Trabajo Fin de Grado
Ingeniería de la Energía

Desarrollo de una herramienta destinada a la adquisición y procesamiento de señales

Autor: Miguel Ángel Tagua Navarrete

Tutor: Javier Serrano Reyes

Dpto. Ingeniería Energética
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado
Ingeniería de la Energía

Desarrollo de una herramienta destinada a la adquisición y procesamiento de señales

Autor:

Miguel Ángel Tagua Navarrete

Tutor:

Javier Serrano Reyes

Profesor Sustituto Interino

Dpto. de Ingeniería Energética

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2022

Proyecto Fin de Carrera: Desarrollo de una herramienta destinada a la adquisición y procesamiento de señales

Autor: Miguel Ángel Tagua Navarrete

Tutor: Javier Serrano Reyes

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2022

El Secretario del Tribunal

A mi familia

A mis amigos

A mis profesores

Agradecimientos

Han sido 5 años de duro trabajo para poder llegar a escribir estas palabras, año tras año he ido aprendiendo y mejorando tanto personal como profesionalmente. Como decía Isaac Newton "*Si consigo ver más lejos es porque he conseguido auparme a hombros de gigantes"*. Todo esto ha sido posible a todos los que me han rodeado en el camino y me han empujado a ser lo que soy ahora.

Gracias a mi familia, mi madre Estrella por su apoyo incondicional y esfuerzo para verme llegar lejos, mi padre Miguel porque gracias a él soy una persona capacitada para cualquier reto que se interponga en el camino, de ellos he aprendido a no rendirme, a mis hermanos, Estrella y Aithor por que han sabido aguantar mi manera de ser y mis berrinches. Gracias también a Diego, Víctor e Israel por ser mis apoyos morales en los momentos más difíciles, siempre hemos sido una verdadera piña y nadie ha podido con nosotros, por mis Termoenergéticos. Gracias sobre todo a ti, Eva, has sabido ser mi alma gemela durante todo este trayecto todo lo que he conseguido es logro tuyo, por saber de mi más que nadie, mi protección ante cualquier catástrofe que me ha surgido y sobre todo hacerme sentir lo que valgo. No quiero terminar estas palabras de agradecimiento sin mencionar a Javier, Francisco, Juan y Antonio por confiar en mi desde el principio y por enseñarme tanto en tan poco tiempo.

Miguel Ángel Tagua Navarrete

Sevilla, 2022

Resumen

La obtención de información es una necesidad, cada vez es más importante contar con sistemas que puedan lograr extraer datos de ella de manera que puedan ser mostrados, analizados y transformados para tomar decisiones. Las industrias requieren de herramientas para captar los datos, por ejemplo, conocer la temperatura para el control de un reactor en una planta química, para ello, es necesario poseer sensores que puedan adquirir las señales para visualizarlos y estudiarlos. Este proceso de obtención de información a través de un fenómeno físico medible y poder transformarlo en un valor es posible conseguirlo con un sistema de adquisición de datos (DAQ). La utilización de esta herramienta también es apropiada para la aplicación en el laboratorio de máquinas y motores térmicos de la Escuela Técnica Superior de Ingeniería, pues los trabajos realizados en este requieren el uso de este instrumento considerando que, se realizan ensayos a motores de combustión interna alternativos (MCIA), se efectúan trabajos de investigación en los que tomar medidas y los alumnos de la escuela realizan sus prácticas en distintas asignaturas.

El objetivo de este proyecto es la realización de un DAQ debido a la necesidad de remodelar el sistema actual que se encuentra en el laboratorio. Además, se ha fabricado cumpliendo con los requisitos establecidos en cuanto a versatilidad, una estructura firme y económico. Durante este trabajo, se muestra la información relativa al proceso de montaje y los elementos que han sido utilizados, siendo Arduino el hardware de adquisición elegido. También, se ha diseñado una aplicación que pueda controlar está herramienta con MATLAB y, por último, se ha comprobado el funcionamiento del sistema en el laboratorio.

Abstract

Obtaining information is a necessity, and it is increasingly important to have systems that can extract data from it so that it can be displayed, analysed, and transformed to make decisions. Industries require tools to capture data, for example, to know the temperature for the control of a reactor in a chemical plant, for this, it is necessary to have sensors that can acquire the signals to visualise and study them. This process of obtaining information through a measurable physical phenomenon and being able to transform it into a value can be achieved with a data acquisition system (DAQ). The use of this tool is also appropriate for the application in the laboratory of machines and thermal engines of the School of Engineering, as the work carried out there requires the use of this instrument, considering that alternative internal combustion engine tests (AICE) are carried out, research work is carried out in which measurements are taken and the students of the school carry out their practical work in different subjects.

The objective of this project is the realization of a DAQ due to the need to remodel the current system in the laboratory. In addition, it has been manufactured in compliance with the established requirements in terms of versatility, firm structure, and economy. During this work, information regarding the assembly process and the elements that have been used is shown, Arduino being the acquisition hardware of choice. Also, an application has been designed that can control this tool with MATLAB and, finally, the operation of the system has been tested in the laboratory.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

DAQ	Data Acquisition System
MCIA	Motor de combustión interna alternativo
PCI	Peripheral Component Interconnect
ECG	Electrocardiograma
PXI	PCI eXtensions for Instrumentation
RTD	Resistance Temperature Detector
NI	National Instrument
I/O	Input/Output
ADK	Assessment and Deployment Kit
PnP	Plug and play
mV	Milivoltios
V	Voltios
PWM	Pulse Width Modulation (Modulación por ancho de pulsos)
KB	Kilobyte
Hz	Hercio
mA	Miliamperio
IDE	Entorno de Desarrollo Integrado
GUI	Interfaz Gráfica de Usuario
MECH	Motor encendido por chispa
MEC	Motor encendido por compresión
ECU	Unidad de control del motor
DC	Corriente continua
AC	Corriente alterna

1 OBJETO

La necesidad del muestreo de datos en un laboratorio se vuelve compleja cuando el número de señales es elevado, esto es común en laboratorios de investigación o en plantas de potencia que requieren monitorizar constantemente las señales que llegan de los diferentes procesos.

El laboratorio de Máquinas y Motores Térmicos de la Universidad de Sevilla, alojado en la Escuela Técnica Superior de Ingeniería cuenta con 4 celdas de ensayo, de las cuales dos de ellas están diseñadas para el estudio y ensayo de motores de combustión interna alternativos (MCIA) de diferente potencia, las otras dos restantes utilizadas para proyectos de diversa aplicación, pero principalmente enfocados a la energía, también tiene un enfoque didáctico ya que se realizan prácticas para alumnos de los grados universitarios de la escuela. El laboratorio actualmente dispone de un sistema de adquisición de datos (DAQ) de la compañía Dewetron modelo DEWE-2601, este se utiliza para medir parámetros de ensayo de MCIA. En este momento existe una capacidad limitada del número de señales que se puede medir y con visión a futuro se pretenden incorporar otras adicionales. Por ello, se enfoca el objetivo principal de este trabajo al desarrollo de un sistema de adquisición de datos que consiga obtener las señales que sean de utilidad para las aplicaciones que se requieran y poder, tras su lectura, procesarlos y almacenarlos para que puedan ser utilizados como fuente de información para trabajos de investigación, prácticas u otro tipo de usos.

El objeto de este trabajo será, partiendo de las bases teóricas de los DAQ, la descripción del montaje de un sistema que pueda medir una variedad de señales junto con la programación de una aplicación destinada al control del DAQ y, por último, la comprobación de funcionamiento mostrando experimentos realizados en el laboratorio.

2 ALCANCE

El alcance de este proyecto es obtener un DAQ realizado de forma manual en el que se pueda adquirir la información de las señales que se tratan en el laboratorio de forma multidisciplinar, como por ejemplo son las de temperatura, presión, caudalímetros y régimen de giro para obtener señales de un MCIA.

En primera instancia existía la posibilidad de la compra de material comercial con módulos PCI ("Peripheral Component Interconnect") que están directamente conectados a un PC y que incluye un software propio de adquisición de datos. Por otro lado, se realiza una búsqueda intensiva sobre soluciones o alternativas al hardware comercial. Se opta por esta segunda opción ya que al ser un trabajo académico trata de mostrar un sistema de buenas prestaciones desarrollado íntegramente por el alumno para que pueda ser modificado en cualquier momento por otro usuario que necesite realizar algún cambio.

Por tanto, es necesario identificar cada una de las señales y elegir un hardware para medirlas, además, debe estar conectado al ordenador para visualizar la información recogida y que pueda ser registrada con el objeto de hacer históricos y análisis posteriores a la adquisición.

Específicamente se obtendrán como resultados de este trabajo:

- La creación de un sistema mediante un controlador que puede ser modificado en cualquier momento gracias a un montaje realizado a mano y una codificación de fácil uso.
- Acondicionamiento de las señales al DAQ.
- Adaptación para utilizar el sistema únicamente como DAQ o como amplificador de señales de temperatura para otro DAQ comercial.
- Lectura y registro de datos para la visualización, almacenaje, procesado o elaboración de informes de verificación de MCIA.

3 INTRODUCCIÓN

Este capítulo describe el marco teórico sobre los DAQ, la teoría de las señales, los tipos de señales que se pueden obtener según el fenómeno físico que se quiera medir, los diferentes hardware de adquisición de datos que existen y cuál es el utilizado en este trabajo y, por último, el software. Para finalizar, se pondrá en contexto diferentes aplicaciones y en concreto la aplicación de los MCIA.

3.1 Sistemas de adquisición de datos

Los DAQ son dispositivos encargados de leer la información que es transmitida por una serie de elementos, comúnmente sensores, y que es capaz de mostrar en un entorno programable, los fenómenos físicos que son medidos. Estos dispositivos han conseguido mejorar la automatización de recogida de datos ya que anteriormente ésta se realizaba en papel o gráficos de los cuales se recuperaba la información que se obtenía de los fenómenos físicos que se quería medir. El trabajo para medir señales se ha vuelto una tarea más cómoda y esto se puede ver con el uso de herramientas como son los polímetros u osciloscopios, que son capaces de obtener un valor midiendo, por ejemplo, cualquier componente electrónico.

3.1.1 Fundamentos de los DAQ

Los DAQ son necesarios para aplicaciones tan importantes como medir la frecuencia cardíaca de un paciente mediante un electrocardiograma (ECG), máquina EKG [1]. Sus elementos son iguales que los que podemos encontrar en otro tipo de aplicaciones, electrodos que se colocan en el cuerpo como sensores, un electrodo de referencia y un sistema que recoge esta información mostrando resultados que puedan ser interpretados por un profesional.

Esto puede ser extrapolado a la verificación de la normativa aplicable a ensayos de MCIA que hace que este tipo de sistemas sean de un alto interés por parte de los fabricantes o profesionales que realizan este tipo de ensayos.

Como se muestra en la figura 1 y 2, estos sistemas tienen un esquema básico que cualquier DAQ debe cumplir ya que todos tienen en común los elementos que se muestran.

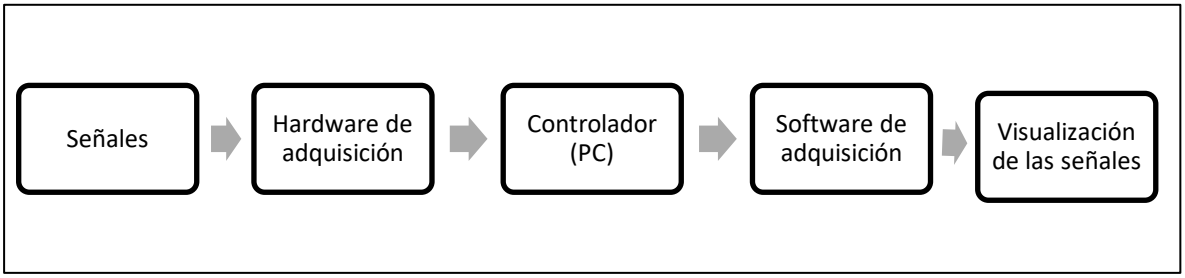


Figura 1. Diagrama de bloques de un DAQ

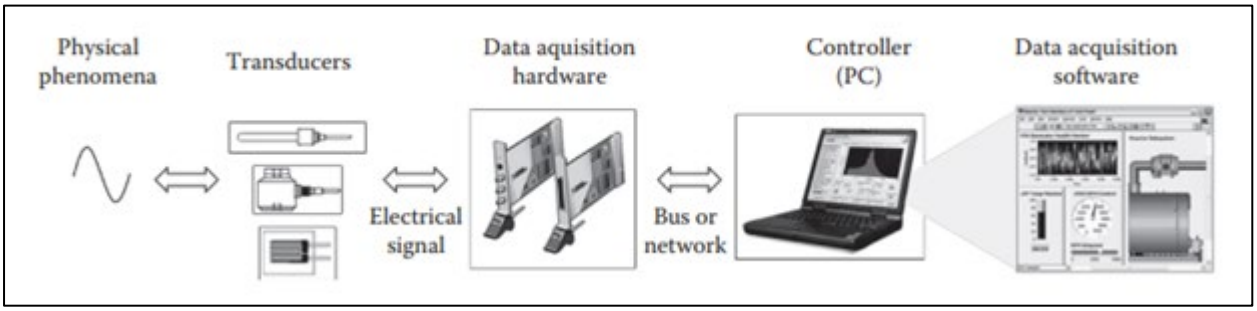


Figura 2. Esquema de un DAQ

Según Mohammed Abdallah [2], existen tres grandes grupos en los que se clasifican los DAQ.

1. Utilizando el ordenador como DAQ, en él se realizan todas las operaciones necesarias tras la toma de los datos, tanto la recepción de las señales, manipulación y visualización. Principalmente se puede instalar de forma interna o externa, normalmente con módulos PCI que son colocados en el propio ordenador. Es importante indicar que la tecnología va constantemente evolucionando y que los elementos del propio equipo quedan obsoletos con el paso de los años, esto conlleva una compra de nuevos equipos o módulos de adquisición de datos.

   Existen muchas empresas que tienen soluciones para este caso, como puedan ser *Dewesoft* o *National Instrument (NI).*

2. El segundo grupo importante es el que se ejecuta mediante microcontroladores, los sistemas basados con este esquema son portables, compactos y normalmente de bajo coste, pero su principal desventaja es encontrarse con un sistema que está capado puesto que no es sencillo realizar cambios en él ya que suele ser un montaje fijo e incluso tras su fabricación no existe la posibilidad de reconfigurarlo.

3. Por último, existe un tipo de hardware que es reconfigurable, llamado FPGA (*Field Programmable Gate Array)*, normalmente instalados en módulos PCI o PXI ("PCI eXtensions for Instrumentation") para instalarlos directamente en PC o en algún lector de módulos PCI. Comercialmente lo distribuye la compañía *NI* que son ejecutadas con los que se mencionan en el primer grupo. Este tipo de montaje es algo más complejo ya que requiere un alto conocimiento de programación de este tipo de hardware, pero tiene como ventaja principal que es totalmente configurable por el usuario para la aplicación que se quiera instalar y existe software para la programación de este tipo de módulos como son *LabView.*

Para poner en contexto la figura que se muestra anteriormente, se pueden definir los diferentes elementos para un correcto entendimiento de cada punto de los DAQ [3].

a) **Fenómeno físico**

b) **Transductores o sensores**

c) **Hardware de adquisición**

d) **Controlador**

3.2 Fenómeno físico y teoría de señales

Este punto describe los fenómenos físicos y la teoría de señales en los que se darán conceptos teóricos para comprender el concepto de señal, cómo se transmite y su modelización matemática. Además, se muestran las medidas de señales más comunes y con qué tipo de elemento de medida se realiza, con esto se desarrollan los puntos a y b que se proponen según Potter [3].

La señal se conceptualiza como una magnitud física que depende de una o más variables independientes, como magnitud física se tiene el voltaje, presión, intensidad lumínica, corriente eléctrica y como variable independiente el tiempo o espacio.

Las señales se clasifican, de forma común, como unidimensionales o multidimensionales y por otro lado las discretas o continuas. Para el primero de los tipos, la señal unidimensional, la magnitud física que se mide depende únicamente de una variable independiente, es decir, la temperatura en función del tiempo, para las multidimensionales, como su nombre indica, depende de varias variables independientes como por ejemplo el espacio y el tiempo. Las señales discretas toman valores de forma finita por lo que el conjunto de valores está definido a lo largo de la variable independiente, normalmente el tiempo. Finalmente, el último tipo son las señales continuas que pueden tomar cualquier valor ya que la variable dependiente es continua. En la figura 3 se muestra la diferencia entre las señales continuas, que muestran una línea suave de los valores en función del tiempo, y señales discretas, en las que los valores van tomando puntos sin continuidad en función del tiempo. [4].

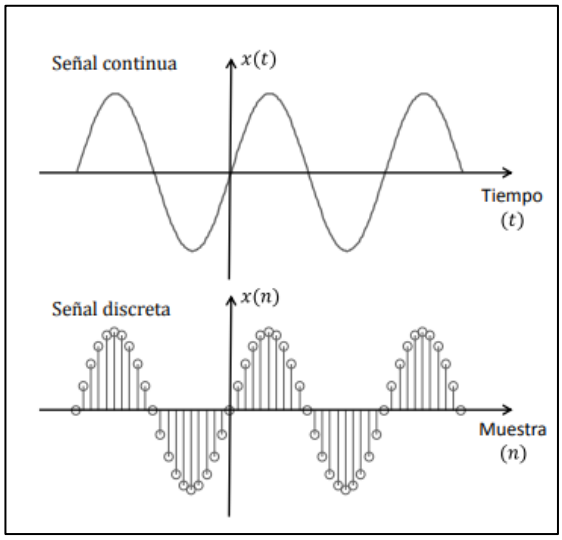


Figura 3. Señal continua y discreta

Existen diferentes tipos de señales elementales que gracias a su combinación producen señales complejas que modelan los fenómenos físicos los cuales queremos medir.

a) Escalón unidad
b) Pulso rectangular
c) Pulso triangular
d) Función rampa
e) Función seno
f) Función sinusoidal compleja

Existen otras señales que se prolongan en el tiempo y generan una señal que se repite de forma periódica, de forma que *x* es el valor dependiente del tiempo, *n* un número entero y *T* el periodo en el que se genera la señal periódica según la ecuación 1.

$$x(t) = x(t + nT) \quad -\infty < t < \infty \qquad (1)$$

En la figura 4 se muestra a la izquierda una señal periódica sinusoidal en función del tiempo, esta señal medida con el periodo T se mide de pico a pico de cada una de sus oscilaciones. Por otro lado, en la derecha se muestra una señal periódica rectangular en función del tiempo en la que se mide su periodo T desde su flanco de subida hasta el siguiente o de bajada hasta el próximo.

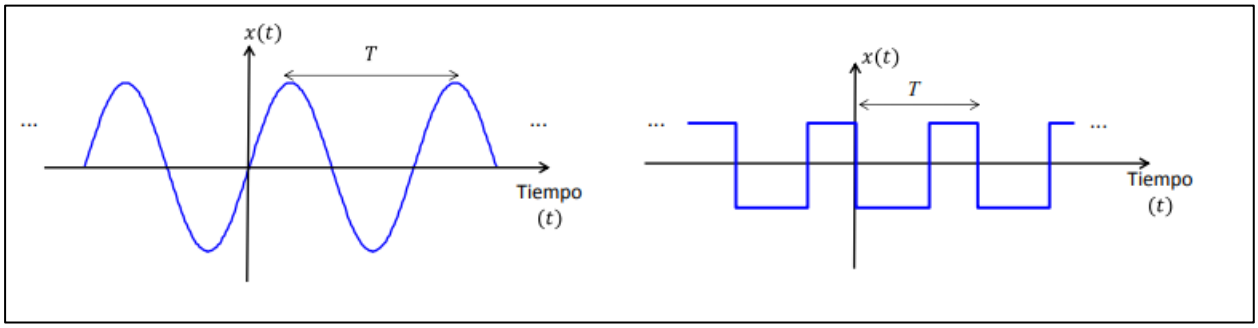


Figura 4. Señal periódica sinusoidal y rectangular

Los fenómenos físicos que se quieren medir se pueden modelar mediante una respuesta ya que existe una excitación que lo produce. O si se quiere ver de otra manera, el fenómeno físico sería la entrada y la salida sería la magnitud que queremos medir, normalmente suele ser una magnitud eléctrica como es el voltaje o la corriente eléctrica. La señal de entrada se transmite hasta el sistema que interpretará la señal, la transformará y será enviada como señal de salida. La relación entre la señal de salida y entrada se puede representar mediante una función matemática de la ecuación 2, siendo $x(t)$ la señal de entrada e $y(t)$ la señal de salida. Al igual que para el caso anterior de un sistema continuo, existen los sistemas discretos que transforman las entradas en salidas de tiempo discreto según $y[n] \rightarrow x[n]$. La figura 5 muestra la representación gráfica de un sistema de señales en los que se muestra la señal de entrada y salida en función del tiempo.

$$y(t) = f[x(t)] \qquad (2)$$

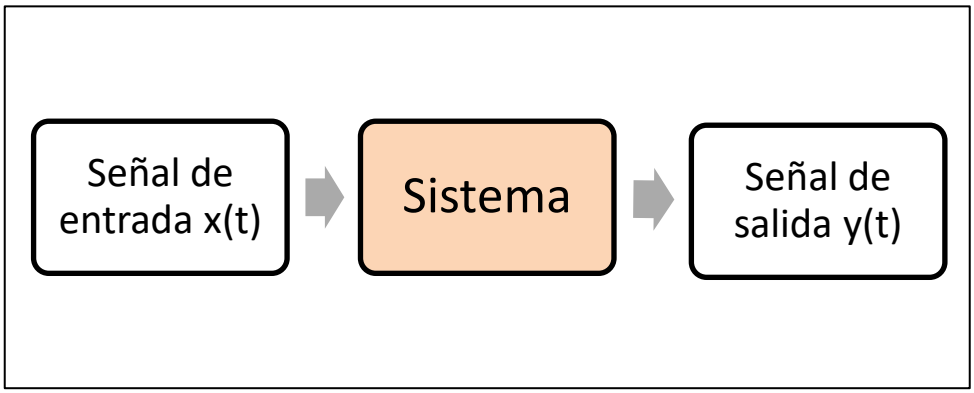


Figura 5. Representación gráfica de un sistema de señales

Los sistemas existentes pueden funcionar de manera combinada produciendo así uno más complejo que opere con las diferentes señales de entrada y salida para obtener el que se desee en su conjunto.

3.2.1 Tipos de señal más comunes

3.2.1.1 Temperatura

La monitorización de la temperatura se puede dar con distintos transductores o sensores, los más comunes son los siguientes:

1. Termopares: El instrumento de medida tiene dos metales diferentes que se unen en una punta final, cuando estos metales se calientan o se enfrían crean un voltaje que se puede correlacionar con la temperatura. Los termopares son la tecnología más utilizada para medir la temperatura, sobre todo en aplicaciones industriales. Tiene diferentes tipos, el más común es el tipo K, que es capaz de medir en un rango de entre -270 ºC hasta los 1400 ºC. En la figura 6 se muestra un ejemplo de termopar en la que dos cables conectan al metal de la unión caliente con la fría para obtener un valor de voltaje que con el cálculo de su correlación se obtiene el valor de la temperatura [5].
    a. Tipo K: Níquel-Cromo + Níquel Aluminio. Temperaturas [-270 ºC a 1372 ºC]
    b. Tipo T: Cobre + Cobre-Níquel. Temperaturas [-270 ºC a 400 ºC]
    c. Tipo J: Hierro + Cobre-Níquel. Temperaturas [-210 ºC a 1200 ºC]
    d. Tipo L: Cobre + Cobre-Níquel. Temperaturas [-270 ºC a 400 ºC]
    e. Tipo N: Níquel-Cromo-Silicio + Níquel-Silicio-Magnesio. Temperaturas [-270 ºC a 1300 ºC]
    f. Tipo R: Platino-Rodio (13%) + Platino. Temperaturas [-50 ºC a 1768 ºC]
    g. Tipo S: Platino-Rodio (10%) + Platino. Temperaturas [-50 ºC a 1768 ºC]
    h. Tipo B: Platino-Rodio (30%) + Platino-Rodio (6%). Temperaturas [0 ºC a 1820 ºC]

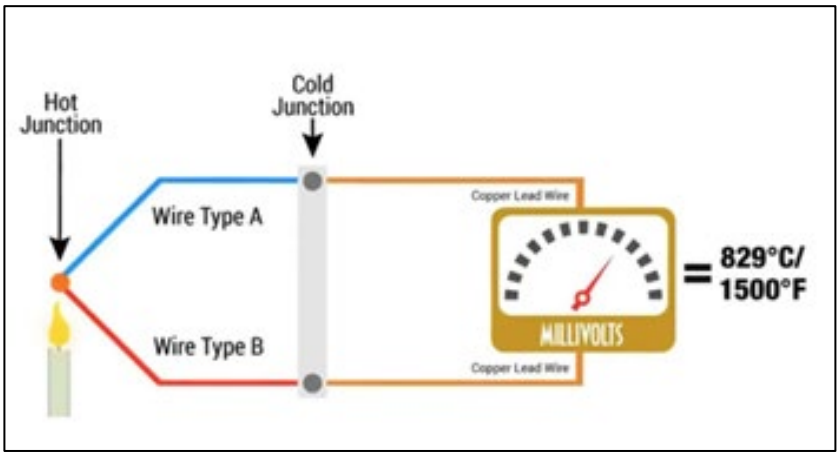


Figura 6. Esquema de funcionamiento de un termopar

2. RTDs (“Resistance Temperature Detector”): Es un metal cuyo valor de resistencia cambia en función de la temperatura [6]. Este dispositivo no muestra una señal como lo hacen los termopares, si no que con otro dispositivo electrónico podemos medir la corriente que pasa por él y calcular su correlación, este elemento de medida puede ser un polímetro o medidor amperimétrico.
3. Termistores: Es un elemento fabricado con un material semiconductor que es sensible a los cambios de temperatura y se puede medir de la misma forma que lo hacen los RTDs, mediante la medida en la resistencia. En comparación con los RTDs en el que la variación de la resistencia se reduce al aumentar

su temperatura, el termistor varía su resistencia reduciéndose según aumenta la temperatura. En la figura 7, la relación con respecto a los termopares y los RTDs, los termistores son completamente opuestos. Los termistores requieres el uso de un coeficiente de temperatura y una medición más precisa de la temperatura la proporciona la ecuación de Steinhart-Hart, que se describe de la siguiente forma. Comúnmente, para una temperatura de 25ºC se obtiene una resistencia de 3 kΩ [7].

$$\alpha(t) = \frac{1}{R(T)}\frac{dR}{dT} \quad (3) \qquad \frac{1}{T} = a + b * Ln(R) + c * Ln^3(R) \quad (4)$$

$$R = e^{\left(x-\frac{y}{2}\right)^{\frac{1}{3}}-\left(x+\frac{y}{2}\right)^{\frac{1}{3}}} \quad (5) \qquad x = \sqrt{\left(\frac{b}{3c}\right)^3 + \frac{y^2}{4}} \quad (6) \qquad y = \frac{a-\frac{1}{T}}{c} \quad (7)$$

$$a = 1.40 * 10^{-3} \;\; b = 2.37 * 10^{-4} \;\; c = 9.90 * 10^{-8} \quad (8)$$

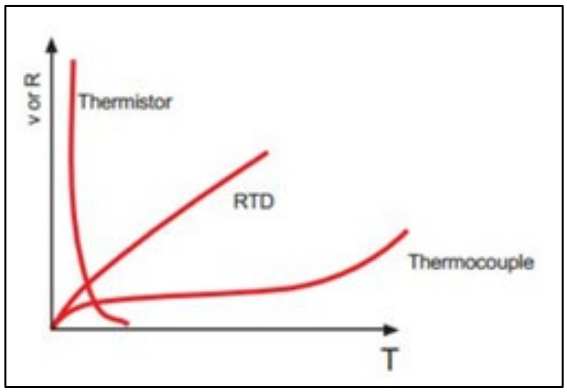


Figura 7. Comparación entre diferentes elementos de medida de temperatura

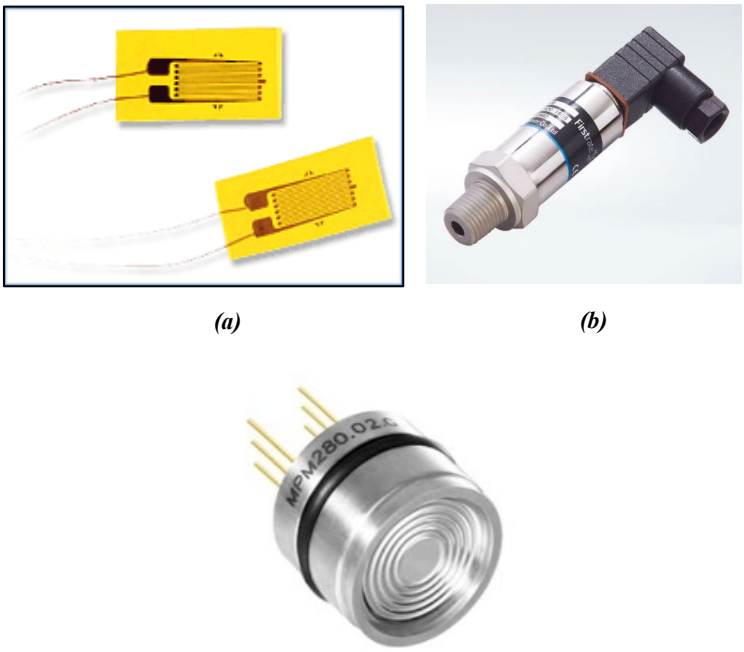
3.2.1.2 Presión

Los sensores de presión miden la fuerza por unidad de área que se ejerce en el punto que se quiera medir. Como principio físico consiste en medir los cambios de capacitancia o de resistencia de una galga o un elemento piezoeléctrico que tiene correlación con la presión que se quiere medir [8]. Según la necesidad de medida que se tenga existe una variedad que ofrecen los diferentes fabricantes, ya sea para medir altas o bajas presiones.

1. **Galgas extensiométricas:** Su medida se basa en la variación de una resistencia cuando es sometida a un esfuerzo mecánico que se traduce como el efecto piezoeléctrico, este transforma una magnitud física en una magnitud eléctrica [9]. Son objetos sometidos a una tensión y estrés determinados a los que se le aplica una fuerza externa [10]. El elemento de medición suele ser una membrana o una lámina metálica.
2. **Sensores de presión capactivos:** Este tipo de sensores están construidos con una base parecida a la de un condensador, en la punta tiene una placa metálica que está conectada al interior del circuito en la que tiene un oscilador que crea un campo electrostático. El funcionamiento es simple, al objeto en el que se encuentre el fenómeno físico que se quiere medir se le acerca el sensor capacitivo y este produce un enfrentamiento de las oscilaciones que aumenta hasta que se active la salida [11].
3. **Sensor de presión piezorresistivo:** Un elemento piezorresistivo produce una variación de una resistencia eléctrica a causa de una variación mecánica, por lo que los sensores de presión piezorresistivos tiene el mismo modo de funcionamiento que tendría un material piezorresistivo, cambia

su resistencia eléctrica según el cambio mecánico que se esté produciendo, en este caso un cambio de fuerza por unidad de área como es la presión. Este tipo de sensores suelen ser membranas, con los que podemos medir la presión ejercida debido a la deformación de esta.

En la figura 8 se muestra una galga extensiométrica en la que está el hilo activo junto con dos terminales de conexión, todo ello, en una base flexible. Los otros dos sensores que se muestran corresponden a uno capacitivo y otro piezorresistivo, estos dos últimos se muestran como son vendidos comercialmente [12]–[14].



(a) *(b)*

(c)

Figura 8. (a) Galga extensiométrica (b) Sensor de presión capacitivo (c) Sensor piezorresistivo

3.2.1.3 Caudal

El caudal o gasto de un fluido se mide a través del caudalímetro para determinar la cantidad de masa o volumen que circula por unidad de tiempo. Este instrumento puede ser de dos tipos, mecánico o eléctrico, y esto influye en su construcción ya que existen caudalímetros que miden de forma directa o de forma indirecta. La primera de ellas mediante desplazamiento positivo y el segundo tipo puede ser por presión diferencial, velocidad, variación del área.

Según García Gutiérrez [15], existen diferentes tipos de caudalímetros que podemos encontrar en el mercado:

1. **Medida de presión diferencial**
    a. Placas de orificio
    b. Toberas
    c. Tubos Venturi
    d. Tubos Pitot
    e. Tubos Annubar
    f. Codos

g. Medidores de área variable

h. Medidores de placa

Este tipo de medidores de caudal suelen ser los más comunes y utilizados en diferentes aplicaciones y el elemento más utilizado es el de placas de orificio ya que la medida de presión diferencial es más sencilla que en los otros tipos.

2. Medidores de accionamiento mecánico

a. Medidores de desplazamiento positivo

b. Medidor de pistón oscilante

c. Medidor de paletas deslizantes

d. Medidor de engranajes

La mayoría de los medidores accionados mecánicamente suelen ser utilizados para caudales de líquido.

3.2.1.4 Régimen de giro

Estos sensores recogen la señal de régimen de giro de un motor o de cualquier otra máquina rotativa. Existen dos tipos que muestran la señal de forma diferente, el primero de ellos adquiere la señal mediante el Efecto *Hall* y el segundo con el efecto inductivo o pasivo. Ambos son utilizados para medir el régimen de giro, es decir, las vueltas por minuto que genera una máquina cuando realiza un giro rotativo. La principal diferencia entre ambos es la alimentación del sensor, en el Efecto Hall se requiere alimentación de una tensión determinada, mientras que, el de efecto inductivo no lo precisa.

El sensor por efecto Hall, muestra una salida de señal cuadrada de entre, normalmente, 0 a 5V o en algunas ocasiones de hasta 12 V. Esto se consigue al crearse un pulso al paso de una rueda dentada, como pueda ser un volante de inercia, en la que el dispositivo de medida contiene un imán que crea un campo magnético que hace crear un voltaje. La figura 9 muestra cuál es la forma de señal que crea un sensor de régimen de giro por efecto Hall en la que al paso de un diente de la rueda se produce un escalón con un valor aproximado con el que se alimenta el sensor y, también, se puede observar el esquema de posicionamiento de este tipo de sensores.

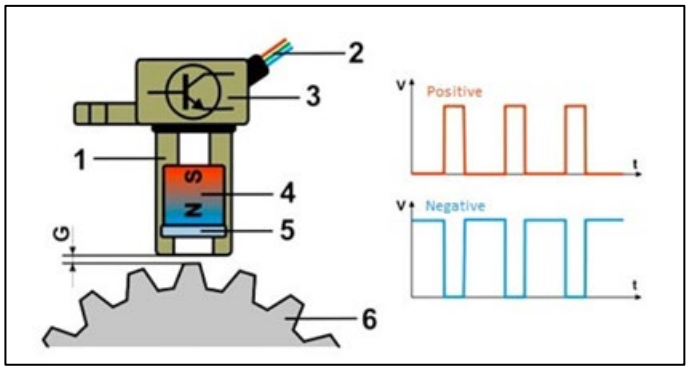


Figura 9. Señal de régimen de giro tomada por efecto Hall

Por otro lado, tenemos la señal que se adquiere mediante el efecto inductivo, esta se crea cuando el diente pasa por el sensor y es producida a causa de un cambio de campo magnético, pero al contrario que el caso anterior crea una señal en forma senoidal y según la frecuencia a la que está la onda se puede obtener el régimen de giro en revoluciones por minuto. En la figura 10, se muestra la señal generada por el sensor de efecto inductivo descrito anteriormente y un esquema del sensor.

