UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES

ARAGÓN

# "ELABORACIÓN DE UN SISTEMA DIGITAL DE ALTA RESOLUCIÓN PARA LA MEDICIÓN DE DEFORMACIONES POR MEDIO DE LA TÉCNICA DE EXTENSOMETRÍA ELÉCTRICA"

# TESIS

QUE PARA OBTENER EL TÍTULO DE:

**INGENIERO MECÁNICO ELECTRICISTA**

(ÁREA ELÉCTRICA-ELECTRÓNICA)

PRESENTAN:

**MAURICIO ONTIVEROS SALGADO**

**RICARDO SÁNCHEZ SÁNCHEZ**

ASESOR DE TESIS

**M. en C. ARTURO OCAMPO ÁLVAREZ**

FES Aragón

MÉXICO 2012

Un agradecimiento especial al Laboratorio de Mecánica Aplicada del Centro Tecnológico Aragón a través del convenio ADS Mexicana-Universidad Nacional Autónoma de México.

Por el apoyo recibido al proyecto ***Investigación y desarrollo tecnológico para mejorar la calidad de los productos y la prevención de fallas durante el servicio Segunda fase*****,** numero de convenio **29588-1668-19-VIII-11**

# Dedicatoria

A mis padres:

**Martin Sánchez Javier y Cleotilde Sánchez Ramírez**

Por darme la vida y haberme criado de la mejor manera posible. Por todo el cariño y amor recibido en estos 26 años. Por el esfuerzo hecho para ser el hombre que soy. Por su apoyo incondicional y también por soportar mis tropiezos.

A mis hermanas

**Alma Sánchez, Carla Sánchez, Mariana Mendoza**

Por su comprensión y apoyo, por ser la motivación que me mueve a ser mejor en el día a día.

A mi tío

**Félix Sánchez Ramírez**

Por su apoyo y gran esfuerzo por el bien de la familia en momentos de dificultad.

**Este trabajo es por y para ustedes!**

*Ricardo Sánchez Sánchez*

*Mayo 2012*

# Agradecimientos.

**A nuestro asesor el M. en C. Arturo Ocampo Álvarez**

Por el apoyo recibido durante la realización del proyecto.

**Al Ing. Javier Campillo Ortiz**

Por haberme permitido el tiempo necesario para la realización este proyecto.

**A mis amigos:**

Eric Rodríguez Juárez, Martha Esther López Rodríguez, Celeste Guadalupe Mosqueda Piña, Noé Benumea Aguilar, Martin Estrada Arcos, Iván Leos, Ángel Romero por haberme permitido compartir momentos agradables y especiales con ustedes.

**A Mauricio Ontiveros Salgado**

Por su amistad y apoyo durante la realización de este trabajo.

*Ricardo Sánchez Sánchez*

*Mayo 2012*

# Dedicatoria

**A mis padres:**

Margarita Salgado Luna y Benjamín Ontiveros Loa por su apoyo y comprensión ya que Ustedes también forman parte de esto. Gracias por todo, los amo!

**A mis hermanos:**

Laura y Gabriel por estar siempre a mi lado.

# Agradecimientos

A mi asesor, el profesor Arturo Ocampo por apoyarme en mi formación profesional.

Al profesor Narciso Acevedo por su apoyo y experiencias que me han sido muy útiles en mi formación académica.

A todos los compañeros del laboratorio de Mecánica Aplicada, especialmente a Ricardo Sánchez por su amistad y paciencia durante el desarrollo de este trabajo.

*Mauricio Ontiveros Salgado*

A nuestros sínodales:

Dr. Alejandro Antonio Vega Ramírez

M. en I. Fidel Gutiérrez Flores

Ing. Ramón Patiño Rodríguez

Un agradecimiento especial al Dr. Jacinto Cortez Pérez, al Mtro. Juan Gastaldi Pérez, al Ing. Adrian Paredes, al Ing. Raúl Barrón y al Ing. Julio Bernal por su apoyo durante el desarrollo del proyecto.

A mis amigos Eric Rodríguez, Noé Benumea, Martín Estrada, Ángel Romero, Jacinto Vargas, José Luis Mera, Iván Leos, Felipe Palomeque, Dulce Morales y Mauricio Morales por compartir momentos importantes.

*Mauricio Ontiveros Salgado*

*"La felicidad no se logra acumulando riquezas, títulos o fama, sino estando bien con nuestras conciencias, con nosotros mismos y con el prójimo"*

# INDICE

**CAPÍTULO 1**

**Antecedentes y Generalidades**

## CAPÍTULO 2

## Diseño del Hardware de sistema de adquisición

**CAPÍTULO 3**

**Diseño de Software para el procesamiento de datos**

## CAPÍTULO 4

## Implementación y pruebas del sistema

# Introducción

Para poder llevar a cabo el cálculo y análisis de los diferentes componentes de una máquina o de una estructura sometidos a diferentes cargas, es preciso conocer la distribución de fuerzas que tienen lugar en dichos componentes. La resistencia de materiales proporciona la base teórica necesaria para predecir estas características, permitiendo comprender el comportamiento de estos elementos sometidos a cargas o esfuerzos.

Sin embargo, en algunos casos puede ocurrir que el análisis teórico no sea suficiente, y por lo tanto es preciso llevar a cabo una serie de medidas de carácter experimental para poder completar el diseño o análisis.

La extensometría es la técnica más utilizada para el análisis experimental de tensiones. Su fundamento básico es la variación de la resistencia producida en un hilo conductor cuando se alarga o se contrae, y se emplea en otras aplicaciones como por ejemplo la construcción de transductores.

Las galgas extensométricas sólo son capaces de medir deformaciones locales en el lugar en donde se adhieran y pueden ser manufacturadas tan pequeñas como para permitir un análisis  y realizar un estudio de esfuerzos sobre el material en el que se instrumente.

El puente de Wheatstone es el circuito ideal para la medición de cambios de resistencias en escalas mínimas y es el más común en los equipos comerciales utilizados en el campo de la extensometría.

La elaboración de un sistema de adquisición de datos con características especificas y desarrollado a un bajo costo permitirá el análisis de esfuerzos y deformaciones por medio de la técnica de extensometría eléctrica con resultados similares a los proporcionados por los equipos comerciales.

En los siguientes cuatro capítulos se describen los conceptos generales de los elementos de una tarjeta de adquisición de datos, los fundamentos teóricos  de la extensometría eléctrica y el diseño y la elaboración del sistema digital de alta resolución.

En el capítulo uno se describe de forma general los conceptos teóricos de los dispositivos utilizados en el sistema de alta resolución  así como los fundamentos teóricos de la extensometría eléctrica.

En el capítulo dos se describe el diseño del hardware del sistema de alta resolución y se hace una explicación general del funcionamiento del código del microcontrolador.

En el capítulo tres se explica el software en la computadora que se utiliza para el procesamiento y almacenamiento de los datos.

Por último en el capítulo cuatro se describe el diseño y desarrollo de los circuitos impresos utilizados en el sistema de alta resolución. También se presentan las pruebas realizadas al sistema y los resultados finales obtenidos.

# Justificación

Actualmente existen en el mercado diversos equipos capaces de medir deformaciones basados en la técnica de extensometría, estos equipos poseen características que los haces ideales para la medición precisa de las deformaciones en los materiales. Por las características tan precisas de estos equipos sus costos son demasiado elevados.

En el laboratorio de Mecánica Aplicada del Centro Tecnológico Aragón y de acuerdo al proyecto "**Investigación y desarrollo tecnológico para mejorar la calidad de los productos y la prevención de fallas durante el servicio Segunda fase"** se ha diseñado un sistema digital de alta resolución económico y que tiene el mismo principio de funcionamiento que un equipo comercial. Además de tener características especificas en la aplicación de medición de deformaciones de los materiales utilizando galgas extensométricas en un arreglo de puente de Wheatstone en configuración puente completo.

# CAPÍTULO 1

## Antecedentes y Generalidades

En este capítulo se expondrán de forma general los conceptos teóricos de los dispositivos implementados en la tarjeta de adquisición de datos y de las comunicaciones serie utilizadas.

### 1.1 Instrumentación.

La medida de variables con precisión y fiabilidad adecuadas constituye uno de los puntos clave en cualquier tecnología.

El proceso de medición generalmente requiere el uso de un instrumento como medio físico para determinar la magnitud de una variable. Los instrumentos constituyen una extensión de las facultades humanas y en muchos casos permiten a las personas determinar el valor de una cantidad desconocida la cual no podría medirse utilizando solamente los sentidos físicos. Por lo tanto se puede definir a un instrumento como un dispositivo capaz de determinar la magnitud de una variable física. Un instrumento electrónico se basa en principios y fundamentos eléctricos para llevar a cabo la medición de la variable y puede ser tan sencillo como complejo. Para esto, se deben de comprender los principios de operación y valorar la importancia para las aplicaciones deseadas.

#### 1.1.1. Conceptos generales para la instrumentación.

A continuación se presentan algunos conceptos importantes con los que se cuenta en la instrumentación:

Exactitud: Se refiere a la capacidad del instrumento de medir o determinar el valor más cercano al real de la variable a medir.

Precisión: Es referente a la capacidad del instrumento de llevar a cabo una serie de mediciones y siempre obtener mediciones similares de la variable que se esté midiendo.

Sensibilidad: Capacidad del instrumento de medir la variable respecto a la de a variable medida.

Resolución: Cambio más pequeño de la variable que el instrumento puede medir.

### 1.1.2. Extensometría Eléctrica

La extensometría eléctrica es una técnica de medida de deformaciones que utiliza un sensor llamado galga extensométrica.

Una galga extensométrica consiste en una fina película metálica en forma de hilo plegado depositada sobre una lámina de plástico aislante de algunas micras de espesor.

Cuando la galga se pega sobre la superficie de una pieza, si se aplican cargas y la pieza se de forma, la galga también lo hace. Así el hilo metálico experimenta un alargamiento o acortamiento que modifica su resistencia eléctrica. La variación de resistencia se puede evaluar con precisión en un equipo de medida, y es proporcional a la deformación de la galga según la dirección en la que está situada.

Por tanto, una galga extensométrica puede medir la deformación longitudinal unitaria en un punto P en que se encuentra situada y según la dirección en la que está orientada.

## 1.2 Galgas extensométricas

Una galga extensométrica (Strain Gage) es un dispositivo de medida universal que se utiliza para la medición electrónica de diversas magnitudes mecánicas como pueden ser la presión, carga, torque, deformación, posición, etc. Se entiende por Strain o esfuerzo a la cantidad de deformación de un cuerpo debida a la fuerza aplicada sobre él. En términos matemáticos Strain se define como la fracción de cambio en longitud, como se muestra en la figura 1.1.

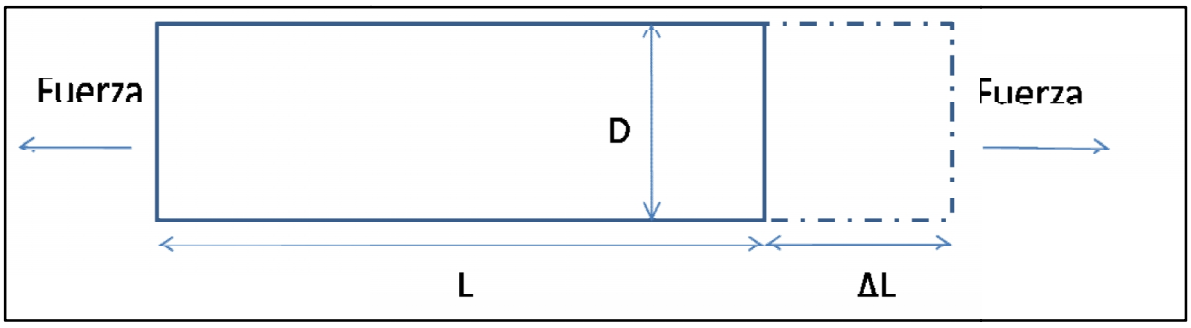



*Figura 1.1*

El parámetro Strain puede ser positivo (tensión) o negativo (compresión), en cualquier caso el valor es adimensional. En la práctica los valores de Strain suelen ser demasiado pequeños por lo que se expresa como microdeformación (micro-strain) με, que es ε ×10 $^{-6}$

El Strain gage mas utilizado es el confinado a papel metálico o *"blonded metallic strain gage"* Este tipo de strain gage consiste en un cable muy fino o papel aluminio dispuesto en forma de parrilla la cual maximiza la cantidad de metal sujeto al esfuerzo en la dirección paralela. La rejilla está pegada a un fino respaldo llamado "*carrier*" el cual está sujeto directamente a la pieza bajo medida. Por lo tanto el esfuerzo experimentado por la pieza es transferido directamente a la galga extensométrica la cual responde con cambios lineales de resistencia. Los strain gages se encuentran en el mercado con valores nominales de resistencia de 30 a 3000Ω.

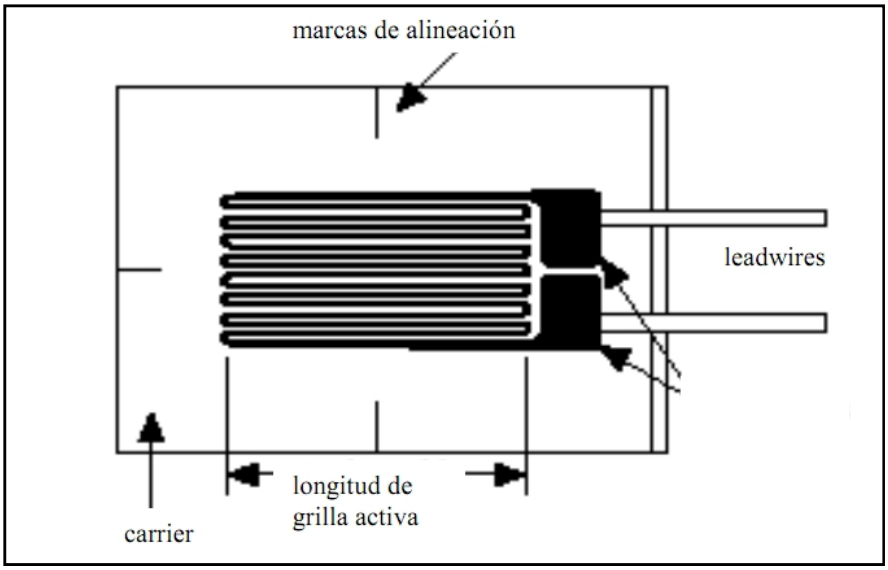



*Figura 1.2 Strain gage metálico*

Es de suma importancia que el strain gage sea apropiadamente montado sobre la pieza para que el esfuerzo sea transferido adecuadamente desde la pieza a través del adhesivo y el material de respaldo hasta la misma rejilla metálica.

