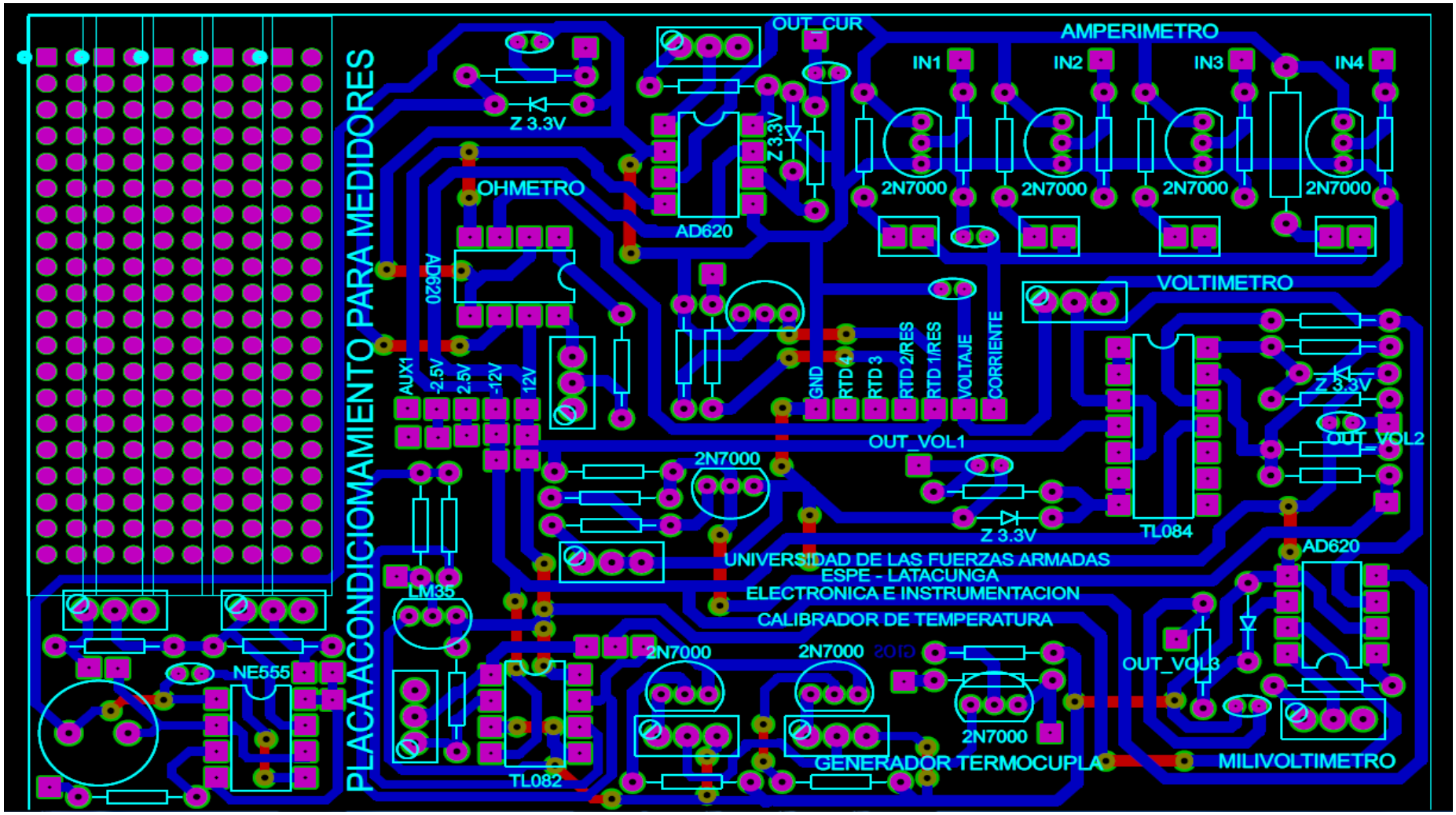
ANEXO D

CIRCUITOS IMPRESOS

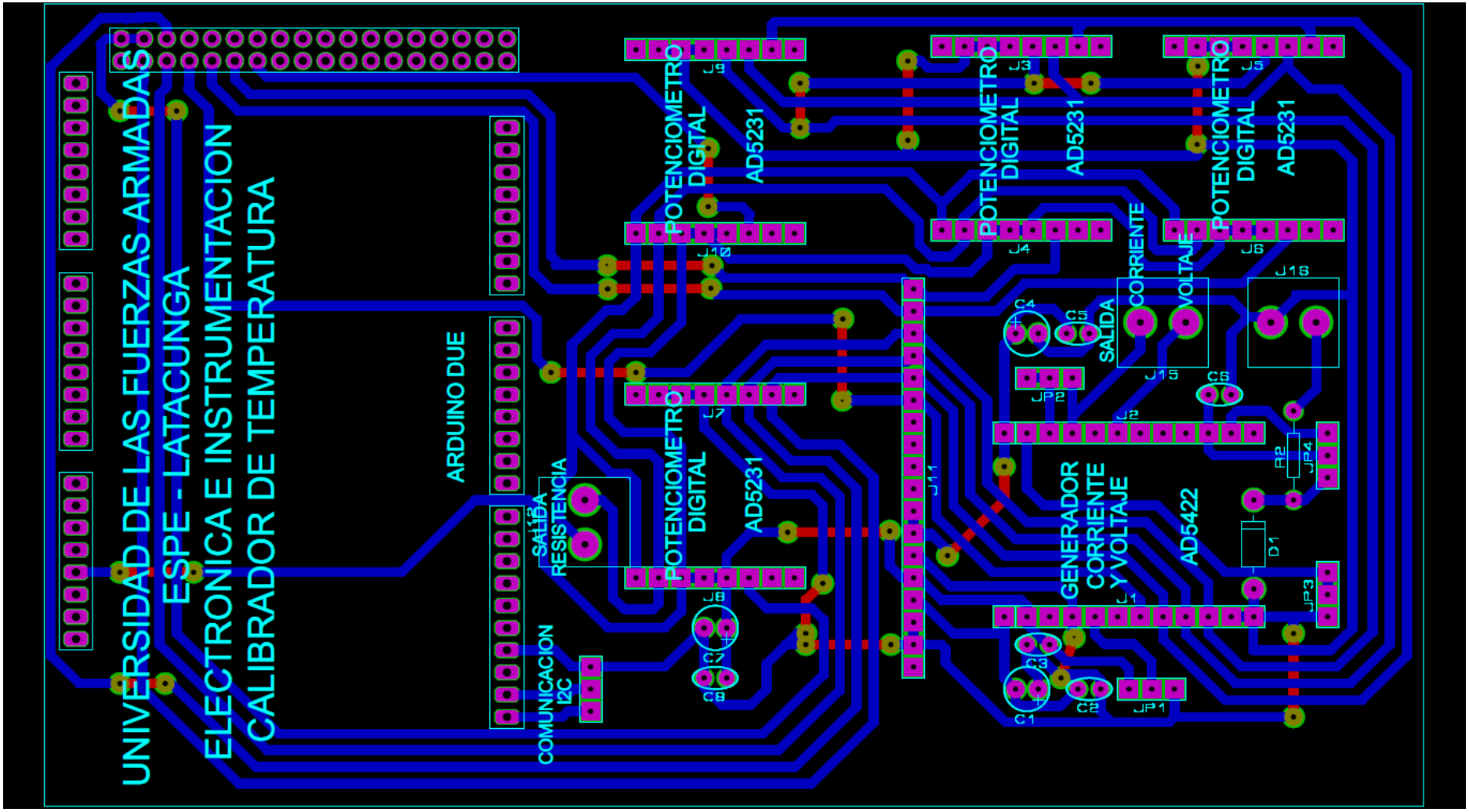
PLACA GENERAL



MEDIDORES

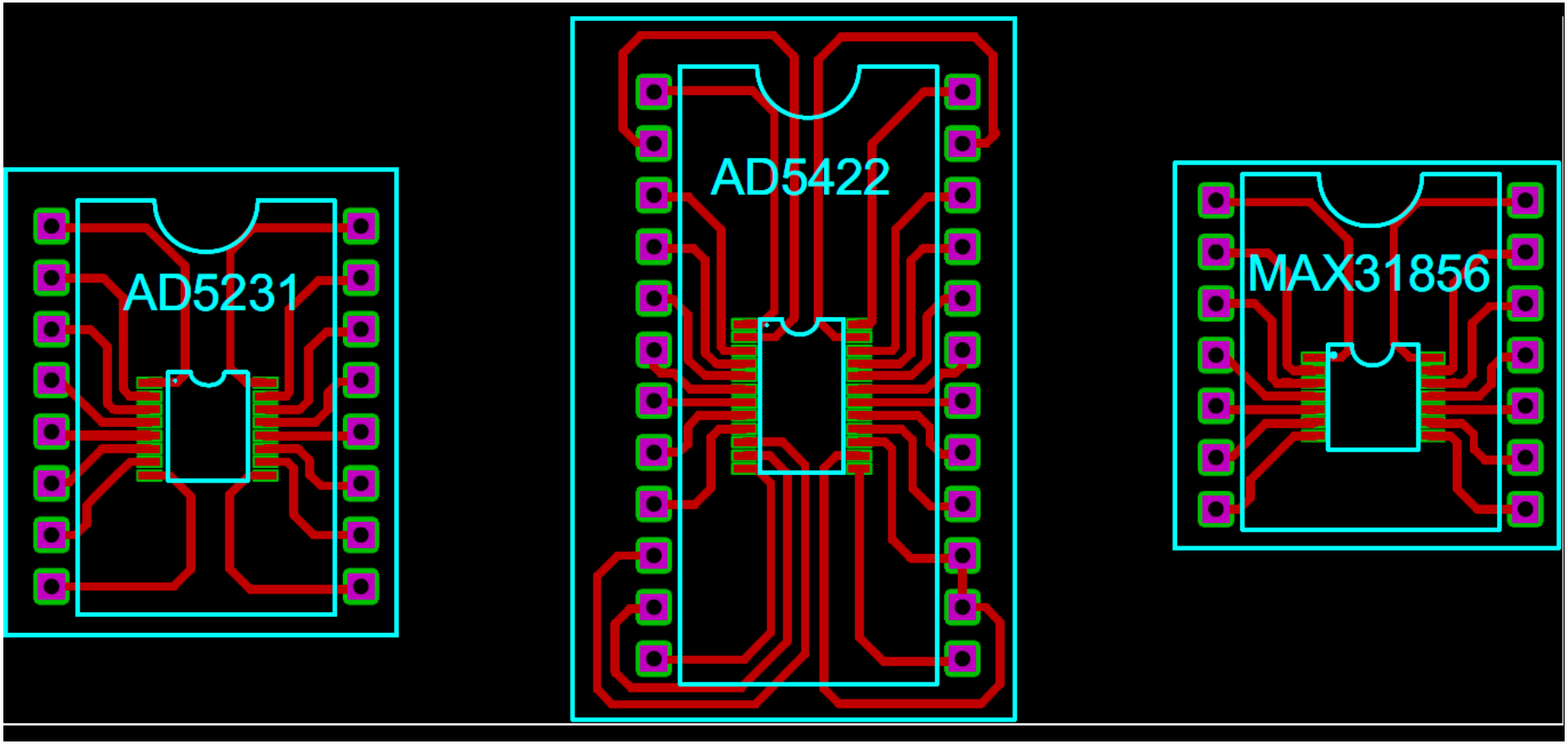

PLACA ACONDICIOMAMIENTO PARA MEDIDORES
OHMETRO
AMPERIMETRO
VOLTIMETRO
OUT_CUR
IN1
IN2
IN3
IN4
2N7000
AD620
Z 3.3V
AUX1
-2.5V
2.5V
-12V
12V
GND
RTD 4
RTD 3
RTD 2/RES
RTD 1/RES
VOLTAJE
CORRIENTE
OUT_VOL1
OUT_VOL2
OUT_VOL3
TL084
LM35
NE555
TL082
UNIVERSIDAD DE LAS FUERZAS ARMADAS
ESPE - LATACUNGA
ELECTRONICA E INSTRUMENTACION
CALIBRADOR DE TEMPERATURA
GENERADOR TERMOCUPLA
MILIVOLTIMETRO