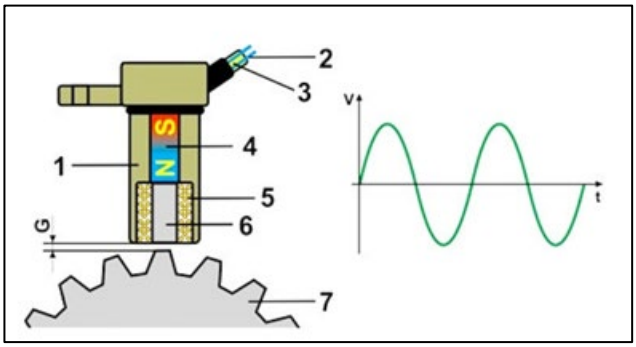


Figura 10. Señal de régimen de giro tomada por efecto inductivo

Las condiciones más importantes para este tipo de equipos de medida de régimen de giro es que requiere una distancia muy pequeña entre el sensor y el diente. Esto es debido a que existe una distancia de sensado [16], provocando que la señal de salida se obtenga con un rango de amplitud distinto al que fuera la medida real. Esto depende del material y del tamaño de objeto a detectar como se representa en la figura 11.

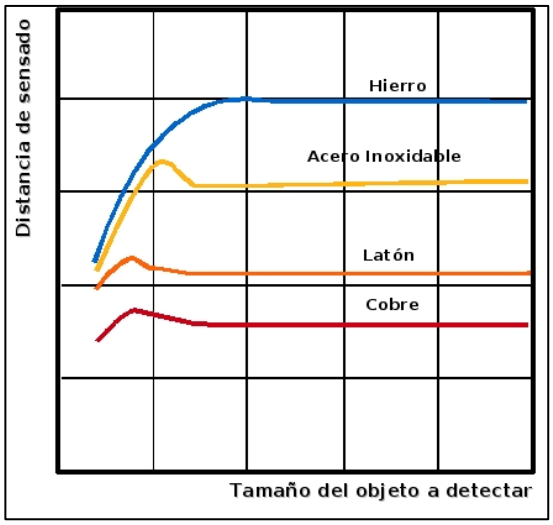


Figura 11. Distancia de sensado para efecto inductivo

3.2.1.5 Par

El par es el momento ejercido en un eje de transmisión de potencia, es decir, es la fuerza que tiende a ejercer haciendo girar el objeto conectado a un eje. El par se mide en Newton por metro ($N \cdot m$) y es proporcional al calculo de la potencia según la velocidad angular del eje y del propio par, siguiendo la ecuación 9, siendo la potencia (P) medida en W, el par (M) en Nm y la velocidad angular (ω) en radianes por segundo.

$$P = M \cdot \omega \qquad (9)$$

Esta magnitud se suele medir para obtener curvas características de los MCIA que comparan el par ejercido con respecto a la potencia obtenida por el sistema. Por otro lado, también son utilizadas para medir el par en turbinas de gas, aunque también el par puede ser medido para realizar cálculos mecánicos sobre otro tipo de ejes rotativos. Es interesante conocer este tipo de magnitud, pues está muy relacionada con el ámbito al que se puede destinar este trabajo teniendo en cuenta que pretende ser una herramienta capaz de leer distintas magnitudes. Es importante recalcar que, el voltaje es la magnitud adecuada para medir señales, por ello, se ha preparado el

equipo atendiendo a esta característica. De esta forma a la hora de calcular el par obtenido, se requiere una conversión en la señal y amplificarla para poder ser leída por el sistema.

3.2.1.6 Aceleración

La aceleración lineal o angular se mide con un dispositivo llamado acelerómetro, este es capaz de determinar la fuerza de aceleración en unidad g en diferentes planos del espacio, plano X, Y o Z. Si no se perturba con una aceleración externa, este debe medir la fuerza gravitatoria puesto que sería la única ejercida en el elemento a medir.

Existen distintos tipos de acelerómetros, entre ellos están los capacitivos, piezorresistivos y piezoeléctricos. En primer lugar, el efecto capacitivo funciona colocando en el dispositivo el peso montado en un resorte o muelle que está anclado a un condensador, el otro extremo está unido al peso que en el que es montado, esto hace mover el resorte provocando una diferencia de distancia entre el condensador y la masa. En la figura 12, se muestra este fenómeno de movimiento del resorte debido a la aceleración de la masa suspendida [17]

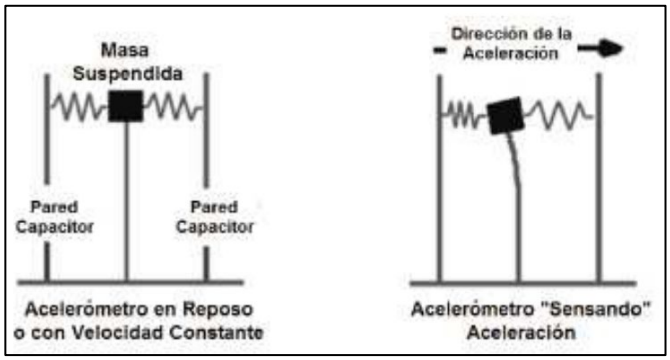


Figura 12. Esquema de funcionamiento de un acelerómetro capacitivo

En segundo lugar, los acelerómetros piezorresistivos utilizan un sustrato, este elemento genera una variación en la resistencia debido a los cambios de aceleración y al igual que otros sensores de esta categoría son basados en el puente de Whetstone. Sin embargo, los acelerómetros piezoeléctricos no tienen el elemento llamado sustrato, en su lugar se fabrican usando un cristal piezoeléctrico. Este cristal debido a la deformación en su estructura cristalina produce una corriente eléctrica, este cambio se produce por la perturbación de una aceleración en el dispositivo. La magnitud eléctrica que se mide es una corriente eléctrica, aunque gracias a los convertidores es posible transformarlas en magnitud de tensión que es más sencilla de manipular en los DAQ. La cantidad de carga que es generada es proporcional al esfuerzo mecánico al que el acelerómetro se somete, en la figura 13 es posible entender la dependencia de la fuerza frente a la carga eléctrica [18].

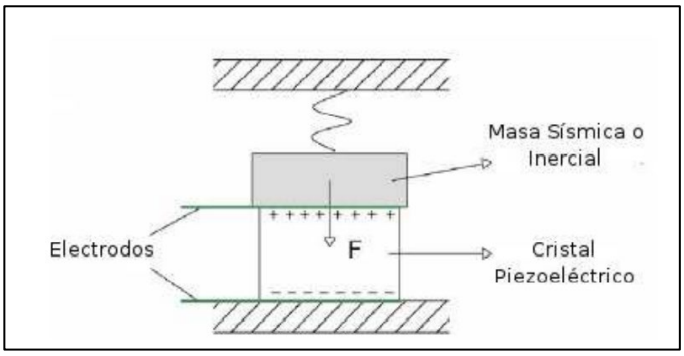


Figura 13. Esquema de funcionamiento de un acelerómetro piezoeléctrico.

3.3 Hardware de adquisición

Los sistemas de adquisición de datos requieren un hardware de adquisición de las señales procedentes de los sensores. Sus funciones principales son:

1. Recepción: La señal que envían los dispositivos de medida suele ser en voltios. Es recibida por el hardware de adquisición de datos que ha de ser capaz de leerla y traducirla para que el ordenador pueda mostrar su valor.
2. Procesar: Tras la llegada de la señal desde el elemento de medida, el hardware tiene que encargarse de transformar el valor de una magnitud a bits o información que pueda ser leída por el ordenador en el que este el DAQ instalado.

Por tanto, el hardware de adquisición de datos es el encargado principal de convertir la señal analógica a una señal digital que pueda ser representada por un ordenador. Tal como explica el libro "Data Acquisition Handbook" [19], un sistema ideal de adquisición de datos tendría un único canal por cada convertidor de señales o hardware de adquisición de datos, con esto conseguiríamos que la señal sea adquirida y acondicionada sin que existan interferencias entre otras señales, pero esto no es lo que ocurre en la realidad.

Los DAQ cuentan con una gran cantidad de señales, esto produce la necesidad de que exista un dispositivo o función electrónica que realice la función de lectura de señales de forma simultánea. La solución más idónea es el uso de multiplexores, estos son un conjunto de interruptores electromecánicos de estado sólido o relés electromecánicos conectados a varios canales en los que va pasando de uno en uno por cada canal.

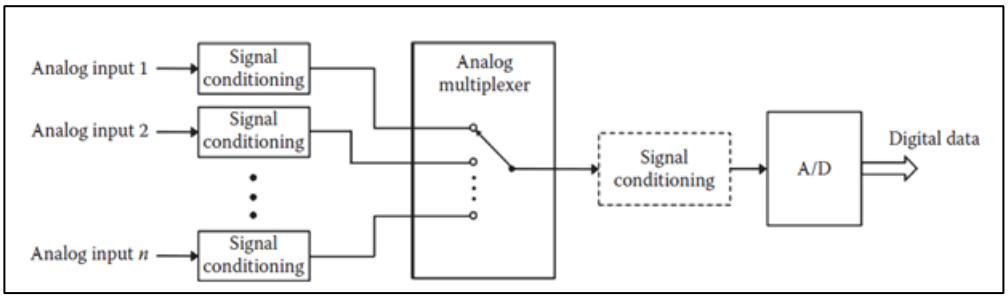


Figura 14. Funcionamiento de un Multiplexor

Esta es la forma teórica de adquirir los datos y son implementados en multitud de sistemas comerciales, las principales marcas suelen distribuir módulos o sistemas completos para la adquisición de datos, pero lo más común es el uso de sistemas portables que se puedan instalar en ordenadores para que no queden obsoletos. Las marcas más importantes de estos sistemas de hardware de adquisición son:

1. NI: Cuenta con productos muy conocidos, en especial su software *LabView*, son los principales vendedores de sistemas de adquisición de datos en todo el mundo. Cuenta con una gran variedad de hardware de adquisición e incluso como se menciona en apartados posteriores cuenta con conexiones propias como son el PXI. En la figura 15, se muestra un módulo de entrada de voltaje que irá conectado a otro hardware como pueda ser el CompactDAQ de la compañía.

Figura 15. Módulos de entrada de voltaje y CompactDAQ de NI

2. Dewesoft: Es otra de las compañías más importantes de sistemas de adquisición de datos, con más de 20 años en el mercado tiene productos muy específicos para las funciones de adquisición. Al igual que la compañía NI, cuenta con módulos específicos de medidas y sistemas completos y portátiles para el uso de campo. La compañía también cuenta con su propio software de adquisición de datos llamado *DewesoftX.*

Figura 16. Modelo R1DB y R2DB de Dewesoft

La elección del hardware de adquisición de datos es muy relevante en este proyecto ya que es el encargado de tomar las señales y transformarlas para mostrarlas en el software, por ello, en este trabajo, que se cuyo pilar es el desarrollo de un DAQ económico, versátil y sencillo, se elige Arduino como principal hardware para la recogida de señales.

3.3.1 Arduino

Arduino es un hardware libre de código abierto, montado en una placa que contiene una cantidad de entradas y salidas I/O ("input/output") para la conexión de pines. La programación propia del microcontrolador es sencilla y existe una gran cantidad de información en la que los usuarios se pueden basar para realizar sus propios proyectos [20].

Arduino comienza a desarrollarse en Ivrea (Italia) en el año 2005, donde varios investigadores entre ellos Massimo Banzi y David Cuartielles desarrollaron un prototipo de hardware y software libre con el fin de que lo pudieran utilizar con los alumnos de una manera económica y sin restricciones de programación. El éxito de este proyecto los llevó a colaborar con Google en el desarrollo del ADK ("Assessment and Deployment Kit") [21] para poder comunicar la placa con Android. En sus inicios, los primeros microprocesadores de Arduino fueron de Atmel, que, aunque no son hardware libre el fabricante publicó las librerías para que pudiera ser modificado por cualquier usuario [22].

Entre los objetivos marcados se encontraban: Establecer un precio económico, que se ensamblara en una placa azul, que fuera Plug&Play (PnP), es decir, conectar y funcionar, y que fuera compatible con diferentes sistemas operativos [23].

Arduino fue muy aceptado, en el año 2011 [24], el fundador Massimo Banzi realizaba una estimación de unos 250.000 Arduino vendidos, pero en el año 2021, según la noticia publicada por "Control Design" [25] los datos que manejaba la compañía giraban en torno a más de 10 millones de placas Arduino UNO. Esto nos lleva a pensar que gracias a un proyecto que nació de la necesidad de crear un hardware de forma libre con el objetivo de que los usuarios puedan crear sus proyectos sin que influya el tema económico.

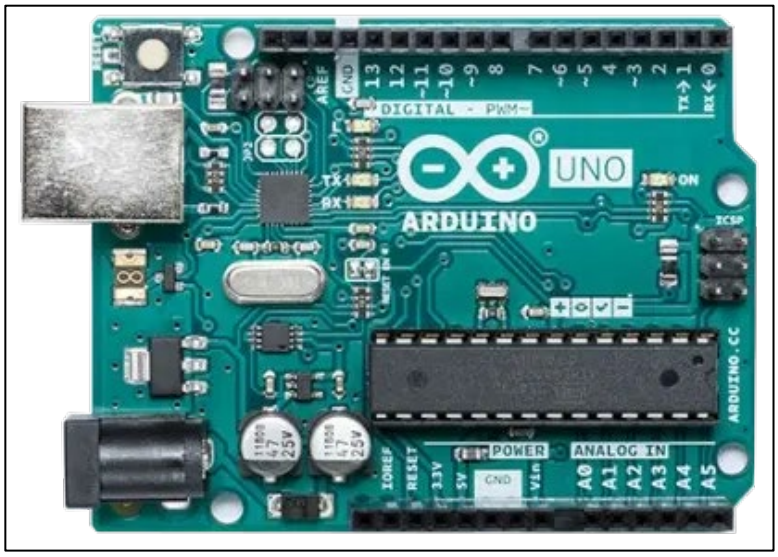


Figura 17. Arduino UNO

La compañía Arduino apostó por lanzar diferentes modelos de placa para que se pudieran adaptar a las necesidades de cada proyecto ya que existen proyectos de mayor envergadura y otros que requieren placas pequeñas para poder adaptarlas al espacio que se les facilita.

Comenzando con el modelo más pequeño, el Arduino Nano, con una superficie de 8 cm^2, cuenta con 14 pines digitales de los que 6 son PWM ("Pulse Width Modulation") y 8 analógicos, en este caso han de ir soldados, cuenta con un microcontrolador ATMega 328P con una velocidad de reloj de 16 MHz. El segundo modelo siendo el más vendido y conocido en el mercado de Arduino es el UNO, tras la salida de su versión R3 cuenta con microcontrolador ATMega 328P, al igual que su modelo más pequeño el Nano, con 14 pines digitales de los que 6 son PWM y 6 pines analógicos, dos menos que el Nano. La versión más grande es el Arduino Mega 2560 con una superficie aproximada de 54,1 cm2, cuenta con un microcontrolador ATMega 2560 con una velocidad de reloj de 16 MHz, 54 pines digitales de los cuales 15 son PWM y 16 pines analógicos. Existen otros modelos como son el Leonardo o el 101 que son variaciones de los modelos principales que se han comentado anteriormente.

Para este trabajo se utilizará como modelo de Arduino el MEGA2560, este tiene las siguientes características:

Tabla 1. Características Arduino MEGA 2560

Microcontrolador	ATmega2560
Voltaje de operación	5 V

Voltaje de entrada (Recomendado)	7-12 V
Voltaje de entrada (Límite)	6-20 V
I/O pines digitales	54 (14 PWM)
I/O pines analógicos	16
Corriente DC por cada pin I/O	40 mA
Corriente DC 3.3 V por pin	50 mA
Memoria flash	256 KB
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 Hz

3.3.2 Amplificador de señal de temperatura

Debido a que Arduino lee las señales en voltaje, se requiere una conversión para adaptarlas a tensión. Muchos de los elementos de medición ya transmiten la señal en voltios (V), pero los termopares dan una medida en milivoltios (mV), debido a esto, se requiere un amplificador para que la señal leída por Arduino sea del orden de magnitud óptimo. Para ello, en este trabajo se utilizará el amplificador AD8495.

El amplificador AD8495, será el encargado de realizar la compensación de la unión fría para los termopares tipo K que se van a monitorizar. La compañía que distribuye el amplificador es *Adafruit*, pero el circuito integrado es de *Analog Devices.*

La compensación de la unión fría se realiza para añadir o disminuir voltaje de la salida del termopar y así llegue a una temperatura aproximada de 0 ºC [26]. Al amplificarse la señal que leemos del AD8495, requerimos de la ecuación 10 para convertir la tensión en voltios a temperatura en grados Celsius. En la ecuación 10 tenemos V_{out} como voltaje de salida del módulo AD8495 y V_{ref} como voltaje de referencia. Este último según la hoja de especificaciones del fabricante equivale a 1,25 V, pero al medirse con un polímetro se obtiene un valor aproximado de 1,24 V por lo que el cálculo se realizará con esa referencia.

$$T\ (ºC) = \frac{V_{out} - V_{ref}}{0{,}005\,\frac{V}{ºC}} \qquad (10)$$

Aunque la hoja del fabricante nos facilita la ecuación anterior [27], existe una más exacta también obtenida del mismo documento donde aparece un término de voltaje de error para logar que a 1,25 mV se obtengan 25 ºC. Añadir a esto que el término que aparece en el denominador es debido a la función de transferencia que se ha calculado para este módulo medido en Voltios por grado Celsius.

$$T(ºC) = \frac{V_{out} - V_{ref} - V_{offset}}{0{,}005\,\frac{V}{ºC}} \qquad (11)$$

El amplificador tiene un esquema sencillo para la conexión de sus terminales como se puede ver en la figura 18. En el podemos ver dos terminales en los que se conectan los dos cables que vienen del termopar, positivo y negativo, en la parte derecha de la placa tendremos cuatro terminales, por orden desde la parte superior hasta la

inferior tenemos; Marcado como $V+$ y GND la conexión de una fuente de alimentación de corriente continua que suministre 5 V. Los otros dos, son los terminales de salida en los que se conectará hasta el elemento de medida, en este caso Arduino, donde se medirá V_{out} para realizar el cálculo y obtener la temperatura, estos dos marcados como $OUT\ y\ GND$. En el capítulo 5 de montaje se especifica que cables son los que se conectan a este amplificador desde el termopar hasta Arduino

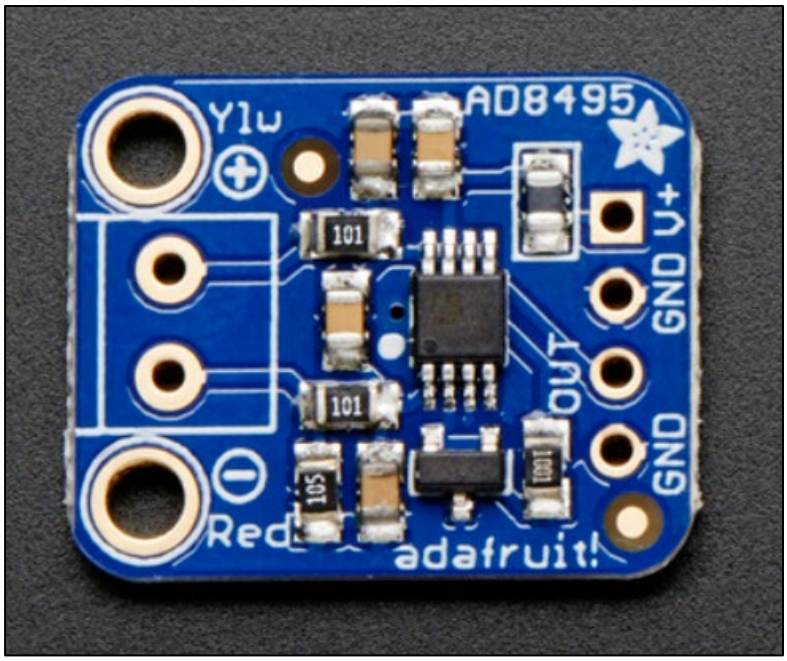


Figura 18. Amplificador AD8495

3.4 Controlador y software

La comunicación entre el hardware y el controlador es un pilar fundamental para un DAQ. En virtud de ello, cuando se habla del hardware del DAQ se menciona la conexión con el ordenador considerando que este último, será el principal director que controlará el sistema por completo desde la conexión con el hardware, el inicio de las medidas y el guardado de los datos.

La forma más común de comunicación es mediante puerto serial, estos puertos son interfaces de comunicación de datos que conectan ordenadores con los llamados periféricos, que son elementos conectados al ordenador de manera externa. La particularidad de este tipo de comunicación es que el puerto serial transmite la información con un solo bit a la vez. Este tipo de puerto serial trabaja con el protocolo RS-232 y transmiten según tres tipos de comunicación en serie, Simplex, Dúplex y Semi-Duplex. Los puertos series alcanzan velocidades desde 75 hasta 256000 bit/s, aunque las más comunes son de 2400 bit/s, 9600 bit/s, 115200 bit/s para los DAQ.

El puerto serial es leído por algún software que controla la adquisición de datos, como se menciona en el apartado de hardware de adquisición, una de las marcas comerciales más importantes como NI tienen su propio software y además este en concreto, *LabView*, es uno de los más utilizados en el panorama ingenieril ya que es posible crear interfaces para la monitorización de señales y es posible crear mediante lenguajes de programación ampliar los procedimientos estándar que se puede realizar en el programa. Está basado en un conjunto de diagramas de bloques que gracias a la incorporación de las señales que se recogen mediante los diferentes protocolos de comunicación se puede programar de forma sencilla o más avanzada para obtener el sistema que se quiere. Tras

esto también es posible guardar resultados en diferentes formatos para tratar los datos a posterior, esto hace una característica esencial para que se convierta en un software ejemplar para los DAQ.

3.4.1 MATLAB como software DAQ

MATLAB con las siglas procedentes de Matrix Laboratory, es un sistema de computación numérica de lenguaje de alto nivel propio y tiene un denominado IDE (Entorno de Desarrollo Integrado). La principal función de este software es el cálculo numérico mediante matrices, algoritmos, etc, pero existe la opción de la comunicación con dispositivos hardware, permitiendo así el desarrollo de aplicaciones multidisciplinares. Adicionalmente existen herramientas que se pueden incorporar a MATLAB como son Simulink, que es un entorno de programación visual en el que se puede modelizar sistemas que posteriormente se simularan, enfocado a sistemas de control. Por otro lado, tenemos los llamados GUI (Interfaz gráfica de usuario), que son aplicaciones creadas para facilitar el uso de la programación mediante funcionalidades como son botones, gráficos interactivos y otros similares.

La particularidad de MATLAB, en concreto para la aplicación que se quiere desarrollar, es el uso de cajas de herramientas, conocido con el término inglés "*Toolbox"* donde, programadores ya sean de la compañía o terceros, crean funciones adicionales al funcionamiento de MATLAB y permite añadir nuevas funcionalidades al software nativo.

Para este trabajo, ya que se va a utilizar Arduino como hardware de adquisición de datos, se requiere uno de los *Toolbox* que ofrece MATLAB, *support package for Arduino Hardware* [28], para poder realizar una comunicación entre el dispositivo Arduino y el software. Arduino al establecer comunicación con el computador lo hace mediante puerto serial, sin embargo, el paquete de soporte que ofrece MATLAB se basa en la comunicación con la placa mediante un programa ejecutado en ella misma y hace llegar mediante un protocolo de entrada y salida de información que haya sido programada.

Para comprender como es el funcionamiento de este paquete de soporte de Arduino, existen funciones preprogramadas en el paquete de herramientas que se utilizan para realizar la comunicación.

- *arduino(port, board).* Con esta función realizaremos la conexión mediante un puerto de comunicación serial creada por la conexión de USB. El argumento de entrada *port* será el creado por el PC, como por ejemplo el *'COM5'*, el siguiente argumento es *board* en el que se especifica el tipo de placa Arduino utilizada y que sea soportada por el paquete, por ejemplo *'Uno'*. Existen otros argumentos como la dirección de Bluetooth, pero no es objeto de este trabajo ya que la comunicación se crea mediante USB.
- *readVoltage(a, pin).* Esta función es utilizada para medir el voltaje entre un pin, analógico, y la tierra que se conectan a la placa arduino. El primer argumento *a* será el que indique la conexión a la que se haya llamado a arduino con la función anterior y el argumento *pin* será el pin referido a la conexión que estemos realizando, por ejemplo *'A2'*. Cabe destacar que según el modelo de placa de Arduino se nombrará de una forma, pero el estándar es desde *A0 hasta A7* y si es el modelo MEGA2560 que se constituye de más pines será entre el *A0 y A15*.
- *writeDigitalPin(a, pin, value).* Escribe un valor en un pin digital al que esté conectado en Arduino. Esta función está pensada principalmente para encender o apagar algún elemento que se conecte, como por ejemplo un LED, ya que el argumento *value* toma valores o 0 o 1, es decir, *true* o *false*.

Recapitulando, este paquete permite la comunicación entre MATLAB y las placas Arduino de diferentes modelos y es posible leer, escribir señales a sensores que están conectados a la placa, con este paquete nos ofrece una posibilidad de poder realizar una programación más sencilla ya que no requiere utilizar el lenguaje creado por Arduino si no el de MATLAB.

3.4.2 *Appdesigner* de MATLAB

Gracias a la conexión que existe entre el hardware de Arduino con el software MATLAB, es posible crear un sinfín de aplicaciones que puedan ser utilizadas por un usuario para distintas aplicaciones. En este trabajo se pretende ir un paso más allá de la programación entre ambos, para ello se realizará una aplicación con la herramienta denominada *Appdesigner* de MATLAB. Con ello se pretende crear una GUI, en la que sea posible crear de forma visual y con programación basada en MATLAB, un sistema de adquisición de datos que conecte Arduino con un ordenador mediante el software MATLAB.

Appdesigner es un entorno interactivo en el que se puede crear una infinidad de aplicaciones con componentes que se pueden programar de forma sencilla, esta incluye, gráficos, imágenes, cuadros de texto y numéricos y otros muchos. Los componentes codificados mediante los llamados *Callbacks*, son funciones programadas para realizar cualquier tipo de operación en el que se puede realizar una entrada y salida de datos, ya sea para mostrar o introducir algún dato, es denominado como I/O.

La librería de componentes con los que se pueden ir creando la aplicación requiere del conocimiento de algunos códigos para su programación, este se basa en el lenguaje de programación de MATLAB, por lo que la escritura del código es sencilla. La librería incluye diferentes componentes, a continuación, se muestran denominados al igual que en el programa, los más importantes y los que han sido utilizados para personalizar la interfaz.

- *Axes*: Son componentes para la creación de gráficos, estos requieren de variables que sean introducidas como datos con la finalidad de crear cualquier tipo de gráfico que sea compatible con MATLAB. El DAQ requiere la muestra de gráficos en vivo para mostrar las señales que se miden y otros que muestren históricos del experimento.
- *Button:* Los botones son programables que responden cuando el usuario presiona y suelta. Cambia un valor de una propiedad, ya sea para modificar la apariencia, iniciar un proceso u otra acción a realizar.
- *Check box:* Las casillas de selección son utilizados para marcar una opción o preferencia. Se diferencian de los botones por quedarse activos cuando están marcados e inactivos en caso contrario, de esta manera se facilita la elección de opciones en el programa.
- *Edit field (text y number):* Se trata de cuadros de texto programables y existen dos tipos diferentes, pero ambos son utilizados para leer o escribir datos numéricos o caracteres de texto, según el tipo que se elija. Para el caso particular del DAQ se utilizará para mostrar los datos en directo de valores de las señales, temperatura, presión, etc.
- *Menu bar:* Este componente crea una barra de herramientas en la parte superior de la interfaz, esta se puede programar organizando diferentes funciones de acceso rápido.
- *Tab group:* Crea un grupo de pestañas en la interfaz para poder ordenar los componentes en diferentes espacios. Con ello, podremos organizar diferentes visualizaciones de la aplicación como por ejemplo pestañas solo con gráficos, con datos o combinando ambos.

La figura 19 muestra un ejemplo de aplicación creada con *Appdesigner*, en él se pueden ver diferentes componentes como son *Axes*, *Button,* para determinar un histograma de imágenes.

Esta herramienta ofrece muchas ventajas para la creación de una interfaz gráfica con la que modelar el DAQ, entre ellas:

- Vinculación entre Arduino y MATLAB mediante el paquete de soporte.
- Creación de una interfaz similar a un SCADA
- Tratamiento y almacenamiento de datos para futuro análisis en el propio software
- Versatilidad de programación

- Facilidad de uso para el usuario final

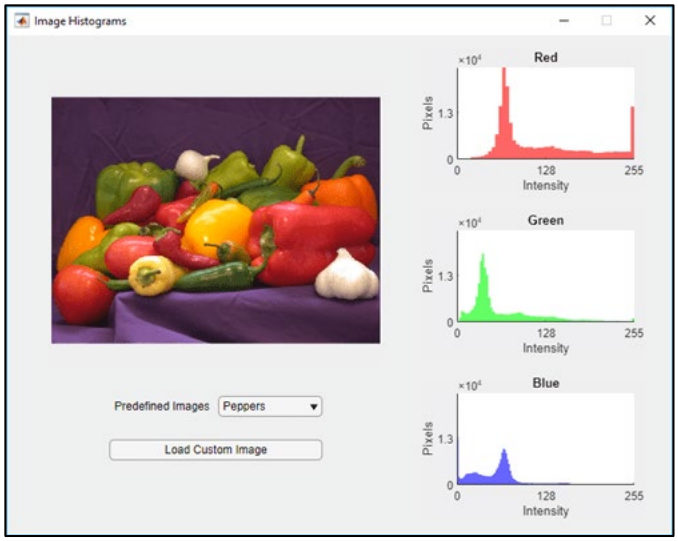


Figura 19. Ejemplo de aplicación diseñada en Appdesigner

4 APLICACIONES DE LOS SISTEMAS DE ADQUISICIÓN DE DATOS

El capítulo cuatro describe las diferentes aplicaciones en las que se frecuenta el uso de los DAQ, se detallan algunas en las que tienen más importancia y se analizan trabajos similares al que se está desarrollando para comparar el estado del arte de los DAQ utilizando el hardware Arduino. Por último, se describe la aplicación específica de los DAQ para MCIA en los que se describirá con más profundidad cuáles son las señales requeridas para un ensayo y cuáles son los instrumentos de medida utilizados para ello.

4.1 Aplicaciones generales de los sistemas de adquisición de datos

Se tiende a pensar que los DAQ solo se utilizan en aplicaciones industriales, pero eso no es del todo cierto. Con el objeto de entender mejor el uso que se le puede dar a un DAQ en los diferentes ámbitos de trabajo, se recapitulan los diferentes usos que se le puede dar a estos sistemas y como de importante son en cada uno de los sectores [29].

1. Energía
    a. Solar: El ángulo en el que se posicionan los paneles solares son esenciales para aumentar la captación de la radiación solar, este ángulo se ha de medir que los motores encargados de la rotación de los ejes en los que se apoyan los módulos fotovoltaicos puedan corregir su posición. Por otro lado, también es posible la medición de la radiación solar con otros dispositivos como son piranómetros, que son capaces de captar y transmitir la señal de radiación para que puedan ser leídos por equipos informáticos. Con estos datos suelen realizarse bases de datos y mapas de radiación en los diferentes países con objeto de realizar estimaciones de cálculos de potencia fotovoltaica u otro interés.
    b. Plantas de potencia: En las grandes plantas de potencia podemos encontrar cientos o miles de sensores encargados de controlar el estado de la propia planta, esto acarrea un problema para los operarios ya que son los que tienen que supervisar constantemente los parámetros de la planta. Gracias a los DAQ la tarea de realizar un control de la planta se vuelve más sencillo ya que todas las señales que se adquieren son traducidas en los sistemas de adquisición de datos de la planta que se encargará de visualizar a los operarios e ingenieros todos los parámetros que se requieran para poder tener el control operativo de la planta en todo momento.
    c. Máquinas térmicas: De cualquiera de las máquinas térmicas existentes, MCIA, turbinas de gas o de vapor, se necesita conocer los valores de los parámetros que se suelen medir en este tipo de tecnologías como son la presión, temperatura, caudales. La investigación de estos equipos también requiere de los DAQ para poder conocer todo lo posible de ellos y así tener caracterizada la máquina térmica.
2. Médicas: Los ECG, como explica Young en su estudio [1], son una de las aplicaciones principales de un DAQ, pero hay otras como son los ultrasonidos o incluso el uso de cámaras endoscópicas que también son consideradas como DAQ, pero en formato de imagen.
3. Otras aplicaciones como puedan ser la geolocalización, los radares de proximidad, medición de conexión a internet, contador de partículas Alpha para aplicaciones nucleares son alguna de las muchas

que se pueden realizar con los DAQ.

Se ha realizado un estudio a través de varios proyectos, para conocer la profundidad de Arduino utilizado en un DAQ.

Una investigación en esta línea es la desarrollada por Abdukarimov [30], en el que se realiza un proyecto basado en un DAQ conectado por Arduino implementado en un acelerómetro de tres ejes. Este autor realiza la conexión de Arduino en una interfaz GUI, basada en la versión antigua, en la que conecta Arduino mediante un protocolo serial de comunicación. Otro ejemplo esta desarrollado por Khan [31], en la que también desarrolla un trabajo similar al anterior ya que utiliza Arduino para conectar un acelerómetro y mostrarlo mediante el software MATLAB, con la particularidad que no utiliza un multiplexor para señales como lo hace el anterior. Por último, Casauco Liger [32], describe la utilización de un DAQ para la toma de muestras de luminosidad en un acuario. Este trabajo pretende mostrar cómo es posible la conexión de un sensor de luz a través de Arduino para obtener la señal que se podrá leer a través del software MATLAB y así tener controlado en todo momento la luz que se requiere en el acuario e incluso mediante el propio Arduino controlar la luz.

La idea fundamental en la que se desarrolla este trabajo es novedosa ya que, aunque existen varios autores como son Sinaga y Lozano-Garzón [33], [34], no muestran de manera completa el sistema de adquisición de datos como se mostrará en este trabajo ya que su aplicación es transversal para diferentes aplicaciones, basadas principalmente en un laboratorio en el que existen señales de temperatura, presión, régimen y otras, en la que se crea el sistema completamente desde cero y se modela de forma que puedan llegar cualquier tipo de señal medida en magnitud de voltaje y posteriormente se pueda modificar en un software como es MATLAB en el que la programación se hace de una forma sencilla. Como ventaja más importante, partiendo de la base de su versatilidad, es el precio al que se puede construir este sistema de adquisición de datos, es mucho menor al que existe comercialmente y aunque su frecuencia de adquisición sea menor es posible competir con ellas.

4.2 Aplicación específica en MCIA

El trabajo pretende desarrollar un DAQ multidisciplinar, pero el experimento realizado se hará en un MCIA ya que esta aplicación requiere la utilización de toma de datos para ensayos tanto de verificación de parámetros del motor, investigación, etc.

En la figura 20 se muestra el esquema de un MCIA, específicamente, un motor de encendido por chispa turbo alimentado. El diagrama muestra el conjunto de elementos mecánicos del que dispone la máquina, algunos móviles y otros fijos. Gracias a esto, es posible que a partir de una mezcla de aire y combustible se produzca una combustión que transforme la energía térmica en un movimiento lineal del pistón. Este está soportado por la biela que, a su vez, se apoya en el cigüeñal. El conjunto está acoplado a un volante que acumula las inercias de sucesivas explosiones con el objeto de realizar el giro lo más suave posible. Para realizar este ciclo, es necesario la utilización de otros elementos auxiliares como es, el turbocompresor, que es el encargado de alimentar el aire que se requiere para la mezcla aire-combustible a una presión superior a la que es tomada del exterior. Otro elemento sería la bomba de alta presión, la cual alimenta el circuito de combustible a presiones elevadas para que pueda ser inyectado en la cámara de combustión cuando el pistón realiza la etapa de compresión, en la que, en el caso del diagrama, la combustión es activada por una chispa generada por la bujía, pero existen motores de encendido por compresión (MEC) en los que el combustible se activa por efecto de la alta presión y temperatura que se genera dentro de la cámara de combustión.