Un parámetro fundamental de los strain gages es la sensibilidad al esfuerzo, expresado cuantitativamente como el factor de galga (GF). El factor de galga es definido como la relación de variación fraccional de resistencia eléctrica y la variación fraccional de longitud.

$$GF = \frac{\Delta R/R}{\Delta L/L} = \frac{\Delta R/R}{\varepsilon}$$

El factor de galga típico para un strain gage metálico es de aproximadamente 2.

### 1.2.1. Tipos de aleaciones para strain gages.

El componente principal que determina las características de operación de un strain gage es la aleación sensible al esfuerzo que compone la grilla de papel metálico.

El proveedor *Vishay Micro-Measurements* ofrece la siguiente variedad de aleaciones de strain gages:

• Aleación A: *Constantan*, una aleación de cobre y nickel, autocompensado por temperatura. Provee la mejor combinación global de propiedades necesarias para la mayoría de las aplicaciones de los strain gages. Esta aleación tiene una alta sensibilidad al esfuerzo, o factor de galga, y es relativamente insensible a la temperatura. Su resistividad es lo suficientemente alta como para lograr valores adecuados de resistencia aún para pequeñas grillas y su coeficiente de temperatura para resistencias no es excesivo.

• Aleación P: *Constantan* templado. Para la medición de esfuerzos muy grandes, 5% (50000 microstrain) o más, el *constantan* templado (aleaciones P) es seleccionado normalmente como el material de la grilla. El *constantan* es muy dúctil y en strain gages con longitudes de 3mm o más, pueden ser tensados más de un 20%. Se debe tener en cuenta, sin embargo, que bajo grandes esfuerzos cíclicos la aleación P exhibirá cambios permanentes en la resistencia eléctrica provocando un cambio en la graduación del cero en la galga.

• Aleación D: Aleación isoelástica de cromo y nickel. Cuando las medidas de esfuerzo son puramente dinámicas – esto es, cuando no es necesario mantener estable una referencia cero – la aleación isoelástica (aleación D) ofrece ciertas ventajas. Dentro de las principales, podemos citar una larga vida útil soportando la fatiga comparada con las aleaciones A, y un alto factor de galga aproximadamente 3.2 que mejora la relación señal a ruido en pruebas dinámicas.

• Aleación K: Aleación de cromo y nickel; Karma autocompensado por temperatura. La aleación Karma o Karma modificada, con su amplia área de aplicación representa a un miembro importante en la familia de aleaciones de strain gages. Esta aleación se caracteriza por una buena vida útil y excelente estabilidad y es la selección preferida para mediciones estáticas de esfuerzo de alta precisión durante largos períodos de tiempo (meses o años) a temperatura ambiente.

### 1.2.2. Material de respaldo o “carrier”

La confección de strain gages se realiza mediante un grabado del papel metálico sobre un material de respaldo o “carrier” que cumple con las siguientes funciones:

- Proveer el medio de sustento a la grilla metálica durante la instalación.
- Presentar una superficie para confinar y pegar la galga al material de prueba.
- Proveer un aislamiento eléctrico entre la grilla y el material de prueba.

Los materiales de respaldo provistos por Micro-Measurements para sus strain gages son básicamente de dos tipos: polímeros y epoxy-fenólicos reforzados con fibra de vidrio. En el caso de las aleaciones sensibles al esfuerzo, los materiales de respaldo no son parámetros independientes, se presentan en combinaciones de aleaciones y material de respaldo con características constructivas especiales a los que llaman sistemas y se les aplican designadores de series. Como resultado, cuando se llega a un tipo óptimo de galga para una aplicación en particular, el proceso no permite la combinación arbitraria de aleación con material de respaldo sino que requiere la especificación de una de las series disponibles en particular. A continuación se detallan los diferentes materiales de respaldo para cada una de las series disponibles.

*Polímeros:*

El polímero clase E es un material de respaldo duro, extremadamente flexible y puede ser contorsionado para caber en pequeños radios. Además, debido a la gran resistencia del conjunto material de respaldo-aleación a base de polímeros, estas galgas son mucho menos sensibles a ser dañadas durante su instalación. Gracias a su durabilidad e idoneidad para uso sobre rangos de temperatura que van desde -195 a +175 °C, los materiales de respaldo a base de polímeros son una opción ideal para medidas de esfuerzo tanto estáticas como dinámicas. Este material de respaldo es capaz de soportar grandes elongaciones y puede ser utilizado para medir elongaciones plásticas excesivas de un 20%.

*Epoxi-fenólicas:*

Los materiales de respaldo epoxi-fenólicos reforzados con fibra de vidrio son la mejor elección para un excepcional desempeño sobre un amplio rango de temperaturas. Estas materiales pueden ser usados tanto para medidas estáticas como dinámicas desde - 269 a +290°C. En aplicaciones de corta duración, la temperatura superior puede ser extendida hasta los -750°C. La máxima elongación aceptable por el material es limitada al 1 o 2.

### 1.2.3. Características de un strain gage.

*Longitud de una galga:*

Es la región activa o longitud de la grilla sensible al esfuerzo de una galga. Los codos y almohadillas de soldadura no se consideran sensibles al esfuerzo debido a su gran sección transversal y su baja resistencia eléctrica.

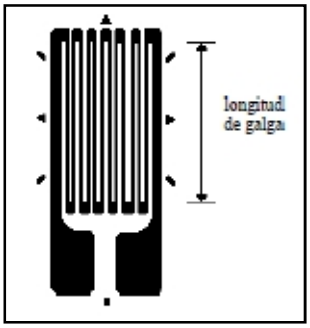



*Figura 1.3*

*Concentración del esfuerzo:*

La longitud de la galga es usualmente un factor muy importante a la hora de determinar su rendimiento bajo ciertas circunstancias. Por ejemplo, las medidas de esfuerzo son realizadas, en general, sobre las piezas o estructuras críticas de una máquina (sometidas a mayor esfuerzo). Y muy frecuentemente, las piezas más fatigadas son las que se encuentran sometidas mayor esfuerzo, donde el gradiente de esfuerzo es más pronunciado y el área de mayor esfuerzo se circunscribe a una pequeña región. Los strains gages tienden a integrar, o promediar, el área cubierta por la grilla. Puesto que el promedio de la distribución de un esfuerzo no uniforme es siempre menor al máximo, un strain gage que es más larga que la máxima región de esfuerzo, indicará una magnitud de esfuerzo muy bajo. La figura siguiente ilustra de forma representativa la distribución de esfuerzo en la vecindad de la concentración de esfuerzo y demuestra el error en el esfuerzo indicado para un strain gage demasiado largo con respecto a la zona de máximo esfuerzo.

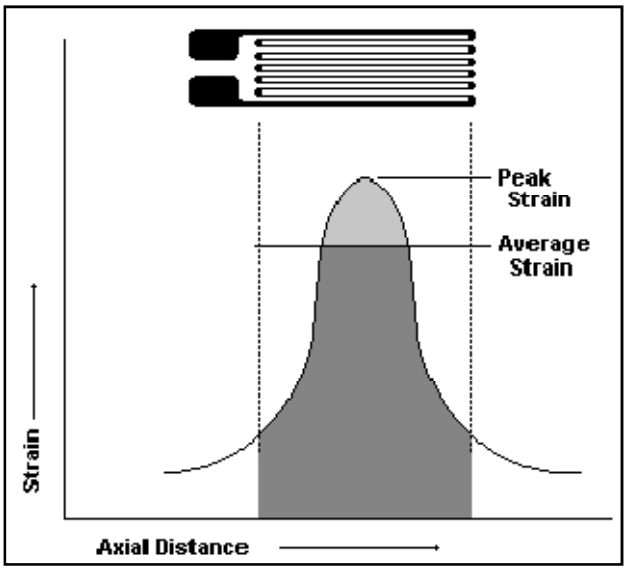



*Figura 1.4*

Como una regla general, en lo posible, la longitud de la galga no debe ser mayor a la dimensión de la causa del esfuerzo para que la medición sea aceptable. Cuando la causa del esfuerzo es pequeña, digamos del orden de 13mm, la regla general conduciría a longitudes de galgas muy chicas. Puesto que el uso de galgas muy pequeñas introduce otros tipos de problemas, se tiene que llegar a una relación de compromiso.

*Galgas cortas:*

Los strain gages cuya longitud es de alrededor de 3mm tienden a exhibir su rendimiento un tanto degradado (particularmente con respecto a su máxima elongación, su estabilidad bajo esfuerzo estático y su durabilidad cuando está sometida a esfuerzo cíclico alternativo). Cuando cualquiera de estas características empobrece la precisión de la medición en mayor medida que el promedio del esfuerzo se justifica la utilización de una galga de mayor longitud.

*Galgas largas:*

Las galgas largas ofrecen ciertas ventajas que valen la pena mencionar. Son, casi siempre, más fáciles de manipular en todos los aspectos de la instalación y cableado que las galgas miniatura (13mm). Más aún, las galgas largas proveen una mejor disipación de calor porque debido a su resistencia nominal tienen menor potencia por unidad de área de grilla. Estas consideraciones pueden ser muy importantes a la hora de trabajar sobre materiales plásticos u otros materiales con pobre disipación de calor. Una inadecuada disipación de calor trae aparejada una sobre elevación de temperatura en la grilla, material de respaldo, adhesivo y superficie de prueba, y puede afectar notablemente el rendimiento y la precisión.

*Patrón de grilla:*

El patrón de grilla se refiere a la forma de la grilla, el número y orientación de las grillas en las galgas multi-grillas o rosetas, la configuración de las almohadillas y varias características constructivas que son estandar para un patrón particular. La gran variedad de patrones disponibles se han diseñado para satisfacer el amplio rango de instalaciones medidas a través de strain gages.

*Galgas uniaxiales:*

Consiste en una galga de simple grilla, con patrón apropiado a una aplicación particular que depende particularmente de:

*Almohadillas*:

Deben ser, por supuesto, compatibles en tamaño y orientación con el espacio disponible. Además, es importante que el arreglo de almohadillas sea tal que facilite al operario realizar las conexiones pertinentes.

*Ancho de grilla:*

Cuando existen severos gradientes de esfuerzo perpendiculares al eje de la galga sobre la superficie de prueba, una grilla estrecha minimizará el error por promediación. Las grillas amplias, cuando sean apropiadas para la instalación, mejorarán la disipación de calor y reforzarán la estabilidad (particularmente cuando la galga se instale sobre un material con pobre transferencia de calor)

*Resistencia de galga:*

En ciertas instancias, la única diferencia entre dos galgas disponibles de la misma serie, es la resistencia eléctrica (típicamente 120 ohms contra 350 ohms). Cuando existen estas opciones, la galga con mayor resistencia se prefiere pues reduce la disipación la generación de calor en un factor de tres (el mismo voltaje se aplica a la galga). También se tiene como ventaja a la disminución de ciertos efectos debido a las pistas como la menor sensibilidad del circuito gracias a la resistencia de las pistas tanto como a las variaciones de señal indeseadas a causa de los cambios de resistencia con fluctuaciones de temperatura. Además, cuando las galgas incluyen llaves, bucles o fuentes aleatorias de resistencia variable, la relación señal a ruido se ve mejorada con galgas con resistencia mayor para el mismo nivel de potencia. En análisis experimentales de esfuerzos, una galga de simple grilla se utilizaría si sólo si se conoce de forma fehaciente que el esfuerzo en el punto de medida es uniaxial y la dirección de dichos esfuerzos es conocida con una precisión razonable. Estos requerimientos limitan seriamente el campo de aplicación de los strain gages de simple grilla en el análisis de esfuerzos y la no consideración de esfuerzos biaxiales pueden llevar a grandes errores de medición.

*Autocompensación de temperatura:*

Una importante propiedad compartida tanto por los strain gages de *constantan* y Karma modificado es su respuesta a procesos especiales gracias a su autocompensación de temperatura. Las galgas autocompensadas son diseñadas para producir una salida térmica mínima (la temperatura induce esfuerzos aparentes) sobre un rango de temperatura que va desde los -45ºC a los +200ºC. Cuando se elige un strain gage compuesto tanto de *constantan* (aleación A) o un Karma modificado (aleación K), se debe especificar el número STC. Este número es una aproximación al coeficiente de expansión térmico en ppm/ºF del material estructural sobre el cual la galga mostrará una mínima salida térmica (figura 1.5).