GENERADORES



SHIELD PARA ARDUINO DUE

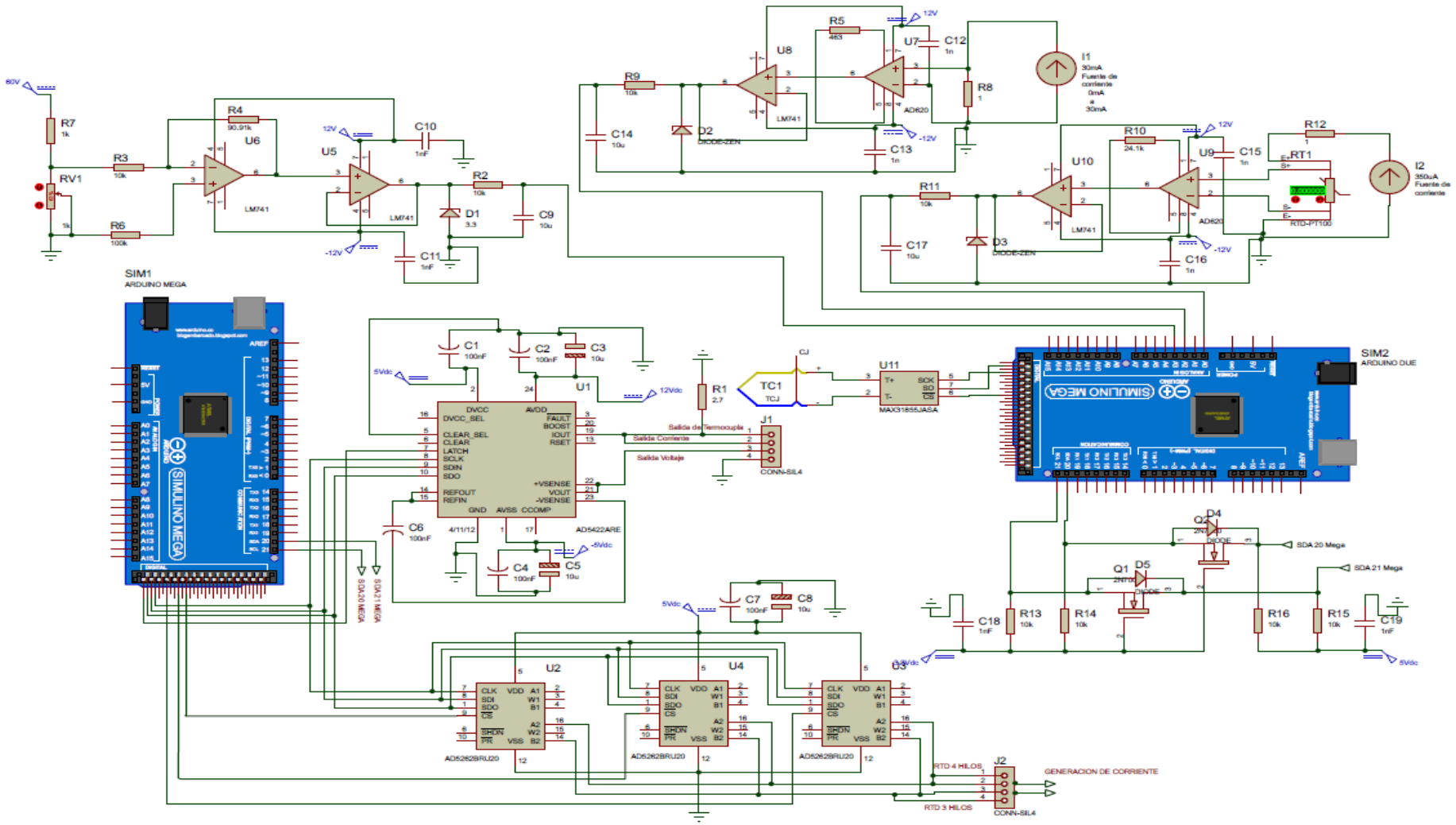
BASES PARA LOS INTEGRADOS CMD



ANEXO E

DIAGRAMA ESQUEMÁTICO GENERAL

DIAGRAMA ESQUEMATICO GENERAL



ANEXO F

CERTIFICADO DE CALIBRACIÓN

CERTIFICADO DE CALIBRACIÓN

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE - LATACUNGA

CERTIFICADO DE CALIBRACION

PORCENTAJE %	TEMPERATURA C	MEDICION mA	ERROR %
0	0.0	3.86	-0.875
25	25.0	8.15	0.9375
50	50.0	11.81	-1.1875
75	75.0	16.14	0.875
100	100.0	20.19	1.1875
75	75.0	16.13	0.8125
50	50.0	11.82	-1.125
25	25.0	8.16	1.0
0	0.0	3.86	-0.875

Cliente: Laboratorio de Redes Industriales y Control de Procesos de la UFA-ESPEL

Direccion: Avenida Quijano y Ordonez y Hermanas Paez

Representante: Ing. Edwin Pruna

Telefono: 2810-208

Equipo: Transmisor de Temperatura

Marca: ROSEMOUNT

Modelo: 3144-EMERSON

Temp. Ambiente: 22 C

Fecha Inicial: 12-3-2017

Fecha Final: 12-3-2017

INSTRUMENTO PATRON:

PROTOTIPO IMPLEMENTADO (TESIS)