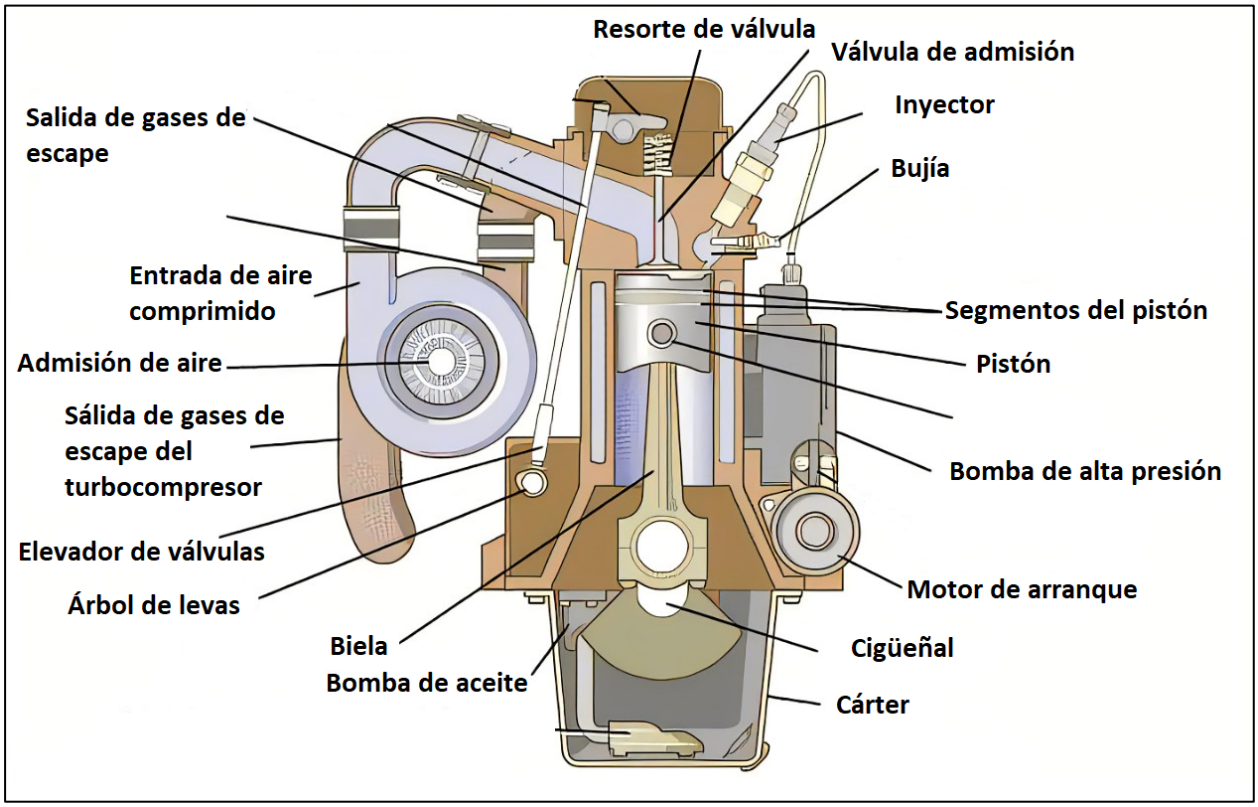


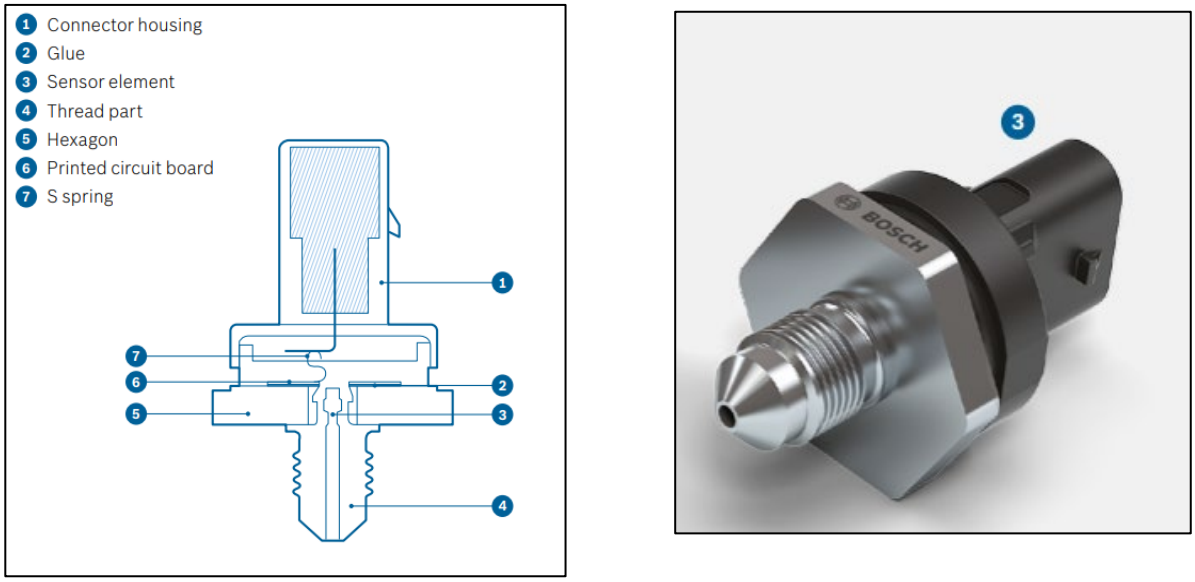
Figura 20. Corte frontal de un MCIA.

El control de estas máquinas es esencial para un correcto funcionamiento, ejemplo de ello es la utilización de sensores en los vehículos que están conectados a una unidad de control llamada ECU (Unidad de control del motor o en sus siglas en ingles *"Engine Control Unit"*). Esta es la encargada de recibir todas las señales de los diferentes elementos de medida que están en el motor. Al ser una máquina térmica, las señales que se han de medir suelen ser temperaturas, presiones, régimen de giro, par, consumo del combustible y caudal de aire. Estos parámetros son medidos, normalmente, con una señal de voltaje que genera el instrumento de medida o sensor. Para comprender el funcionamiento de la medida de estas señales, se describe cada uno de los fenómenos físicos que se han de medir en esta aplicación y específicamente cuales son los sensores utilizados y cuál es su funcionamiento.

4.1.1 Presión

Una de las señales más importantes es la de presión en puntos específicos del MCIA, como son: En el turbocompresor donde el aire que se toma del exterior aumenta su presión debido a que, el flujo de gases procedentes de la poscombustión hace girar la turbina que, a su vez, gira el compresor sobrealimentando de aire el motor. La medida de presión en la salida de la parte del compresor se realiza con el objetivo de conocer la presión de aire antes de que se produzca su entrada en la cámara de combustión. A más presión, mayor cantidad de oxígeno obteniéndose una mayor potencia. Atendiendo a la descripción para la toma de presión del turbocompresor, también es útil la medida de presión de la admisión de aire que procede del exterior pues la ECU realiza cálculos con este parámetro para realizar la inyección. Además de eso, es necesario la medida de presión de combustible que proviene de la bomba de alta presión. Existen vehículos que incorporan un sistema llamado *common rail* en el que los inyectores comparten la presión de combustible distribuyéndose desde una rampa. En este punto también es necesario la medida ya que la ECU requiere conocer si el combustible está entrando a una presión suficiente.

Estos sensores de presión contienen un diafragma en su interior, dependiendo del rango de presión este elemento es más grueso o delgado, cuando se aplica presión generan una tensión eléctrica debido a la deformación de la membrana, suele ser del orden de magnitud de mV, pero, se amplifica hasta el orden de 0 a 5 V.



(a) *(b)*

Figura 21. (a) Esquema del sensor de presión BOSCH PS-HPS5 (b) Sensor de presión BOSCH PS-HPS5

Los sensores de presión generan una señal de tensión que es proporcional a la presión que se está midiendo. Los fabricantes, como Bosch, en sus hojas de datos del sensor muestra una función recta lineal en la que está calibrado el sensor obteniendo la proporción mencionada, esto se puede observar en la figura 22 [35].

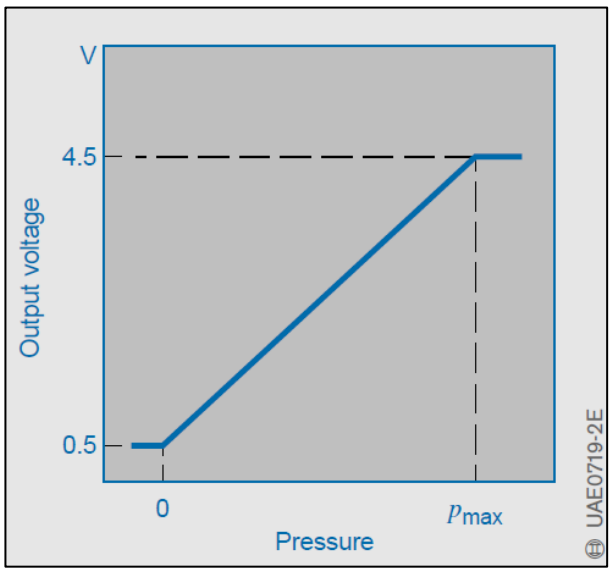


Figura 22. Curva de un sensor de presión genérica.

4.1.2 Caudal

Para obtener un control preciso en la relación aire-combustible, es necesario la medida de masa de aire. El caudalímetro de hilo caliente es el encargado de medir el gasto de aire de forma electrónica. Esta construido en un tubo de medición, donde se coloca un rectificador o rejilla rectificadora compuesta por un plástico y una malla metálica, que está diseñada para que la medición del flujo de aire sea uniforme Este instrumento está compuesto por un filamento que es calentado por una resistencia con una temperatura aproximada de 120 ºC, debido a la variación de temperatura que existe entre la entrada y la salida de aire, es posible determinar la cantidad de aire que pasa por el caudalímetro. Por otra parte, la medida de voltaje del caudalímetro determinado por la diferencia de temperatura es a través del puente de Wheatstone, esto se ve reflejado en la figura 23 en la que se muestra la curva característica de un caudalímetro. Esta curva muestra en el eje horizontal el flujo de aire que es medido con respecto al voltaje medido [36].

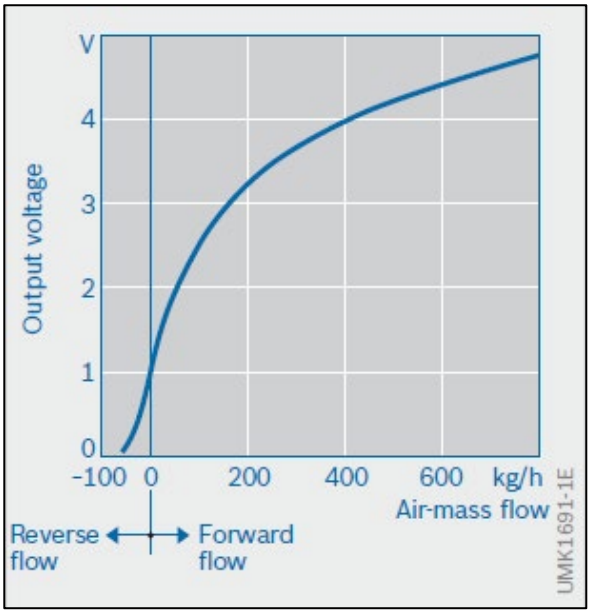


Figura 23. Curva característica de un caudalímetro

Figura 24. Caudalímetro de hilo caliente

4.1.3 Régimen giro

En los MCIA los sensores para determinar el régimen de giro se suelen colocar en dos posiciones importantes a medir, la primera es en el árbol de levas en la que se conocerá su velocidad y también comprobar si el elemento está girando ya que crea una sincronización entre la apertura de válvulas, inyección y el desplazamiento de los pistones. Por otro lado, se colocará en el volante de inercia, con el obtendremos el régimen de giro del motor. Esta señal de medida está pensada para obtener información las revoluciones a las que gira el motor por minuto, en la que la ECU se encargará de obtener dicha información para regular el sistema de aire, inyección y otros involucrados en el funcionamiento del vehículo y será mostrada en el velocímetro al usuario.

Para diferenciar los dos tipos de sensores que son instalados en el motor y como se explica en el apartado anterior en la que se han explicado las señales más comunes, se encuentran los sensores que trabajan por efecto Hall, estos son construidos con 3 pines en su conector ya que requieren alimentación, siendo los otros dos pines los terminales con los que se transmite la señal al dispositivo responsable de su medida, ya sea el DAQ o la ECU. Con respecto al siguiente tipo de sensor de régimen de giro, el inductivo, tiene 2 terminales, ya que este último carece de alimentación, por lo que ambos terminales son la salida transmisora de señal al dispositivo de medida. Ambos se pueden ver en la figura 25, en la que en el lado izquierdo se muestra un sensor por efecto Hall, siendo el pin 3 el positivo de la señal de salida, el pin 2 la tierra y el pin 1 el terminal de alimentación del sensor, en cambio, en el lado derecho se muestra el sensor por efecto inductivo, que al no necesitar alimentación tiene el pin 1 y 2 que son positivo y tierra.

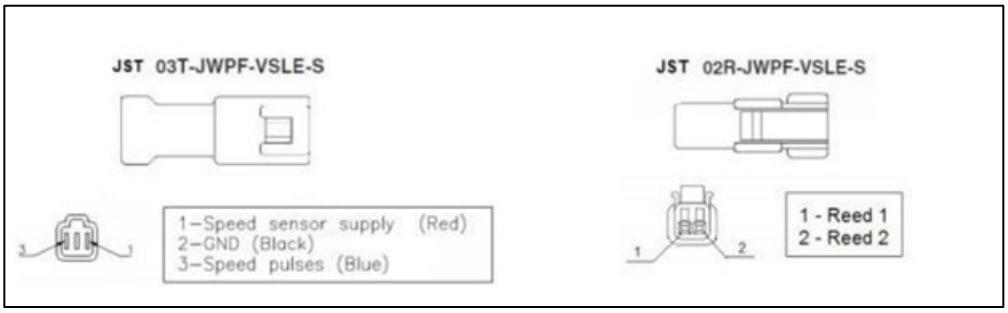


Figura 25. Sensor de efecto Hall (Derecha) y sensor de efecto inductivo o pasivo (Izquierda)

4.1.4 Par

El par motor es una medida necesaria para obtener la potencia de este, ya que, como se define en la ecuación 9, está definido como el producto del par y la velocidad de giro. La lectura del par es útil para determinar curvas características de los motores, en ellas se representan potencia efectiva, consumo especifico, y el par en función del régimen de giro. Esto determina la energía producida y el combustible utilizado para obtener el punto de funcionamiento de un MCIA.

Bautista [37] realizo un trabajo para el laboratorio donde se pretende implementar la solución, en el que calculó para un freno de motor de 10 kN que se encuentra en una celda de ensayos, el voltaje que representa el par que ejerce el freno. Para ello, acondicionó la señal ya que la generada por la célula de carga que tiene instalada registra una tensión del orden de mV y la óptima sería de ordenes de V. El siguiente paso fue el cálculo del brazo que hay instalado para poder realizar cambios en el freno hidráulico, ya que tiene una palanca que dependiendo de su posición el freno aumenta los pesos del freno.

Para calcular y obtener las tensiones dependiendo del par que se está ejerciendo, utilizo la ecuación 12, siendo E la distancia del eje a la galga, G la distancia de la galga hasta el brazo, S la distancia entre el brazo y el soporte, m sería el peso que se quiere poner al freno y g la fuerza gravitatoria.

$$M_0 = [(E + G + S) * 10^{-3}] * m_{peso} * g \quad (12)$$

Según el peso que se vaya aumentando al freno el par motor aumenta y se obtienen diferentes voltajes que son representados en la figura 27 para poder obtener una relación entre par y voltaje que sea representativa para el DAQ.

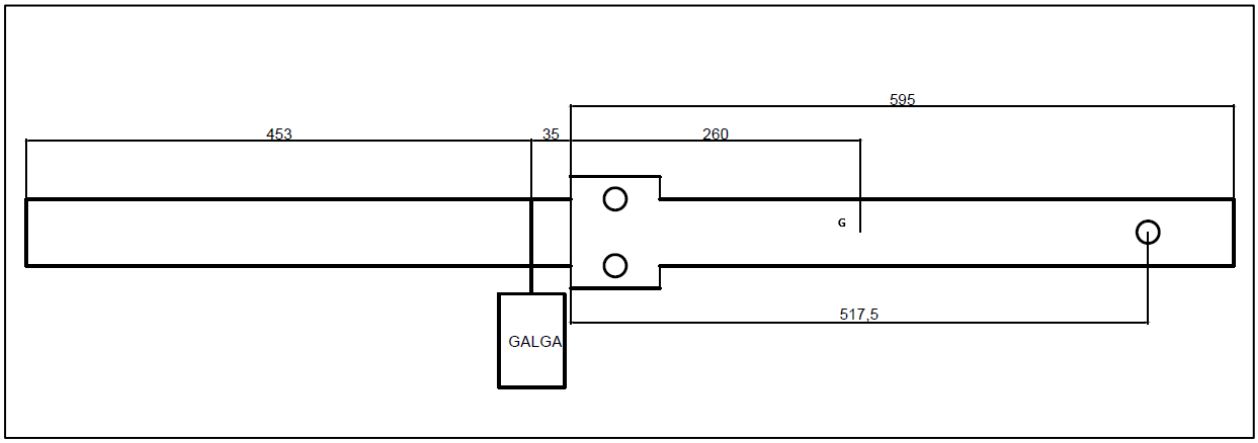


Figura 26. Brazo para freno hidráulico

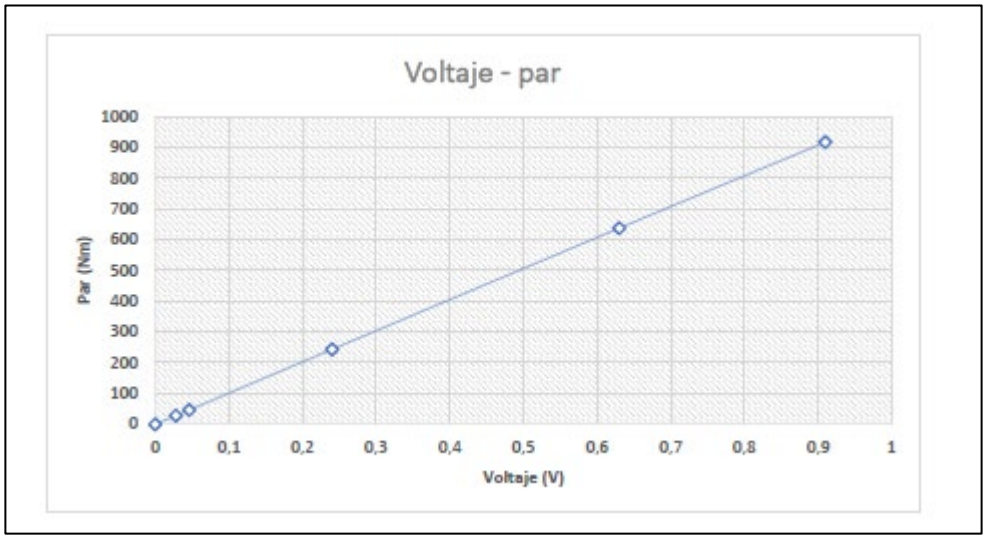


Figura 27. Gráfico par-voltaje generado para un freno hidráulico

5 HARDWARE Y SOFTWARE DEL SISTEMA DE ADQUISICIÓN DE DATOS

El objetivo principal de este trabajo es el desarrollo de un DAQ, en este capítulo se pretende describir los principales elementos que tiene el sistema. Se comienza con una breve introducción sobre los objetivos a desarrollar para la fabricación de este sistema y posteriormente se detalla cada uno de los componentes que tiene instalado el DAQ.

5.1 Objetivos para desarrollar

Se pretende que el DAQ sea robusto, manipulable y sencillo de utilizar, que cuente con una cantidad alta de entradas de señales, en comparación con los comerciales, y que tenga una conexión rápida y segura. Por otra parte, otro de los puntos clave a tener en cuenta es que sea manipulable, es decir, con la visión futura de que otro usuario pueda realizar cambios en el sistema y que no sea de una dificultad alta ni se requiera de la compra de equipos económicamente altos. Estos objetivos que se plantean crean unos requisitos para el sistema que se han de abordar, pero gracias a ello obtenemos un producto que sea competitivo frente a los que se encuentran en el mercado.

Las señales que se quieren adquirir son principalmente, temperatura (termopares), presión, caudal y régimen de giro. Los sensores están colocados en un bloque de motor, soplante y otras aplicaciones en el laboratorio, pero en el experimento que se realizará se encuentran en diferentes puntos de un MCIA en diferentes puntos para poder leer los parámetros durante los ensayos.

5.2 Montaje del hardware del DAQ

Se han elegido diferentes materiales para la fabricación del DAQ. Para comenzar, la base en la que se colocarán todos los materiales de forma que ningún elemento se desprenda es la caja ABS, con ella protegeremos y ordenaremos correctamente cada uno de los componentes del sistema. Dentro de la caja ABS, se colocarán tres módulos de Arduino MEGA 2560, 16 amplificadores de señal para termopares AD8495, estas requieren alimentación por lo que se colocarán dos fuentes de alimentación a 5 V en corriente continua. Adicionalmente, cableado con el que se realicen las conexiones entre Arduino, amplificadores, fuente de alimentación y conectores, de los que para estos últimos se colocarán 32, estos se especificarán más adelante. Para recapitular toda la información que se describe en este capítulo, en la tabla 2 se muestra el material utilizado para la fabricación del DAQ.

Tabla 2. Material utilizado en el montaje

ELEMENTO	DESCRPICIÓN	UNIDADES
Caja estanca ABS	Será la encargada de alojar todos los componentes y aislarlos del exterior, también funcionará como estructura para la conexión de los sensores.	1
Arduino MEGA 2560		3

Terminal block shield board Arduino MEGA	Se coloca en la parte superior de Arduino actuando como puente entre los pines de la placa y los terminales del bloque de conexión. Estos terminales tienen tornillos para que el cable que se conecte quede seguro.	3
Fuente de alimentación 5 V AC/DC		2
Placa PCB perforada 15 x 15 cm		4
Adafruit AD8495	Son amplificadores de señal para termopares, con ellos conseguiremos una señal que sea correctamente legible para Arduino u otro medidor de voltaje.	16
Conector GX – 12 4 PIN	Serán los conectores colocados en el exterior de la caja en los que podremos realizar la conexión con los sensores. Al ser un conector libre podemos colocar de forma libre sus pines, aunque en el montaje de la caja como se especificará posteriormente existe un código de colores para la conexión interna.	32
Fuente de alimentación regulable	Esta fuente es utilizada para dar alimentación a sensores que requieran más de 5 V.	1
Macho a presión IEC 14 con interruptor	Se requiere alimentación a corriente alterna para poder conectar la fuente de alimentación de 5 V AC /DC, por ello, se elige este estándar de conexión al ser un cable común.	1
HUB USB de 4 puertos	Se conectarán los 3 Arduino de forma simultánea a un mismo puerto USB al PC.	1
Ventilador 60x60 mm	La ventilación es necesaria en la caja ya que se instalan elementos electrónicos que disipan calor, para ello se colocarán en la parte posterior para crear un flujo de aire en el interior de la caja.	2
LEDs	Serán LEDs de color rojo, naranja y verde que mostrará al usuario el estado del sistema sin tener que visualizar el PC.	3
Resistencias	Para la colocación de los LEDs mencionados anteriormente, se requiere la utilización de resistencias.	3
Tornillería y separadores		-

5.2.1 Bloques de terminales para Arduino

La placa de Arduino tiene unas entradas físicas llamadas pines que están construidas en la propia placa para poder insertar un cable con una punta metálica que realice la conexión con la clavija. Estos pines colocados en la placa tienen una numeración, ya que dependiendo de en cuál realices la conexión puede ser digital, analógico,

para alimentación, tierra, etc. En el anexo A vemos el mapa de conexiones del Arduino MEGA 2560 en el que están definidas las salidas y entradas de este Arduino que es el utilizado para este proyecto.

Se necesita un elemento de conexión para comunicar el dispositivo de medida con Arduino y dado que esta conexión es inestable físicamente, se ha decidido la utilización de un bloque de terminales el cual actúa como escudo entre la placa y el resto de los elementos y a su vez facilita la conexión de cables de forma fija y segura ya que este incorpora unos tornillos en sus terminales.

En la figura 28 se muestra la adaptación del bloque de terminales al Arduino MEGA 2560. En la parte inferior conecta los pines a Arduino mediante unas patillas que están conectadas en una placa perforada hasta unos terminales de conexión numerados por cada pin que ha sido conectado. En el lateral deja paso para la conexión de USB y en la parte superior un pulsador de RESET.

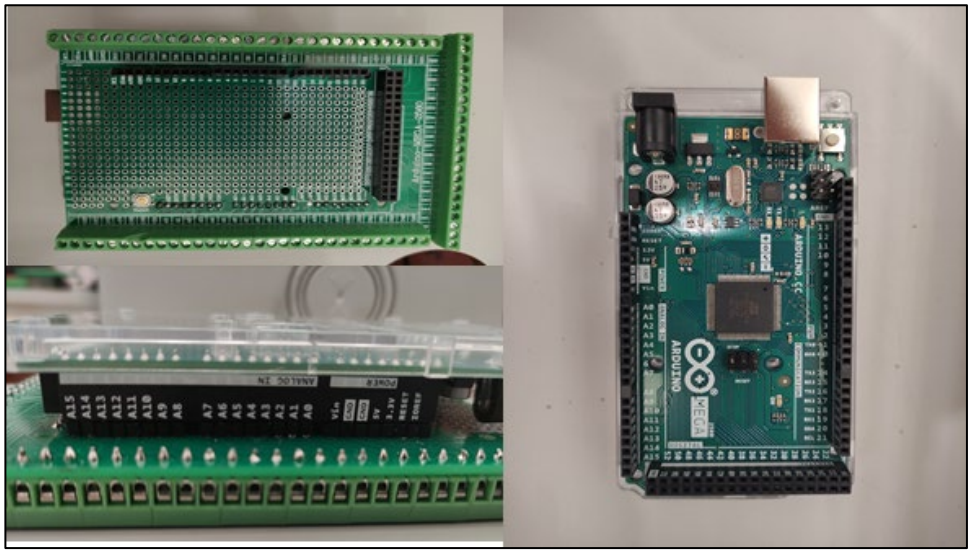


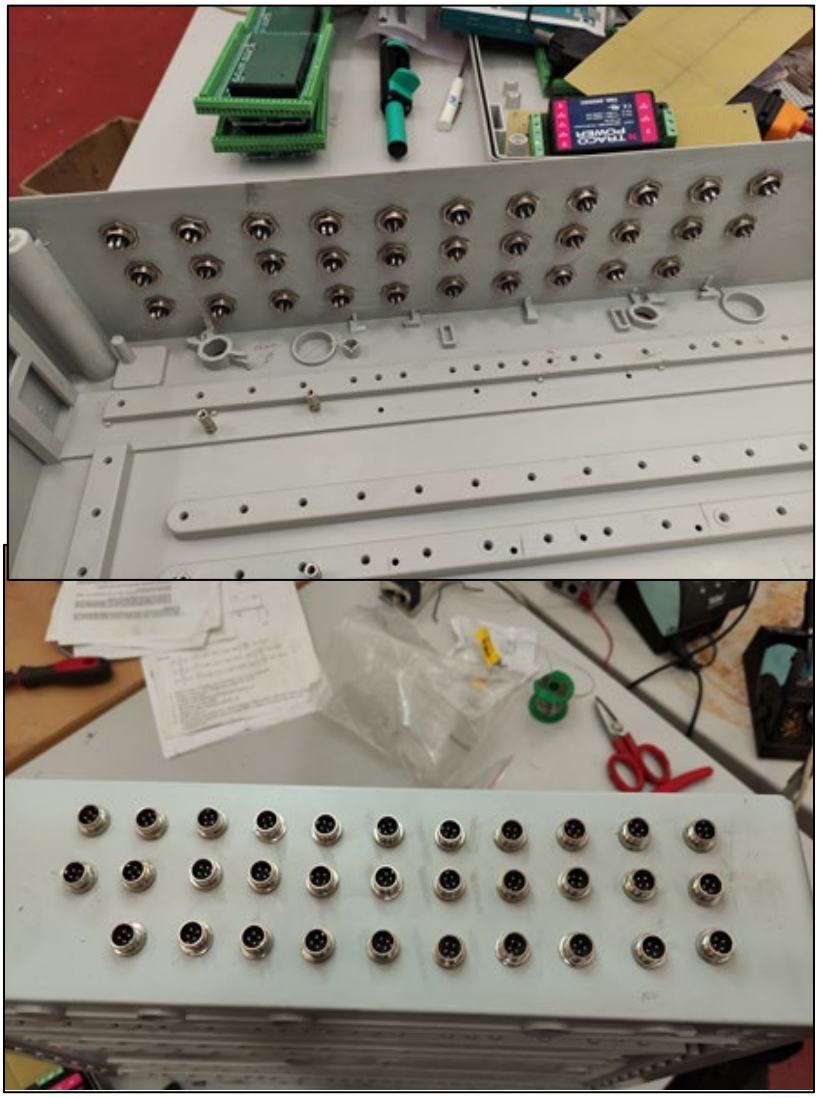
Figura 28. Arduino MEGA 2560 y bloque de terminal para conexión.

Este fue el primer paso para la fabricación del DAQ, esto es debido a que se requiere una ligera modificación soldando patillas a la placa perforada para que sean conectados a Arduino.

5.2.2 Mecanizado de caja ABS

Se utiliza una caja de material ABS, similar a la utilizada para cuadros eléctricos, para que sirva como estructura principal del DAQ. En ella se colocarán elementos en el interior y exterior, debido a esto, se ha de mecanizar partes que no son necesarias para este dispositivo ya que es una caja universal con elementos de sujeción en sus partes laterales interiores que han de ser lijadas y eliminadas para abarcar más espacio en su interior y, por otro lado, se ha de realizar varios orificios en sus paredes para colocar conectores para sensores y alimentación.

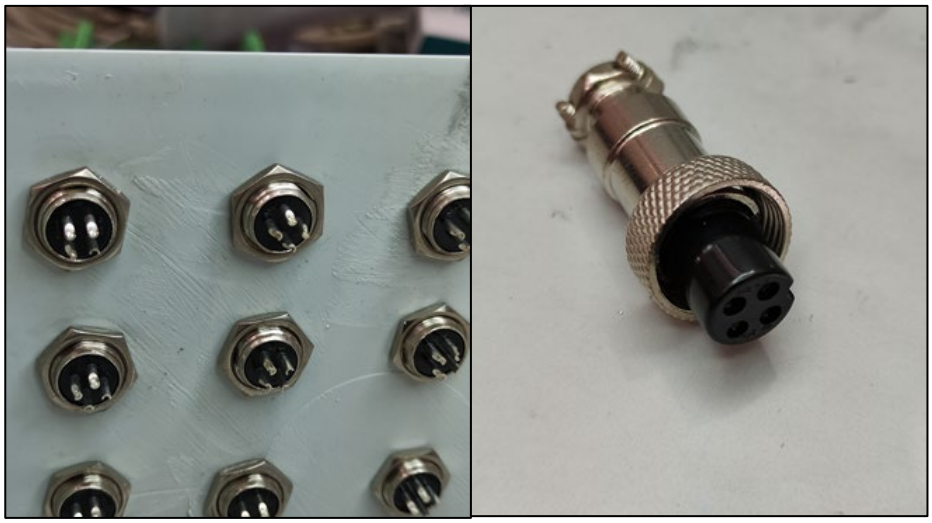
En la figura 29 y 30, se muestra el mecanizado realizado para los conectores GX – 12 de 4 pines, estos han de ir en la parte frontal de la caja ya que son los encargados de realizar la conexión entre los sensores o dispositivos de medida con el interior de la caja. La conexión de estos conectores se explica en un paso posterior a este ya que se requiere la soldadura de los pines con cables que irán en su interior. Se han colocado un total de 32 conectores GX – 12 los cuales se han distribuido dependiendo del tipo de sensor.



(a)

(b)

Figura 29. (a) Interior de la caja ABS con conectores GX -12 (b) Detalle exterior de las conexiones de la caja



(a) *(b)*

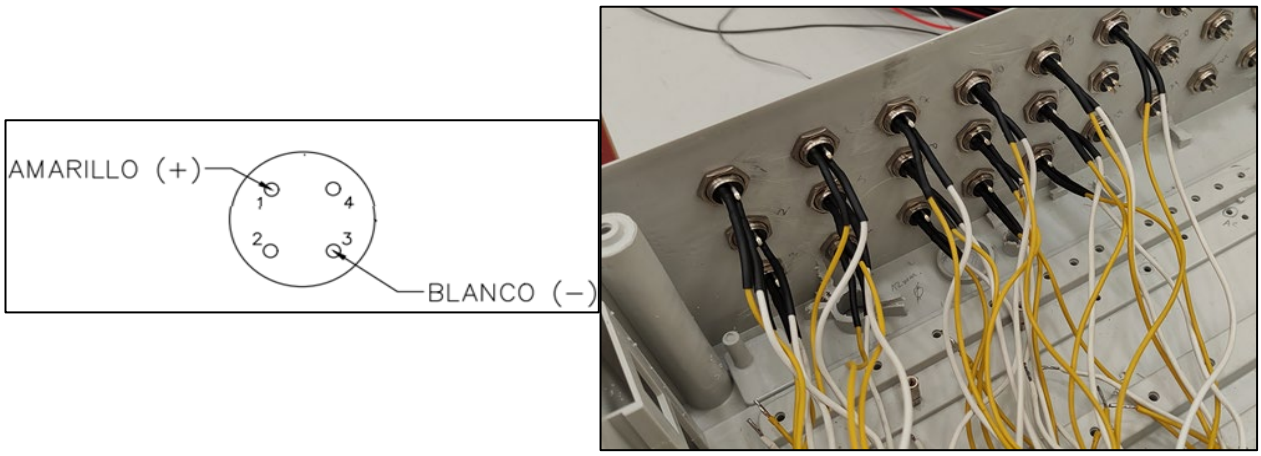
Figura 30. (a) Detalle interior de los conectores GX – 12 (b) Detalle de un conector GX – 12

5.2.3 Conexión GX – 12 con Arduino

Los conectores GX – 12 actúan como puente conectando los dispositivos de medida o sensores y la placa Arduino. Para ello, cuenta con 4 pines los cuales son de libre configuración ya que tienen que ser soldados, esto es una ventaja para este trabajo ya que al ser de libre conexión podemos utilizar la configuración que se desee. Cada tipo de sensor es diferente ya que no tienen el mismo número de salidas, esto se puede ver en el apartado de régimen de giro en la que los sensores por efecto Hall no son iguales que los de efecto inductivo.

En este caso se van a diferenciar los conectores utilizados para termopares, que requieren de dos pines de salida y sensores que han de ser alimentados con una tensión de 5 V que será obtenida de la fuente de alimentación que habrá en el interior de la caja, este tipo de sensores pueden ser los de presión o caudalímetros.