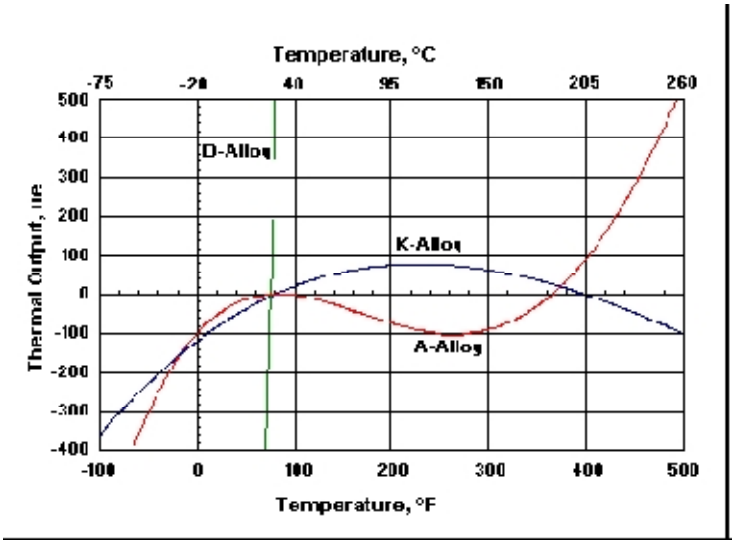



*Figura 1.5*

El gráfico térmico de salida ilustra las características térmicas típicas de salida para aleaciones de tipo A y K. La salida térmica de aleaciones no compensadas isoelásticas (D) se incluye en el mismo gráfico con propósitos comparativos. En la práctica, el número STC para una galga tipo A o K, se elige los más próximo posible al coeficiente de expansión termal de la pieza de prueba. Sin embargo, las curvas térmicas de salida para estas aleaciones, pueden ser rotadas alrededor de la temperatura de referencia ambiente para favorecer un rango de temperatura particular. Esto se hace errando el número STC y el coeficiente de expansión a propósito en la dirección apropiada. Cuando el número STC seleccionado es menor que el coeficiente de expansión, la curva es rotada en sentido contrario al de las agujas del reloj. Errando en sentido opuesto la curva es rotada en sentido de las agujas del reloj. Si el número STC es incorrecto, las curvas de salida térmica para aleaciones tipo A y K (suministradas con cada paquete de strain gages) no son aplicables, y será necesario calibrar la salida térmica de la instalación como un función de la temperatura.

Para la realización de este proyecto los strain gages utilizados son de la marca Vishay Micro-Measurements con un valor de resistencia de 350.0±0.5% a 24ºC, con un factor de galga (GF) de 2.060±0.2% a 24ºC y con una sensibilidad transversal de +0.9±0.2% a 24ºC

### 1.3 Acondicionamiento de las galgas extensométricas por medio del puente de Wheatstone

*Puente Wheatstone*

El puente de Wheatstone tiene cuatro ramas resistivas (figura 1.6), junto con una fuente de alimentación y un detector de cero, que generalmente es un medidor sensible a la corriente. La corriente a través del medidor depende de la diferencia de potencial entre los puntos c y d. Se dice que el puente esta balanceado (o en equilibrio) cuando la diferencia de potencial a través del medidor es 0V, de forma que no hay paso de corriente a través de él. Esta condición se cumple cuando el voltaje del punto c al punto a es igual que el voltaje del punto d al punto a; o bien, tomando como referencia la otra terminal de la fuente de alimentación, cuando el voltaje del punto c al punto b es igual que el voltaje del punto d al punto b. Por lo tanto, el puente esta en equilibrio cuando

$$I_1 R_1 = I_2 R_2$$

Si la corriente del medidor es cero, la siguiente condición también se cumple

$$I_1 = I_3 = \frac{E}{R_1 + R_3}$$

$$I_2 = I_4 = \frac{E}{R_2 + R_4}$$

Al combinar las ecuaciones y simplificándolas se obtiene

$$\frac{R_1}{R_1 + R_3} = \frac{R_2}{R_2 + R_4}$$

De la cual

$$R_1 R_4 = R_2 R_3$$

La ecuación anterior es conocida para el equilibrio del puente de Wheatstone. Si tres de las resistencias tienen valores conocidos, la cuarta puede establecerse a partir de la ecuación anterior.

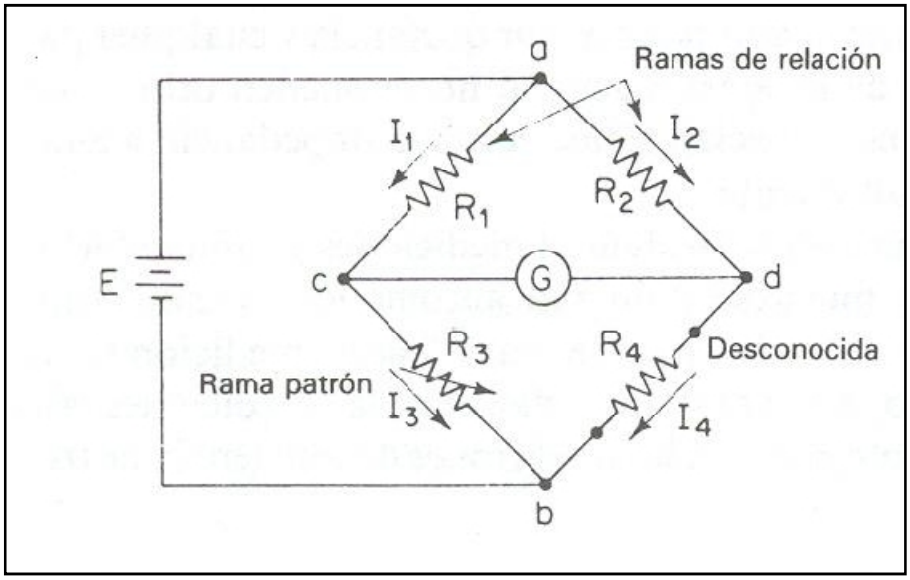



*Figura 1.6*

La resistencia R3, se denomina rama patrón del puente, y las resistencias R1 y R2 tienen el nombre de ramas de relación.

La medición de la resistencia desconocida R, es independiente de las características o de la calibración del galvanómetro detector de cero, puesto que el detector de cero tiene suficiente sensibilidad para indicar la posición de equilibrio del puente con el grado de precisión requerido.

Para medir la resistencia, primero debe encontrarse una técnica para convertir el cambio de resistencia en una corriente o voltaje para exhibirla en un amperímetro o multímetro. Si se tiene que medir un pequeño cambio de resistencia, se obtendrá un cambio muy mínimo de voltaje o corriente. Por lo tanto se necesita un circuito que permita amplificar solo la diferencia en voltaje a través del sensor de deformación causado por un cambio en su resistencia.

### 1.3.1. Puente Básico de resistencia.

Colocando la  galga extensométrica se coloca en un brazo del puente, como se muestra en la figura siguiente). Suponiendo que la galga no se encuentra deformada de tal manera que su resistencia = *R*. DE igual manera R1, R2, R3 son iguales a *R.* En estas circunstancias E1 = E2 = E/2 y E1-E2=0. Se dice que el

puente esta balanceado. Si el sensor se deformación se comprime, R disminuirá en ΔR y el voltaje diferencial E1 – E2 está definido por

$$E_1 - E_2 = E\frac{\Delta R}{4R}$$

Esta aproximación es válida porque 2ΔR « 4R para el sensor de deformación. La ecuación anterior muestra que E debe hacerse grande para maximizar el voltaje diferencial de la salida, E1-E2.

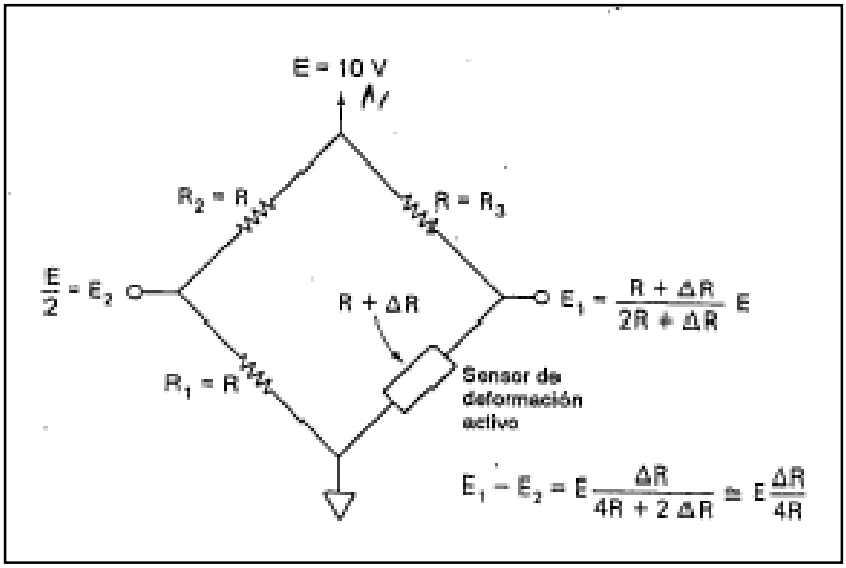



*Figura 1.7 El arreglo de resistencias en puente y e voltaje de alimentacion E convierte un cambio en la resistencia ΔR del sensor de deformacion en un voltaje de salida diferencial E1 – E2.*

### 1.3.2. Efectos térmicos en el balance del puente.

Aun cuando se logra balancear el circuito anterior, no se mantendrá en balance porque los ligeros cambios de temperatura del sensor de deformación producirán cambios de resistencia iguales o mayores que los ocasionados por la deformación. Este problema se resuelve al montar otro sensor idéntico contiguo al activo, de manera que ambos compartan un mismo ambiente térmico. Por consiguiente, al variar la temperatura, los cambios de resistencia del sensor agregado corresponden exactamente a la resistencia del sensor activo. El sensor agregado

produce compensación automática de temperatura, por lo cual se le denomina sensor de compensación de temperatura o falso sensor.

El sensor de compensación de temperatura se monta con su eje transversal perpendicular al eje transversal del sensor de trabajo, como se muestra en la siguiente figura. El nuevo sensor está colocado en lugar de la resistencia R1 en el circuito puente. Una vez que el puente se ha balanceado, la resistencia R del sensor de compensación de temperatura y la del sensor de trabajo se siguen una a la otra para mantener el puente en balance, cualquier desbalanceo, esta provocado estrictamente por ∆R del sensor de trabajo debido a la deformación,

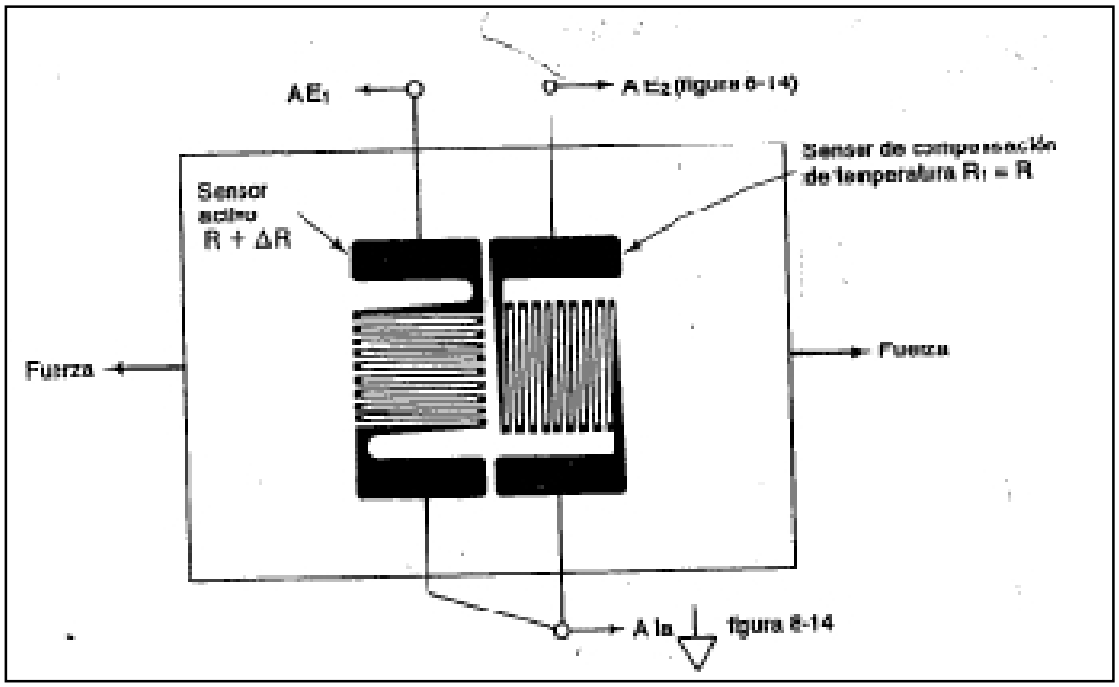



*Figura 1.8 El sensor de deformacion para compensacion de temperatura tiene los mismos cambios de resistencia que el sensor activo al incrementarse la temperatura. Solamente el sensor activo cambia su resistencia con la deformacion.*

### 1.3.3. Incremento en la salida del puente de deformación

Con un solo sensor de deformación activo y un sensor de compensación de temperatura producen una salida en puente diferencial

$$E_1 - E_2 = E\frac{\Delta R}{4R}$$

El voltaje de salida del puente E1-E2 puede duplicarse al duplicarse el número de sensores activos. Los sensores 1-2 y 5 y 6 son los activos y se incrementa su resistencia (tensión) si aplica la fuerza como se muestra. Al arreglar los sensores activos en los brazos opuestos del puente y los sensores de compensación de temperatura en los otros brazos, la salida del puente es:

$$E_1 - E_2 = E\frac{\Delta R}{2R + \Delta R} \cong E\frac{\Delta R}{2R}$$

Si los miembros estructurales experimentan flexiones, se puede obtener una mayor estabilidad del puente. El lado superior de la barra será alargado (tensión) para incrementar la resistencia del sensor activo en (+) ΔR. El lado inferior de la barra se acortara (comprensión) para reducir la resistencia del sensor en (-) ΔR.

Los sensores de tensión 1-2 y 5 -6 están conectados en los brazos opuestos del puente. Los sensores en compresión 3-4 y 7-8 están conectados en los lados opuestos remanentes del puente. Los sensores también se compensan por temperatura unos con otros. La salida del arreglo del cuatro sensores de deformación se cuadruplica con respecto al puente de un solo sensor a

$$E_1 - E_2 \cong E\frac{\Delta R}{2R}$$

### 1.4 Convertidor Analógico Digital

Un convertidor analógico a digital es el que nos permite transformar una señal lineal (voltaje) en una representación digital (código binario) para poder ser tratada por los sistemas de procesamiento tal como se muestra en la figura 1.9.