Hora de Calibracion: 1:55:7

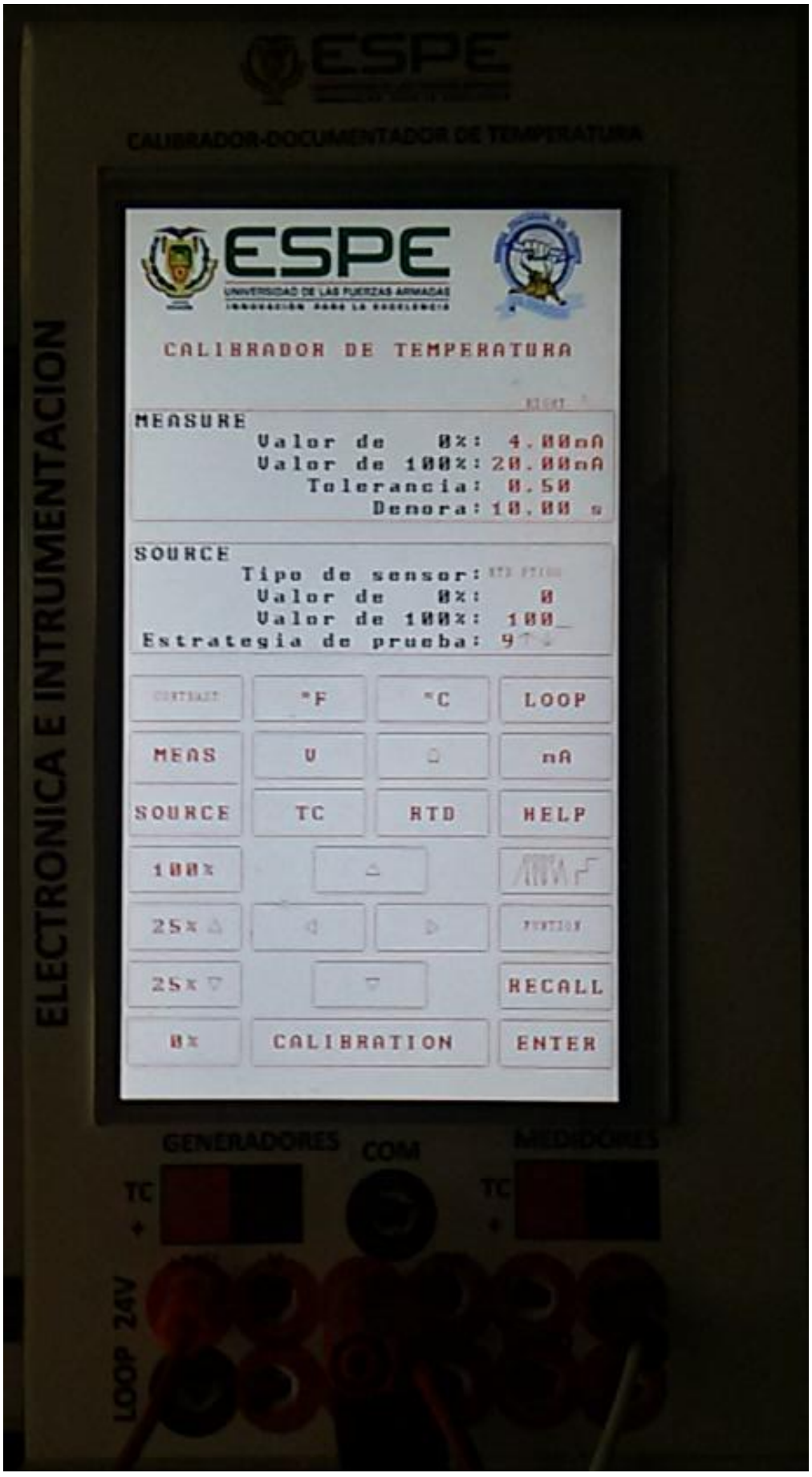
Humedad Relativa: 55%

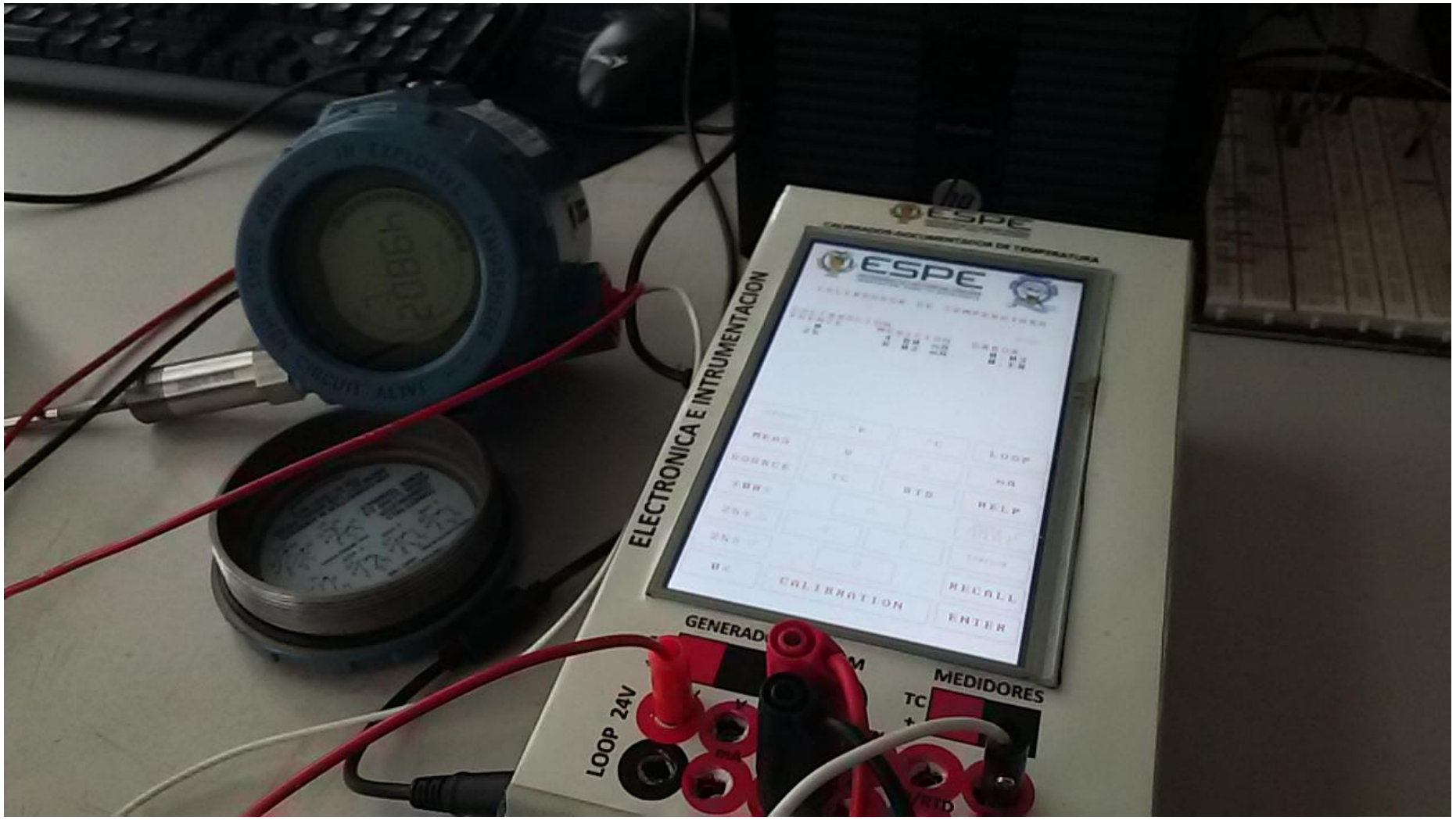
RESPONSABLES:

GERMAN ONCE JONATHAN RIVERA ING. EDWIN PRUNA

ANEXO G

PROTOTIPO IMPLEMENTADO COMPLETAMENTE


ESPE
CALIBRADOR-DOCUMENTADOR DE TEMPERATURA
ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
CALIBRADOR DE TEMPERATURA
MEASURE
Valor de 0%: 4.00nA
Valor de 100%: 20.00nA
Tolerancia: 0.50
Demora: 10.00 s
SOURCE
Tipo de sensor:
Valor de 0%: 0
Valor de 100%: 100
Estrategia de prueba: 9
°F
°C
LOOP
MEAS
V
nA
SOURCE
TC
RTD
HELP
100%
25%
25%
RECALL
0%
CALIBRATION
ENTER
ELECTRONICA E INTRUMENTACION
GENERADORES
COM
MEDIDORES
TC
TC
LOOP 24V


ESPE
ELECTRONICA E INTRUMENTACION
ESPE
LOOP
RTD
HELP
CALIBRATION
RECALL
ENTER
MEDIDORES
TC
LOOP 24V
RTD

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA E INSTRUMENTACIÓN.

CERTIFICACIÓN

Se certifica que el presente trabajo fue desarrollado por los señores **ONCE SUCUZHAÑAY GERMÁN ISRAEL** y **RIVERA COMINA JONATHAN JAVIER.**

En la cuidad de Latacunga a los 13 días del mes de marzo del 2017

Ing. Edwin Pruna

DIRECTOR DEL PROYECTO

Aprobado por:

Ing. Franklin Silva

DIRECTOR DE CARRERA

Dr. Rodrigo Vaca

SECRETARIO ACADEMICO