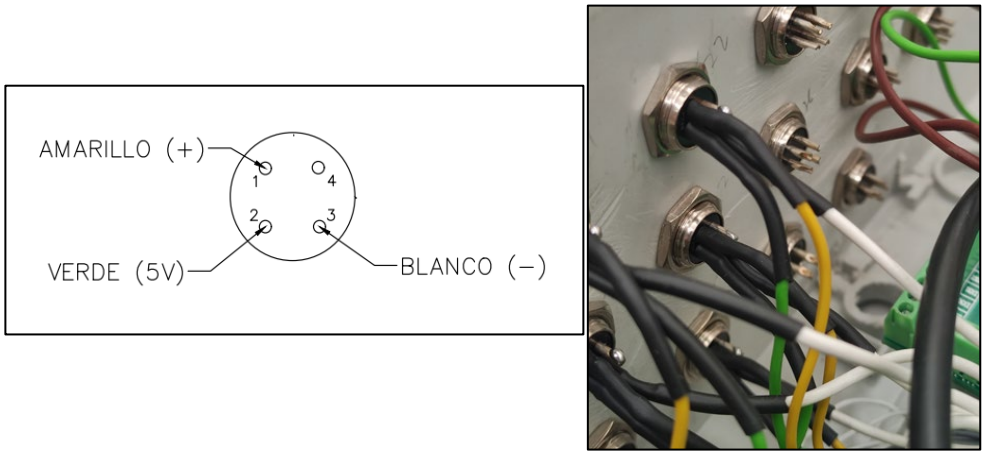
Para la conexión de termopares se utilizarán un total de 16 conectores GX – 12, en la figura 31 se muestra el esquema de conexión de los termopares, los cuales requieren dos terminales uno positivo y otro negativo. Estos dos terminales, por convenio utilizado en el trabajo, será el pin 1 para el positivo y el pin 3 para el negativo en los que se conectarán con un cable amarillo y blanco respectivamente.



(a) *(b)*

Figura 31. (a) Esquema de conexión de termopares en conectores GX-12 (b) Detalle de conexión de cableado interior para conectores GX – 12 - Termopares

En la figura 32, encontramos el esquema de conexión para sensores que requieren ser alimentados, este tendrá 3 terminales, se utilizará el convenio anterior para los pines positivo y negativo, siendo estos el pin 1 y 3 respectivamente y el pin 2 será el que alimentará con 5 V el sensor. El código de colores utilizado para este tipo de sensores será de amarillo para el pin 1 positivo, blanco para el pin 3 negativo y finalmente el verde para el pin 2 de alimentación. Si en algún caso el sensor requiere de alimentación de más de 5 V se utilizará una fuente de alimentación externa a la caja que se fabrica ya que esta no incorporará alimentación a una tensión superior a 5 V, esto se debe a que la mayoría de los elementos utilizados en la caja son alimentados a esta tensión.



(a) *(b)*

Figura 32. (a) Esquema de conexión para sensores alimentados con conectores GX -12 (b) Detalle de conexión de cableado interior para conectores GX – 12 de sensores alimentados

5.2.4 Amplificadores AD 8495

Uno de los elementos más importantes del DAQ será el AD8495 de Adafruit, como se explica en el apartado 3.3.2 es el encargado de amplificar la señal de temperatura de termopares. Debido a la necesidad de adquirir señales de temperatura y con el objetivo de tomar el mayor número de señales posible, se decide la utilización de un módulo de Arduino exclusivo para medir temperatura.

En este proyecto se colocarán 16 amplificadores, con la visión de obtener el mayor número de señales de temperatura. La organización de estos módulos AD8495 será la siguiente; Se cortarán las placas PCB perforadas y se montarán en dos estantes, en cada uno de ellos se atornillarán 6 módulos, los últimos cuatro módulos se sostienen en una placa perforada separada de estos dos estantes, esto es debido a la falta de espacio a lo alto de la caja y para futuras reparaciones sea sencillo de acceder a todos los amplificadores.

En la figura 33 se muestra la colocación de los amplificadores AD8495 en las placas perforadas, en ella, también, podemos observar en la parte trasera del módulo la conexión de cables que irán a Arduino y a alimentación y en la parte frontal se conectarán los cables que van directos al conector GX – 12 para en la parte exterior realizar el enlace con los termopares o dispositivos de medida de temperatura.

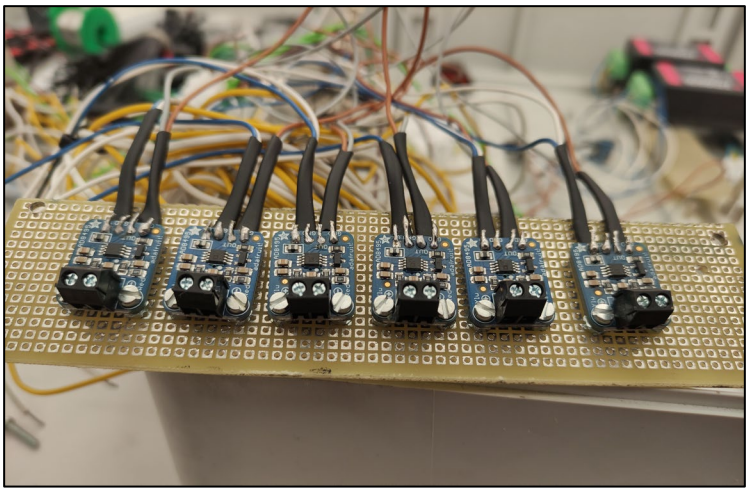


Figura 33. Montaje de los amplificadores AD8495 en el interior de la caja.

Por otro lado, en la figura 34 se muestra en detalle la conexión de los cables mencionados anteriormente. Para

que exista un consenso para una futura modificación se ha decidido la utilización de diferentes colores para organizar mejor las conexiones entre el amplificador con Arduino y la alimentación:

- $V+$: Para el terminal de alimentación positivo que va desde la fuente hasta el módulo amplificador a 5 V, se ha utilizado el cable azul.
- GND: Para el terminal que está conectado a la tierra de la fuente de alimentación, se ha utilizado el cable blanco.
- OUT: En el terminal de salida en el que es conectado con Arduino, se ha utilizado el cable marrón.
- GND: El terminal de tierra de la señal de salida que está conectado con Arduino, se utiliza el cable gris.

Para tener una referencia de este esquema de conexión, es recomendable fijarse en la figura 18 en la que se muestra en detalle el amplificador.

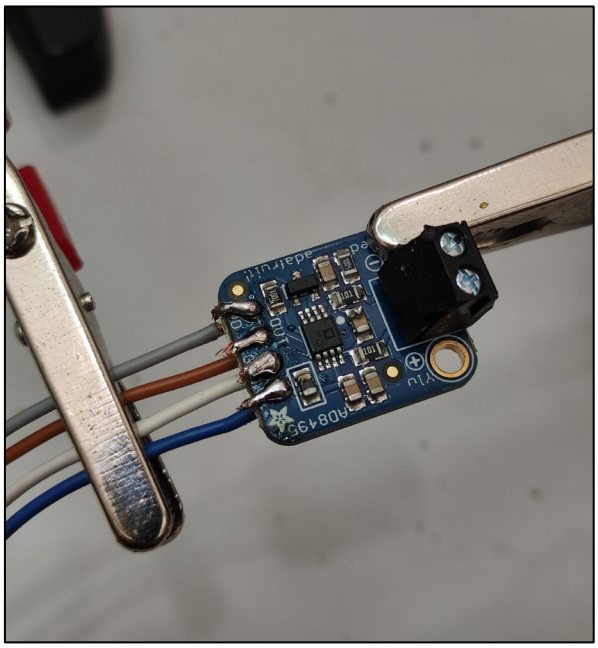


Figura 34. Detalle de conexión a amplificador AD8495

5.2.5 Conexión entre elementos

Tras mostrar el enlace de conexión de los sensores con los conectores y la amplificación de señal para termopares, se han de conectar los elementos mencionados con la alimentación y a Arduino para recibir las señales.

Comenzando por los amplificadores de señal, con el amarillo y blanco, indicado en la figura 31, se conectarán a los dos terminales positivo y negativo respectivamente del amplificador. Con esto obtendremos que una señal en mV procedente de los termopares, se pueda transmitir a través de los cables marrón y gris, indicados en la figura 34, hasta Arduino. Para cada uno de los cables marrones, es decir, el de salida de señal, se conectará a una unión de empalme entre cables, en ella, saldrán dos cables, uno de ellos será amarillo, con este conectaremos finalmente el amplificador de señal con Arduino en los diferentes terminales desde el A0 hasta el A15. Por otro lado, saldrá otro cable blanco con el que podremos realizar una conexión con un conector BNC u otro tipo para que pueda ser utilizado en un dispositivo diferente al DAQ utilizado, como, por ejemplo, un osciloscopio. Esto hace que el sistema sea aún más manipulable y versátil ya que tiene la posibilidad de utilizar una parte del sistema y que sea conectado en otro sin necesidad de utilizar Arduino ni el software para controlarlo.

Continuando con lo anterior, otra de las salidas del amplificador es la tierra, este se conectará con todos los cables gris que unen el terminal GND de salida de señal en una ficha de empalme para sacar un solo cable de color gris que irá conectado a Arduino ya que esta tierra es común para todos los amplificadores. Este conjunto será para el primer Arduino, que será exclusivo para señales de temperatura ya que se utilizarán todas las entradas analógicas.

Tendremos otro Arduino con el que conectaremos otro tipo de señales, como puedan ser de presión o caudalímetros, estás serán alimentadas con un conector del tipo mostrado en la figura 32. El cable amarillo y blanco será, al igual que con los termopares, el que enlace el conector GX – 12 con el resto de los elementos de la caja, pero en este caso la diferencia es que este irá directo a Arduino ya que no requiere amplificarse debido a que el propio dispositivo de medida al ser alimentado lo amplifica. Esto último es la razón principal por la que se utilizará un cable adicional, el verde, con el que alimentaremos directamente desde la fuente de alimentación hasta el conector que será el que transmita al pin correspondiente los 5 V. Por otro lado, es necesario una conexión con tierra, esta se realizará con un puente entre el cable blanco, el cable marrón (procedente de la fuente de alimentación como tierra) y Arduino.

5.2.6 LEDs

Para mostrar cambios en el software de adquisición de forma visual en la caja, se han colocado tres LEDs que ayudarán al usuario a conocer el funcionamiento del DAQ. Para esta instalación se ha utilizado 3 LEDs, con diferentes colores, verde, naranja y rojo, con los que se dará una respuesta diferente según la acción que se esté ejecutando en el DAQ.

La patilla de ánodo del LED irá conectado a una resistencia que limitará la potencia de este para que no se sobrealimente, esta resistencia tendrá un valor de 300 Ohmios para todos los LEDs, el otro terminal de la resistencia irá conectado a un pin digital de Arduino, con este pin podremos crear una señal de 5 V que estará en ON para un valor de 1 y en OFF para un valor de 0. Por otro lado, se conectará el cátodo al pin de tierra del Arduino que será común para todos los LEDs. La figura 35 muestra un esquema utilizado para un Arduino UNO en la que se conectan los LEDs, las resistencias a la placa, cabe destacar que, aunque el esquema muestre el modelo de Arduino UNO el esquema es compatible al utilizado con el Arduino MEGA 2560 en este proyecto.

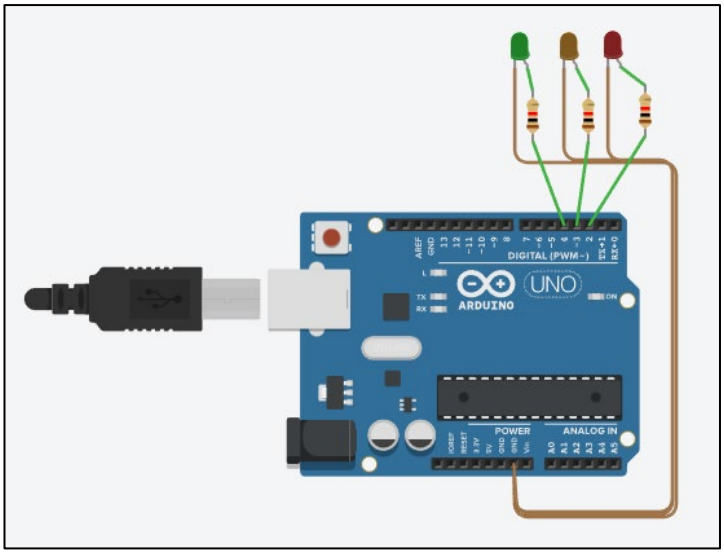


Figura 35. Conexión de Arduino con LED

5.2.7 Conexión interna del DAQ

En la figura 36 se muestra como ha sido el montaje completo de la caja en la que se integran todos los componentes, incluyendo, adicionalmente al sistema ya descrito, ventilación que irá alimentada por la fuente. En estas imágenes se observa una gran cantidad de cables que han sido ordenados por colores y unidos con bridas para una correcta organización, en la tabla 3 se detalla cada uno de los colores y conexiones de forma que pueda ser reconocida de una forma más sencilla como recapitulación de los anteriores puntos.

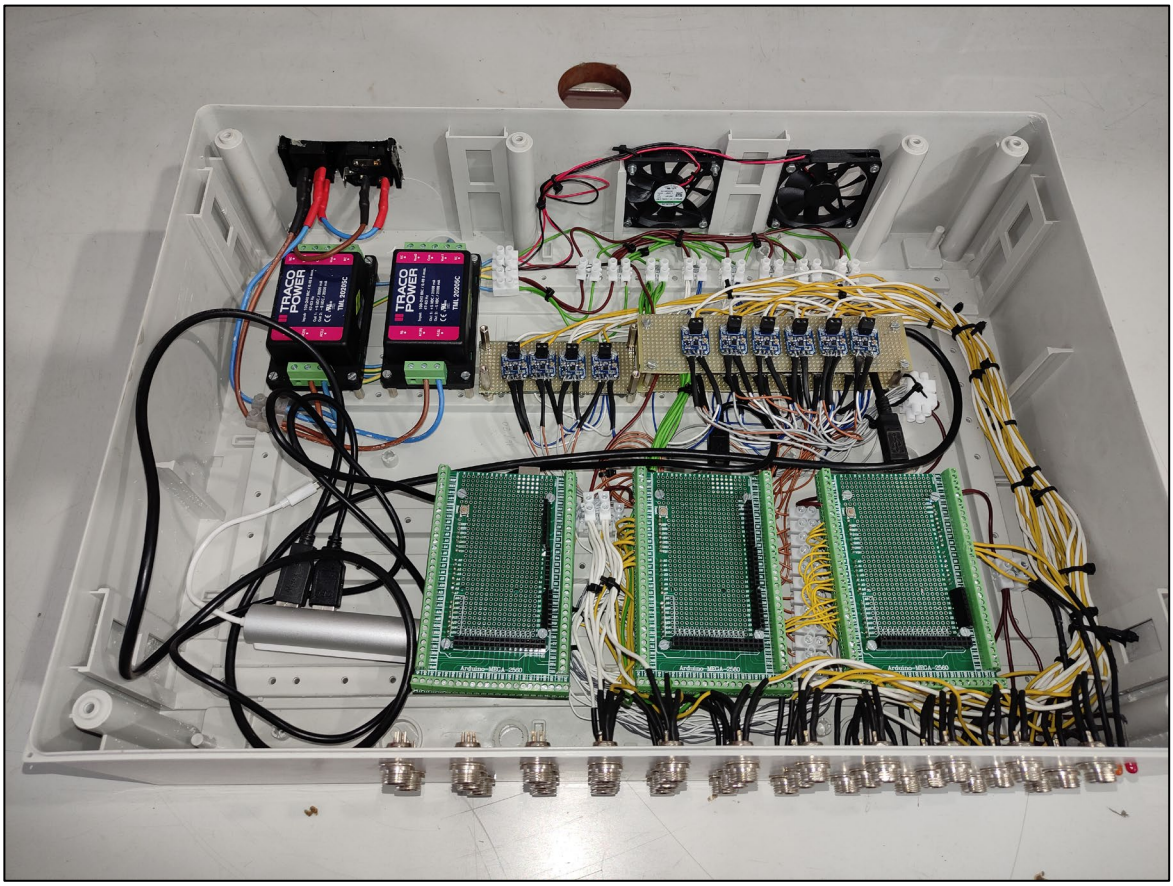


Figura 36. Montaje completo del DAQ

Tabla 3. Código de colores para la conexión interna del DAQ

Color de cable	Elementos que conecta
Amarillo 1	Positivo termopar en conector GX – 12 con amplificador AD8495
Blanco 1	Negativo termopar en conector GX – 12 con amplificador AD8495
Amarillo 2	Positivo sensor alimentado en conector GX – 12 con Arduino
Blanco 2	Negativo sensor alimentado en conector GX – 12 con tierra común conectada a Arduino
Verde 1	Alimentación a 5 V en conector GX – 12
Azul 1	Alimentación a 5 V procedente de la fuente del amplificador AD8495 en terminal V+

Blanco 3	Tierra procedente de la fuente del amplificador AD8495 en terminal GND
Marrón 1	Tensión de salida del amplificador AD8495 en terminal OUT conectado a enlace por modo de adquisición dual
Gris 1	Tierra de la señal de salida del amplificador AD8495 en terminal GND conectado a tierra común de Arduino
Amarillo 3	Conexión en el enlace de la señal de salida del amplificador AD8495 a Arduino
Blanco 4	Conexión en el enlace de la señal de salida del amplificador AD8495 para otro DAQ
Gris 2	Conexión en el enlace de la tierra de la señal de salida del amplificador AD8495 para otro DAQ
Verde 2	Conexión para alimentación a la fuente de 5 V en corriente continua
Marrón 2	Conexión para tierra a la fuente de 5 V en corriente continua
Azul 2	Polo positivo de la fuente de alimentación lado DC
Amarillo/Verde	Polo negativo de la fuente de alimentación lado DC
Azul 3	Polo positivo de la fuente de alimentación lado AC
Marrón 3	Polo negativo de la fuente de alimentación lado AC

5.3 Coste de material del sistema

El coste de este sistema es de 1.093,19 €, desglosado en el anexo B, donde podremos ver la lista de elementos que han sido comprados y el precio de estos. En comparación con un sistema comercial, tomando como referencia un sistema de NI completo con unas funcionalidades parecidas a las que se diseñan en este trabajo, el precio asciende hasta los 4.401,98 €, sin contar el software. Este último precio también se muestra en el anexo B para realizar una comparativa.

5.4 Descripción del software diseñado en *Appdesigner* de MATLAB

Para que el sistema DAQ sea un conjunto completo, se ha de conectar el hardware a un software junto con un controlador que sea capaz de transmitir la información para que pueda ser tratada, visualizada y guardada. En el apartado 3.4 se describe como es el funcionamiento de un software de adquisición de datos y la descripción del que se va a utilizar en este trabajo, *Appdesigner* de MATLAB.

Se ha programado una aplicación en este software para poder controlar el DAQ, esto es gracias al paquete de herramientas con el que conectamos Arduino a MATLAB. El diseño de esta aplicación pretende ser sencilla en la que podamos, con pocos botones, poder medir y visualizar las señales adquiridas y generar un archivo en el que guardar el histórico.

5.4.1 Interfaz principal del software

Para comenzar, en la figura 37 se muestra la interfaz gráfica principal en la que se ha diseñado el controlador

principal del DAQ. En esta ventana principal podemos ver una barra de herramientas en la parte superior, dos botones con los que iniciaremos y detendremos la adquisición de datos, 16 *check box* utilizadas para la elección de los termopares conectados, 8 *check box* para sensores de presión y el resto serán utilizados para otro tipo de señales adicionales. En la figura 38 vemos como al activar la casilla de selección de la señal correspondiente que se vaya a utilizar, se mostrará un cuadro de texto y numérico en el que aparecerá el valor de la señal medida, por ejemplo, en grados Celsius o bares de presión. Por otro lado, tenemos un grupo de pestañas en las que se puede navegar para encontrar gráficos para ver las medidas en vivo, el gráfico generado al final de la adquisición para antes de comprobar que los datos se han generado correctamente y por último una pestaña en la que poner llamados instrumentos de medida como son los calibres o en el término inglés *gauge*.

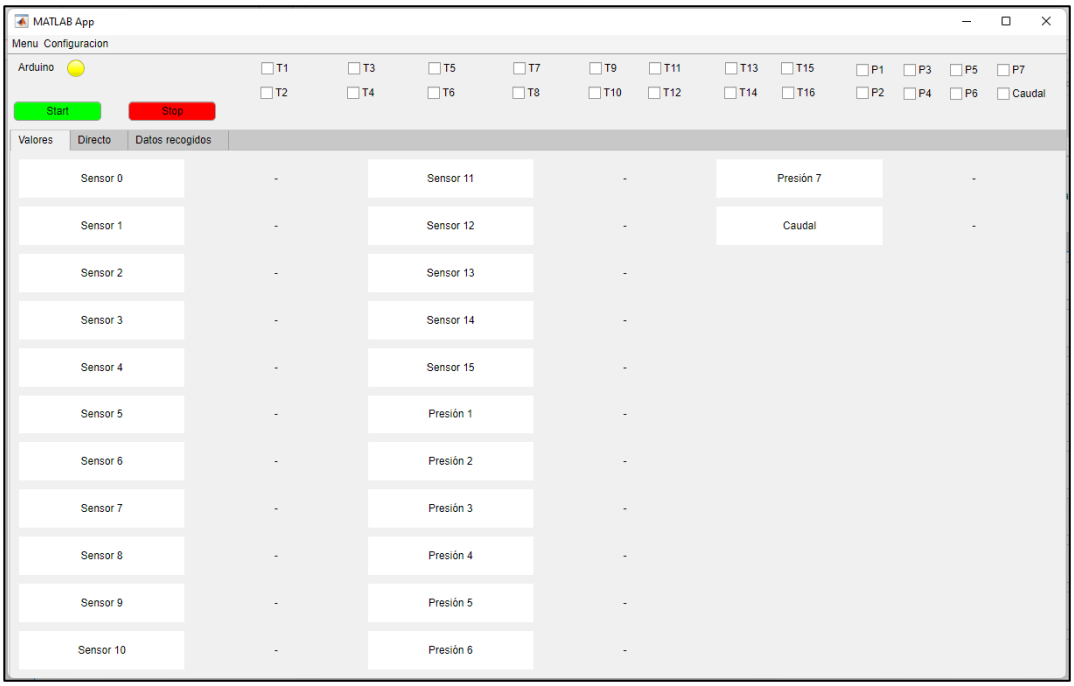


Figura 37. Interfaz principal del DAQ

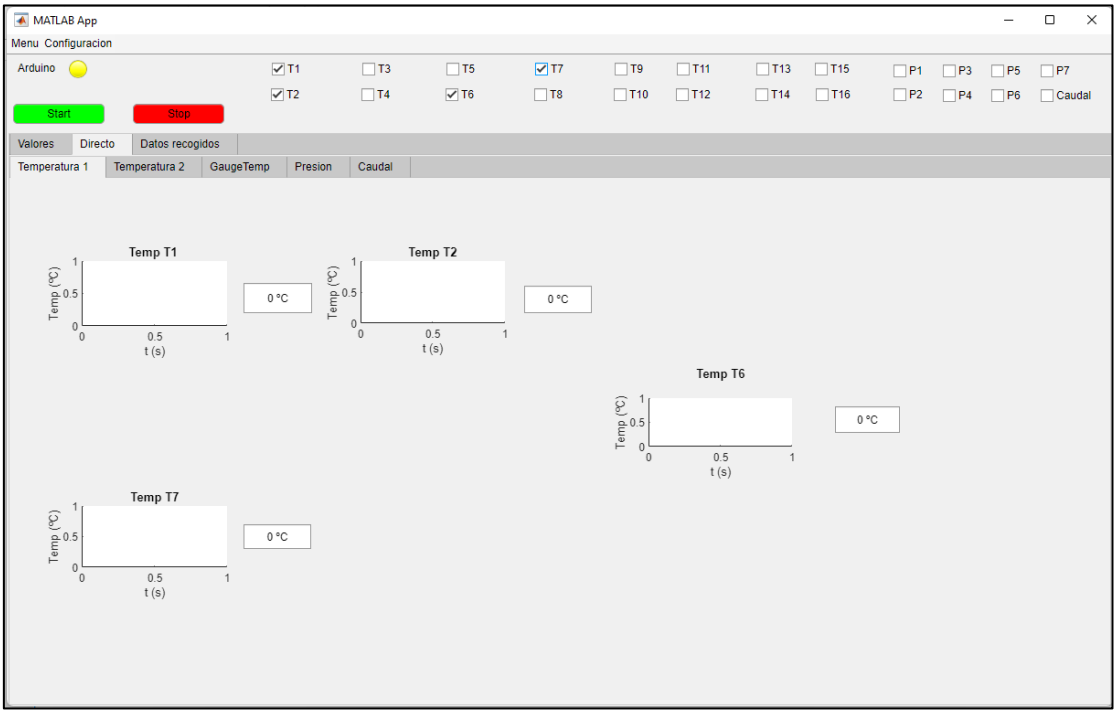


Figura 38. Gráficos en vivo en el interfaz del DAQ

En la barra de herramientas superior se puede acceder a dos pestañas independientes, en la primera denominada menú, encontraremos el acceso a la conexión con Arduino; Este botón realiza mediante la programación con el paquete de herramientas de Arduino el enlace entre MATLAB y el propio dispositivo. El segundo será el botón de guardado, al parar con el botón *STOP* se terminará la adquisición, y para realizar un guardado de los datos se presionará dicho botón. Por último, el botón de "reiniciar" está diseñado para que al acabar una medida hay que utilizarlo para volver a adquirir las señales, esto es debido a que el programa tiene una complejidad a la hora de volver a restaurar los datos de inicio, por ello, este botón está programado para que al pulsarlo se realice un cierre del programa y se inicie de nuevo sin que el usuario tenga que cerrar y acceder en su carpeta raíz. La segunda pestaña denominada configuración, aloja el botón con el que cambiaremos los parámetros de presión, este se describirá en los siguientes puntos de manera más detallada. Toda esta información la podemos ver en la figura 38 en la que se muestra la barra de herramientas.

Se debe mencionar que, al guardar los datos, encontraremos dos formatos; El primero será un archivo con extensión "*.xls"* compatible con Excel, está pensado como base de datos y creación de gráficos externo al software MATLAB, por otro lado, se creará un archivo con extensión *".mat"* con el que podremos trabajar con las medidas obtenidas dentro del propio MATLAB para, por ejemplo, filtrar la señal u obtener otra información. El nombre con el que se guardará este archivo será de la siguiente forma;

- Archivo Excel: "*Temperatura_yyyy-MM-ddTHHmmss.xls"*
- Archivo MATLAB: "*Temperatura_yyyy-MM-ddTHHmmss.mat"*

Siendo *yyyy* el año, *MM* el mes y *dd* el día. Por otro lado, *HH* la hora, *mm* los minutos y *ss* los segundos. Esta fecha y hora será la actual a la que se ha guardado los datos.

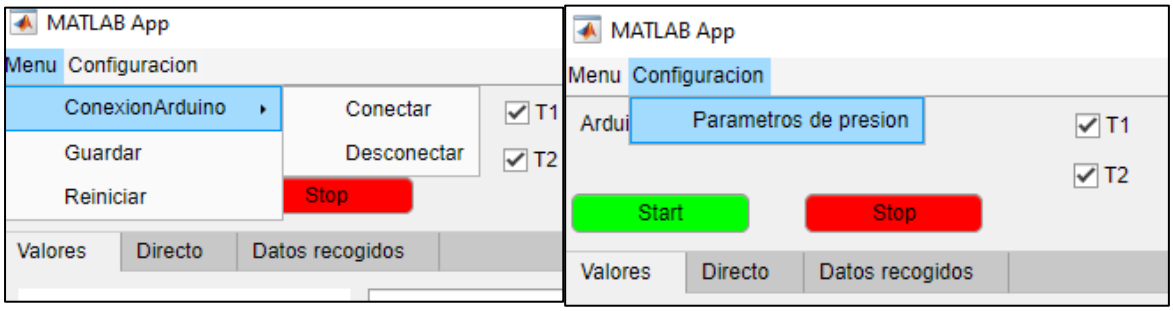


Figura 39. Barra de herramientas del DAQ.

5.4.2 Mensajes para el usuario

El programa ha de mostrar al usuario mensajes de alerta al realizar algunas de las siguientes acciones; La primera de ellas es la confirmación de una correcta conexión de MATLAB con Arduino, está pensado para que el usuario sepa si es posible comenzar a realizar la toma de señales. En segundo lugar, al finalizar la adquisición pulsando el botón *STOP*, se muestra un mensaje recordatorio al usuario para que guarde los datos antes de salir del programa. Al guardar los datos, con el botón en la barra de herramientas "guardar", aparecerá otro mensaje que confirmará el correcto guardado de los datos. Por último, al cerrar el programa o reiniciar, el sistema preguntará si desea cerrarlo, esto se hará como medida de seguridad para que no se pierdan los datos que están almacenados. Todos los mensajes aparecen en la figura 40 ordenados de forma en la que se procede en el programa.

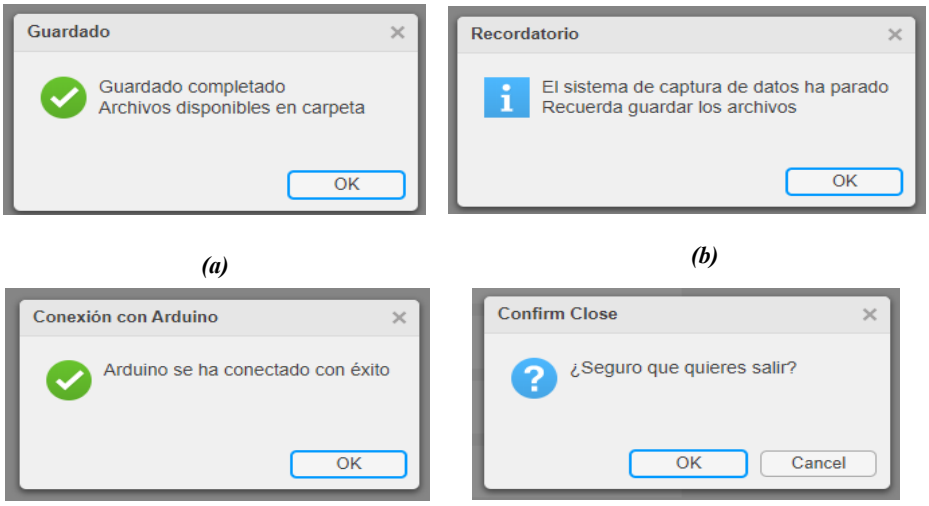


Figura 40. (a) Mensaje de conexión de Arduino (b) Mensaje recordatorio para guardado de datos (c) Mensaje de comprobación de guardado de datos (d) Confirmación del usuario para cerrar la interfaz

5.4.3 Configuración de los parámetros de presión

Dentro del software se ha diseñado una configuración específica para la configuración de los sensores de presión, esto se debe a que, según el tipo de sensor y el fabricante, existe una recta de calibración para este tipo de dispositivos de medida. La recta es obtenida en la ficha técnica del sensor, esta suele mostrar en el eje horizontal la presión y en el eje vertical el voltaje, este último va desde 0 V hasta 5 V ya que son alimentados a esta tensión. En la figura 22 se muestra una recta de estas características la cual esta parametrizada con la ecuación 13, donde p es la presión medida en bares, $U\ (V)$ es la tensión medida por el sensor, a y b son parámetros definidos por la recta de calibración en la ficha técnica.

$$p\ (bar) = a * U(V) + b \qquad (13)$$

El programa, por tanto, al seleccionar la configuración de los parámetros de presión, abrirá una pestaña en la que se podrá incorporar esos dos parámetros a y b que ofrece el fabricante en su ficha de características técnicas. Esta pestaña está reflejada en la figura 41.

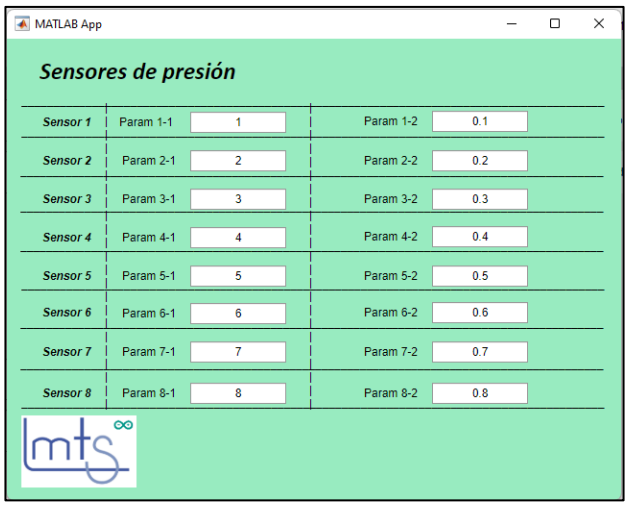


Figura 41. Interfaz para la configuración de los sensores de presión

5.4.4 Visualización de la toma de datos

A la hora de tomar resultados, la interfaz mostrará en las diferentes pestañas las señales ha medir elegidas, es decir, si se eligen dos de temperatura y una de presión, en la pestaña principal aparecerán los valores en vivo en ºC y en bar marcando los elegidos, el resto no aparecerán. En la segunda pestaña, se podrá ver los gráficos en directo de las señales tomadas. Esto se puede ver en la figura 42 que muestra el funcionamiento descrito anteriormente para la interfaz del DAQ, en la siguiente, la figura 43, se plasma la pestaña en directo en la que se puede observar el gráfico junto al valor de la medida.

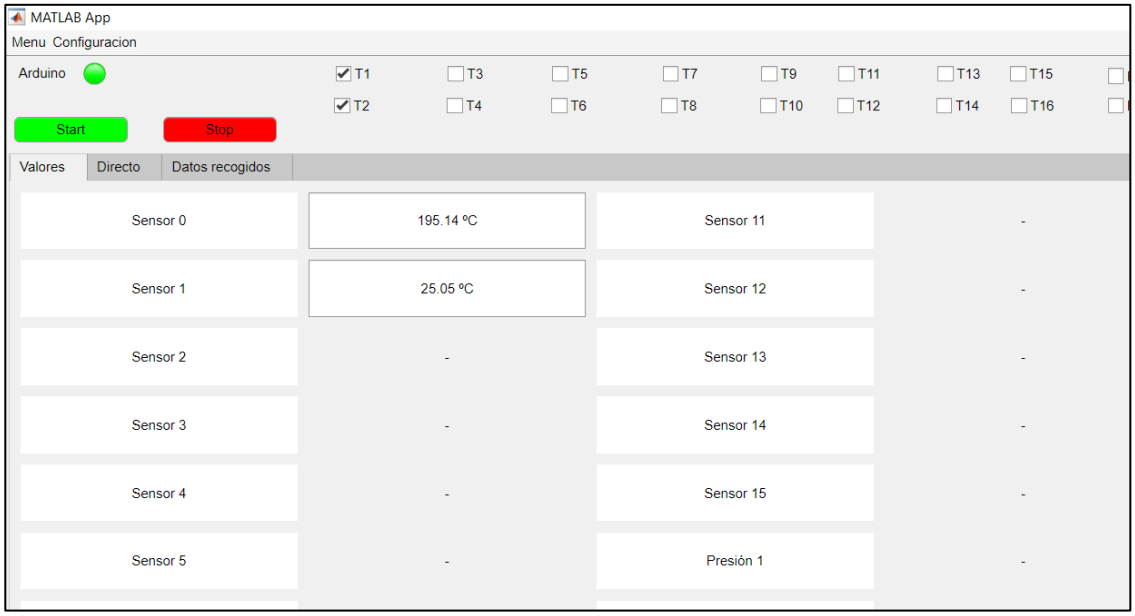


Figura 42. Muestra de la pestaña de valores en la interfaz

Figura 43. Muestra de la pestaña directo en la interfaz

6 EXPERIMENTO

Para verificar que el sistema DAQ que se ha construido incluyendo el hardware y el software programado, funciona de forma correcta, en este capítulo se muestra la realización de varios experimentos en los que se han sometido diferentes sensores en tres medios distintos. El primero consistirá en conocer el gasto de un soplante utilizando un caudalímetro, en segundo lugar, la medida de temperatura de escape de un MCIA y ambiente. En último lugar, se obtendrá la presión medida en un regulador de presión con manómetro para realizar una comprobación entre el propio dispositivo y el sensor.

6.1 Experimento de señal de caudal

En primer lugar, se ha tomado la medida de caudal en un soplante centrifugo, figura 44. Este elemento está construido como una caracola, toma el aire en un lateral y gracias a su rodete impulsa el flujo de aire de forma radial o centrifuga aumentando su velocidad. El soplante de aire no tiene incorporado un variador de velocidad, por ello, utilizaremos válvulas manuales con la que poder controlar el caudal de aire que es impulsado para pasar por el caudalímetro.