Las magnitudes que se obtienen del mundo en el que vivimos son parámetros físicos mientras que los circuitos integrados procesan información en forma digital. De ahí la necesidad de establecer esta transformación de señales.

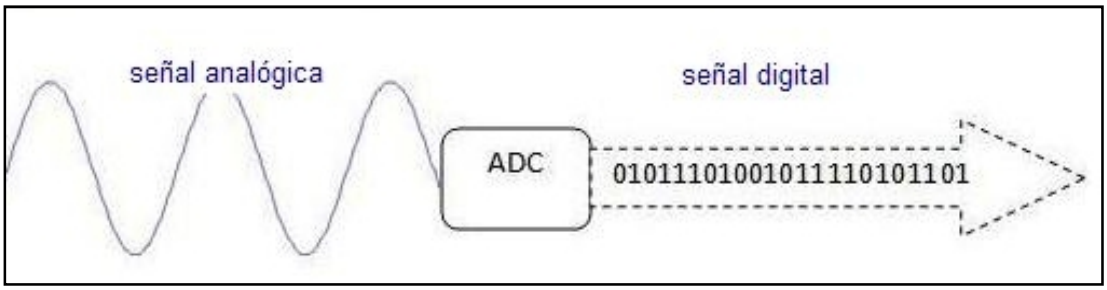



*Figura 1.9 Conversión Analógica-Digital*

Existen tres procesos que intervienen en la conversión analógica a digital y son el muestreo, la cuantificación y codificación.

### 1.4.1. Muestreo de la señal

En el proceso de muestreo se toma una muestra de la señal analógica y se mantiene el tiempo necesario para que el convertidor A/D la transforme en una señal digital. Básicamente es un condensador que se conecta en paralelo con la entrada durante el tiempo de muestreo este se carga con la tensión analógica por lo que mantiene la muestra durante el tiempo de la conversión al formato digital.

La transformación más habitual es que la señal muestreada esté formada por los valores de la señal original en instantes de tiempo equiespaciados[1], se habla entonces de muestreo uniforme.

### 1.4.2. Muestreo Natural

En la figura 1.10 se representa un modelo grafico del muestreo uniforme, cuando se considera que la señal muestreadora *m(t)* es un tren de impulsos de amplitud uno y duración finita t (de aquí la denominación de muestreo natural) separados un tiempo T, que se denomina periodo de muestreo.

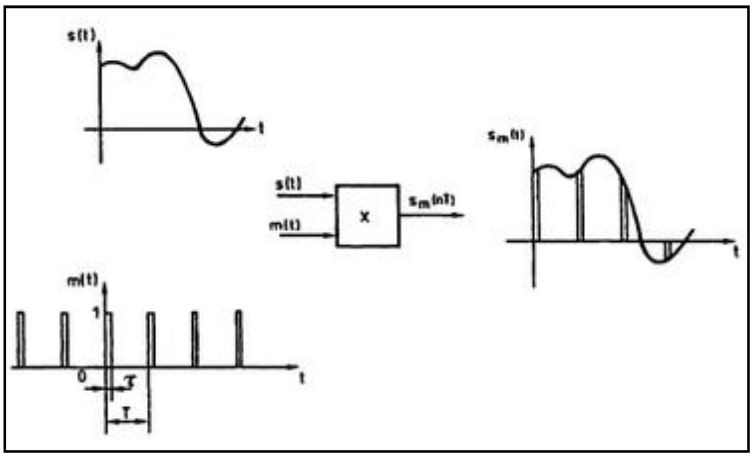



*Figura 1.10*

La señal a muestrear *s(t)* se supone que es de banda limitada con máxima componente frecuencial*fM* y mínima 0 (0-*fM* en su descripción mediante análisis de Fourier). La señal muestreada Sm es un tren de impulsos modulados en amplitud, es decir, es discreta en el tiempo pero continua en amplitud y puede interpretarse como el producto de la señal de entrada por la muestreadora.

$$Sm = s(t)\, m(t)$$

El teorema de muestreo establece la relación que debe haber entre la máxima componente frecuencialfM y el periodo de muestreo T.

Otra forma de interpretar el teorema de muestreo es que la señal se debe muestrear cuando menos dos veces en cada ciclo o periodo de su componente mayor de frecuencia.

$$T < {}^{1}/_{(2fM)seg}$$

### 1.4.3. Cuantificación

La cuantificación a diferencia del muestreo es un proceso no lineal que convierte una señal de amplitud continua en otra amplitud discreta, es decir, con un número finito de valores o niveles de amplitud. En cada muestra se observa en que rango de voltaje se encuentra la señal y en función de esto se le asigna un nivel de voltaje a la salida.

En la figura 1.11 se representa un proceso de cuantificación uniforme. En este ejemplo, al conjunto de valores de entrada de entre 0 y q/2 se les ha asignado el

nivel o estado 1, a los valores entre q/2 y 3q/2 el estado 2, a los valores entre 3q/2 y 5q/2 el estado 3 y así sucesivamente.

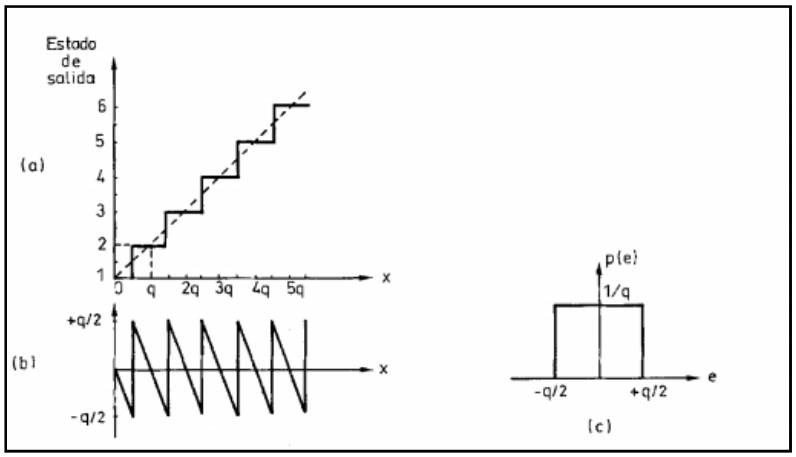



*Figura 1.11*

Toda la gama de amplitudes que pueden tomar las muestras se divide en intervalos por lo que el valor q se denomina intervalo de cuantificación, y coincide con la diferencia entre el mayor y menor valor de la entrada a los que se les asigna el mismo estado de salida.

El número de estados de salida expresado en bits (n) determina la resolución del cuantificador; $2^n$=N es el numero de estados. Si por ejemplo se trabaja con 16 bits la resolución seria de 65,536.

### 1.4.4. Codificación

En el proceso de codificación se representa de forma biunívoca cada uno de los estados de salida del cuantificador por un símbolo elegido de un alfabeto. Cuando el alfabeto consta sólo de cifras 1 o 0, y la posición de cada cifra dentro de un número se corresponde con una potencia entera de 2, se habla entonces de códigos binarios. Cada cifra o código binario se denomina bit.

## 1.5 Convertidores Analógicos-Digitales

Existen diferentes tipos de convertidores A/D, entre los más usuales se encuentra el convertidor tipo Delta-Sigma, el de aproximaciones sucesivas y el tipo flash.

### 1.5.1. Convertidor Delta-Sigma

También conocido como convertidor de sobremuestreo son muy usados en aplicaciones en donde se requiere una alta resolución. Contiene dos partes principales un modulador integrador y un filtro digital. El primer bloque realiza la resta de la señal analógica de entrada y la proveniente de un convertidor D/A (de

ahí la delta) y posteriormente el resultado se integra (de ahí la sigma) y introduce a un comparador cuya salida es una secuencia de unos y ceros a alta velocidad. El filtro digital elimina el ruido de alta frecuencia introducido por el modulador analógico. El diezmador ofrece las muestras de salida a una velocidad menor de la disponible a la salida del comparador, pero con mayor resolución

### 1.5.2. Convertidor de aproximaciones sucesivas

Este tipo de convertidor suele usarse en aplicaciones en donde se requiere trabajar a altas velocidades. En la figura 1.12 se muestran los elementos básicos para el funcionamiento del convertidor. La conversión en este dispositivo consiste en ir comparando la tensión de entrada con una tensión analógica generada internamente con un convertidor D/A, cuya entrada digital se incrementa o decrementa según que el resultado de la comparación indique.

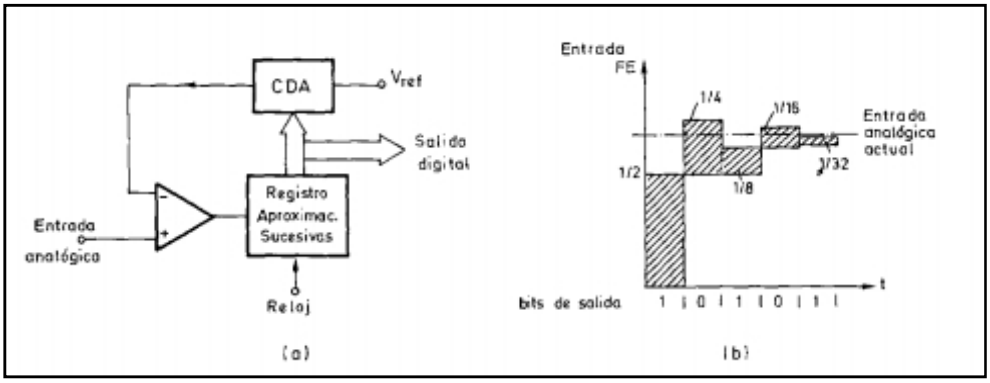



*Figura 1.12 En el esquema (a) esta simplificado un convertidor A/D basado en el algoritmo de aproximaciones sucesivas. En el esquema (b) se representa la asignación del valor de los bits de salida en comparaciones sucesivas.*

## 1.6 Amplificador operacional

El amplificador operacional es un dispositivo acoplado directamente y responde sólo a la diferencia de voltaje entre las dos terminales de entrada y no al potencial común de éstas, es decir, con respecto a tierra. Una señal positiva en la entrada inversora (-) produce una señal negativa a la salida, esa misma señal de entrada a la terminal no inversora (+), producirá una señal positiva.

La función que realiza un amplificador operacional es tomar la información de los voltajes en las terminales de entrada, en la forma de voltaje diferencial de entrada

(Vd) y multiplicar a éste por la ganancia de voltaje en lazo abierto ($A_0$) del amplificador, entregando el voltaje de salida.

$$V_o = A_o V_d$$

Por lo que la salida del amplificador ideal con entrada diferencial, solo depende de la diferencia de los voltajes aplicados a las dos terminales de entrada.

### 1.6.1. Amplificador de ganancia programable

Es un amplificador cuyo comportamiento puede ser modificado mediante un código digital. El parámetro cuya modificación es más interesante es la ganancia pues permite adaptar el margen dinámico de la señal al del convertidor A/D.

El uso de un amplificador de ganancia programable antes de un convertidor A/D con un determinado margen dinámico, permite obtener la misma resolución para todas las señales de entrada independientemente de su amplitud.

### 1.6.2. Amplificador de instrumentación

Muchas aplicaciones en los laboratorios e industria necesitan de la medición de señales analógicas de bajo nivel como por ejemplo las que se obtienen de los termopares, galgas extensométricas sensores biológicos. Estas señales deben amplificarse mediante dispositivos que posean la ganancia, impedancia de entrada y estabilidad adecuadas. Es común seleccionar los amplificadores de instrumentación para esas aplicaciones, porque son amplificadores diseñados específicamente para cumplir con esos requisitos además que están hechos para emplearse siempre que sea difícil la adquisición de una señal útil.

La configuración del amplificador de instrumentación es la siguiente:

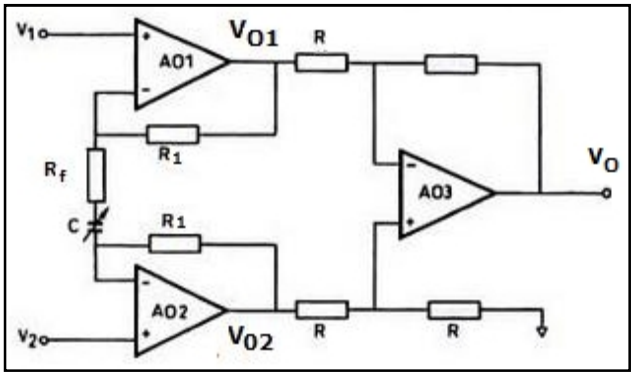



*Figura 1.13*

En donde se demuestra que AO1 y AO2 es una configuración de un seguidor de voltaje y A3 tiene una de restador o amplificador diferencial.

Los voltajes de salida V01 y V02 vienen de la configuración de dos amplificadores configurados como seguidor de voltaje, dichos voltajes entran a un amplificador diferencial en donde se hace la resta de las dos entradas y se multiplica por una ganancia que depende de las resistencias Rf y R1.

$$V_o = (V_{c2} - V_{c1})\left| {}^{R_f}/_{R_1} \right|$$

### 1.6.2.1. Seguidor de voltaje o buffer

Con esta configuración la señal de entrada es igual a la de salida en amplitud y fase por lo que la resistencia de entrada debe ser muy grande con el fin de no cargar la etapa anterior, así la señal de entrada no se verá alterada. Por lo tanto, la salida de esté amplificador se comporta como una fuente ideal de voltaje. Se puede considerar este tipo de configuración como una etapa de protección para dos circuitos.