Figura 44. Soplante centrifuga

En la figura 45 se muestra la instalación montada para la medida del caudal en un soplante, en la parte izquierda se puede observar el soplante, a la que se conecta una tubería con dos válvulas, una de ellas cerrada y a la segunda, a la que se le conectará el caudalímetro, será variable para tomar diferentes medidas de caudal.

También podemos apreciar la caja a la que vamos a conectar el caudalímetro, siendo este uno de los más complejos ya que requiere la conexión de cuatro terminales. Utilizando la conexión del DAQ, tendremos cubierto 3 de los 4 terminales que han de ser conectados, tensión de alimentación a 5 V, señal de salida y tierra. Por otro lado, es necesario alimentar este dispositivo con una tensión de 14 V independiente de la de referencia, es decir la de 5 V, para ello utilizaremos una fuente de laboratorio generando dicha tensión.

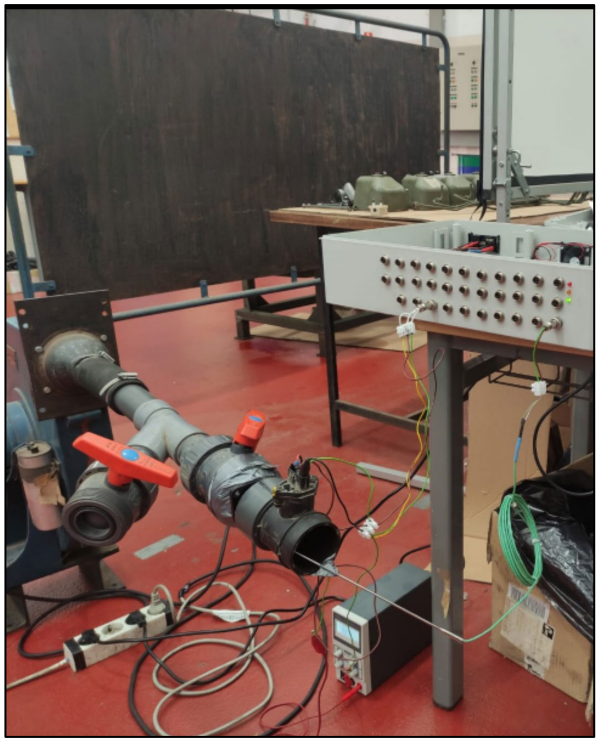


Figura 45. Montaje para medida de caudal con soplante

Se comienza tomando las muestras de caudal variando la válvula que se indica anteriormente. Esta toma será en un periodo de tiempo de aproximadamente 4 minutos en el que, en la figura 46, vemos la variación de caudal debido a la apertura y cierre manual de la válvula. El caudal varía desde los 0 kg/h hasta los 200 kg/h que es el máximo caudal al que puede impulsar el soplante de aire.

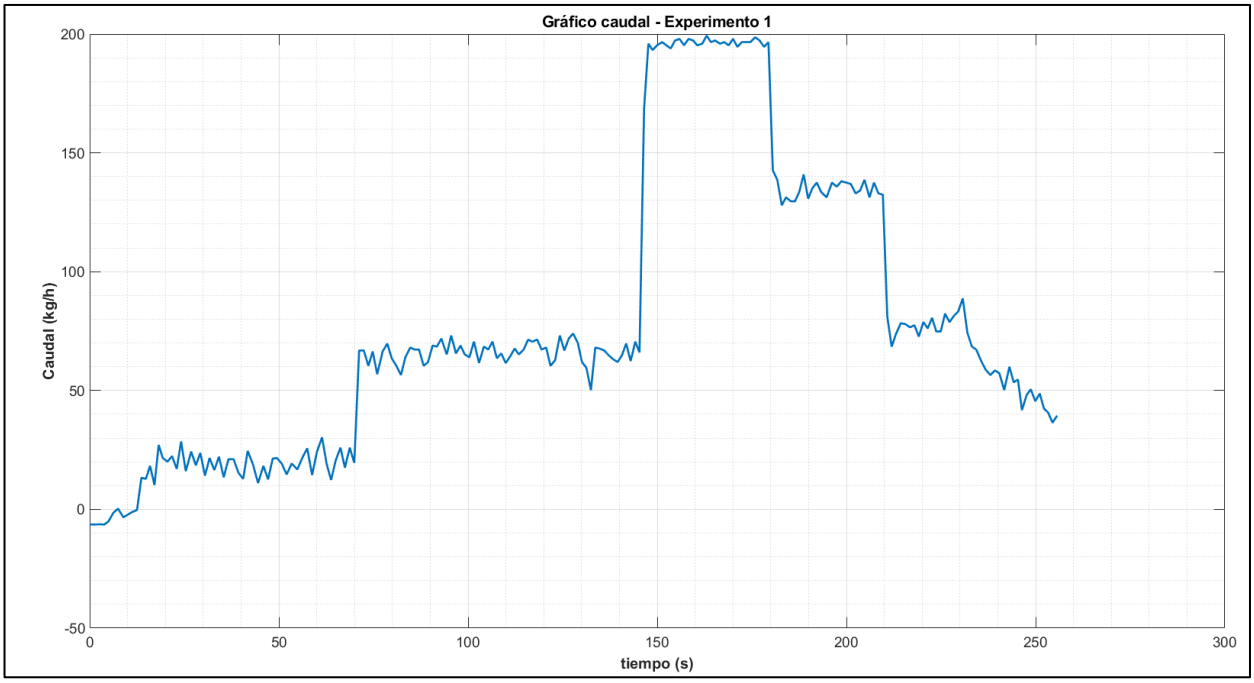


Figura 46. Gráfico de caudal para experimento 1

Ya que existe ruido en la medida de la señal, es posible el filtrado de esta y así dar una muestra de los resultados más suave para poder estudiarla posteriormente. Este filtrado depende del nivel de ruido que pueda ser producido por el dispositivo de medida. Para poder corregir la medida, se ha utilizado un *script* de MATLAB llamado *SmoothData* con el que podemos suavizar la curva generada por el DAQ. Este *script* requiere introducir los datos que han sido guardados, es decir, el caudal medido y el tiempo. Por otro lado, se ha de elegir el método de suavizado, de los cuales el programa da a elegir 8, los tres más utilizados para el filtrado de las señales en este trabajo será; *Gaussian*, *move mean* y *move median.* A esto hay que incluir el factor de suavizado o *Smooth Factor* con el que si es aumentado la curva será más suave y tenderá a ser horizontal, por lo que este factor debe ir alrededor de 0,05 hasta 0,3 para que pueda ajustarse adecuadamente a la curva medida.

Tanto en este experimento como en el resto se utilizará este método para realizar un filtrado de la señal y dar unos resultados gráficos de manera que sea posible su interpretación de una forma más sencilla.

Por todo ello, en la figura 47 se muestra un gráfico mostrando dos líneas, una punteada azul en la que se representa la señal medida sin modificar y la segunda, una línea continua roja que muestra la señal tras realizar el proceso de filtrado.

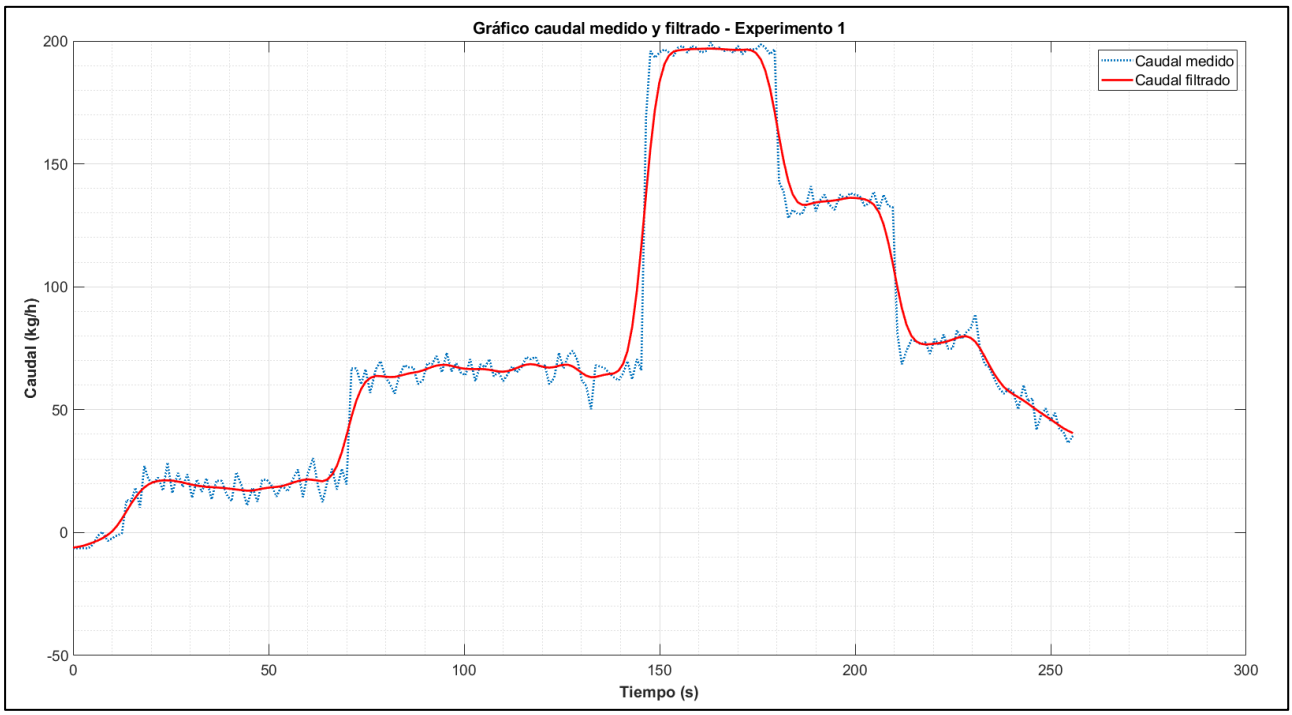


Figura 47. Señal de caudal medido y filtrado para experimento 1

6.2 Experimento de señal de temperatura

El segundo experimento tratará de tomar medidas de temperatura en un MCIA. Este motor es un Scania modelo DS 9 61 A 24 MIL, utilizado por el ejército español de tierra en el vehículo BMR (Blindado Medio sobre Ruedas), montado sobre una bancada en el laboratorio de máquinas y motores térmicos de la Escuela Técnica Superior de Ingeniería de la Universidad de Sevilla para la realización de ensayos de verificación de parámetros del motor. Sus características principales son; Motor de 6 cilindros de 9 litros con turbo diésel y 306 CV de potencia.

El procedimiento de ensayo consiste en la colocación de dos termopares para la toma de temperatura en dos puntos. El primer punto será en el escape del motor, un lugar de alta temperatura puesto que son gases que se producen en la combustión y son expulsados por ese conducto. En un segundo punto, se colocará un termopar

tomando la temperatura del aire que se introduce en el motor. Dependiendo del valor de esta magnitud, puede afectar al rendimiento volumétrico repercutiendo en la potencia del motor. En consecuencia, esta medida es importante para un ensayo de un MCIA pues es un parámetro que influye a los factores correctores de las magnitudes medidas en el motor.

En la figura 48 se muestra el motor colocado sobre la bancada dentro de una celda del laboratorio. Esta bancada cuenta con un freno hidráulico que está acoplado al motor mediante un elemento elástico al volante motor. Más en detalle, se puede observar el tubo plateado en el que está conectado el escape del motor, en ese punto es donde se realizará la medición de temperatura, por otro lado, en la celda se colocará el otro termopar para, como se ha explicado anteriormente, tomar la medida de temperatura ambiental. La figura 49 muestra en detalle la posición del termopar para tomar la temperatura en el escape del motor. Ambas figuras se han marcado con un recuadro rojo para diferenciar el termopar colocado en el motor.

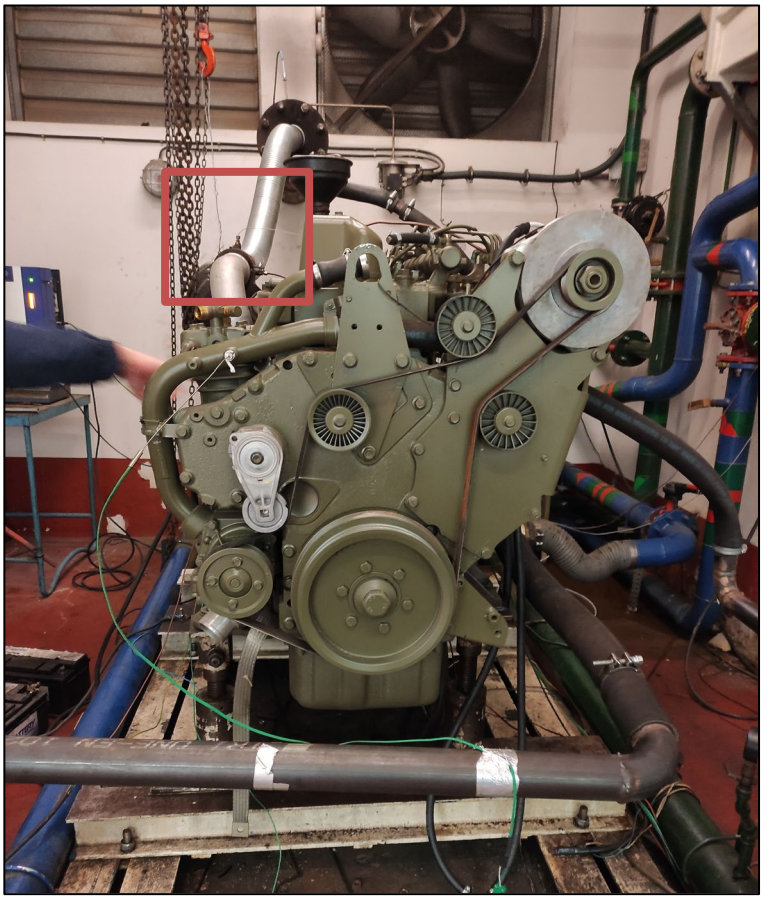


Figura 48. Motor Scania DS 9 61 A 24 MIL

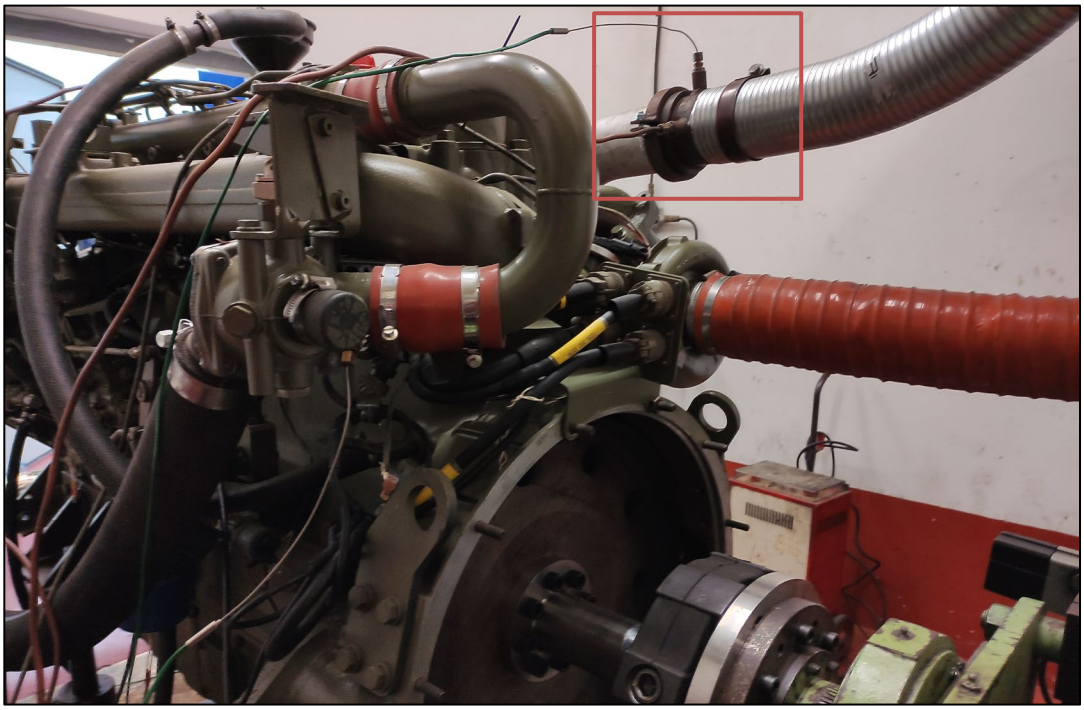
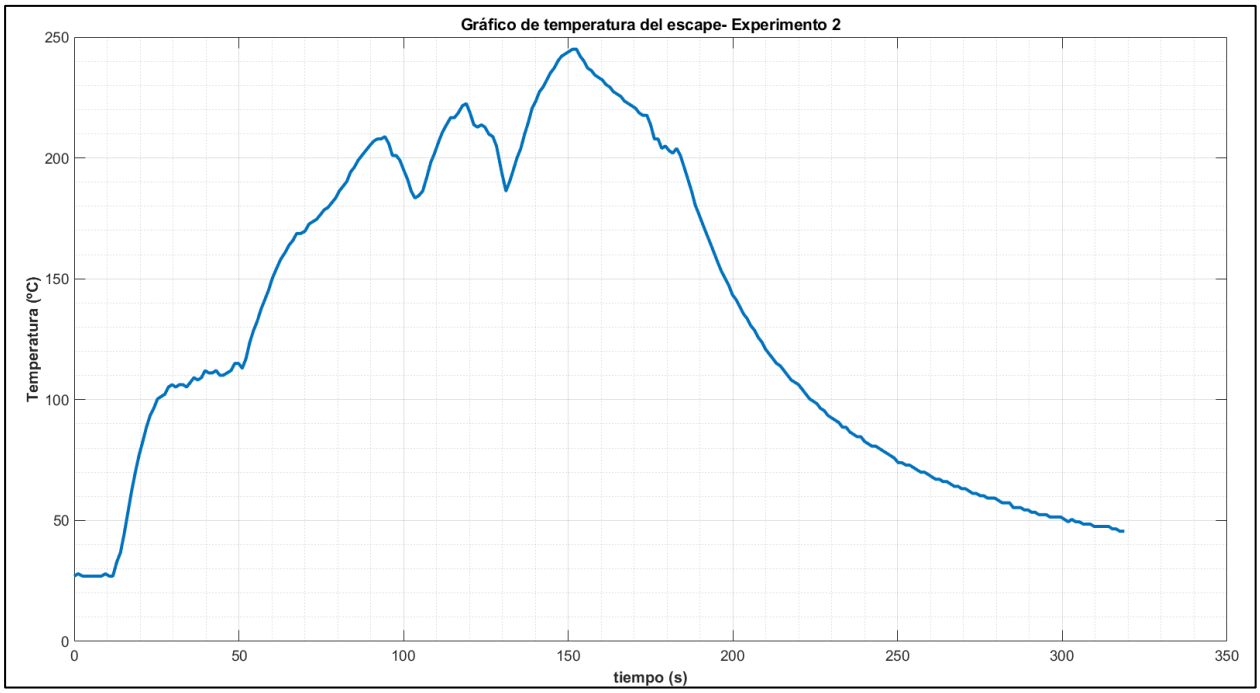
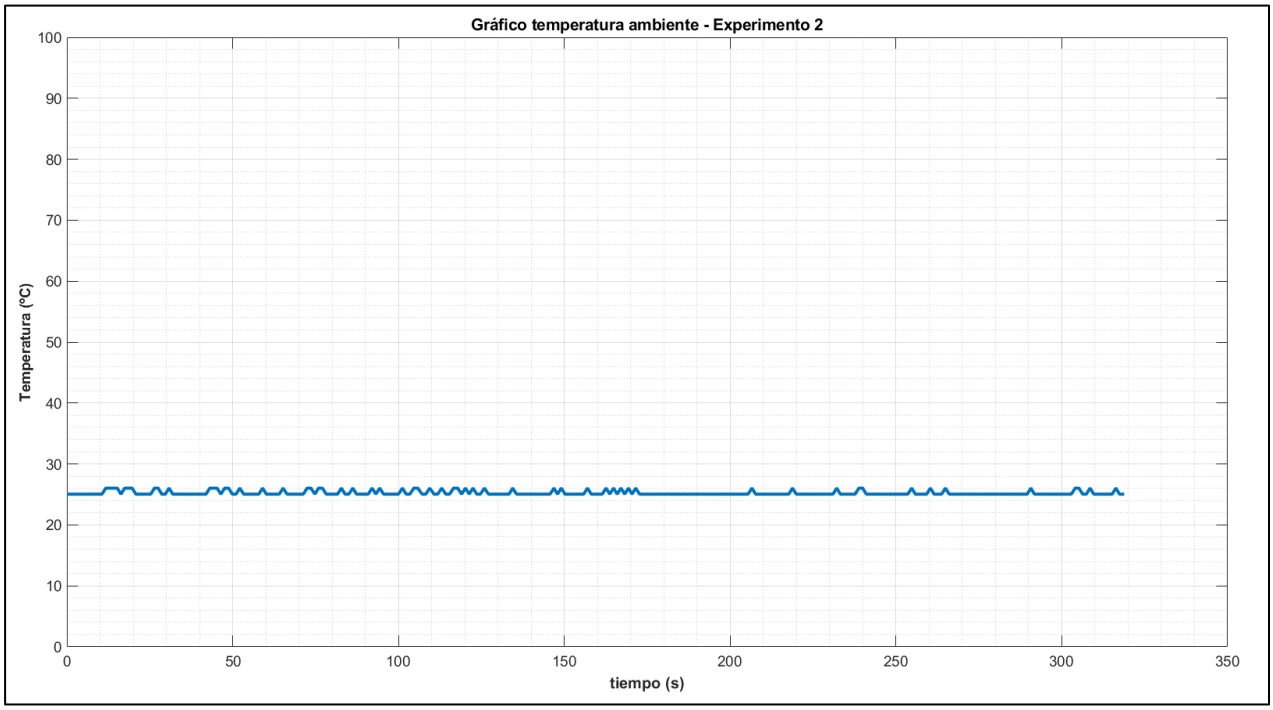


Figura 49. Posición del termopar para la toma de temperatura en el escape del motor

Tras adquirir la señal de temperatura obtenemos la figura 50 que representan la temperatura en función del tiempo del escape del motor y temperatura ambiente.



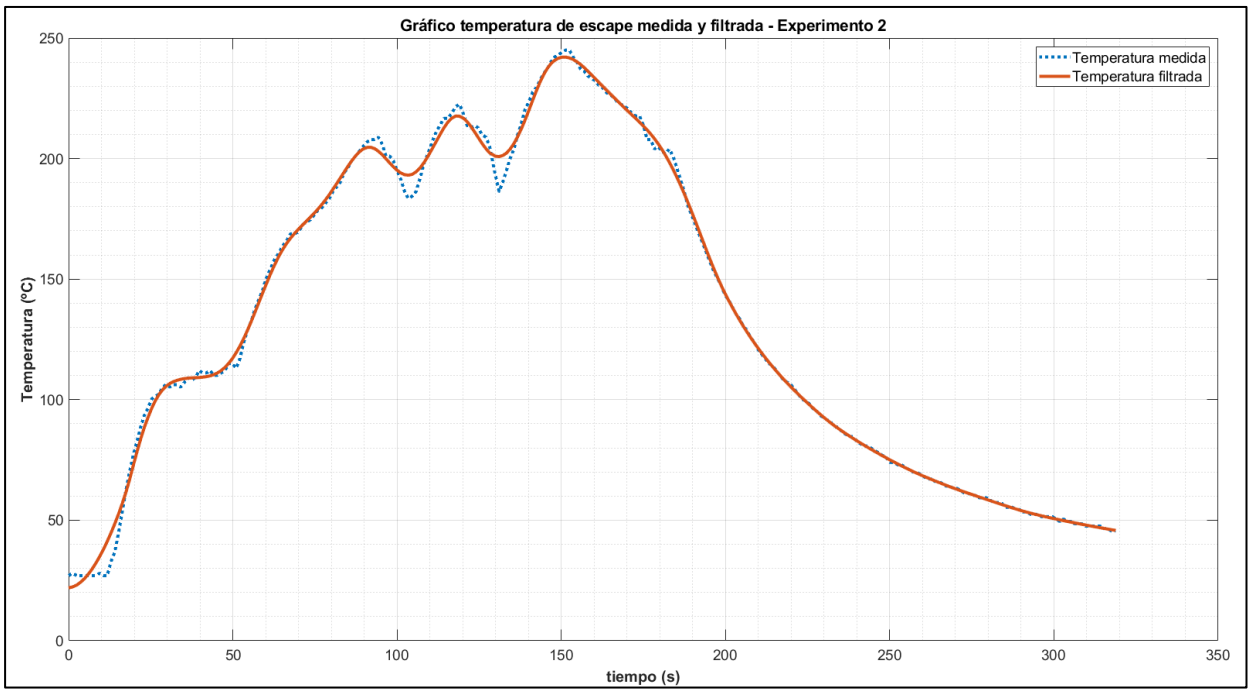
(a)



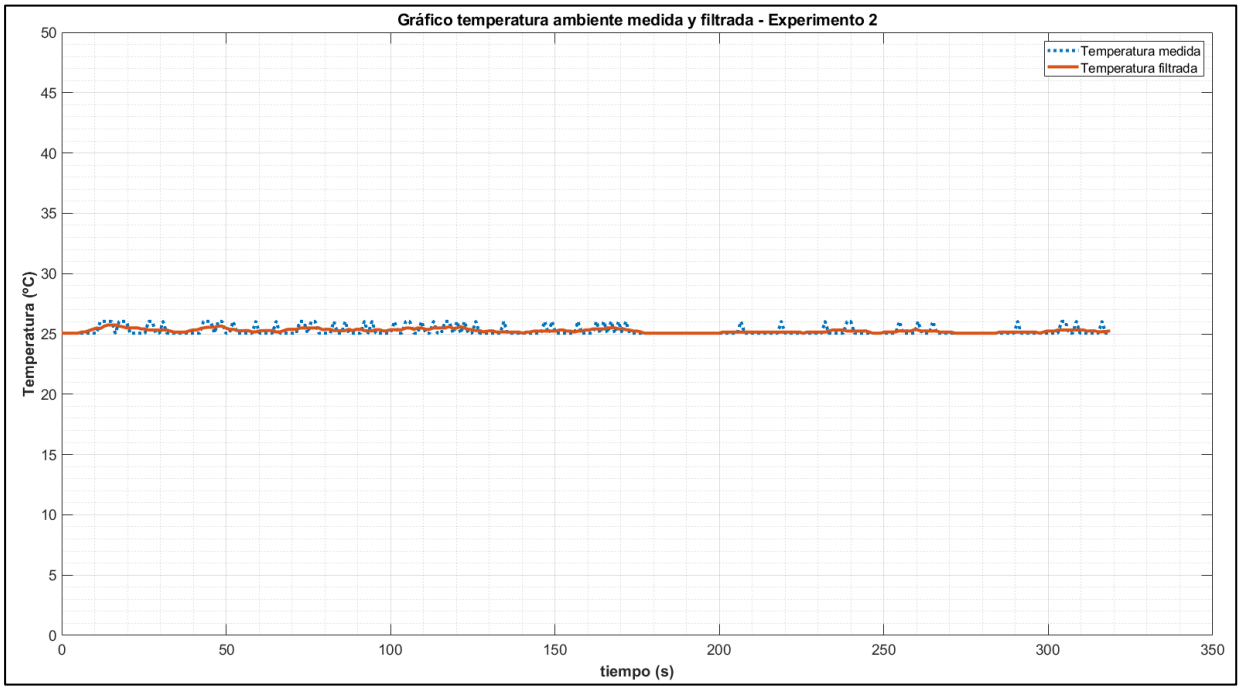
(b)

Figura 50. (a) Temperatura de escape del motor (b) Temperatura ambiente de la celda

Por otro lado, como se describe en el experimento 1, se vuelve a tomar los datos medidos y se filtran con el *script* para obtener una curva suave que será utilizada para tomar un histórico de los resultados. Por tanto, en la figura 51 se representan las curvas medidas en línea punteada azul y en línea continua roja la filtrada. Cabe destacar que el cambio de temperatura tomada en el escape se debe a el arranque del motor durante un tiempo para que subiera la temperatura y seguido de esto se para el motor y así ver como se disipa la temperatura por la tubería de escape.



(a)



(b)

Figura 51. (a) Señal medida y filtrada de la temperatura de escape (b) Señal medida y filtrada de la temperatura ambiente

6.3 Experimento de señal de presión

El tercer experimento consiste en la toma de presión a la salida de un regulador de presión, es decir, se utiliza una línea de aire a presión que se encuentra presente en el laboratorio y se conecta al regulador con manómetro con el que será posible controlar la presión de aire y así tomar la medida. El objetivo de esto será conocer si se corresponde la presión medida en el manómetro con la que mide el sensor de presión.

El sensor utilizado para medir la presión se corresponde con un sensor MAP (*Manifold Absolute Pressure*), de los que se ha hablado en el capítulo 4.1.1, que son utilizados para la medida de presión en el colector de admisión de los vehículos. Dependiendo del modelo, este puede medir presión y temperatura, este, sin embargo, solo es posible utilizarlo para medir presión. Como ya es sabido, requieren una recta de calibración puesto que la medida es en voltaje y se ha de convertir a presión. En este caso, los parámetros a introducir en el programa, diseñado en el punto 5.3.3, son para $a = 0,6$ y para $b=-0,06$ que se ha de incorporar en la ecuación 13. Este sensor de Bosch de hasta 3 bar de presión para los 5 V se muestra en la figura 52. Adicionalmente, se añade en la figura 53 la curva con la que se obtienen los parámetros *a y b* mencionados anteriormente.

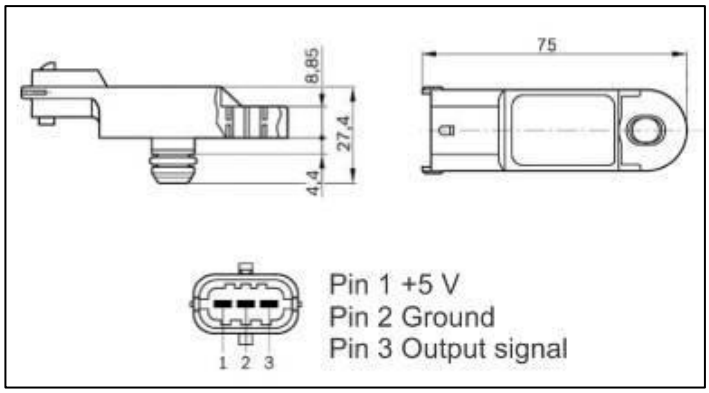


Figura 52. Sensor de presión Bosch utilizado para experimento 3

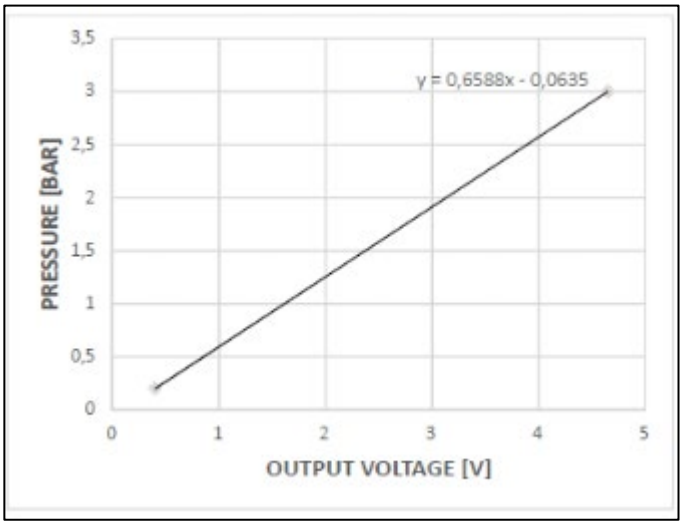


Figura 53. Recta de calibración del sensor de presión

El experimento comienza con la válvula del regulador de presión cerrada, por lo que, en primer lugar, se medirá una presión muy próxima a la atmosférica. Siguiendo con la dinámica, se irá aumentando la presión y por tanto abriendo la válvula hasta ir tomando medidas de entre 1 y 3 bar. Finalmente, se vuelve al inicio cerrando la válvula al completo en el que se comprueba que volvemos a la presión atmosférica. Gracias a esta prueba podemos corroborar que el sistema DAQ funciona con la señal de presión y que el sensor utilizado funciona correctamente.

En siguiente figura, número 54, se representa la señal medida de presión durante una duración determinada con el procedimiento anteriormente descrito. Por otro lado, en la figura 55 se muestra la representación de la curva de presión medida por el sistema DAQ, en color azul punteado, y la curva filtrada con una línea roja y continua.

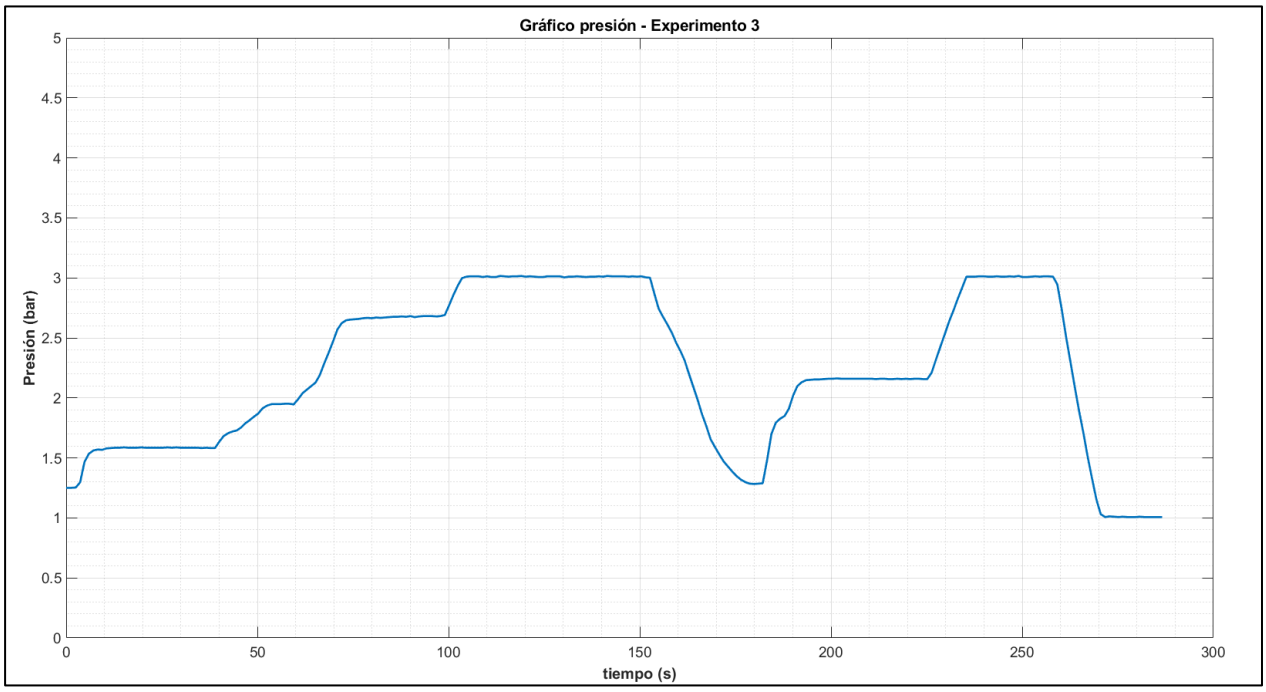


Figura 54. Gráfico de presión para experimento 3

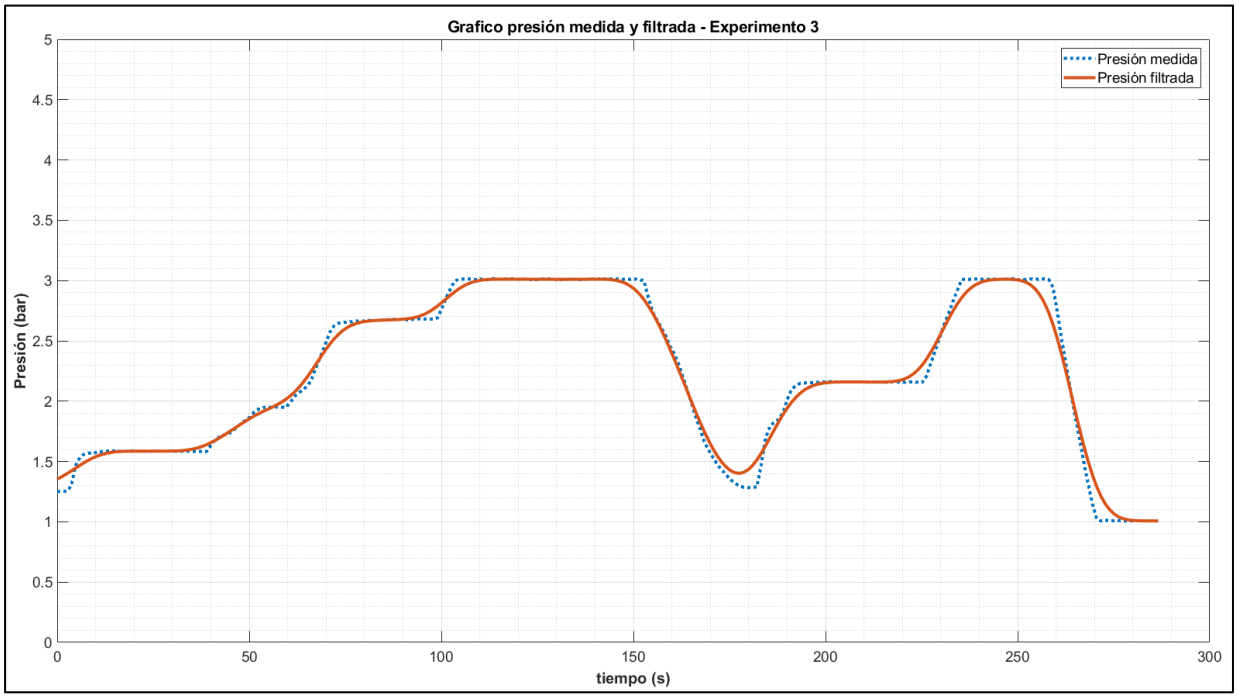


Figura 55. Señal medida y filtrada para experimento 3

7 CONCLUSIONES

Este capítulo concluye el trabajo de fin de grado con las conclusiones globales, los objetivos cumplidos y describiendo las limitaciones del proyecto.