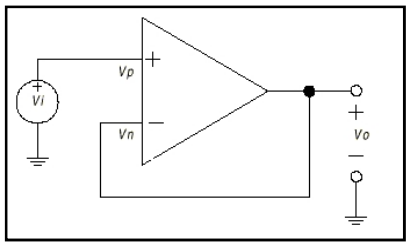



*Figura 1.14*

Para lograr estas características tomamos como referencia el amplificador de voltaje no inversor:

$$A_v = {}^{V_c}/_{V_i} = 1 + {}^{R_2}/_{R_1}$$

Para que la ganancia sea unitaria y no afecte a la salida del circuito, R2 puede ser cero o R1 tener una resistencia infinita:

$$A_v = {}^{V_c}/_{y_i} = 1$$

Por lo tanto se logra que la entrada sea igual a la salida:

$$V_o = V_i$$

### 1.6.2.2. Amplificador diferencial

Este amplificador también es conocido como restador ya que su salida $V_0$es igual a la diferencia entre el voltaje aplicado a su entrada inversora $V_1$ (-) y a la no inversora $V_2$ (+), multiplicada por una ganancia que depende de las resistencias de entrada y la resistencia de retroalimentación.

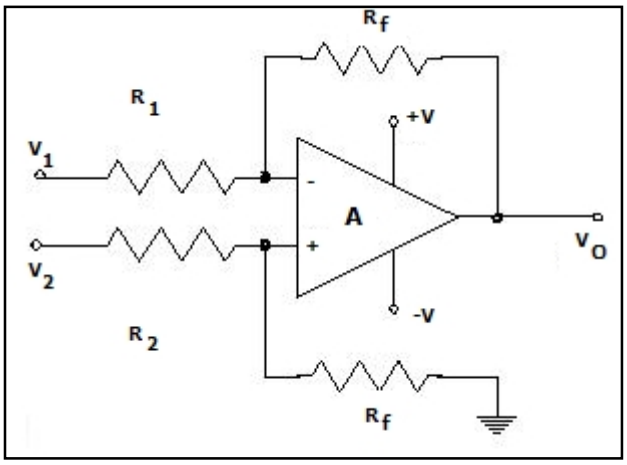



*Figura 1.15*

Por lo que el voltaje de salida:

$$V_o = (V_2 - V_1)\left[ {}^{F_f}/_{R_1} \right]$$

Se considera que este amplificador es la base del amplificador de instrumentación.

En esta sección se explicara de manera general las características del microcontrolador PIC18f4550 y de los módulos internos que lo conforman, aunque

solo se profundizará en aquellos que fueron utilizados para la elaboración del sistema de adquisición de datos.

### 1.7 Microcontrolador

Un microcontrolador es un circuito integrado programable que contiene en su interior las 3 unidades funcionales de una computadora las cuales son: un microprocesador, memoria y unidades de entrada y salida. Por lo que se considera que es una microcomputadora. En su memoria solo reside un programa destinado a gobernar una aplicación determinada; sus periféricos de entrada/salida soportan el conexionado de sensores del dispositivo a controlar. El microprocesador lee los valores de la memoria, los interpreta como ordenes y las ejecuta en el orden establecido.

#### 1.7.1. Microprocesador

El microprocesador es un circuito integrado que contiene la Unidad Central de Proceso (CPU), la cual está formada por una unidad de control y una unidad aritmético-lógica además de unos registros internos. El microprocesador se le considera un sistema abierto porque su configuración es variable de acuerdo con la aplicación a la que se destine, por lo que un μP es simplemente un componente que forma parte de un microcontrolador.

#### 1.7.2. Memoria de programa

El microcontrolador necesita un programa para su funcionamiento. Este programa se almacena en una memoria de valores fijos (ROM, EEPROM o Flash). Como puede ser el caso de una memoria ROM (memoria de solo lectura)de una computadora, la cual no puede ser modificada por el usuario y posee la información necesaria para el arranque de la PC. Los microcontroladores poseen memorias de tipo EEPROM (memoria ROM eléctricamente programable) en la cual puede programarse o borrarse eléctricamente, por consiguiente esta memoria es aplicable como memoria no volátil de lectura y escritura. Actualmente los μC poseen una memoria Flash, que también puede ser programada eléctricamente, estas suelen ser de bajo consumo y disponer de mayor capacidad a diferencia de las EEPROM. Por razón de ventajas está sustituyendo a la EEPROM convencional.

### 1.7.3. Memoria de datos

Se necesita una memoria que de escritura/lectura para almacenar datos que varían continuamente como lo son valores de cálculo y de señales por lo que la memoria RAM (memoria de acceso aleatorio) es la adecuada, aunque sea volátil.

Los nuevos microcontroladores contienen memorias tipo SRAM (RAM estática) la cual almacena de forma temporal variables utilizadas en el programa. Al desconectar la unidad de control, esta memoria perderá todos los datos almacenados por eso se le llama memoria volátil.

### 1.7.4. Unidades de entrada y salida.

Los microcontroladores poseen pines de entrada/salida para la comunicación con el exterior, como lo puede ser sensores, actuadores y otros dispositivos electrónicos.

Las líneas de entrada y salida se agrupan en conjuntos llamados Puertos. Estos puertos están implementados como direcciones de memoria, de forma que escribir en ellos es como enviar datos por los correspondientes pines del μC y leer esas direcciones es como leer el estado de los pines del chip.

En las aplicaciones de control y comunicaciones, en las que se utilizan los microcontroladores, se deben ejecutar gran cantidad de tareas de entrada/salida lo que hace necesario que dispongan de un número considerable de terminales dedicados a dichas tareas.

### 1.7.5. Arquitectura Harvard y Von Neumann

En la memoria de un ordenador, un microcontrolador o un microcomputador, s almacenan instrucciones y datos. Las instrucciones deben pasar secuencialmente a la CPU para su decodificación y ejecución, en tanto que algunos datos en memoria son leídos por la CPU y otros son escritos en la memoria desde la CPU. Las arquitecturas Von Neumann y Harvard son modelos generalas del hardware de los ordenadores que representan dos soluciones diferentes al problema de la conexión de la CPU con la memoria y a la organización de la memoria como almacén de instrucciones y datos.

*Arquitectura Von Neumann:* Esta arquitectura toma el nombre del matemático John Von Neumann que propuso la idea de un ordenador con el programa almacenado (stored-program computer). Neumann trabajo en el equipo de diseñadores de la computadora ENIAC (*Electronic Numerical Integrator and Calculator*) diseñada en la universidad de Pennsylvania durante la Segunda Guerra Mundial.

La arquitectura Von Neumann utiliza una memoria única para instrucciones y datos (figura 1.16). Esto significa que con un mismo bus de direcciones se localizan (direccionan) instrucciones y datos y que por un único bus de datos transitan tanto instrucciones como datos. La misma señal de control que emite la CPU para leer un dato, sirve para leer una instrucción. No hay señales de control diferentes para datos e instrucciones. Debe quedar claro que aunque se use una memoria ROM para almacenar el programa y RAM para los datos, para la CPU no hay tal distinción, sino que ROM y RAM forman un conjunto único (una memoria de lectura y escritura) para el cual la CPU emite señales de control, de dirección y datos. En la figura 1.16 se representa la arquitectura Von Neumann.

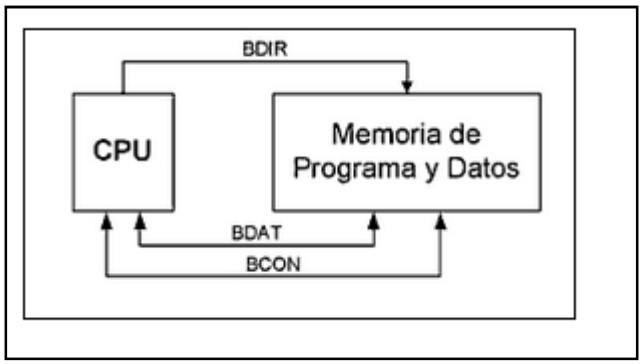



*Figura 1.16*

*Arquitectura Harvard:* El termino arquitectura Harvard se debe al nombre del lugar donde Howard Aiken diseño los ordenadores Mark I, II, III y IV. Estos ordenadores fueron los primeros en utilizar memorias separadas para instrucciones y datos, lo que fue una concepción diferente al ordenador de programa almacenado.

En la figura 1.17 se muestra la arquitectura tipo Harvard la cual utiliza memorias separadas para instrucciones y datos. En este caso la memoria de programa (que almacena las instrucciones) tiene su propio bus de direcciones, su propio bus de datos y su bus de control. La memoria de datos tiene sus propios buses de direcciones, datos y control, independientes de los buses de la memoria de programa. Como ya se había mencionado anteriormente la memoria de datos es de solo lectura, mientras que en la de datos se puede leer y escribir.

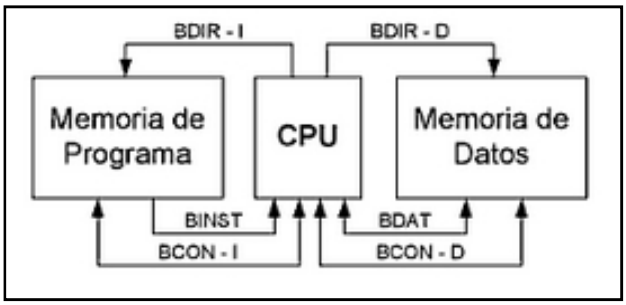



*Figura 1.17*

### 1.7.6. Procesador CISC y RISC

La arquitectura CISC se caracteriza por disponer de un grupo muy amplio de instrucciones muy complejas y potentes. Es más antigua que la arquitectura RISC y por tanto su diseño está marcado por la tecnología existente en los años sesenta.

*Procesador CISC:* En la arquitectura CISC (computador de conjunto de instrucciones complejas) se cuenta con un procesador con instrucciones maquina muy largas y sofisticadas para poder reducir el número de accesos que realizan a memoria.

Son muchos los procesadores CISC existentes hoy en día, y aún de hecho constituyen la variedad de arquitectura más extendida en el mercado. Valga citar las familias de Intel 80x86 y la de Motorola 680X0 como los ejemplos más claros de esta afirmación.

*Procesador RISC:* En la arquitectura RISC (computador de conjunto de instrucciones reducido), el microprocesador dispone de un repertorio corto de instrucciones sencillas. Cada instrucción realiza operaciones muy simples, como mover un dato entre la CPU y la memoria, pero a alta velocidad. La complejidad del microprocesador disminuye, por lo que es fácil aumentar la frecuencia de oscilación y con ello la velocidad de las instrucciones. La mayoría de las instrucciones tienen la misma longitud. Los microcontroladores PIC son un ejemplo de dispositivos con arquitectura RISC.

## 1.8 Comunicaciones seriales en los microcontroladores

Los sistemas serie, en comparación con los paralelos, tienen las siguientes características: transmisión a mayor distancia, menor costo y más sencillos en canto a hardware necesario. Suelen ser comunicaciones punto a punto.

El procesador de comunicaciones utilizado depende de la norma elegida por el fabricante del microcontrolador, el cual suministra información detallada de la forma de configurarlo para llevar a cabo la transmisión de información a distancia.

### 1.8.1. Estándar RS232

RS232 significa Estándar Recomendado 232, definido así por el ANSI (American National Standard Institution) como “la interface entre un equipo terminal de datos y un equipo de comunicación de datos utilizando un intercambio binario en modo serie”.

Los niveles de voltaje que maneja la interface RS232 están definidos para extenderse entre -3V y -15V para el 1 lógico, y entre +3V y +15V para el cero lógico, con lo que se puede observar que esta interface utiliza la lógica negativa. Típicamente se establece en -12V y +12V.

Las características más importantes del estándar son:

- Velocidad máxima original era 20 Kbps, hay aplicaciones que llegan a 116 kbps.
- Longitud máxima del cable de 15m
- Un emisor y un receptor
- Transmisión síncrona o asíncrona. (usando líneas extra para el reloj)

En la transmisión síncrona se permite la comunicación continua de datos y no existe un límite de tamaño. Trabaja en modo semi-duplex en el cual la comunicación serie se establece a través de una única línea, en ambos sentidos, pero no se puede transferir información en ambos sentidos de forma simultánea

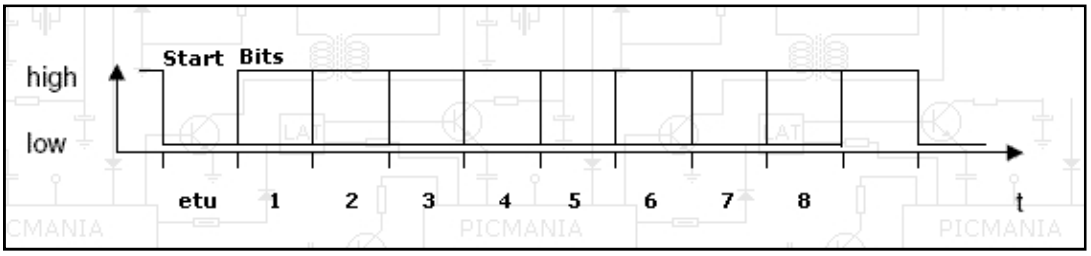



*Figura 1.18. Transmisión síncrona*

En la transmisión asíncrona se emplean relojes tanto en el emisor como en el receptor. Ambos relojes deben ser de igual frecuencia y deben estar sincronizados. Cada trama de datos tiene un tamaño fijo y poseen un bit inicial (bit de start) y un bit de parata (stop) que permiten realizar la sincronización. Esta

transmisión se realiza en modo full-dúplex en el que cual se deben utilizar dos líneas, una de transmisión y otra de recepción de datos, con lo que se puede transferir y recibir información de forma simultánea.

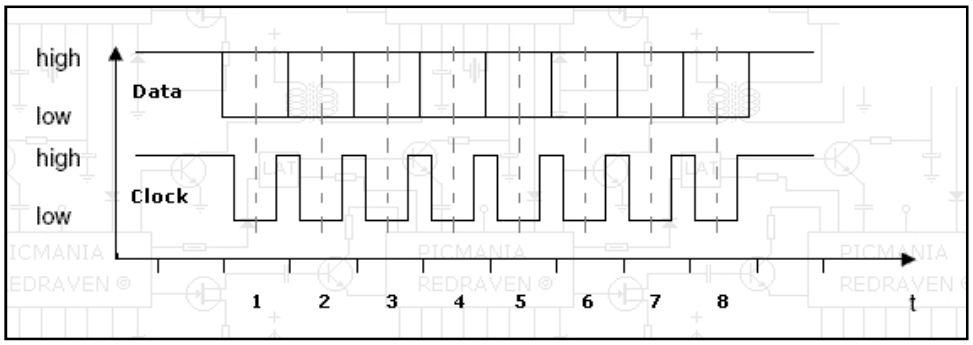



*Figura 1.19 Transmisión asíncrona*

### 1.8.2. Señales RS232

Un dispositivo llamado DTE (equipo terminal de datos) es un equipo que genera o recibe datos, mientras que un dispositivo llamado DCE (equipo de comunicación de datos) es quien transporta los datos. El DTE controla las señales TD, RTS y DTR; y el DCE controla las señales RD, CTS, DCD, DSR, y RI. La referencia de todas las señales es un pin común (GND).

| Señal | Sentido | Descripción |
|---|---|---|
| TD (Transmit Data) | DTE→DCE | Datos a transmitir |
| RD (Received Data) | DTE←DCE | Datos recibidos |
| RTS (Request To Send) | DTE→DCE | El terminal desea transmitir |
| CTS (Clear to Send) | DTE←DCE | El terminal puede transmitir |
| DTR (Data Terminal Ready) | DTE→DCE | El terminal está operacional |
| DSR (Data Set Ready) | DTE←DCE | El modem está operacional |
| DCD (Data Carrier Detect) | DTE←DCE | El modem recibe portadora del modem remoto |
| RI (Ring Indicator) | DTE←DCE | Se recibe un llamado por la línea telefónica |

## 1.9 Estándar USB

El USB (bus serie universal) se invento para conectar dispositivos de entrada/salida como el teclado y el ratón a la computadora. Este bus utiliza un conector pequeño de cuatro alambres, dos de los cuales suministran energía eléctrica a los dispositivos USB. El USB es un bus centralizado en el que un dispositivo raíz consulta a los dispositivos de E/S cada milisegundo para ver si tiene tráfico. Todos los dispositivos USB comparten un solo controlador de dispositivo USB, lo que hace innecesario instalar un nuevo controlador para cada dispositivo USB nuevo. Por ello, es posible añadir dispositivos USB a la computadora sin tener que reiniciarla.

No hace muchos años apareció en el mercado el estándar USB para comunicaciones serie y actualmente la mayoría de las computadoras lo incorporan, llegando a sustituir incluso a los famosos conectores serie.

La estructura de este bus ha sido definida por un consorcio de líderes de la industria para permitir la conexión de múltiples periféricos de ordenador a baja y media velocidad, entre los que se incluyen módems, ratones, impresoras, escáner, teclados, joysticks, etc.

El USB es un nuevo bus externo estándar que soporta la transferencia de datos a velocidades de 12 Mbit/s en el USB 1.1 (1.5Mbit/s para dispositivos que requieren poca velocidad) como joystick y ratones) hasta 450 Mbit/s en el USB 2.0. Con este bus los usuarios no se tienen que preocupar acerca de que puerto elegir, no tienen que instalar ninguna tarjeta de expansión ni configurar puentes físicos. Otra ventaja importante es que este bus proporciona la alimentación a los periféricos hasta una distancia de 5 metros.

A principios de 1996 unos pocos fabricantes de ordenadores empezaron a incluirlo en sus productos y desde 1997 el USB se ha extendido considerablemente llegando a remplazar a los puertos serie y paralelo. En la actualidad ya son más de 400 fabricantes los que lo han adoptado.

### 1.9.1. Conexión del USB

La conexión de un dispositivo USB es muy sencilla, ya que solo basta con insertar el conector rectangular USB en el puerto de la computadora. Además de conectar dispositivos al ordenador, el USB también permite aumentar el número de puertos RS-232 o paralelo. Hay convertidores USB-serie y USB-paralelo con la electrónica incorporada en los propios conectores.

### 1.9.2. Componentes del USB

El sistema de bus serie universal USB consta de cuatro componentes:

- **Controlador**: Se encuentra dentro de la PC y es responsable de las comunicaciones entre los periféricos USB y el procesador del PC, y de la admisión de los periféricos dentro del bus, tanto si se detecta una conexión como una desconexión. Para cada periférico, el controlador determina su tipo y le asigna una dirección lógica. Si se producen errores durante la conexión, el controlador lo comunica al procesador que a su vez lo transmite al usuario. Una vez que se ha producido la conexión correctamente, el controlador asigna al periférico los recursos del sistema que este precise para su funcionamiento.
- **Concentradores o hubs**: Son distribuidores inteligentes de datos y alimentación que hacen posible la conexión a un único puerto USB de 127 dispositivos. De una forma selectiva reparten datos y alimentación entre ellos. Además del controlador, el PC también contiene el concentrador raíz. Este es el primer concentrador de toda la cadena que permite a los datos y a la energía pasar uno o dos conectores USB del PC, y de allí a los 127 periféricos que como máximo puede soportar el sistema mediante concentraciones adicionales.
- **Periféricos**: El USB soporta periféricos de baja y media velocidad, empleando dos velocidades (1,5 y 12 Mbps) para la transmisión d datos según las necesidades de los periféricos.
- **Cables y conectores:** USB transfiere señales y energía a los periféricos utilizando un cable de 4 hilos y una longitud máxima de 5 metros.

## 1.10. Interfaz de bus SPI (Serial Peripheral Interface)

El SPI es un protocolo de comunicación desarrollado por Motorola y posteriormente adoptado por otras industrias. El SPI también suele ser llamado bus serial a “cuatro hilos”.

El bus SPI es una interfaz serial de comunicaciones a 4 hilos utilizada por muchos microcontroladores/microprocesadores que habilita y controla dispositivos periféricos para comunicarse unos a otros.

El bus SPI, el cual opera en modo *full dúplex,* (las señales que envían datos pueden ir en ambas direcciones simultáneamente), es un tipo síncrono de enlace con una interfaz Maestro-Esclavo y puede soportar hasta 1 megabaud o 10 Mbps de velocidad. Ambos protocolos, maestro único y multimaestro son posibles con

SPI. Pero el bus multimaestro es raramente utilizado y limitado a un dispositivo simple esclavo.

El bus SPI es usualmente utilizado solo en el PCB (PrintedCircuitBoard). Hay muchas razones que nos impiden el uso fuera del área del PCB. El bus SPI fue diseñado para transferir datos entre varios circuitos integrados a muy altas velocidades. Debido a estas altas velocidades, las líneas del bus no pueden ser muy largas, porque la reactancia se incrementa demasiado y el bus llega a ser inservible.

Los periféricos pueden ser relojes en tiempo real, convertidores ADC/DAC, módulos de memoria como EEPROM, FLASH, sensores, LCD, UART, USB control entre otros.

### 1.10.1. Líneas de control y datos en SPI y las conexiones básicas.

El protocolo SPI especifica 4 hilos de señal:

1.- Master Out Salve In (MOSI) – La señal MOSI es generada por el dispositivo Maestro y recibida por el dispositivo esclavo,

2.- Master In Slave Out (MISO) – La señal MISO es generada por el dispositivo esclavo y recibida por el dispositivo maestro.

3. - Serial Clock (SCLK o SCK) - La señal SCLK es generada por el dispositivo maestro para la sincronización de los datos.

4.- Slave Select (SS) del maestro a Chip Select (CS) del esclavo – La señal SS es generada por el dispositivo maestro para seleccionar el esclavo individual. Las señales SS y CS con activadas con un valor bajo (binario).

Hay muchas maneras de nombrar a las señales antes mencionadas, por ejemplo SDI (Serial Data In) se puede usar en lugar de MOSI y SDO (Serial Dara Out) puede ser reemplazada por MISO.

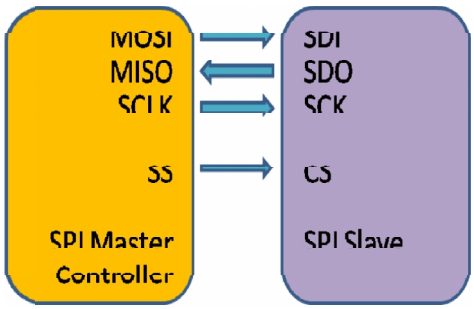



*Figura 1.20 Implementación bus SPI*

De estas cuatro señales lógicas, dos de ellas, MOSI y MISO, pueden ser agrupadas en líneas de daros y las otras dos, SS y SCLK, como líneas de control.

En el bus SPI solo puede haber un dispositivo maestro con múltiples esclavos. En el protocolo maestro único, usualmente un dispositivo SPI actúa como el maestro y controla el flujo de datos mediante la generación de la señal de reloj (SCLK) y activa el esclavo con el que se quiera comunicar con la señal *select-slave*(SS), entonces recibe o trasmite los datos vía las dos líneas de datos. Un maestro, normalmente provee la señal de reloj a todos los dispositivos conectados en el bus.

El uso de cada una de estas 4 líneas depende de cada dispositivo. Por ejemplo, el pin SDI puede no estar presente si el dispositivo no requiere una entrada, o el pin SDO puede no estar presente si el dispositivo no requiere una salida. Si un microcontrolador solo necesita comunicarse con un dispositivo SPI, entonces el pin CS puede ser conectado directamente a tierra (GND). Con múltiples dispositivos esclavos, es necesaria una señal independiente SS del dispositivo maestro para cada uno de los esclavos.

### 1.10.2. La comunicación entre dispositivos.

La comunicación es iniciada por el maestro todo el tiempo. El maestro primero configura la señal de reloj, usando una frecuencia, la cual es menos o igual a la máxima frecuencia que el esclavo soporta. El maestro entonces selecciona el esclavo deseado para la comunicación colocando en la línea *chip-select* (SS) un nivel lógico bajo. Si se requiere de un periodo de espera, el maestreo deberá esperar por lo menos durante ese periodo de tiempo antes de comenzar a emitir ciclos de reloj.


*Figura 1.21*

Los dispositivos esclavos conectados al bus SPI que no han sido activados por el maestro utilizando la señal *select-slave* pasaran por alto la señal de reloj y la señal MOSI generadas por el maestro y no debe contemplar la señal MISO. Esto significa que el maestro solo puede seleccionar un esclavo a la vez.

Muchos de los dispositivos (periféricos) tienen salidas de tres estados, la cual pasa por el estado de alta impedancia (desconectado) cuando el dispositivo no es seleccionado.

Una transmisión de datos *full dúplex* puede ocurrir durante cada ciclo de reloj. Esto significa que el maestro envía un bit en el pin MOSI; el esclavo lo lee y envía un bit en el pin MISO

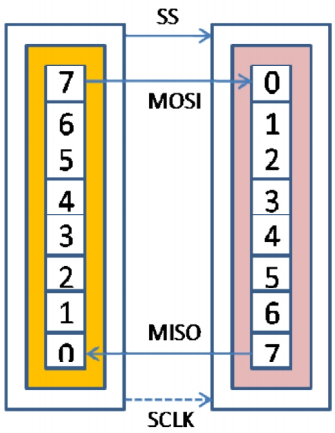



*Figura 1.22 Configuración para la comunicación utilizando dos registros de desplazamiento.*

Los datos transferidos son organizados por medio del uso de registros de desplazamiento con un ancho de palabra que puede ser de 8 bits en maestro como en esclavo. Están conectados en anillo. Mientras el valor de los registros del maestro se desplaza a través de la línea MOSI, el esclavo desplaza los datos dentro de su registro de desplazamiento.

Los datos sin comúnmente desplazados con el MSB (More Significant Bit) primero, mientras se va desplazando un nuevo LSB (LessSignificant Bit) dentro del mismo registro. Después de que el valor del registro ha sido desplazado el maestro y el esclavo han intercambiado sus valores de registro. Entonces cada dispositivo toma ese valor y hace la operación necesaria con ese valor (por ejemplo escribir el valor en la memoria). Si existen más datos por ser intercambiados, los registros de desplazamiento son actualizados con los nuevos datos y el proceso es repetido. Cuando no hay más datos para ser transmitidos, el maestro para su reloj.

Existe un “modo de flujo continuo de bytes” con el bus SPI. En este modo el maestro puede desplazar bytes continuamente. En este caso el selector de esclavo (SS) se mantiene en valor lógico bajo hasta el proceso de flujo haya terminado.

# CAPÍTULO 2

## Diseño del Hardware de sistema de adquisición

Este capítulo se divide en dos partes, en la primera se hará referencia a los elementos físicos que componen la tarjeta de adquisición de datos, es decir al hardware y en la segunda parte al funcionamiento que se tiene entre los dispositivos, por lo que se describirá el desarrollo del código de programación para el microcontrolador. . El diagrama a bloques de la tarjeta de adquisición se muestra en la figura 2.1, cabe mencionar que para simplificar el entendimiento solo se representa el acondicionamiento y procesamiento de un solo canal.