7.1 Conclusiones globales

Tras la finalización de este trabajo, se ha demostrado lo siguiente:

- Hay pocos estudios en este campo, por lo que, sería interesante seguir indagando en esta línea y realizar nuevas aportaciones y proyectos que desarrollen DAQ más avanzados desacoplados a los comerciales que permitan interpretar otros tipos de señales con distintas magnitudes.
- El montaje de un DAQ 3 veces más económico que uno comercial con unas características similares y con una posibilidad de actualización a largo plazo superior debido a que los comerciales no pueden cambiar sus características físicas.
- Se logró fabricar un sistema versátil y multidisciplinar pues se puede adaptar con facilidad y rapidez en diferentes aplicaciones, esto quiere decir, que no se limita su uso a adquirir señales de un MCIA si no que es posible utilizar para medir señales en otras máquinas o prácticas de laboratorio para el alumnado y que puedan visualizar y comprender los datos.
- Se ha creado un prototipo modificable con el objetivo de que cualquier usuario pueda alterar sus componentes, atendiendo a las especificaciones plasmadas en este trabajo. Es decir, que pueda añadir, quitar o cambiar los dispositivos que se encuentran en su interior. A su vez, la aplicación ha sido diseñada para ser configurada de manera distinta a la versión original.

8 DESARROLLOS FUTUROS

Aunque el resultado del proyecto ha sido satisfactorio, hay aspectos que deben ser mejorados en un futuro para que este sistema pueda ofrecer un servicio más completo e incluso competir con los comerciales. Sería interesante seguir esta línea de investigación para aportar más valor e información sobre la aplicación y el desarrollo de los DAQ en diferentes áreas y conseguir que este sistema tenga un mayor reconocimiento y uso.

Con respecto a esto, se propone:

- Incorporar convertidores de señales para transformar distintas magnitudes a voltaje que puedan ser leídas por Arduino, esto permitirá el uso de nuevos dispositivos de medida añadidos a los que han sido utilizados en este trabajo.
- Mejora de la aplicación diseñada en *Appdesigner* integrando funcionalidades adicionales para que la toma de datos sea más rápida y precisa. Esto será posible con la modificación del código integrado en la herramienta.

REFERENCIAS

[1] Simon S. Young, "Computerized Data Acquisition and Analysis for the Life Sciences," *Cambridge University Press*, 2001, doi: https://doi.org/10.1017/CBO9780511609558.

[2] O. E. Mohammed Abdallah, "A Survey on Data Acquisition systems DAQ," *IEEE*, Apr. 2009, doi: 10.1109/ICC.2009.24.

[3] D. Potter, "Data Acquisition Fundamentals," *Instrument Engineers' Handbook*, no. January 2012, pp. 330–341, 2011, doi: 10.1201/b11093-23.

[4] I. Bosch Roig, J. Gosálbez Castillo, R. Miralles Ricós, and L. Vergara Domínguez, "Señales y Sistemas Teoría y problemas," 2015, [Online]. Available: www.lalibreria.upv.es

[5] "Thermocouple Hub." https://www.omega.com/en-us/resources/thermocouple-hub (accessed Feb. 03, 2022).

[6] "What is an RTD | Understanding RTD Sensors | TE Connectivity." https://www.te.com/usa-en/industries/sensor-solutions/insights/understanding-rtds.html (accessed Feb. 03, 2022).

[7] "¿Qué es un termistor? | OMEGA Engineering." https://es.omega.com/prodinfo/termistores.html (accessed Feb. 03, 2022).

[8] A. Abacus, "Pressure Sensors : The Design Engineer ' s Guide MEMS pressure sensors," p. 97, 2019.

[9] C. D. Alonso, "ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN UNIVERSIDAD DE CANTABRIA Proyecto Fin de Carrera ANÁLISIS COMPARATIVO ENTRE INSTRUMENTACION CLÁSICA DE GALGAS EXTENSIOMÉTRICAS Y SENSORES ÓPTICOS PARA LA MONITORIZACIÓN EN UNA CÉLULA DE CARGA (Comparative analysis between the classical approach by means of gauge extensometers and optical sensors for the monitoring of load cells) Para acceder al Título de INGENIERO DE TELECOMUNICACIÓN."

[10] "¿Qué es un sensor de presión? | HBM." https://www.hbm.com/es/7646/que-es-un-sensor-de-presion/ (accessed May 19, 2022).

[11] "¿Qué es un sensor capacitivo? - INFORMACIÓN DE AUTOMATIZACIÓN." https://automation-insights.blog/2017/06/07/what-is-a-capacitive-sensor/ (accessed Jun. 06, 2022).

[12] "¿Qué son las galgas extensiométricas? ¿Cómo se usan?" https://es.omega.com/prodinfo/galgas-extensiometricas.html (accessed May 19, 2022).

[13] "China Capacitive Pressure Sensor Electronic Air Pressure Sensor for Compressor Manufacturers, Suppliers - Factory Direct Price - Firstrate." https://www.firstrate-sensor.com/showroom/capacitive-pressure-sensor-electronic-air-pressure-sensor-for-compressor.html (accessed Jul. 01, 2022).

[14] "MPM280 03 G 0 L 1 Micro Sensor | Sensores y transductores | DigiKey Marketplace." https://www.digikey.es/es/products/detail/micro-sensor/MPM280-03-G-0-L-1/14544347 (accessed Jul. 01, 2022).

[15] L. García Gutiérrez, "TEORÍA DE LA MEDICIÓN DE CAUDALES Y VOLÚMENES DE AGUA

E INSTRUMENTAL NECESARIO DISPONIBLE EN EL MERCADO."

[16] J. C. Massón, "Inductivo distancia.jpg - Wikimedia Commons," 2006. https://commons.wikimedia.org/wiki/File:Inductivo_distancia.jpg (accessed Jun. 06, 2022).

[17] M. Arenas Mas, "Sistema para la adquisición y monitorización de aceleraciones mediante microprocesador," Sevilla, 2008.

[18] E. Bayona, "Atlas de aleraciones del cuerpo humano en movimiento de carrera y trabajo en escaleras," Cantabria, 2012. [Online]. Available: https://www.researchgate.net/publication/233425138

[19] Measurement Computing, "Data Acquisition Handbook," 2004.

[20] N. Zlatanov, "Arduino and Open Source Computer Hardware and Software," *Journal of Water, Sanitation and Hygiene for Development*, vol. 10, no. 11, pp. 1–8, 2016, doi: 10.13140/RG.2.1.1071.7849.

[21] RogerBit, "Google presenta ADK, interfaz basada en Arduino para Android." https://rogerbit.com/wprb/2016/10/google-presenta-adk-interfaz-basada-en-arduino-para-android/ (accessed May 11, 2022).

[22] J. T. Calvo, "Desarrollo de una comunicación alfa arduino mediante un SCADA," pp. 1–106, 2015.

[23] A. Avr, H. Libre, and E. Ide, "1.1 ¿Qué es Arduino?," pp. 1–3.

[24] Adafruit, "How many Arduinos are 'in the wild?' About 300,000 « Adafruit Industries – Makers, hackers, artists, designers and engineers!" https://blog.adafruit.com/2011/05/15/how-many-arduinos-are-in-the-wild-about-300000/ (accessed May 11, 2022).

[25] ControlDesign, "10M Arduino Uno boards sold worldwide." https://www.controldesign.com/industrynews/2021/10m-arduino-uno-boards-sold-worldwide/ (accessed May 11, 2022).

[26] J. Williams, "Thermocouple measurement," *Analog circuit design. A tutorial guide to applications and solutions*, pp. 596–613, 2011, doi: 10.1016/B978-0-12-385185-7.00030-5.

[27] A. Devices, "AD8494/AD8495/AD8496/AD8497 (Rev. C) Precision Thermocouple Amplifiers with Cold Junction Compensation," 2010. [Online]. Available: www.analog.com

[28] T. Mathworks, "MATLAB ® Support Package for Arduino ® Hardware User ' s Guide," p. 182, 2020.

[29] "Data Acquisition Industry Applications ." https://www.gage-applied.com/data-acquisition/applications/index.htm (accessed Jan. 31, 2022).

[30] N. Abdukarimov, "Engineering Master of Science on Mechatronics Easy DAQ System Development with Arduino ® and Tri-Axis Accelerometer – Model ADXL335," no. July, 2019, doi: 10.13140/RG.2.2.23474.04804.

[31] F. Khan and A. Masood, "Arduino Based Control And Data Acquisition System Using Python Graphical User Interface (GUI)," no. November, 2021.

[32] J. Casauco Liger, "Relación entre temperatura y luz en un acuario, analizando datos adquiridos con Arduino y Matlab," Sevilla, 2017.

[33] N. Sinaga, B. Yunianto, D. Purba, Syaiful, and A. Nugroho, "Design and Manufacture of a Low-Cost Data Acquisition Based Measurement System for Dual Fuel Engine Researches," in *IOP Conference Series: Materials Science and Engineering*, Sep. 2019, vol. 598, no. 1. doi: 10.1088/1757-

899X/598/1/012031.

[34] C. A. Lozano-Garzón and C. Gustavo Alvarez, “Low cost data adquisition system to register fuel consumption in diesel engine vehicles,” *IEEE*, 2018.

[35] K. Reif Ed, “Diesel Engine Management Systems and Components Bosch Professional Automotive Information.”

[36] R. Bosch GmbH, “Bosch Automotive Electrics and Automotive Electronics.”

[37] M. C. Bautista, “Aplicación para el ensayo en serie de MCIA,” 2020.

Anexo A: Código de *Appdesigner*

- Cambios para realizar en MATLAB para el correcto funcionamiento de Arduino
- Códigos de *Appdesigner*
    - Properties
    - StartupFcn
    - StartButtonPushed
    - StopButtonPushed
    - GuardarMenuSelected
    - ConectarMenuSelected
    - DesconectarMenuSelected
    - T1CheckBoxValueChanged
    - UIFigureCloseRequest
    - ReiniciarMenuSelected
    - ParametrosdepresionMenuSelected

Cambios para realizar en MATLAB para el correcto funcionamiento de Arduino

Debido a que Arduino puede que no funcione correctamente en un ordenador que no haya sido configurado anteriormente, se ha de seguir los pasos que se describen a continuación para que funcione correctamente:

1. Escribir en *Command Window* de MATLAB el siguiente código:

```
which -all arduino
```

2. Si en la ventana de comando no aparece ProgramData si no que aparece otro como ProgramFiles, escribir en la barra de dirección de búsqueda lo siguiente, en caso contrario, copiar la dirección que nos proporciona el comando anterior:

```
C:\ProgramData\MATLAB\SupportPackages\R2021b\toolbox\matlab\hardware\supportpackages\arduinoio\@arduino\arduino.m
```

   Se recomienda utilizar esta dirección si se ha instalado MATLAB en la dirección local del disco C:\, si no es así cambiar el disco por la letra correspondiente. Realizar el mismo procedimiento si se ha de cambiar la versión instalada.

3. Al abrir el archivo *arduino.m* se ha de encontrar con la herramienta de búsqueda de MATLAB la siguiente línea de código:

```
function delayValue = getResetDelayHook(obj)
  % workaround for Arduino Mega on Linux taking longer time to reset, explained in g1638539
  if(obj.ConnectionType==matlabshared.hwsdk.internal.ConnectionTypeEnum.Serial)
    if ispc || ismac
      delayValue = 2;
    else
      delayValue = 10;
    end
  else
    %The reset on the DTR line is via Serial only
    delayValue = 0;
  end
end
```

Al siguiente código con el único cambio en la línea *1002*, si se corresponde con el archivo, en el que *delayValue = 2,* y pasa a ser *delayValue = 10.*

```
function delayValue = getResetDelayHook(obj)

  % workaround for Arduino Mega on Linux taking longer time to reset, explained in
g1638539

  if(obj.ConnectionType==matlabshared.hwsdk.internal.ConnectionTypeEnum.Serial)

    if ispc || ismac

      delayValue = 10;

    else

      delayValue = 10;

    end

  else

    %The reset on the DTR line is via Serial only

    delayValue = 0;

  end

end
```

Códigos de Appdesigner

Properties

```
properties (Access = private)
        %Variables para arduino
        a;
        a1;
        pin1;
        pin2;
        pin3;
        pin4;
        pin5;
        pin6;
        pin7;
        pin8;
        pin9;
        pin10;
        pin11;
        pin12;
        pin13;
        pin14;
        pin15;
        pin16;
        %Graficos
        stop;
        done=0 ;
        %Base de datos
        h1;
        h2;
        h3;
        h4;
        h5;
        h6;
        h7;
        h8;
        h9;
        h10;
        h11;
        h12;
        h13;
        h14;
        h15;
        h16;
        hp1;
        hp2;
        hp3;
        hp4;
        hp5;
        hp6;
        hp7;
        hp8;

        %timelogs
          timeLogs;
```

```
        timeLT;
        timeLogs_p;
        timeLTP;
        timeSecs;
        timeST;
        timeSecs_p;
        timeSTP;

      %logs
        TempLogs;
        TempLT;
        PresLogs;
        PresLT;

      %tablas
        tablaTemp;
        tablaPres;
        tablaTiempo;

        P;

      %fecha
         fecha;
         cfecha;
      %Lectura de parametros de presión

         px11;
         px12;
         px21;
         px22;
         px31;
         px32;
         px41;
         px42;
      end
end

methods (Access = public)

function confirmClose(v2_arduinocopia, ~, event)
            % Determine which dialog button the user clicked
            answer = event.SelectedOption;

            % Close the app if the user clicks OK
            if strcmp(answer,'OK')
                delete(v2_arduinocopia);
            end
        end
    end
```

StartupFcn

```
app.ArduinoLamp.Color='yellow';
   %Lectura de parametros presion
```

```
pcomp=readtable('ParametrosPresion.xlsx');

app.px11=table2array(pcomp(1,2));
         app.px12=table2array(pcomp(1,3));
         app.px21=table2array(pcomp(2,2));
         app.px22=table2array(pcomp(2,3));
         app.px31=table2array(pcomp(3,2));
         app.px32=table2array(pcomp(3,3));
         app.px41=table2array(pcomp(4,2));
         app.px42=table2array(pcomp(4,3));
         app.px51=table2array(pcomp(5,2));
         app.px52=table2array(pcomp(5,3));
         app.px61=table2array(pcomp(6,2));
         app.px62=table2array(pcomp(6,3));
         app.px71=table2array(pcomp(7,2));
         app.px72=table2array(pcomp(7,3));
         app.px81=table2array(pcomp(8,2));
         app.px82=table2array(pcomp(8,3));
```

StartButtonPushed

```
%-----------------Crear gráficos----------------


%------Temperaturas-----
            %h1
            app.h1=animatedline(app.LiveT);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 100];


            ylim(app.LiveT,[0 100]);
            app.LiveT.YGrid='on';


            app.LiveT.XGrid='on';


            %h2
            app.h2=animatedline(app.LiveT_2);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 100];


            ylim(app.LiveT_2,[0 100]);
```

```
app.LiveT_2.YGrid='on';
app.LiveT_2.XGrid='on';

%h3
app.h3=animatedline(app.LiveT_3);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_3,[0 100]);
app.LiveT_3.YGrid='on';
app.LiveT_3.XGrid='on';

%h4
app.h4=animatedline(app.LiveT_4);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_4,[0 100]);
app.LiveT_4.YGrid='on';
app.LiveT_4.XGrid='on';

%h5
app.h5=animatedline(app.LiveT_5);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_5,[0 100]);
app.LiveT_5.YGrid='on';
app.LiveT_5.XGrid='on';

%h6
app.h6=animatedline(app.LiveT_6);
ax=gca;
```

```
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_6,[0 100]);
app.LiveT_6.YGrid='on';
app.LiveT_6.XGrid='on';

%h7
app.h7=animatedline(app.LiveT_7);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_7,[0 100]);
app.LiveT_7.YGrid='on';
app.LiveT_7.XGrid='on';

%h8
app.h8=animatedline(app.LiveT_8);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_8,[0 100]);
app.LiveT_8.YGrid='on';
app.LiveT_8.XGrid='on';

%h9
app.h9=animatedline(app.LiveT_9);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_9,[0 100]);
app.LiveT_9.YGrid='on';
app.LiveT_9.XGrid='on';
```

```
%h10
app.h10=animatedline(app.LiveT_10);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_10,[0 100]);
app.LiveT_10.YGrid='on';
app.LiveT_10.XGrid='on';

%h11
app.h11=animatedline(app.LiveT_11);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_11,[0 100]);
app.LiveT_11.YGrid='on';
app.LiveT_11.XGrid='on';

%h12
app.h12=animatedline(app.LiveT_12);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_12,[0 100]);
app.LiveT_12.YGrid='on';
app.LiveT_12.XGrid='on';

%h13
app.h13=animatedline(app.LiveT_13);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];
```

```
ylim(app.LiveT_13,[0 100]);
app.LiveT_13.YGrid='on';
app.LiveT_13.XGrid='on';

%h14
app.h14=animatedline(app.LiveT_14);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_14,[0 100]);
app.LiveT_14.YGrid='on';
app.LiveT_14.XGrid='on';

%h15
app.h15=animatedline(app.LiveT_15);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_15,[0 100]);
app.LiveT_15.YGrid='on';
app.LiveT_15.XGrid='on';

%h16
app.h16=animatedline(app.LiveT_16);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 100];

ylim(app.LiveT_16,[0 100]);
app.LiveT_16.YGrid='on';
app.LiveT_16.XGrid='on';
```

```
%-------Presion------
            %hp1
            app.hp1=animatedline(app.LiveP_1);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 10];

            ylim(app.LiveP_1,[0 10]);
            app.LiveP_1.YGrid='on';
            app.LiveP_1.XGrid='on';

            %hp2
            app.hp2=animatedline(app.LiveP_2);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 10];

            ylim(app.LiveP_2,[0 10]);
            app.LiveP_2.YGrid='on';
            app.LiveP_2.XGrid='on';

            %hp3
            app.hp3=animatedline(app.LiveP_3);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 10];

            ylim(app.LiveP_3,[0 10]);
            app.LiveP_3.YGrid='on';
            app.LiveP_3.XGrid='on';

            %hp4
            app.hp4=animatedline(app.LiveP_4);
            ax=gca;
            ax.YGrid='on';
            ax.YLim=[0 10];
```

```
ylim(app.LiveP_4,[0 10]);
app.LiveP_4.YGrid='on';
app.LiveP_4.XGrid='on';

%hp5
app.hp5=animatedline(app.LiveP_5);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 10];

ylim(app.LiveP_5,[0 10]);
app.LiveP_5.YGrid='on';
app.LiveP_5.XGrid='on';

%hp6
app.hp6=animatedline(app.LiveP_6);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 10];

ylim(app.LiveP_6,[0 10]);
app.LiveP_6.YGrid='on';
app.LiveP_6.XGrid='on';

%hp7
app.hp7=animatedline(app.LiveP_7);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 10];

ylim(app.LiveP_7,[0 10]);
app.LiveP_7.YGrid='on';
app.LiveP_7.XGrid='on';
```

```
%hp8
app.hp8=animatedline(app.LiveP_8);
ax=gca;
ax.YGrid='on';
ax.YLim=[0 200];

ylim(app.LiveP_8,[0 200]);
app.LiveP_8.YGrid='on';
app.LiveP_8.XGrid='on';

app.stop=false;
startTime=datetime('now');

%Paso de parametros de presion para entrar al bucle
    ap1=app.px11;
    ap2=app.px12;
    ap3=app.px21;
    ap4=app.px22;
    ap5=app.px31;
    ap6=app.px32;
    ap7=app.px41;
    ap8=app.px42;
    ap9=app.px51;
    ap10=app.px52;
    ap11=app.px61;
    ap12=app.px62;
    ap13=app.px71;
    ap14=app.px72;
    ap15=app.px81;
    ap16=app.px82;


%Muestra en vivo
while ~app.stop
```

```
%Señales de arduino
                v0=readVoltage(app.a,'A0');
                v1=readVoltage(app.a,'A1');
                v2=readVoltage(app.a,'A2');
                v3=readVoltage(app.a,'A3');
                v4=readVoltage(app.a,'A4');
                v5=readVoltage(app.a,'A5');
                v6=readVoltage(app.a,'A6');
                v7=readVoltage(app.a,'A7');
                v8=readVoltage(app.a,'A8');
                v9=readVoltage(app.a,'A9');
                v10=readVoltage(app.a,'A10');
                v11=readVoltage(app.a,'A11');
                v12=readVoltage(app.a,'A12');
                v13=readVoltage(app.a,'A13');
                v14=readVoltage(app.a,'A14');
                v15=readVoltage(app.a,'A15');
                vp1=readVoltage(app.a1,'A0');
                vp2=readVoltage(app.a1,'A1');
                vp3=readVoltage(app.a1,'A2');
                vp4=readVoltage(app.a1,'A3');
                vp5=readVoltage(app.a1,'A4');
                vp6=readVoltage(app.a1,'A5');
                vp7=readVoltage(app.a1,'A6');
                vp8=readVoltage(app.a1,'A7');
 %-------------------------------------------------------------
                %Ecuaciones
                TempC1=(v0-1.242-0.00125)/0.005;
                TempC2=(v1-1.242-0.00125)/0.005;
                TempC3=(v2-1.242-0.00125)/0.005;
                TempC4=(v3-1.242-0.00125)/0.005;
                TempC5=(v4-1.242-0.00125)/0.005;
                TempC6=(v5-1.242-0.00125)/0.005;
                TempC7=(v6-1.242-0.00125)/0.005;
                TempC8=(v7-1.242-0.00125)/0.005;
                TempC9=(v8-1.242-0.00125)/0.005;
```

```
            TempC10=(v9-1.242-0.00125)/0.005;
            TempC11=(v10-1.242-0.00125)/0.005;
            TempC12=(v11-1.242-0.00125)/0.005;
            TempC13=(v12-1.242-0.00125)/0.005;
            TempC14=(v13-1.242-0.00125)/0.005;
            TempC15=(v14-1.242-0.00125)/0.005;
            TempC16=(v15-1.242-0.00125)/0.005;

            Press1=ap1*vp1-ap2;
            Press2=ap3*vp2-ap4;
            Press3=ap5*vp3-ap6;
            Press4=ap7*vp4-ap8;
            Press5=ap9*vp5-ap10;
            Press6=ap11*vp6-ap12;
            Press7=ap13*vp7-ap14;
            Caud=23.066*vp8^2-47.067*vp8+14.063;
            Press8=Caud;
    %-------------------------------------------------------
            t=datetime('now')-startTime;

            %Añadir puntos a la grafica
            %Temp
            addpoints(app.h1,datenum(t),TempC1);
            addpoints(app.h2,datenum(t),TempC2);
            addpoints(app.h3,datenum(t),TempC3);
            addpoints(app.h4,datenum(t),TempC4);
            addpoints(app.h5,datenum(t),TempC5);
            addpoints(app.h6,datenum(t),TempC6);
            addpoints(app.h7,datenum(t),TempC7);
            addpoints(app.h8,datenum(t),TempC8);
            addpoints(app.h9,datenum(t),TempC9);
            addpoints(app.h10,datenum(t),TempC10);
            addpoints(app.h11,datenum(t),TempC11);
            addpoints(app.h12,datenum(t),TempC12);
            addpoints(app.h13,datenum(t),TempC13);
            addpoints(app.h14,datenum(t),TempC14);
```

```
addpoints(app.h15,datenum(t),TempC15);
addpoints(app.h16,datenum(t),TempC16);

%Press
addpoints(app.hp1,datenum(t),Press1);
addpoints(app.hp2,datenum(t),Press2);
addpoints(app.hp3,datenum(t),Press3);
addpoints(app.hp4,datenum(t),Press4);
addpoints(app.hp5,datenum(t),Press5);
addpoints(app.hp6,datenum(t),Press6);
addpoints(app.hp7,datenum(t),Press7);
addpoints(app.hp8,datenum(t),Press8);



ax.XLim=datenum([t-seconds(15) t]);
datetick('x','keeplimits');



%Muestra en pantalla
app.S1F.Value=TempC1;
app.TempField.Value=TempC1;
app.TempGauge.Value=TempC1;

app.S2F.Value=TempC2;
app.Temp_2Field.Value=TempC2;
app.Temp_2Gauge.Value=TempC2;

app.S3F.Value=TempC3;
app.Temp_3Field.Value=TempC3;
app.Temp_3Gauge.Value=TempC3;

app.S4F.Value=TempC4;
app.Temp_4Field.Value=TempC4;
app.Temp_4Gauge.Value=TempC4;
```

```
app.S5F.Value=TempC5;
app.Temp_5Field.Value=TempC5;
app.Temp_5Gauge.Value=TempC5;

app.S6F.Value=TempC6;
app.Temp_6Field.Value=TempC6;
app.Temp_6Gauge.Value=TempC6;

app.S7F.Value=TempC7;
app.Temp_7Field.Value=TempC7;
app.Temp_7Gauge.Value=TempC7;

app.S8F.Value=TempC8;
app.Temp_8Field.Value=TempC8;
app.Temp_8Gauge.Value=TempC8;

app.S9F.Value=TempC9;
app.Temp_9Field.Value=TempC9;
app.Temp_9Gauge.Value=TempC9;

app.S10F.Value=TempC10;
app.Temp_10Field.Value=TempC10;
app.Temp_10Gauge.Value=TempC10;

app.S11F.Value=TempC11;
app.Temp_11Field.Value=TempC11;
app.Temp_11Gauge.Value=TempC11;

app.S12F.Value=TempC12;
app.Temp_12Field.Value=TempC12;
app.Temp_12Gauge.Value=TempC12;

app.S13F.Value=TempC13;
app.Temp_13Field.Value=TempC13;
app.Temp_13Gauge.Value=TempC13;
```

```
app.S14F.Value=TempC14;
app.Temp_14Field.Value=TempC14;
app.Temp_14Gauge.Value=TempC14;

app.S15F.Value=TempC15;
app.Temp_15Field.Value=TempC15;
app.Temp_15Gauge.Value=TempC15;

app.S16F.Value=TempC16;
app.Temp_16Field.Value=TempC16;
app.Temp_16Gauge.Value=TempC16;

%Los de presión falta Gauge
app.SP1F.Value=Press1;
app.P_1Field.Value=Press1;

app.SP2F.Value=Press2;
app.P_2Field.Value=Press2;

app.SP3F.Value=Press3;
app.P_3Field.Value=Press3;

app.SP4F.Value=Press4;
app.P_4Field.Value=Press4;

app.SP5F.Value=Press5;
app.P_5Field.Value=Press5;

app.SP6F.Value=Press6;
app.P_6Field.Value=Press6;

app.SP7F.Value=Press7;
app.P_7Field.Value=Press7;

app.SP8F.Value=Press8;
```

```
        app.P_8Field.Value=Press8;

        drawnow

        writeDigitalPin(app.a,'D5',1);
        pause(0.01);
        writeDigitalPin(app.a,'D5',0);
        pause(0.01);

        app.stop=app.done;
    end
```