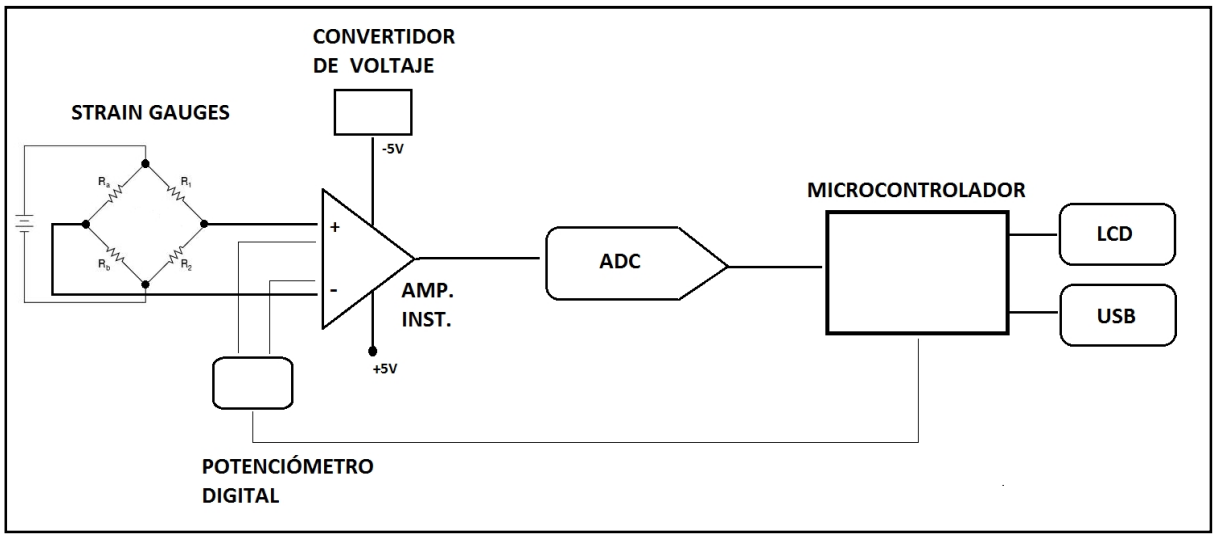



*Figura 2.1*

El dispositivo que controla el funcionamiento de todo el circuito es el microcontrolador, el cual ejecuta un código que tiene previamente programado. Debido a dicho código el microcontrolador permite la interacción de los componentes externos a él.

Para el acondicionamiento de la señales se utilizo un amplificador de instrumentación el cual permite amplificar la señal de entrada para que posteriormente esta pueda ser interpretada por un convertidor analógico digital.

La ganancia para el amplificador de instrumentación se establece con un potenciómetro digital, el cual varía su resistencia de acuerdo a un código que le envía el microcontrolador por medio de la comunicación SPI.

El amplificador de instrumentación necesita estar alimentado por una fuente de voltaje positiva y una negativa. El voltaje positivo se toma del puerto USB (+5V) de una computadora y el voltaje negativo se obtiene con un convertidor de voltaje, con el cual podemos transformar tensiones de +5V a -5V. Para la alimentación de las galgas extensométricas se utiliza un regulador de voltaje de 3.3V.

La conversión de la señal la traducimos con un Convertidor Analógico Digital programable el cual está diseñado para aplicaciones de instrumentación y medición ya que tiene una alta resolución y velocidad para la adquisición de datos, ideal para la lectura de microdeformaciones.

## 2.1 Alimentación del sistema de adquisición de datos.

El sistema digital puede ser alimentado de dos formas: una es con el voltaje del puerto USB para cuando se requiera visualizar y almacenar datos en una computadora y la otra con un voltaje externo para cuando se requiera visualizar los datos en un display LCD.

La selección entre el voltaje otorgado por el puerto USB y un voltaje externo se realiza por medio de un switch (A) de un polo y dos tiros llamado así porque la palanca de este interruptor hace contacto con dos bornes fijos que para este sistema un borne está conectado al pin uno de un conector USB tipo B y el otro borne va a un conector en donde se espera una alimentación de 5V, como se puede mostrar en la figura 2.2.

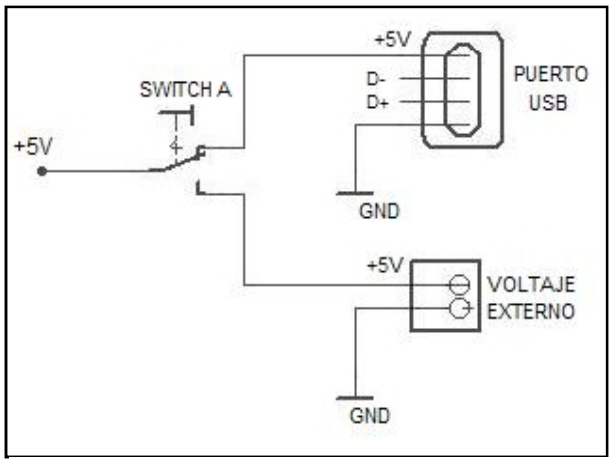



*Figura 2.2*

Con la configuración mostrada en la figura 2.2 toda la tarjeta queda alimentada con un voltaje de 5V. Es importante mencionar que cuando el sistema de datos esta en comunicación con la computadora este no muestra datos en la LCD por lo que el display necesita estar apagado, de lo contrario estará consumiendo un grado importante de corriente (puede llegar a consumir hasta 150mA). Para mantener apagada la LCD se puso a tierra su terminal $V_{cc,}$ esto se realizo por medio de un interruptor (switch B) que permite seleccionar si la LCD queda alimentada con un voltaje externo o queda totalmente inhabilitada colocando su terminal $V_{cc}$ a tierra como se muestra en la figura 2.3.

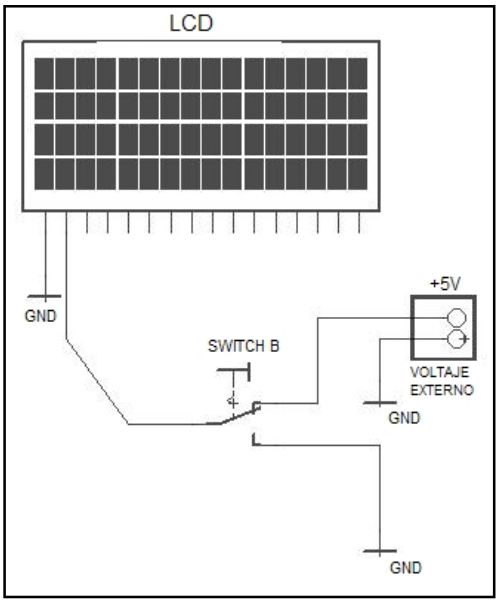



*Figura 2.3*

En la figura anterior se muestra que el display no está conectado al voltaje proveniente del puerto USB sino a un voltaje externo, sin embargo sino se hace esta configuración cuando se haga la comunicación de datos hacia una computadora, la pantalla de la LCD queda encendida y por lo tanto estaría consumiendo corriente sin ser utilizada.

Algunos componentes necesitan un voltaje diferente a los +5V para operar y es el caso del amplificador de instrumentación AD620, el cual debe ser alimentado con +5V y -5V. El voltaje negativo se obtiene del dispositivo TC7660 el cual es un con convertidor de voltaje que transforma una entrada de +5V a -5V con tan solo dos capacitores de 10μf puestos en paralelo entre sus terminales $CAP^+$ y $CAP^-$y otro en paralelo con su terminal de salida $V_{out}$ y tierra. Para este circuito se ocuparon dos convertidores TC7660, cada uno para proveer del voltaje negativo a dos AD620, esto se realizo para mantener el voltaje estable en los amplificadores. La conexión del TC7660 se muestra en la figura 2.4.

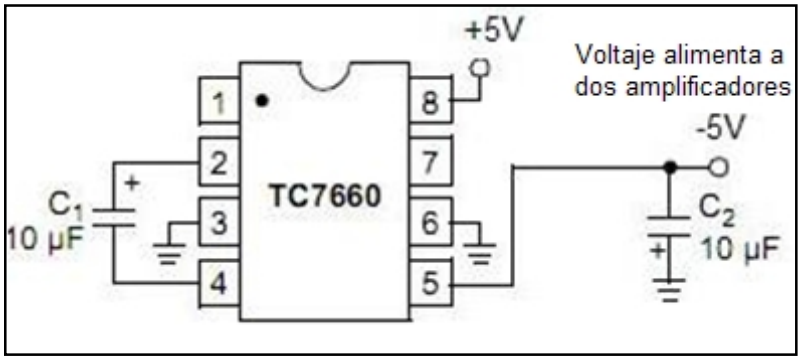



*Figura 2.4*

El TC7660 es utilizado para aplicaciones de instrumentación y en sistemas de adquisición de datos, lo cual es ideal para este circuito.

Para la alimentación de las galgas extensométricas se utilizo un voltaje de 3.3V proporcionado por el regulador de voltaje LF33CV. Este dispositivo mantiene un voltaje fijo a su salida de 3.3V y solo basta con alimentarlo a +5V y colocar un capacitor de 100nf en paralelo con respecto a su entrada y a tierra y otro de 100nf entre su terminal de salida y tierra.

Ya que la tarjeta de adquisición es de cuatro canales, en algún momento el usuario puede llegar a alimentar hasta cuatro arreglos de galgas extensométricas. De acuerdo a las pruebas realizadas previas al diseño final del circuito, se pudo observar que al estar conectados los 4 arreglos de strain gauges y alimentadas

por un solo regulador de 3.3V este disminuía su voltaje hasta 2.7V aproximadamente por lo que se decidió colocar dos LF33CV, cada uno para alimentar dos arreglos de galgas extensométricas como se muestra en la figura 2.5.

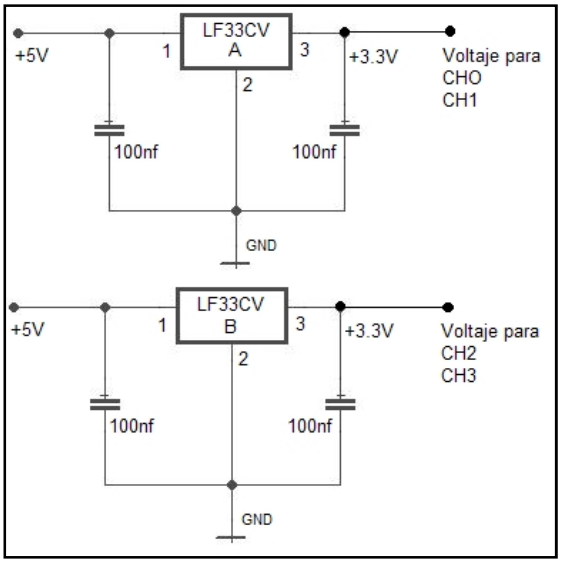



*Figura 2.5*

## 2.2 Etapa de control.

En esta sección se explicara de manera general las características del microcontrolador PIC18F4550 y de los módulos internos que lo conforman, aunque solo se profundizará en aquellos que fueron utilizados para la elaboración del sistema de adquisición de datos.

### 2.2.1. El microcontrolador PIC18F4550

El microcontrolador utilizado para el diseño del circuito es un PIC18F4550 fabricado por la empresa Microchip Techology. Se eligió este chip ya que contiene la cantidad de puertos necesarios de entrada y salida para la comunicación con los demás dispositivos que conforman la tarjeta de adquisición, además de que físicamente es utilizado para enviar y recibir datos por medio del puerto USB de una computadora, por lo se evita el uso de convertidores USB a UART, reduciendo así el costo total del circuito.

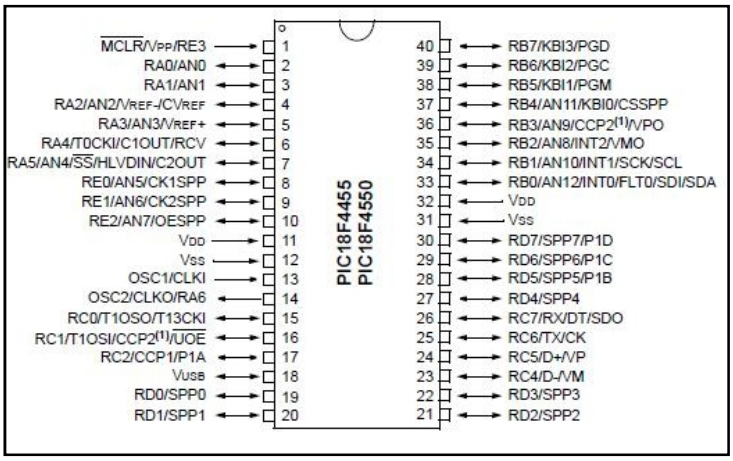



*Figura 2.6 PIC18F4550*

Para la correcta operación el microcontrolador se le alimento con un voltaje de 5V en el pin VDD, este voltaje es la máxima tensión a la que puede operar este circuito integrado y la terminal VSS se conecto a tierra. El microcontrolador puede generar una señal de reset por software si se le aplica un cero lógico en el pin MCLR (Master Clear). Esta terminal está negada por lo cual se conecto directo a 5V como se muestra en la figura 2.5.

En esta sección se explicara de manera general las características del microcontrolador PIC18F4550 y de los módulos internos que lo conforman, aunque solo se profundizará en aquellos que fueron utilizados para la elaboración del sistema de adquisición de datos.

### 2.2.2. Oscilador del microcontrolador

El microcontrolador necesita un circuito que le indique la velocidad a la que este debe trabajar. El circuito es conocido como oscilador de frecuencia. Este dispositivo crea una señal de reloj (señal cuadrada o tren de pulsos) la cual también es utilizada por el microcontrolador para sincronizar su funcionamiento.

El PIC ejecuta una instrucción en un ciclo de máquina (4 periodos de reloj).

En las terminales OSC1 y OSC2 se conecto en paralelo un cristal de 20Mhz y 2 capacitores de 15pf para cada terminal con referencia a tierra como se muestra en la figura 2.7. De acuerdo a la hoja de datos del PIC18F4550 se obtuvieron los valores de los capacitores.

El microcontrolador dispone de un circuito llamado PLL (Circuito de Fase Enlazada). Con la incorporación del PLL en el oscilador, el PIC permite programar una frecuencia mayor a la frecuencia de entrada a través de un preescalador.