StopButtonPushed

```
app.done=1;

    writeDigitalPin(app.a,'D4',0);
    writeDigitalPin(app.a,'D5',1);
    writeDigitalPin(app.a,'D6',0);

    %--------GUARDADO DE DATOS------
    %h1
    if app.T1CheckBox.Value==1
    [app.timeLogs.T1,app.TempLogs.T1]=getpoints(app.h1);
    app.timeSecs.T1=(app.timeLogs.T1-app.timeLogs.T1(1))*24*3600;

    app.timeST.T1=app.timeSecs.T1';
    app.TempLT.T1=app.TempLogs.T1';



    plot(app.RecordT,app.timeSecs.T1,app.TempLogs.T1);
    grid minor;

    app.TmaxField.Value=max(app.TempLogs.T1);
    app.TminField.Value=min(app.TempLogs.T1);
```

```
end

%h2
 if app.T2CheckBox.Value==1
[app.timeLogs.T2,app.TempLogs.T2]=getpoints(app.h2);
app.timeSecs.T2=(app.timeLogs.T2-app.timeLogs.T2(1))*24*3600;

app.timeST.T2=app.timeSecs.T2';
app.TempLT.T2=app.TempLogs.T2';

plot(app.RecordT_2,app.timeSecs.T2,app.TempLogs.T2);
grid minor;


app.Tmax_2Field.Value=max(app.TempLogs.T2);
app.Tmin_2Field.Value=min(app.TempLogs.T2);

end

%h3
 if app.T3CheckBox.Value==1
[app.timeLogs.T3,app.TempLogs.T3]=getpoints(app.h3);
app.timeSecs.T3=(app.timeLogs.T3-app.timeLogs.T3(1))*24*3600;

app.timeST.T3=app.timeSecs.T3';
app.TempLT.T3=app.TempLogs.T3';

plot(app.RecordT_3,app.timeSecs.T3,app.TempLogs.T3);
grid minor;


app.Tmax_3Field.Value=max(app.TempLogs.T3);
app.Tmin_3Field.Value=min(app.TempLogs.T3);
 end
%h4
 if app.T4CheckBox.Value==1
```

```
 [app.timeLogs.T4,app.TempLogs.T4]=getpoints(app.h4);
app.timeSecs.T4=(app.timeLogs.T4-app.timeLogs.T4(1))*24*3600;

app.timeST.T4=app.timeSecs.T4';
app.TempLT.T4=app.TempLogs.T4';

plot(app.RecordT_4,app.timeSecs.T4,app.TempLogs.T4);
grid minor;


app.Tmax_4Field.Value=max(app.TempLogs.T4);
app.Tmin_4Field.Value=min(app.TempLogs.T4);
 end

%h5
 if app.T5CheckBox.Value==1
[app.timeLogs.T5,app.TempLogs.T5]=getpoints(app.h5);
app.timeSecs.T5=(app.timeLogs.T5-app.timeLogs.T5(1))*24*3600;

 app.timeST.T5=app.timeSecs.T5';
app.TempLT.T5=app.TempLogs.T5';

plot(app.RecordT_5,app.timeSecs.T5,app.TempLogs.T5);
grid minor;


app.Tmax_5Field.Value=max(app.TempLogs.T5);
app.Tmin_5Field.Value=min(app.TempLogs.T5);
 end

 %h6
 if app.T6CheckBox.Value==1
 [app.timeLogs.T6,app.TempLogs.T6]=getpoints(app.h6);
app.timeSecs.T6=(app.timeLogs.T6-app.timeLogs.T6(1))*24*3600;

app.timeST.T6=app.timeSecs.T6';
```

```
app.TempLT.T6=app.TempLogs.T6';


plot(app.RecordT_6,app.timeSecs.T6,app.TempLogs.T6);
grid minor;


app.Tmax_6Field.Value=max(app.TempLogs.T6);
app.Tmin_6Field.Value=min(app.TempLogs.T6);
 end

%h7
 if app.T7CheckBox.Value==1
 [app.timeLogs.T7,app.TempLogs.T7]=getpoints(app.h7);
app.timeSecs.T7=(app.timeLogs.T7-app.timeLogs.T7(1))*24*3600;

 app.timeST.T7=app.timeSecs.T7';
app.TempLT.T7=app.TempLogs.T7';

plot(app.RecordT_7,app.timeSecs.T7,app.TempLogs.T7);
grid minor;


app.Tmax_7Field.Value=max(app.TempLogs.T7);
app.Tmin_7Field.Value=min(app.TempLogs.T7);
 end

%h8
 if app.T8CheckBox.Value==1
 [app.timeLogs.T8,app.TempLogs.T8]=getpoints(app.h8);
app.timeSecs.T8=(app.timeLogs.T8-app.timeLogs.T8(1))*24*3600;

 app.timeST.T8=app.timeSecs.T8';
app.TempLT.T8=app.TempLogs.T8';

plot(app.RecordT_8,app.timeSecs.T8,app.TempLogs.T8);
```

```
grid minor;


app.Tmax_8Field.Value=max(app.TempLogs.T8);
app.Tmin_8Field.Value=min(app.TempLogs.T8);
 end

%h9
 if app.T9CheckBox.Value==1
 [app.timeLogs.T9,app.TempLogs.T9]=getpoints(app.h9);
app.timeSecs.T9=(app.timeLogs.T9-app.timeLogs.T9(1))*24*3600;

 app.timeST.T9=app.timeSecs.T9';
app.TempLT.T9=app.TempLogs.T9';

plot(app.RecordT_9,app.timeSecs.T9,app.TempLogs.T9);
grid minor;


app.Tmax_9Field.Value=max(app.TempLogs.T9);
app.Tmin_9Field.Value=min(app.TempLogs.T9);
 end

%h10
 if app.T10CheckBox.Value==1
 [app.timeLogs.T10,app.TempLogs.T10]=getpoints(app.h10);
app.timeSecs.T10=(app.timeLogs.T10-app.timeLogs.T10(1))*24*3600;

 app.timeST.T10=app.timeSecs.T10';
app.TempLT.T10=app.TempLogs.T10';

plot(app.RecordT_10,app.timeSecs.T10,app.TempLogs.T10);
grid minor;


app.Tmax_10Field.Value=max(app.TempLogs.T10);
```

```
app.Tmin_10Field.Value=min(app.TempLogs.T10);
 end

%h11
 if app.T11CheckBox.Value==1
 [app.timeLogs.T11,app.TempLogs.T11]=getpoints(app.h11);
app.timeSecs.T11=(app.timeLogs.T11-app.timeLogs.T11(1))*24*3600;

 app.timeST.T11=app.timeSecs.T11';
app.TempLT.T11=app.TempLogs.T11';

plot(app.RecordT_11,app.timeSecs.T11,app.TempLogs.T11);
grid minor;


app.Tmax_11Field.Value=max(app.TempLogs.T11);
app.Tmin_11Field.Value=min(app.TempLogs.T11);
 end

 %h12
 if app.T12CheckBox.Value==1
 [app.timeLogs.T12,app.TempLogs.T12]=getpoints(app.h12);
app.timeSecs.T12=(app.timeLogs.T12-app.timeLogs.T12(1))*24*3600;

 app.timeST.T12=app.timeSecs.T12';
app.TempLT.T12=app.TempLogs.T12';

plot(app.RecordT_12,app.timeSecs.T12,app.TempLogs.T12);
grid minor;


app.Tmax_12Field.Value=max(app.TempLogs.T12);
app.Tmin_12Field.Value=min(app.TempLogs.T12);
 end

%h13
```

```
 if app.T13CheckBox.Value==1
 [app.timeLogs.T13,app.TempLogs.T13]=getpoints(app.h13);
app.timeSecs.T13=(app.timeLogs.T13-app.timeLogs.T13(1))*24*3600;

 app.timeST.T13=app.timeSecs.T13';
app.TempLT.T13=app.TempLogs.T13';

plot(app.RecordT_13,app.timeSecs.T13,app.TempLogs.T13);
grid minor;


app.Tmax_13Field.Value=max(app.TempLogs.T13);
app.Tmin_13Field.Value=min(app.TempLogs.T13);
 end

%h14
 if app.T14CheckBox.Value==1
[app.timeLogs.T14,app.TempLogs.T14]=getpoints(app.h14);
app.timeSecs.T14=(app.timeLogs.T14-app.timeLogs.T14(1))*24*3600;

 app.timeST.T14=app.timeSecs.T14';
app.TempLT.T14=app.TempLogs.T14';

plot(app.RecordT_14,app.timeSecs.T14,app.TempLogs.T14);
grid minor;


app.Tmax_14Field.Value=max(app.TempLogs.T14);
app.Tmin_14Field.Value=min(app.TempLogs.T14);

 end

%h15
 if app.T15CheckBox.Value==1
 [app.timeLogs.T15,app.TempLogs.T15]=getpoints(app.h15);
app.timeSecs.T15=(app.timeLogs.T15-app.timeLogs.T15(1))*24*3600;
```

```
 app.timeST.T15=app.timeSecs.T15';
app.TempLT.T15=app.TempLogs.T15';

plot(app.RecordT_15,app.timeSecs.T15,app.TempLogs.T15);
grid minor;


app.Tmax_15Field.Value=max(app.TempLogs.T15);
app.Tmin_15Field.Value=min(app.TempLogs.T15);


 end

%h16
 if app.T16CheckBox.Value==1
 [app.timeLogs.T16,app.TempLogs.T16]=getpoints(app.h16);
app.timeSecs.T16=(app.timeLogs.T16-app.timeLogs.T16(1))*24*3600;

 app.timeST.T16=app.timeSecs.T16';
app.TempLT.T16=app.TempLogs.T16';

plot(app.RecordT_16,app.timeSecs.T16,app.TempLogs.T16);
grid minor;


app.Tmax_16Field.Value=max(app.TempLogs.T16);
app.Tmin_16Field.Value=min(app.TempLogs.T16);


 end

 %----Presion---
  %hp1
 if app.P1CheckBox.Value==1
 [app.timeLogs_p.P1,app.PresLogs.P1]=getpoints(app.hp1);
```

```
app.timeSecs_p.P1=(app.timeLogs_p.P1-app.timeLogs_p.P1(1))*24*3600;

   app.timeSTP.P1=app.timeSecs_p.P1';
   app.PresLT.P1=app.PresLogs.P1';

 end

  %hp2
 if app.P2CheckBox.Value==1
 [app.timeLogs_p.P2,app.PresLogs.P2]=getpoints(app.hp2);
app.timeSecs_p.P2=(app.timeLogs_p.P2-app.timeLogs_p.P2(1))*24*3600;

app.timeSTP.P2=app.timeSecs_p.P2';
   app.PresLT.P2=app.PresLogs.P2';
 end

  %hp3
 if app.P3CheckBox.Value==1
 [app.timeLogs_p.P3,app.PresLogs.P3]=getpoints(app.hp3);
app.timeSecs_p.P3=(app.timeLogs_p.P3-app.timeLogs_p.P3(1))*24*3600;

app.timeSTP.P3=app.timeSecs_p.P3';
   app.PresLT.P3=app.PresLogs.P3';
 end

  %hp4
 if app.P4CheckBox.Value==1
 [app.timeLogs_p.P4,app.PresLogs.P4]=getpoints(app.hp4);
app.timeSecs_p.P4=(app.timeLogs_p.P4-app.timeLogs_p.P4(1))*24*3600;

app.timeSTP.P4=app.timeSecs_p.P4';
   app.PresLT.P4=app.PresLogs.P4';
 end

  %hp5
 if app.P5CheckBox.Value==1
```

```
[app.timeLogs_p.P5,app.PresLogs.P5]=getpoints(app.hp5);
app.timeSecs_p.P5=(app.timeLogs_p.P5-app.timeLogs_p.P5(1))*24*3600;

app.timeSTP.P5=app.timeSecs_p.P5';
   app.PresLT.P5=app.PresLogs.P5';
 end

  %hp6
 if app.P6CheckBox.Value==1
 [app.timeLogs_p.P6,app.PresLogs.P6]=getpoints(app.hp6);
app.timeSecs_p.P6=(app.timeLogs_p.P6-app.timeLogs_p.P6(1))*24*3600;

app.timeSTP.P6=app.timeSecs_p.P6';
   app.PresLT.P6=app.PresLogs.P6';
 end

  %hp7
 if app.P7CheckBox.Value==1
 [app.timeLogs_p.P7,app.PresLogs.P7]=getpoints(app.hp7);
app.timeSecs_p.P7=(app.timeLogs_p.P7-app.timeLogs_p.P7(1))*24*3600;

app.timeSTP.P7=app.timeSecs_p.P7';
   app.PresLT.P7=app.PresLogs.P7';
 end

  %hp8
 if app.CaudalCheckBox.Value==1
 [app.timeLogs_p.P8,app.PresLogs.P8]=getpoints(app.hp8);
app.timeSecs_p.P8=(app.timeLogs_p.P8-app.timeLogs_p.P8(1))*24*3600;

   app.timeSTP.P8=app.timeSecs_p.P8';
   app.PresLT.P8=app.PresLogs.P8';
 end
 message = {'El sistema de captura de datos ha parado','Recuerda guardar los
 archivos'};
uialert(app.UIFigure,message,'Recordatorio','Icon','info');
```

GuardarMenuSelected

```
writeDigitalPin(app.a,'D4',0);
            writeDigitalPin(app.a,'D5',0);
            writeDigitalPin(app.a,'D6',1);


            app.fecha=datetime('now','Format','yyyy-MM-dd''T''HHmmss');
            app.cfecha=char(app.fecha);

            TempLogsSave=app.TempLT;
            PresLogsSave=app.PresLT;
            TimeLogsSave=app.timeST;


            %Guardar en .mat
            save("Temperatura"+app.cfecha+".mat",'-struct','TempLogsSave')
            save("Presion"+app.cfecha+".mat",'-struct','PresLogsSave')


           %Guardar Tabla
           app.tablaTiempo=splitvars(table(struct2table(TimeLogsSave)));
           writetable(app.tablaTiempo,"TiempoEnsayo"+app.cfecha+".xls");


           app.tablaTemp=splitvars(table(struct2table(TempLogsSave)));


           writetable(app.tablaTemp,"Temperatura"+app.cfecha+".xls");


           app.tablaPres=splitvars(table(struct2table(PresLogsSave)));
           writetable(app.tablaPres,"Presion"+app.cfecha+".xls");

            writeDigitalPin(app.a,'D4',1);
            writeDigitalPin(app.a,'D5',0);
            writeDigitalPin(app.a,'D6',0);

            message = {'Guardado completado','Archivos disponibles en carpeta'};
            uialert(app.UIFigure,message,'Guardado','Icon','success');
```

ConectarMenuSelected

```
app.a=arduino('COM3','Mega2560','BaudRate',115200); %Conexión Arduino 1
             app.a1=arduino('COM4','Mega2560','BaudRate',115200); %Conexión Arduino 2
             disp('Conectado');
             app.ArduinoLamp.Color='g';
             writeDigitalPin(app.a,'D4',1);
             writeDigitalPin(app.a,'D5',1);
             writeDigitalPin(app.a,'D6',1);
```

```
message = {'Arduino se ha conectado con éxito'};
uialert(app.UIFigure,message,'Conexión con Arduino','Icon','success');

writeDigitalPin(app.a,'D4',1);
writeDigitalPin(app.a,'D5',0);
writeDigitalPin(app.a,'D6',0);
```

DesconectarMenuSelected

```
writeDigitalPin(app.a,'D4',0);
writeDigitalPin(app.a,'D5',0);
writeDigitalPin(app.a,'D6',0);
clear app.a;
app.ArduinoLamp.Color='r';
```

T1CheckBoxValueChanged

Válido para todos los Check Box desde T1-T16, P1-P7 y Caudal.

```
value = app.T1CheckBox.Value;

if value == 1
    app.LiveT.Visible='on';
    app.TempField.Visible='on';
    app.TempGauge.Visible='on';
    app.S1F.Visible='on';
    app.L1.Visible='off';

else
    app.LiveT.Visible='off';
    app.TempField.Visible='off';
    app.TempGauge.Visible='off';
    app.S1F.Visible='off';
    app.L1.Visible='on';
end
```

UIFiguraCloseRequest

```
% Display confirmation dialog box
msg = '¿Seguro que quieres salir?';
uiconfirm(app.UIFigure,msg,'Confirm Close','CloseFcn',@app.confirmClose);
```

ReiniciarMenuSelected

```
APLICACION_DAQ_FINAL();
close(app.UIFigure)
```

ParametrosdepresionMenuSelected

```
ParamPresion();
```

Anexo B: Coste del material

- Coste del material utilizado
- Coste de un DAQ de NI

Coste de material utilizado			
ELEMENTO	**UNIDADES**	**PRECIO UNIDAD (€)**	**PRECIO TOTAL (€)**
Placa bungard CPC0001 100x160 cuadros	3	5,9857	17,96
Fuente TRACO TML-20205C	2	58,32	116,64
Caja de registro estanca 362x500x140	1	71,48	71,48
HUB USB 4 puertos	1	7,52	7,52
Fuente laboratorio 30V 3A Digital	1	76,9	76,9
Termo 1,2 MM 3/63 Pulg rojo	4	0,68	2,74
Termo 1,2 MM 4/64 Pulg negro	4	0,68	2,74
Termo 3,2 MM 1/8 Pulg negro	4	0,7575	3,04
Ficha empalme 4 MM	10	0,605	6,05
Adafruit AD8495	25	12,5	312,5
Arduino MEGA 2560 Rev 3	4	34,5	138,00
Terminal block shield board Arduino MEGA 2560	4	15,9	63,6
Conector GX-12	32	0,9	28,8
Tornillo M2,5x10 DIN 84	100	0,014	1,4
Tuerca M2,5 DIN 934	100	0,0124	1,24
Tornillo M3x16 DIN84	100	0,0129	1,29
Tornillo M3x25 DIN82	100	0,02020	2,02
Separador HEXAG 25MM H/H M3	30	0,1391	4,17
Separador HEXAG 20MM H/H M3	30	0,1010	3,03
Separador HEXAG 15MM H/H M3	30	0,083	2,49
Separador HEXAG 10MM H/H M3	30	0,083	2,49
Ventilador 40x40x10 5VDC	2	6,54	13,08
Ventilador 60x60x10 12 VDC	2	5,64	11,28
Base red+fusible+inter lum doble	1	11,80	11,80
LED rojo 5MM	3	0,1038	0,31
LED verde 5MM	3	0,1038	0,31
LED naranja 5MM	3	0,1038	0,31

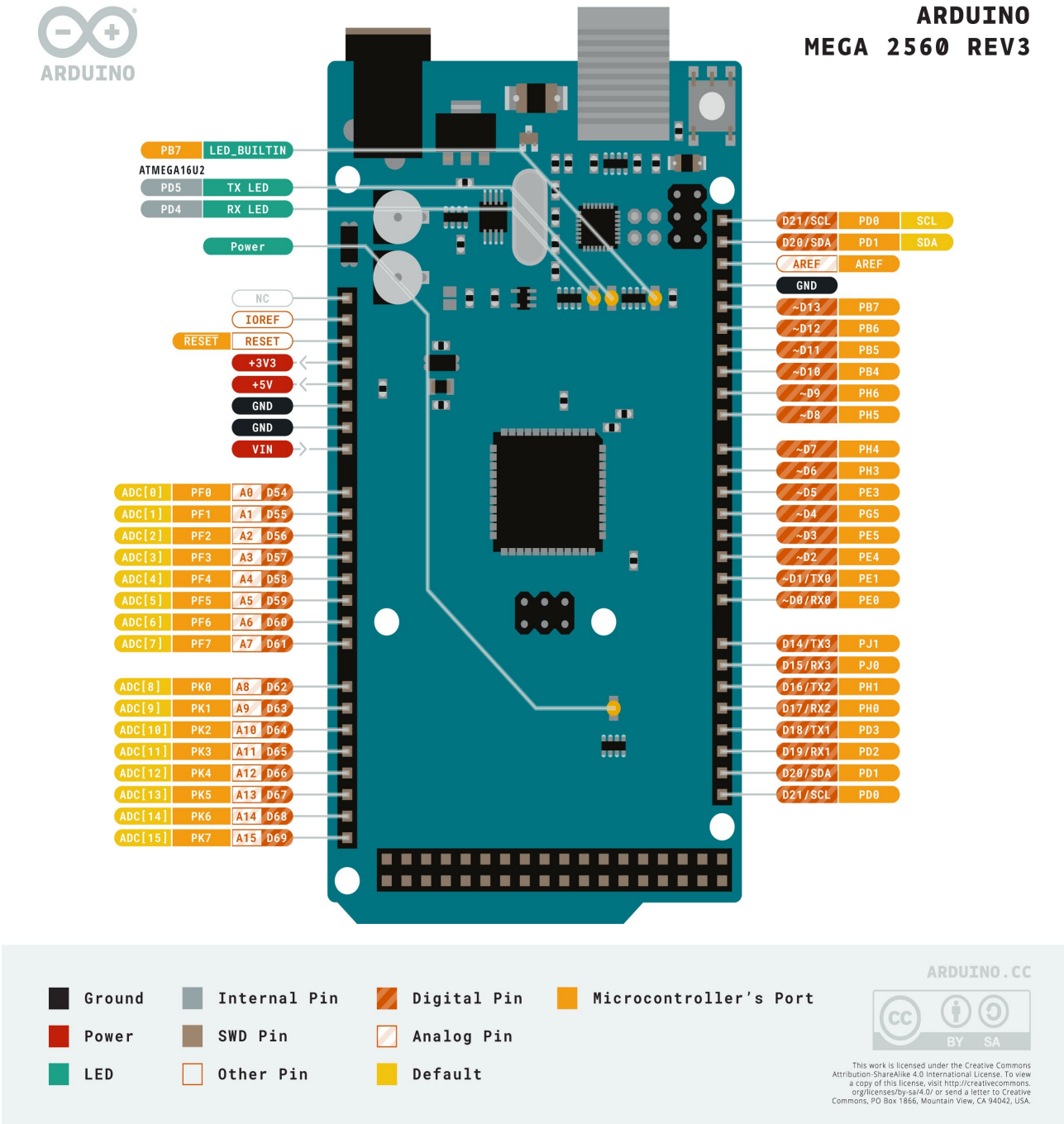
330R Resistencia 1/4W 5%	5	0,018	0,09
680R Resistencia 1/4W 5%	5	0,018	0,09
1K Resistencia 1/4W 5%	5	0,018	0,09
TOTAL IMP	**% IVA**	**IVA**	**TOTAL**
903,46 €	21 %	189,73 €	**1093,19 €**

Coste de un DAQ de NI			
ELEMENTO	**UNIDADES**	**PRECIO UNIDAD (€)**	**PRECIO TOTAL (€)**
Chasis cDAQ-9174	1	1.222,00	1.222,00
NI-9213 16 entradas de temperatura	1	1.778,00	1.778,00
NI-9201 8 entradas de voltaje	1	638,00	638,00
TOTAL IMP	**% IVA**	**IVA**	**TOTAL**
3.638,00 €	21 %	763,98 €	**4.401,98 €**

Anexo C: Ficha técnica de los elementos utilizados en el montaje

- Esquema de conexión Arduino MEGA 2560
- Ficha técnica Adafruit AD8495

Esquema de conexión Arduino MEGA 2560



Precision Thermocouple Amplifiers with Cold Junction Compensation

AD8494/AD8495/AD8496/AD8497

FEATURES

- Low cost and easy to use
- Pretrimmed for J or K type thermocouples
- Internal cold junction compensation
- High impedance differential input
- Standalone 5 mV/°C thermometer
- Reference pin allows offset adjustment
- Thermocouple break detection
- Laser wafer trimmed to 1°C initial accuracy and
  - 0.025°C/°C ambient temperature rejection
- Low power: <1 mW at V_S = 5 V
- Wide power supply range
  - Single supply: 2.7 V to 36 V
  - Dual supply: ±2.7 V to ±18 V
- Small, 8-lead MSOP

APPLICATIONS

- J or K type thermocouple temperature measurement
- Setpoint controller
- Celsius thermometer
- Universal cold junction compensator
- White goods (oven, stove top) temperature measurements
- Exhaust gas temperature sensing
- Catalytic converter temperature sensing

GENERAL DESCRIPTION

The AD8494/AD8495/AD8496/AD8497 are precision instrumentation amplifiers with thermocouple cold junction compensators on an integrated circuit. They produce a high level (5 mV/°C) output directly from a thermocouple signal by combining an ice point reference with a precalibrated amplifier. They can be used as standalone thermometers or as switched output setpoint controllers using either a fixed or remote setpoint control.

The AD8494/AD8495/AD8496/AD8497 can be powered from a single-ended supply (less than 3 V) and can measure temperatures below 0°C by offsetting the reference input. To minimize self-heating, an unloaded AD849x typically operates with a total supply current of 180 µA, but it is also capable of delivering in excess of ±5 mA to a load.

The AD8494 and AD8496 are precalibrated by laser wafer trimming to match the characteristics of J type (iron-constantan) thermocouples; the AD8495 and AD8497 are laser trimmed to match the characteristics of K type (chromel-alumel) thermocouples. See Table 1 for the optimized ambient temperature range of each part.

FUNCTIONAL BLOCK DIAGRAM

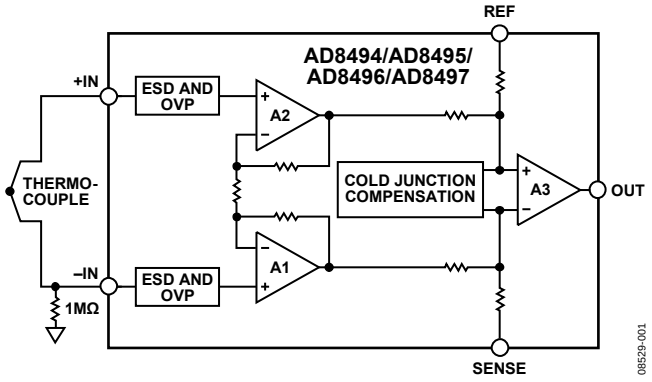


Figure 1.

Table 1. Device Temperature Ranges

Part No.	Thermo-Couple Type	Optimized Temperature Range	
		Ambient Temperature (Reference Junction)	Measurement Junction
AD8494	J	0°C to 50°C	Full J type range
AD8495	K	0°C to 50°C	Full K type range
AD8496	J	25°C to 100°C	Full J type range
AD8497	K	25°C to 100°C	Full K type range

The AD8494/AD8495/AD8496/AD8497 allow a wide variety of supply voltages. With a 5 V single supply, the 5 mV/°C output allows the devices to cover nearly 1000 degrees of a thermocouple's temperature range.

The AD8494/AD8495/AD8496/AD8497 work with 3 V supplies, allowing them to interface directly to lower supply ADCs. They can also work with supplies as large as 36 V in industrial systems that require a wide common-mode input range.

PRODUCT HIGHLIGHTS

1. Complete, precision laser wafer trimmed thermocouple signal conditioning system in a single IC package.
2. Flexible pinout provides for operation as a setpoint controller or as a standalone Celsius thermometer.
3. Rugged inputs withstand 4 kV ESD and provide overvoltage protection (OVP) up to V_S ± 25 V.
4. Differential inputs reject common-mode noise on the thermocouple leads.
5. Reference pin voltage can be offset to measure 0°C on single supplies.
6. Available in a small, 8-lead MSOP that is fully RoHS compliant.

Rev. C

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 **www.analog.com**
Fax: 781.461.3113

TABLE OF CONTENTS

REVISION HISTORY

6/11—Rev. B to Rev. C

Changes to Figure 35 and Figure 36 ... 15

4/11—Rev. A to Rev. B

Changes to Figure 1 ... 1
Changes to Figure 33 and Figure 34 ... 14
Changes to Figure 35 and Figure 36 ... 15
Changes to Ordering Guide ... 16

10/10—Rev. 0 to Rev. A

Changes to Linearity Error of the Thermocouple Section ... 12
Changes to Ambient Temperature Sensor Section ... 14
Changes to Ordering Guide ... 16

7/10—Revision 0: Initial Version

SPECIFICATIONS

$+V_S$ = 5 V, $-V_S$ = 0 V, $V_{+IN} = V_{-IN}$ = 0 V, V_{REF} = 0 V, $T_A = T_{RJ}$ = 25°C, R_L = 100 kΩ, unless otherwise noted. Specifications do not include gain and offset errors of the thermocouple itself. T_A is the ambient temperature at the AD849x; T_{RJ} is the thermocouple reference junction temperature; T_{MJ} is the thermocouple measurement junction temperature.

Table 2.

		A Grade			C Grade			
Parameter	**Test Conditions/Comments**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Unit**
TEMPERATURE ACCURACY								
Initial Accuracy								
AD8494/AD8495	$T_A = T_{RJ} = T_{MJ}$ = 25°C			3			1	°C
AD8496/AD8497	$T_A = T_{RJ}$ = 60°C, T_{MJ} = 175°C			3			1.5	°C
Ambient Temperature Rejection[1]								
AD8494/AD8495	$T_A = T_{RJ}$ = 0°C to 50°C			0.05			0.025	°C/°C
AD8496/AD8497	$T_A = T_{RJ}$ = 25°C to 100°C			0.05			0.025	°C/°C
Gain Error[2, 3]	V_{OUT} = 0.125 V to 4.125 V							
AD8494/AD8495				0.3			0.1	%
AD8496/AD8497				0.3			0.1	%
Transfer Function			5			5		mV/°C
INPUTS								
Input Voltage Range		$-V_S$ – 0.2		$+V_S$ – 1.6	$-V_S$ – 0.2		$+V_S$ – 1.6	V
Overvoltage Range		$+V_S$ – 25		$-V_S$ + 25	$+V_S$ – 25		$-V_S$ + 25	V
Input Bias Current[4]			25	50		25	50	nA
Input Offset Current				1.5			0.5	nA
Common-Mode Rejection	V_{CM} = 0 V to 3 V			1			0.3	°C/V
Power Supply Rejection	$+V_S$ = 2.7 V to 5 V			0.5			0.5	°C/V
NOISE								
Voltage Noise	f = 0.1 Hz to 10 Hz, T_A = 25°C		0.8			0.8		µV p-p
Voltage Noise Density	f = 1 kHz, T_A = 25°C		32			32		nV/√Hz
Current Noise Density	f = 1 kHz, T_A = 25°C		100			100		fA/√Hz
REFERENCE INPUT								
Input Resistance			60			60		kΩ
Input Current			25			25		µA
Voltage Range		$-V_S$		$+V_S$	$-V_S$		$+V_S$	V
Gain to Output			1			1		V/V
OUTPUT								
Output Voltage Range		$-V_S$ + 0.025		$+V_S$ – 0.1	$-V_S$ + 0.025		$+V_S$ – 0.1	V
Short-Circuit Current[5]			7			7		mA
DYNAMIC RESPONSE								
–3 dB Bandwidth								
AD8494			30			30		kHz
AD8495/AD8497			25			25		kHz
AD8496			31			31		kHz
Settling Time to 0.1%	4 V output step							
AD8494			36			36		µs
AD8495/AD8497			40			40		µs
AD8496			32			32		µs
POWER SUPPLY								
Operating Voltage Range[6]								
Single Supply		2.7		36	2.7		36	V
Dual Supply		±2.7		±18	±2.7		±18	V
Quiescent Current			180	250		180	250	µA

Parameter	Test Conditions/Comments	A Grade			C Grade			Unit
		Min	Typ	Max	Min	Typ	Max	
TEMPERATURE RANGE (T_A)								
Specified Performance								
AD8494/AD8495		0		50	0		50	°C
AD8496/AD8497		25		100	25		100	°C
Operational		−40		+125	−40		+125	°C

[1] Ambient temperature rejection specifies the change in the output measurement (in °C) for a given change in temperature of the cold junction. For the AD8494 and AD8495, ambient temperature rejection is defined as the slope of the line connecting errors calculated at 0°C and 50°C ambient temperature. For the AD8496 and AD8497, ambient temperature rejection is defined as the slope of the line connecting errors calculated at 25°C and 100°C ambient temperature.
[2] Error does not include thermocouple gain error or thermocouple nonlinearity.
[3] With a 100 kΩ load, measurement junction temperatures beyond approximately 880°C for the AD8494 and AD8496 and beyond approximately 960°C for the AD8495 and AD8497 require supply voltages larger than 5 V or a negative voltage applied to the reference pin. Measurement junction temperatures below 5°C require either a positive offset voltage applied to the reference pin or a negative supply.
[4] Input stage uses PNP transistors, so bias current always flows out of the part.
[5] Large output currents can increase the internal temperature rise of the part and contribute to cold junction compensation (CJC) error.
[6] Unbalanced supplies can also be used. Care should be taken that the common-mode voltage of the thermocouple stays within the input voltage range of the part.

ABSOLUTE MAXIMUM RATINGS

Table 3.

Parameter	Rating
Supply Voltage	±18 V
Maximum Voltage at −IN or +IN	$+V_S$ − 25 V
Minimum Voltage at −IN or +IN	$-V_S$ + 25 V
REF Voltage	$\pm V_S$
Output Short-Circuit Current Duration	Indefinite
Storage Temperature Range	−65°C to +150°C
Operating Temperature Range	−40°C to +125°C
Maximum IC Junction Temperature	140°C
ESD	
Human Body Model	4.5 kV
Field-Induced Charged Device Model	1.5 kV

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

THERMAL RESISTANCE

θ_{JA} is specified for a device on a 4-layer JEDEC PCB in free air.

Table 4.

Package	θ_{JA}	Unit
8-Lead MSOP (RM-8)	135	°C/W

ESD CAUTION

ESD (electrostatic discharge) sensitive device. Charged devices and circuit boards can discharge without detection. Although this product features patented or proprietary protection circuitry, damage may occur on devices subjected to high energy ESD. Therefore, proper ESD precautions should be taken to avoid performance degradation or loss of functionality.

PIN CONFIGURATION AND FUNCTION DESCRIPTIONS

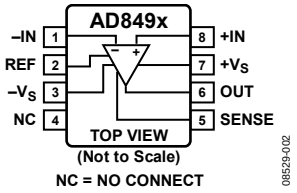


Figure 2. Pin Configuration

Table 5. Pin Function Descriptions

Pin No.	Mnemonic	Description
1	−IN	Negative Input.
2	REF	Reference. This pin must be driven by low impedance.
3	$-V_S$	Negative Supply.
4	NC	No Connect.
5	SENSE	Sense Pin. In measurement mode, connect to output; in setpoint mode, connect to setpoint voltage.
6	OUT	Output.
7	$+V_S$	Positive Supply.
8	+IN	Positive Input.

TYPICAL PERFORMANCE CHARACTERISTICS

T_A = 25°C, $+V_S$ = 5 V, R_L = ∞, unless otherwise noted.

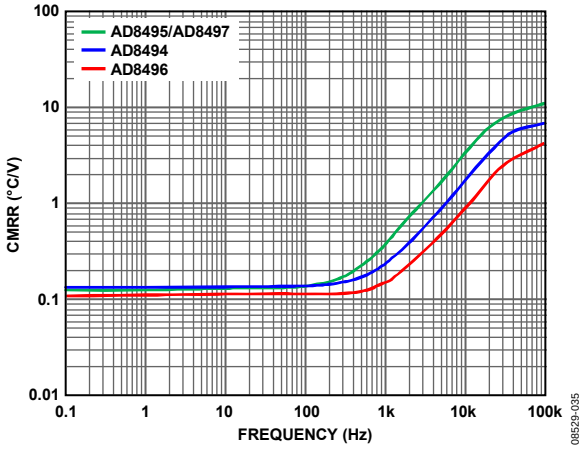


Figure 3. CMRR vs. Frequency

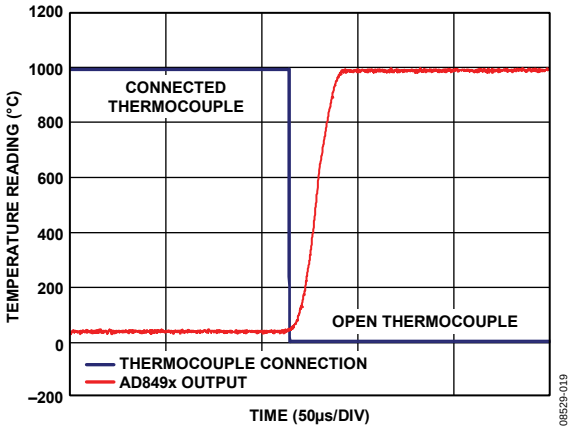


Figure 6. Output Response to Open Thermocouple, –IN Connected to Ground Through a 1 MΩ Resistor

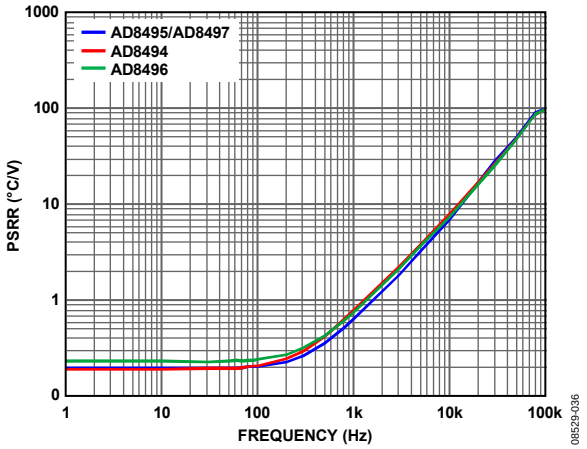


Figure 4. PSRR vs. Frequency

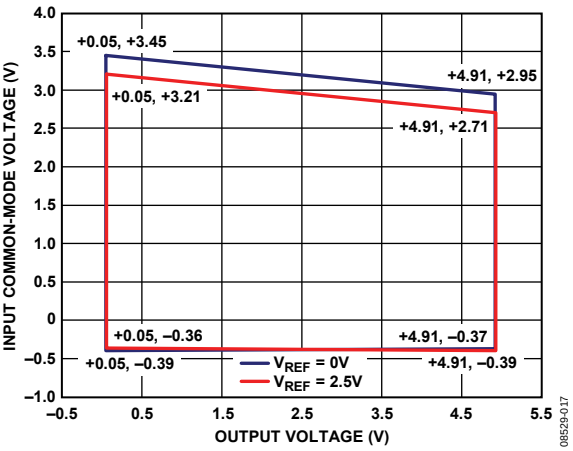


Figure 7. Input Common-Mode Voltage Range vs. Output Voltage, $+V_S$ = 5 V, V_{REF} = 0 V, and V_{REF} = 2.5 V

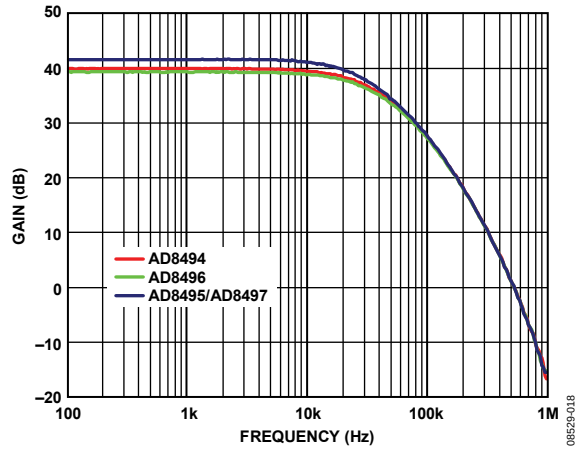


Figure 5. Frequency Response

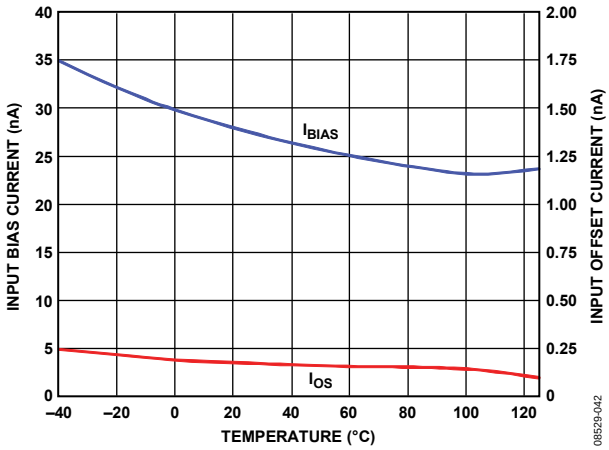


Figure 8. Input Bias Current and Input Offset Current vs. Temperature

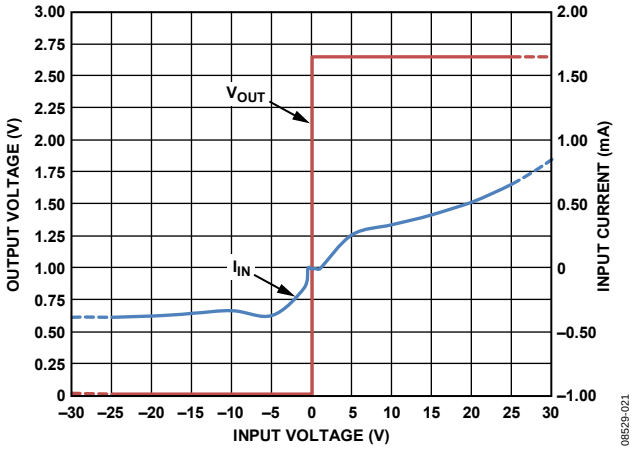


Figure 9. AD8494 Input Overvoltage Performance, $+V_S$ = 2.7 V (Gain = 96.7)

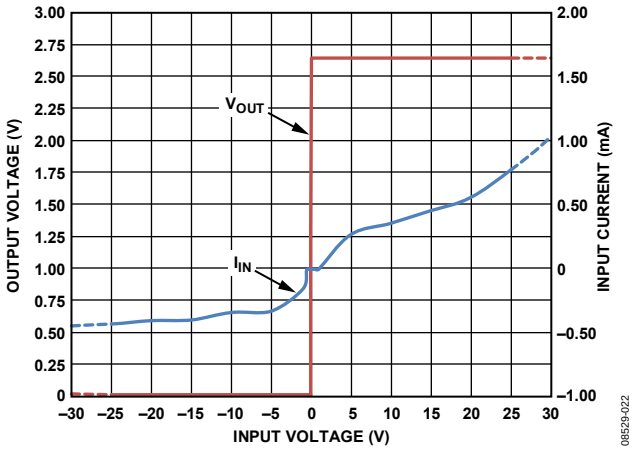


Figure 10. AD8495/AD8497 Input Overvoltage Performance, $+V_S$ = 2.7 V (Gain = 122.4)

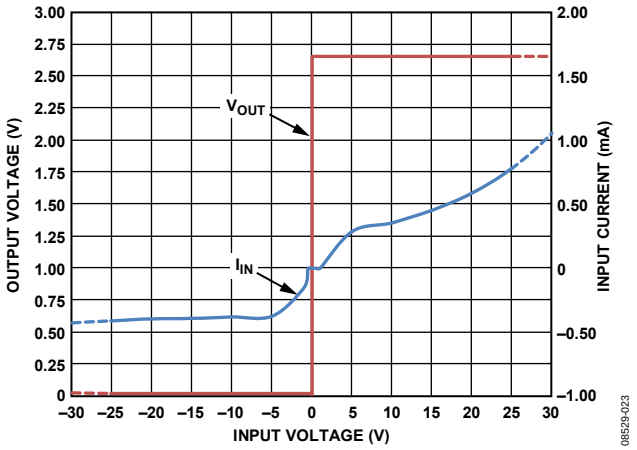


Figure 11. AD8496 Input Overvoltage Performance, $+V_S$ = 2.7 V Gain = 90.35)

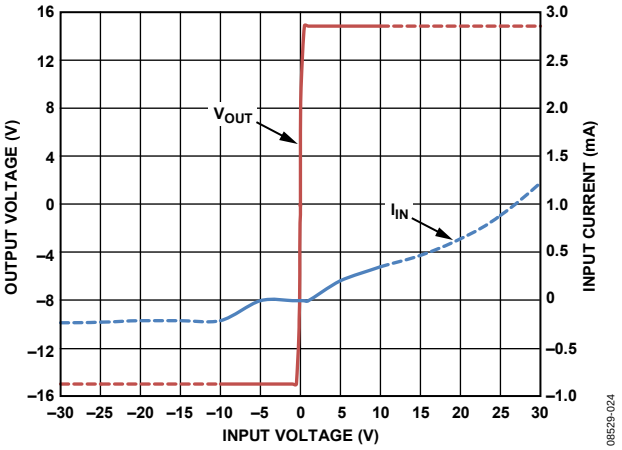


Figure 12. AD8494 Input Overvoltage Performance, V_S = ±15 V (Gain = 96.7)

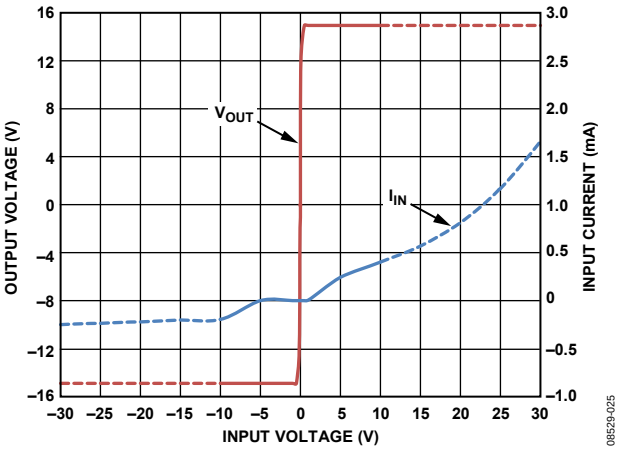


Figure 13. AD8495/AD8497 Input Overvoltage Performance, V_S = ±15 V (Gain = 122.4)

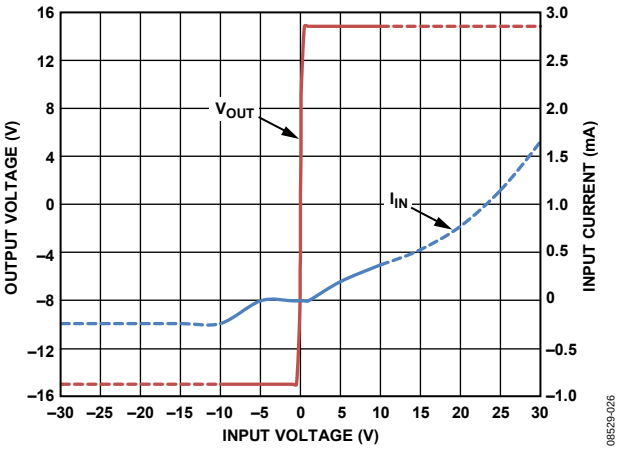


Figure 14. AD8496 Input Overvoltage Performance, V_S = ±15 V (Gain = 90.35)

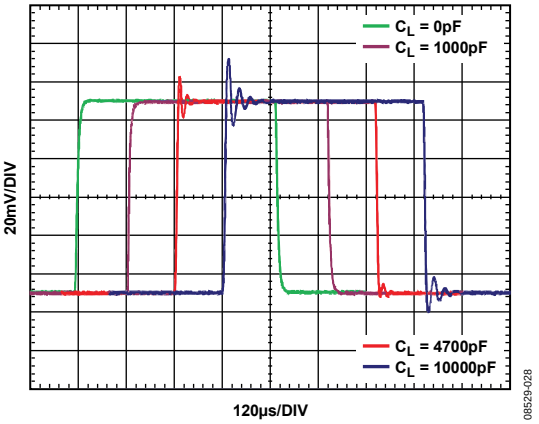


Figure 15. AD8494/AD8496 Small-Signal Response with Various Capacitive Loads

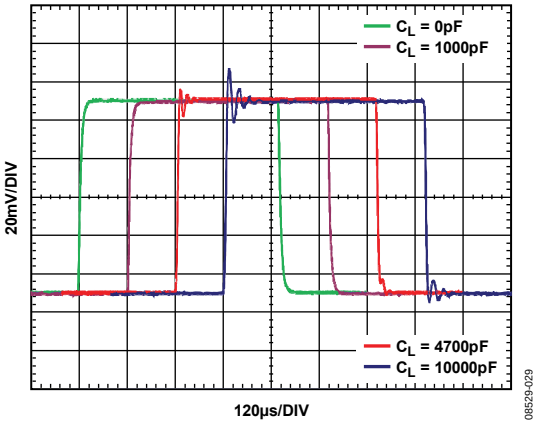


Figure 18. AD8495/AD8497 Small-Signal Response with Various Capacitive Loads

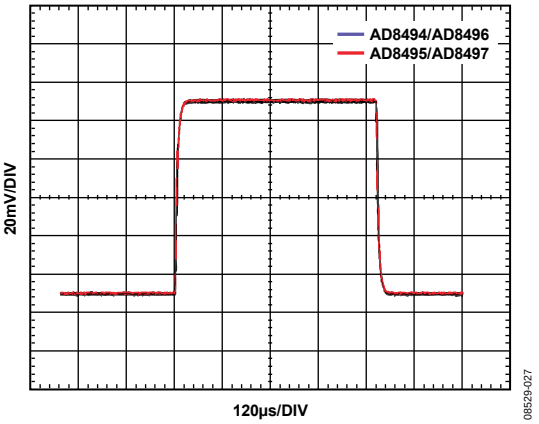


Figure 16. Small-Signal Response, R_L = 100 kΩ, C_L = 1 nF

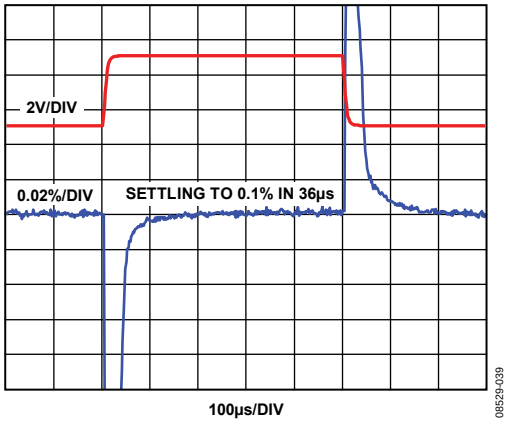


Figure 19. AD8494 Large-Signal Step Response and Settling Time

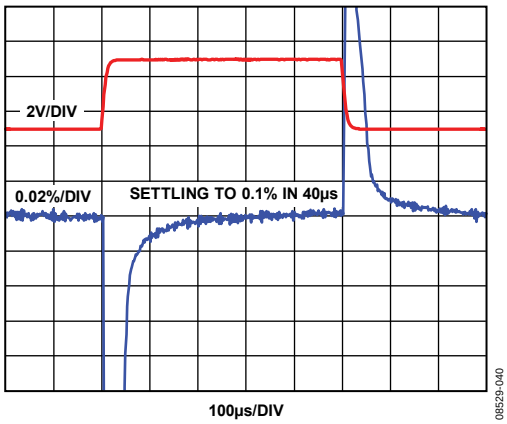


Figure 17. AD8495/AD8497 Large-Signal Step Response and Settling Time

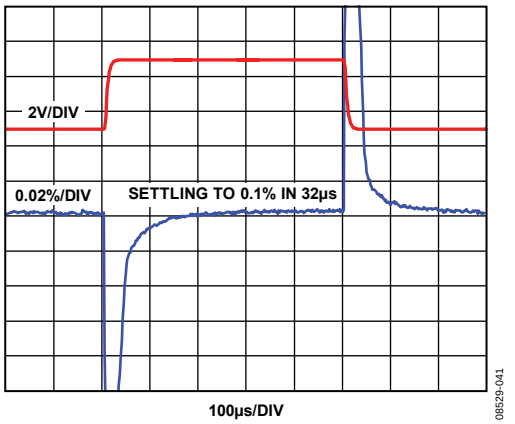


Figure 20. AD8496 Large-Signal Step Response and Settling Time

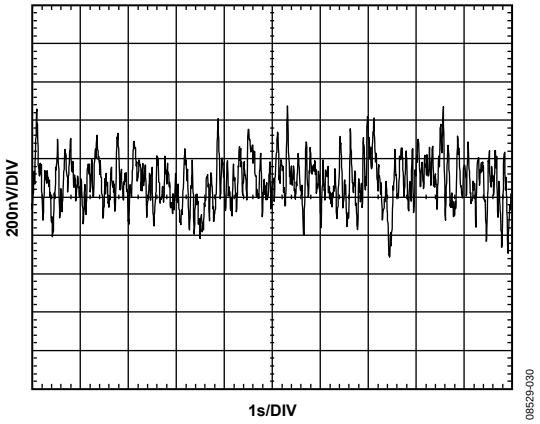


Figure 21. 0.1 Hz to 10 Hz RTI Voltage Noise

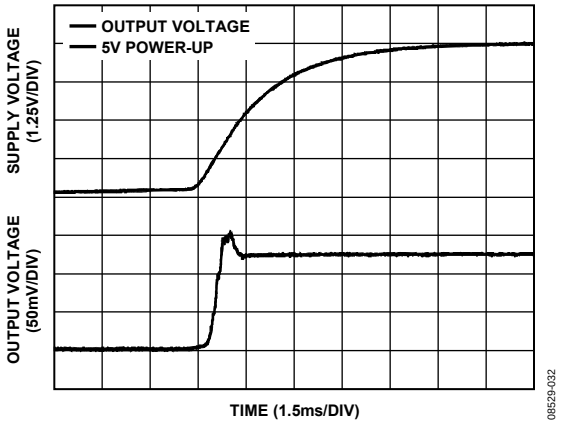


Figure 24. Output Voltage Start-Up

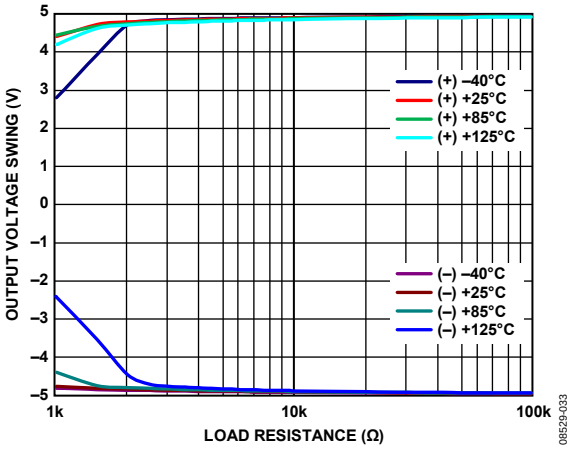


Figure 22. Output Voltage Swing vs. Load Resistance, V_S = ±5 V

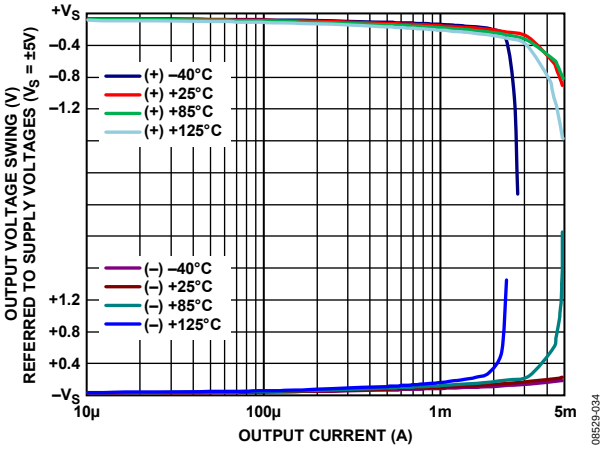


Figure 25. Output Voltage Swing vs. Output Current, V_S = ±5 V

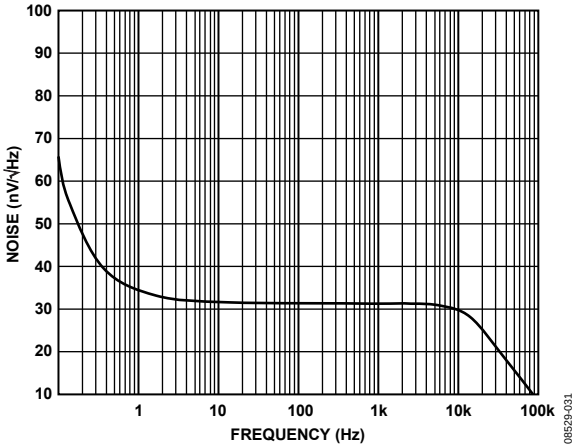


Figure 23. Voltage Noise Spectral Density vs. Frequency

THEORY OF OPERATION

THERMOCOUPLES

A thermocouple is a rugged, low cost temperature transducer whose output is proportional to the temperature difference between a measurement junction and a reference junction. It has a very wide temperature range. Its low level output (typically tens of microvolts per °C) requires amplification. Variation in the reference junction temperature results in measurement error unless the thermocouple signal is properly compensated.

A thermocouple consists of two dissimilar metals. These metals are connected at one end to form the measurement junction, also called the hot junction. The other end of the thermocouple is connected to the metal lines that lead to the measurement electronics. This connection forms a second junction: the reference junction, also called the cold junction.

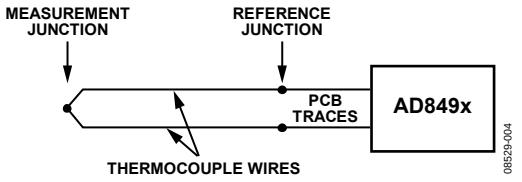


Figure 26. Thermocouple Junctions

To derive the temperature at the measurement junction (T_{MJ}), the user must know the differential voltage created by the thermocouple. The user must also know the error voltage generated by the temperature at the reference junction (T_{RJ}). Compensating for the reference junction error voltage is typically called cold junction compensation. The electronics must compensate for any changes in temperature at the reference (cold) junction so that the output voltage is an accurate representation of the hot junction measurement.

THERMOCOUPLE SIGNAL CONDITIONER

The AD8494/AD8495/AD8496/AD8497 thermocouple amplifiers provide a simple, low cost solution for measuring thermocouple temperatures. These amplifiers simplify many of the difficulties of measuring thermocouples. An integrated temperature sensor performs cold junction compensation. A fixed-gain instrumentation amplifier amplifies the small thermocouple voltage to provide a 5 mV/°C output. The high common-mode rejection of the amplifier blocks common-mode noise that the long thermocouple leads can pick up. For additional protection, the high impedance inputs of the amplifier make it easy to add extra filtering.

Table 6 shows an example of a J type thermocouple voltage for various combinations of 0°C and 50°C on the reference and measurement junctions. Table 6 also shows the performance of the AD8494 amplifying the thermocouple voltage and compensating for the reference junction temperature changes, thus eliminating the error.

Table 6. J Type Thermocouple Voltages and AD8494 Readings

Measurement Junction Temperature (T_{MJ})	Reference Junction Temperature (T_{RJ})	Thermocouple Voltage	AD8494 Reading
50°C	0°C	+2.585 mV	250 mV
50°C	50°C	0 mV	250 mV
0°C	0°C	0 mV	0 mV
0°C	50°C	−2.585 mV	0 mV

AD8494/AD8495/AD8496/AD8497 ARCHITECTURE

Figure 27 shows a block diagram of the AD849x circuitry. The AD849x consists of a low offset, fixed-gain instrumentation amplifier and a temperature sensor.

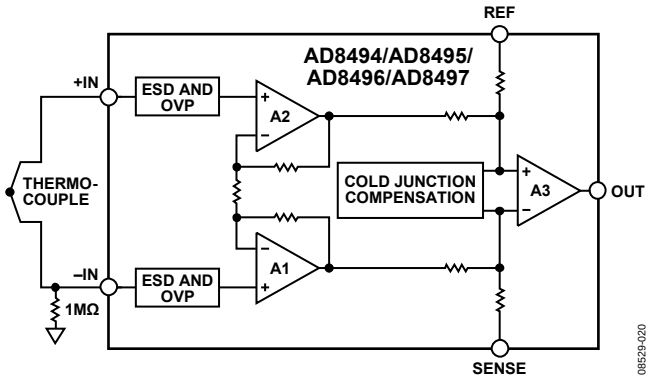


Figure 27. Block Diagram

The AD849x output is a voltage that is proportional to the temperature at the measurement junction of the thermocouple (T_{MJ}). To derive the measured temperature from the AD849x output voltage, use the following transfer function:

$$T_{MJ} = (V_{OUT} - V_{REF})/(5\ \text{mV/°C})$$

An ideal AD849x achieves this output with an error of less than ±2°C, within the specified operating ranges listed in Table 7.

Instrumentation Amplifier

A thermocouple signal is so small that considerable gain is required before it can be sampled properly by most ADCs. The AD849x has an instrumentation amplifier with a fixed gain that generates an output voltage of 5 mV/°C for J type and K type thermocouples.

$$V_{OUT} = (T_{MJ} \times 5\ \text{mV/°C}) + V_{REF}$$

To accommodate the nonlinear behavior of the thermocouple, each amplifier has a different gain so that the 5 mV/°C is accurately maintained for a given temperature measurement range.

- The AD8494 and AD8496 (J type) have an instrumentation amplifier with a gain of 96.7 and 90.35, respectively.
- The AD8495 and AD8497 (K type) have an instrumentation amplifier with a gain of 122.4.

The small thermocouple voltages mean that signals are quite vulnerable to interference, especially when measured with single-ended amplifiers. The AD849x addresses this issue in several ways. Low input bias currents and high input impedance allow for easy filtering at the inputs. The excellent common-mode rejection of the AD849x prevents variations in ground potential and other common-mode noise from affecting the measurement.

Temperature Sensor (Cold Junction Compensation)

The AD849x also includes a temperature sensor for cold junction compensation. This temperature sensor is used to measure the reference junction temperature of the thermocouple and to cancel its effect.

- The AD8494/AD8495 cold junction compensation is optimized for operation in a lab environment, where the ambient temperature is around 25°C. The AD8494/AD8495 are specified for an ambient range of 0°C to 50°C.
- The AD8496/AD8497 cold junction compensation is optimized for operation in a less controlled environment, where the temperature is around 60°C. The AD8496/AD8497 are specified for an ambient range of 25°C to 100°C. Application examples for the AD8496/AD8497 include automotive applications, autoclave, and ovens.

Thermocouple Break Detection

The AD849x offers open thermocouple detection. The inputs of the AD849x are PNP type transistors, which means that the bias current always flows out of the inputs. Therefore, the input bias current drives any unconnected input high, which rails the output. Connecting the negative input to ground through a 1 MΩ resistor causes the AD849x output to rail high in an open thermocouple condition (see Figure 6, Figure 28, and the Ground Connection section).

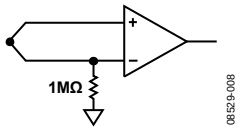


Figure 28. Ground the Negative Input Through a 1 MΩ Resistor for Open Thermocouple Detection

Input Voltage Protection

The AD849x has very robust inputs. Input voltages can be up to 25 V from the opposite supply rail. For example, with a +5 V positive supply and a −3 V negative supply, the part can safely withstand voltages at the inputs from −20 V to +22 V. Voltages at the reference and sense pins should not go beyond 0.3 V of the supply rails.

MAXIMUM ERROR CALCULATION

As is normally the case, the AD849x outputs are subject to calibration, gain, and temperature sensitivity errors. The user can calculate the maximum error from the AD849x using the following information.

The five primary sources of AD849x error are described in this section.

AD849x Initial Calibration Accuracy

Error at the initial calibration point can be easily calibrated out with a one-point temperature calibration. See Table 2 for the specifications.

AD849x Ambient Temperature Rejection

The specified ambient temperature rejection represents the ability of the AD849x to reject errors caused by changes in the ambient temperature/reference junction. For example, with 0.025°C/°C ambient temperature rejection, a 20°C change in the reference junction temperature adds less than 0.5°C error to the measurement. See Table 2 for the specifications.

AD849x Gain Error

Gain error is the amount of additional error when measuring away from the measurement junction calibration point. For example, if the part is calibrated at 25°C and the measurement junction is 100°C with a gain error of 0.1%, the gain error contribution is (100°C − 25°C) × (0.1%) = 0.075°C. This error can be calibrated out with a two-point calibration if needed, but it is usually small enough to ignore. See Table 2 for the specifications.

Manufacturing Tolerances of the Thermocouple

Consult the data sheet for your thermocouple to find the specified tolerance of the thermocouple.

Linearity Error of the Thermocouple

Each part in the AD849x family is precision trimmed to optimize a linear operating range for a specific thermocouple type and for the widest possible measurement and ambient temperature ranges. The AD849x achieves a linearity error of less than ±2°C, within the specified operating ranges listed in Table 7. This error is due only to the nonlinearity of the thermocouple.

Table 7. AD849x ±2°C Accuracy Temperature Ranges

Part	Thermocouple Type	Max Error	Ambient Temperature Range	Measurement Temperature Range
AD8494	J	±2°C	0°C to 50°C	−35°C to +95°C
AD8495	K	±2°C	0°C to 50°C	−25°C to +400°C
AD8496	J	±2°C	25°C to 100°C	+55°C to +565°C
AD8497	K	±2°C	25°C to 100°C	−25°C to +295°C

For temperature ranges outside those listed in Table 7 or for instructions on how to correct for thermocouple nonlinearity error with software, see the AN-1087 Application Note for additional details.

RECOMMENDATIONS FOR BEST CIRCUIT PERFORMANCE

Input Filter

A low-pass filter before the input of the AD849x is strongly recommended (see Figure 29), especially when operating in an electrically noisy environment. Long thermocouple leads can function as an excellent antenna and pick up many unwanted signals.

The filter should be set to a low corner frequency that still allows the input signal to pass through undiminished. The primary purpose of the filter is to remove RF signals, which, if allowed to reach the AD849x, can be rectified and appear as temperature fluctuations.

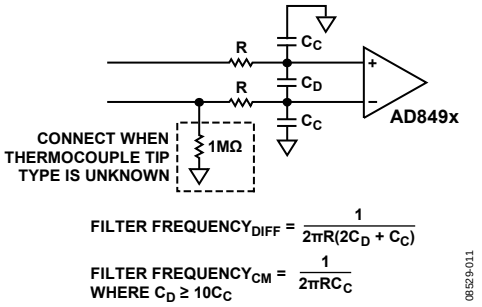


Figure 29. Filter for Any Thermocouple Style

To prevent input offset currents from affecting the measurement accuracy, the filter resistor values should be less than 50 kΩ.

Ground Connection

It is always recommended that the thermocouple be connected to ground through a 100 kΩ to 1 MΩ resistor placed at the negative (inverting) input of the amplifier on the PCB (see Figure 30). This solution works well regardless of the thermocouple tip style.

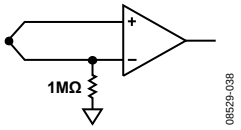


Figure 30. Ground the Thermocouple with a 1 MΩ Resistor

If there is no electrical connection at the measurement junction (insulated tip), the resistor value is small enough that no meaningful common-mode voltage is generated. If there is an electrical connection through a grounded or exposed tip, the resistor value is large enough that any current from the measurement tip to ground is very small, preventing measurement errors.

The AD849x inputs require only one ground connection or source of common-mode voltage. Any additional ground connection is detrimental to performance because ground loops can form through the thermocouple, easily swamping the small thermocouple signal. Grounding the thermocouple through a resistor as recommended prevents such problems.

Keeping the AD849x at the Same Temperature as the Reference Junction

The AD849x compensates for thermocouple reference junction temperature by using an internal temperature sensor. It is critical to keep the reference junction (thermocouple-to-PCB connection) as close to the AD849x as possible. Any difference in temperature between the AD849x and the reference junction appears directly as temperature error. Temperature difference between the device and the reference junction may occur if the AD849x is not physically close to the reference junction or if the AD849x is required to supply large amounts of output power.

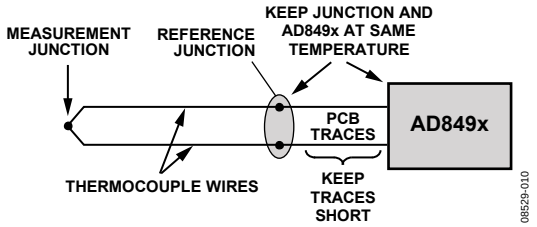


Figure 31. Compensating for Thermocouple Reference Junction Temperature

Driving the Reference Pin

The AD849x comes with a reference pin, which can be used to offset the output voltage. This is particularly useful when reading a negative temperature in a single-supply system.

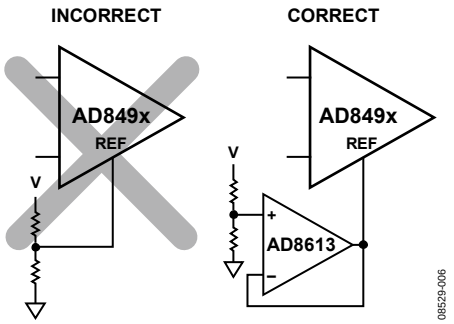


Figure 32. Driving the Reference Pin

For best performance, the reference pin should be driven with a low output impedance source, not a resistor divider. The AD8613 and the OP777 are good choices for the buffer amplifier.

Debugging Tip

If the AD849x is not providing the expected performance, a useful debugging step is to implement the ambient temperature configuration in Figure 34. If the ambient temperature sensor does not work as expected, the problem is likely with the AD849x or with the downstream circuitry. If the ambient temperature sensor configuration is working correctly, the problem typically lies with how the thermocouple is connected to the AD849x. Common errors include an incorrect grounding configuration or lack of filtering.

APPLICATIONS INFORMATION

BASIC CONNECTION

Figure 33 shows an example of a basic connection for the AD849x, with a J type or K type thermocouple input.

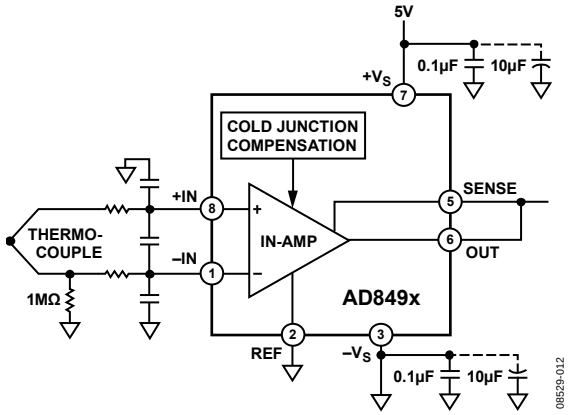


Figure 33. Basic Connection for the AD849x

To measure negative temperatures, apply a voltage at the reference pin to offset the output voltage at 0°C. The output voltage of the AD849x is

$$V_{OUT} = (T_{MJ} \times 5\ \text{mV/°C}) + V_{REF}$$

A filter at the input is recommended to remove high frequency noise. The 1 MΩ resistor to ground enables open thermocouple detection and proper grounding of the thermocouple. The sense pin should be connected to the output pin of the AD849x.

Decoupling capacitors should be used to ensure clean power supply voltages on $+V_S$ and, if using dual supplies, on $-V_S$, also. A 0.1 µF capacitor should be placed as close as possible to each AD849x supply pin. A 10 µF tantalum capacitor can be used farther away from the part and can be shared.

AMBIENT TEMPERATURE SENSOR

The AD849x can be configured as a standalone Celsius thermometer with a 5 mV/°C output, as shown in Figure 34. The thermocouple sensing functionality is disabled by shorting both AD849x inputs to ground; the AD849x simply outputs the value from the on-board temperature sensor.

As a temperature sensor, the AD8494 has a measurement temperature range of −40°C to +125°C with a precision output of

$$V_{OUT} = T_A \times 5\ \text{mV/°C}$$

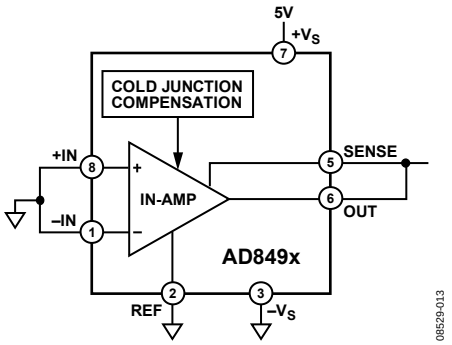


Figure 34. Ambient Temperature Sensor

The AD8494 is the best choice for use as an ambient temperature sensor. The AD8495, AD8496, and AD8497 can also be configured as ambient temperature sensors, but their output transfer functions are not precisely 5 mV/°C. For information about the exact transfer functions of the AD8494/AD8495/AD8496/AD8497, see the AN-1087 Application Note for additional details.

The thermometer mode can be particularly useful for debugging a misbehaving circuit. If the basic connection is not working, disconnect the thermocouple and short both inputs to ground. If the system reads the ambient temperature correctly, the problem is related to the thermocouple. If the system does not read the ambient temperature correctly, the problem is with the AD849x or with the downstream circuitry.

SETPOINT CONTROLLER

The AD849x can be used as a temperature setpoint controller, with a thermocouple input from a remote location or with the AD849x itself being used as a temperature sensor. When the measured temperature is below the setpoint temperature, the output voltage goes to $-V_S$. When the measured temperature is above the setpoint temperature, the output voltage goes to $+V_S$. For best accuracy and CMRR performance, the setpoint voltage should be created with a low impedance source. If the setpoint voltage is generated with a voltage divider, a buffer is recommended.

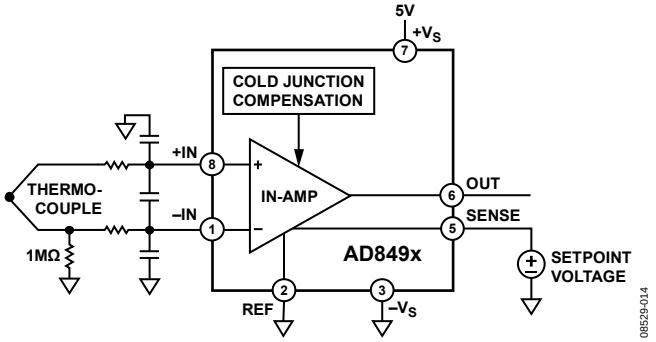


Figure 35. Setpoint Controller

Hysteresis can be added to the setpoint controller by using a resistor divider from the output to the reference pin, as shown in Figure 36. The hysteresis in °C is

$$T_{HYST} = \frac{V_S \times R1/(R1+R2)}{5\,\text{mV}/^\circ\text{C}}$$

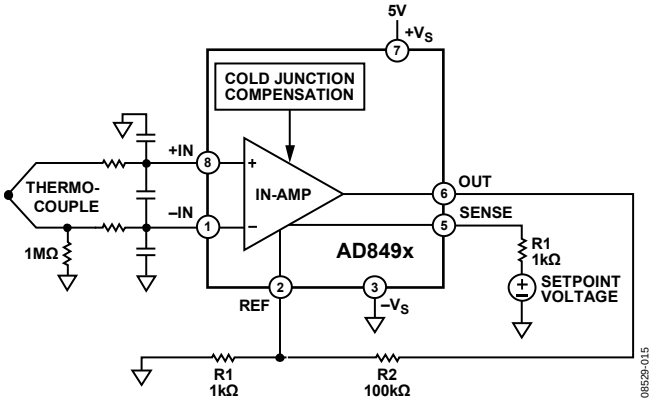


Figure 36. Adding 10 Degrees of Hysteresis

A resistor equivalent to the output resistance of the divider should be connected to the sense pin to ensure good CMRR.

MEASURING NEGATIVE TEMPERATURES

The AD849x can measure negative temperatures on dual supplies and on a single supply. When operating on dual supplies with the reference pin grounded, a negative output voltage indicates a negative temperature at the thermocouple measurement junction.

$$V_{OUT} = (T_{MJ} \times 5\ \text{mV}/^\circ\text{C}) + V_{REF}$$

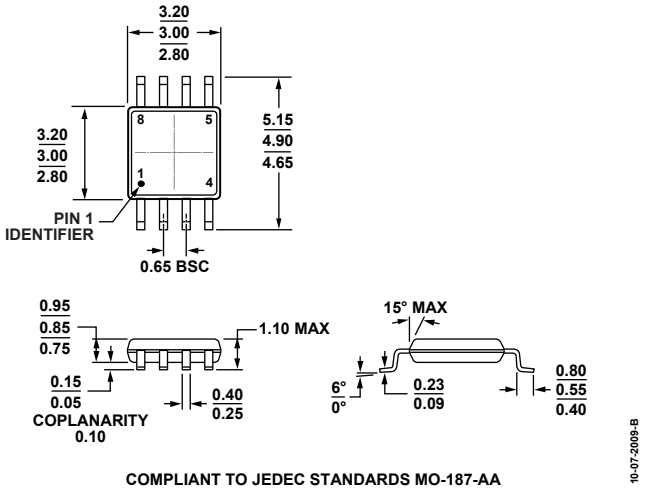
When operating the AD849x on a single supply, level-shift the output by applying a positive voltage (less than $+V_S$) on the reference pin. An output voltage less than V_{REF} indicates a negative temperature at the thermocouple measurement junction.

REFERENCE PIN ALLOWS OFFSET ADJUSTMENT

The reference pin can be used to level-shift the AD849x output voltage. This is useful for measuring negative temperatures on a single supply and to match the AD849x output voltage range to the input voltage range of the subsequent electronics in the signal chain.

The reference pin can also be used to offset any initial calibration errors. Apply a small reference voltage proportional to the error to nullify the effect of the calibration error on the output.

OUTLINE DIMENSIONS



COMPLIANT TO JEDEC STANDARDS MO-187-AA

Figure 37. 8-Lead Mini Small Outline Package [MSOP]
(RM-8)
Dimensions shown in millimeters

ORDERING GUIDE

Model[1]	Temperature Range	Package Description	Package Option	Branding
AD8494ARMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y36
AD8494ARMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y36
AD8494CRMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y37
AD8494CRMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y37
AD8495ARMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y33
AD8495ARMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y33
AD8495CRMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y34
AD8495CRMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y34
AD8496ARMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y3C
AD8496ARMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y3C
AD8496CRMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y3D
AD8496CRMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y3D
AD8497ARMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y39
AD8497ARMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y39
AD8497CRMZ	−40°C to +125°C	8-Lead MSOP	RM-8	Y3A
AD8497CRMZ-R7	−40°C to +125°C	8-Lead MSOP, 7″ Tape and Reel	RM-8	Y3A

[1] Z = RoHS Compliant Part.

D08529-0-6/11(C)