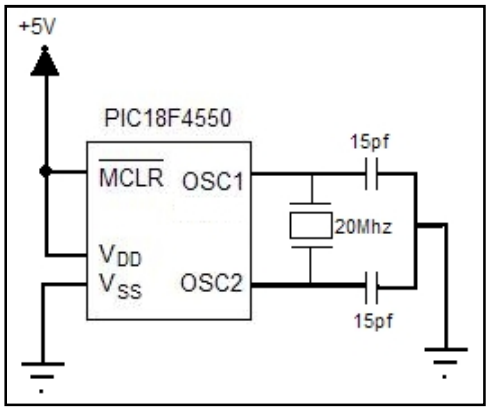



*Figura 2.7*

El PIC18F4550 acepta una amplia variedad de osciladores, como por ejemplo: cristales (XT y HS) y osciladores externos (EC) entre otros; así como una combinación de estos con el modulo PLL presente en el microcontrolador. Son 12 los modos de oscilador que acepta el microcontrolador entre los que más destacan son:

- XT. Cristal Resonador
- XTPLL. Cristal con PLL activado
- HS. Cristal de alta velocidad
- HSPLL. Cristal de alta velocidad con PLL activado
- EC. Reloj externo
- ECPLL. Reloj con reloj activado

El microcontrolador se programó de tal manera que funcionará a la máxima velocidad a la que puede operar que son 48Mhz. Del manual de usuario del PIC18F4550, en la tabla de "Opciones de configuración del oscilador para operaciones con USB" se determinó que el cristal a usar por el PIC debe ser de 20Mhz, con un preescalador PLL de 5 y un modo de oscilador tipo HSPLL.Entonces la frecuencia de oscilación ($F_{osc}$) a la que a la que trabaja el microcontrolador es de 48Mhz. El tiempo en el que se ejecuta un ciclo de reloj se determina:

$$T = {}^{1}/_{F_{OSC}} \; ; \; T_{REJ} = {}^{1}/_{48\,Mhz} = 20ns$$

Como ya se había explicado anteriormente el microcontrolador ejecuta la mayoría de sus instrucciones en un ciclo maquina que equivalen a 4 ciclos de reloj.

$$T_{inst} = T \times 4\ ;\ T_{inst} = 20ns \times 4\ ;\ T_{inst} = 8.3ns$$

Por lo tanto la mayoría de las instrucciones del microcontrolador se ejecutan en 83.3ns salvo las instrucciones de salto que se ejecutan en 2 ciclos maquina que para esta aplicación serán 166.6ns.

### 2.2.3. Características generales

El microcontrolador PIC18F4550 tiene una arquitectura Harvard con 16 bits de bus de programa y 8 bits de bus de datos. Una arquitectura Harvard dispone de dos memorias diferentes, la memoria de datos y la memoria de programa, cada memoria dispone de un bus para su acceso y son direccionadas por un microprocesador (figura 2.8).

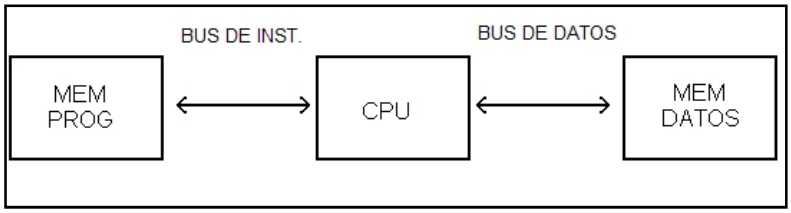



*Figura 2.8*

Dentro de las características principales del PIC18F4550 se encuentran:

- Una memoria de programa de 32 Kbytes de tipo Flash
- Una memoria RAM de datos de 2048 Bytes tipo SRAM
- Una memoria EEPROM de datos de 256 bytes
- 35 pines de entrada/salida repartidos en los puertos A, B, C, D y E
- Frecuencia de oscilación de hasta 48Mhz
- 20 interrupciones por software
- 3 fuentes de interrupción externa
- 4 temporizadores (2 de 8 bits y 2 de 16 bits)
- Dos módulos de comparación/captura/PWM
- Un canal de comunicación serial mejorado (EUSART)
- Un canal USB
- 13 canales para la conversión analógica digital de 10 bits
- 2 comparadores analógicos
- Un Puerto Serie Síncrono Maestro para comunicaciones SPI e $I^2C$
- Un Puerto Paralelo de Transmisión de Datos (SSP)
- 31 registros de 21 bits en la Pila

- Voltaje de operación de 2.0V a 5.5V
- Encapsulados (PDIP de 40 pines, QFN de 40 pines y TQFF de 40 pines)

En los pines RB7, RB6, RB5 y RB4 están conectados cuatro *push button* que permiten seleccionar el número de canales que el usuario requiere. Estos botones están conectados en serie con cuatro resistencias en configuración *pull-up.* Esta configuración por default le entrega al microcontrolador un 1 lógico y cuando se presiona el *push button* cambia a cero lógico. En la figura 2.9 se representa solo una conexión *pull-up*. En el pin RB3 también está conectado un botón con la misma configuración mencionada y este se ocupa para reiniciar al microcontrolador.

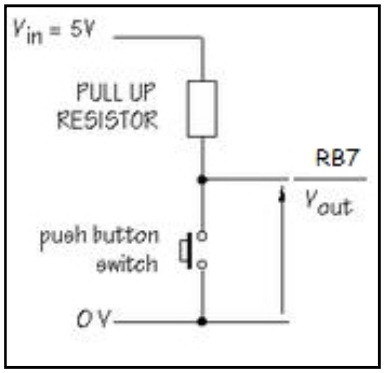



*Figura 2.9*

Estas son las características principales del microcontrolador, pero como ya se había mencionado no todos los módulos se ocupan por lo que a continuación se explicara de manera muy especifica el funcionamiento del modulo SPI y USB del PIC18F4550 y su conexión con la LCD.

### 2.2.4. Módulo SPI del PIC18F4550

El modo SPI del PIC permite utilizar 8 bits de datos para ser transmitidos y recibidos forma simultánea en comunicación síncrona. Los cuatro modos de transmisión de datos de la SPI son compatibles para este microcontrolador.

La SPI se ocupa para realizar la comunicación entre el microcontrolador y 2 convertidores analógicos digitales (MCP3901) y entre el microcontrolador y 4 potenciómetros digitales (MCP41010). Para los dos ADC se utilizaron los pines RA0, RA1, RB0, RB1 y RC7 del microcontrolador y para la comunicación con los potenciómetros se ocuparon los pines RA2, RA3, RA4, RA5, RE0 y RE1. Para

ambos casos el microcontrolador es el dispositivo maestro y los esclavos son los ADC y los potenciómetros.

### 2.2.5. Comunicación con el convertidor analógico digital

Para el caso de la transmisión y recepción de datos con los MCP3901, 3 de los 5 pines utilizados del microcontrolador son especiales para establecer la comunicación SPI.

- Serial Data Out (SDO).RC7/RX/DT/SDO
- Serial Data In (SDI). RBO/AN12/INT0/FLT0/SDI/SDA
- Serial Clock (SCK). RB1/AN10/INT1/SCK/SCL

Por lo cual internamente los pines SDI y SDO están conectados al registro interno SSPSR del PIC. En la figura 2.10 se representa la transmisión de datos entre dos microcontroladores por medio de la SPI.

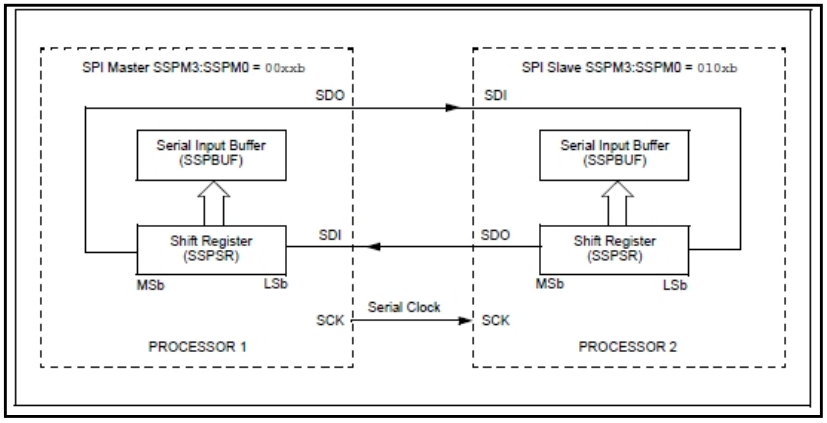



*Figura 2.10*

Cuando se requiere transmitir un dato (8 bits), el byte a enviar debe escribirse en el registro SSPBUF y automáticamente el contenido de ese registro se traspasa al registro SSPSR cuya función es ir desplazando bit a bit el contenido de SSPBUF transmitiendo los datos por el pin SDI ordenadamente al ritmo de la señal de reloj. En la recepción de datos, los bits entran uno por uno a través de la terminal SDO y se agrupan en el registro SSPSR hasta completar 1 byte, esta información pasa inmediatamente al registro SSPBUF y en ese momento se puede leer su contenido.

Para seleccionar con que dispositivo se va comunicar el microcontrolador se utilizaron los pines RA0 y RA1 para el ADC A y el ADC B respectivamente. Por lo

tanto las terminales RA0 y RA1 se ocupan como líneas selectoras de los dispositivos esclavos (Select Slave).

- Select Slave 0 (SS0). RA0/AN0
- Select Slave 1 (SS1). RA1/AN1

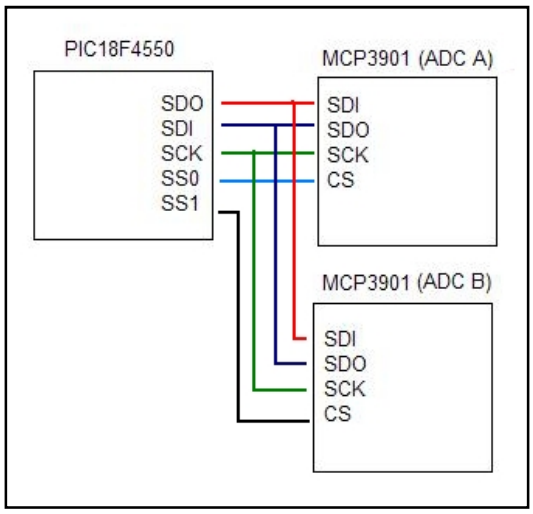



*Figura 2.11 Conexión entre el microcontrolador y los convertidores A/D*

### 2.2.6. Comunicación entre el microcontrolador y los potenciómetros digitales

Las terminales utilizadas del microcontrolador para su comunicación por medio de la SPI con los 4 potenciómetros digitales son las siguientes:

- Serial Data Out (SDOP). RE0/AN5/CK1SSP
- Serial Clock (SCKP). RE1/AN6/CK2SSP
- Select Slave 0 (SSP0). RA2/AN2/VREF-/CVREF
- Select Slave 1 (SSP1). RA3/AN3//VREF+
- Select Slave 2 (SSP2). RA4/T0CKI/C1OUT/RCV
- Select Slave 3 (SSP3). RA5/AN4/SS/HLVDIN/C2OUT

A los pines SDOP y SCKP se les agrego la letra “P” haciendo referencia a los potenciómetros. La terminal SCKP es la que genera la señal de reloj para la sincronización entre el microcontrolador y los cuatro potenciómetros, la terminal SDOP es la que envía desde el PIC el código a los MCP41010 para que estos modifiquen su resistencia, los pines SSP0, SSP1, SSP2 y SSP3 sirven para seleccionar al potenciómetro que se requiere configurar y a cada terminal además

de colocarle la letra “P” también se les designo un número diferente a cada uno para su distinción. La conexión del PIC con los potenciómetros se muestra en la figura 2.12.

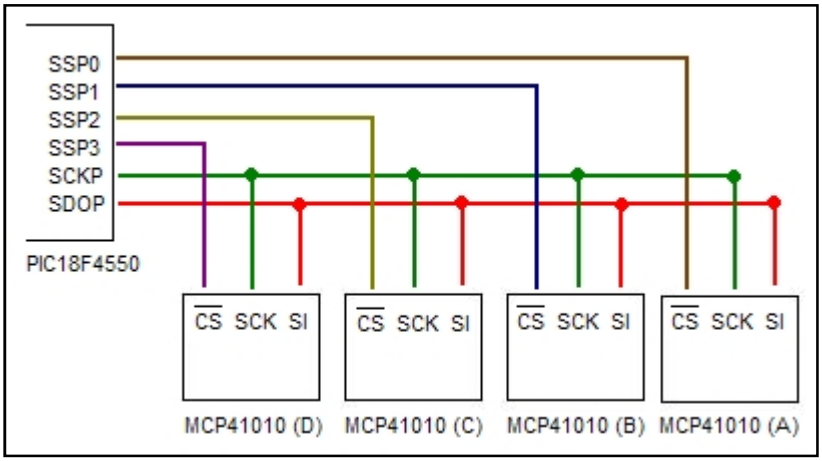



*Figura 2.12*

Ninguno de estos pines está conectado internamente a un registro especial del microcontrolador para la comunicación SPI a diferencia de los pines utilizados para la transferencia de datos con los 2 convertidores analógicos digitales (MCP3901), por lo que en la programación del microcontrolador se creó una librería especial para la comunicación entre el PIC18F4550 y los dispositivos MCP41010, dicha librería se explica en la sección del código del microcontrolador.

Cabe señalar que en vez haber utilizado la señal de reloj SCKP (RE1) y la señal de salida de datos SDOP (RE0) se pudo haber ocupado las terminales SCK (RB1) y SDO (RC7) que si están conectadas a un buffer especial del microcontrolador para establecer la comunicación SPI. Sin embargo para reducir espacio en la tarjeta de adquisición de datos se decidió conectar por separado las señales de control que van tanto hacia los potenciómetros como hacia los convertidores analógicos digitales.

### 2.2.7. Comunicación entre el microcontrolador y la LCD

Como ya se había comentado anteriormente, existen dos formas de visualizar los datos de los cuatro canales, por medio de un display LCD y por medio de la PC.

El display LCD es una pantalla de cristal liquido de la serie JHD204A que puede mostrar cuatro líneas de 20 caracteres cada una, es decir 20X4 = 80 caracteres. Esta LCD es ideal para esta tarjeta de adquisición de datos ya que se muestra el

valor de cada canal por cada una de las líneas. Su consumo de corriente es de aproximadamente 5mA y su voltaje de alimentación es de +5V.

El LCD dispone de una matriz de 5x8 puntos para representar cada carácter. En total se pueden representar 256 caracteres diferentes de los cuales 240 están grabados dentro del LCD y representan las letras mayúsculas, minúsculas, números, signos de puntuación, etc. Ocho caracteres pueden ser definidos por el usuario.

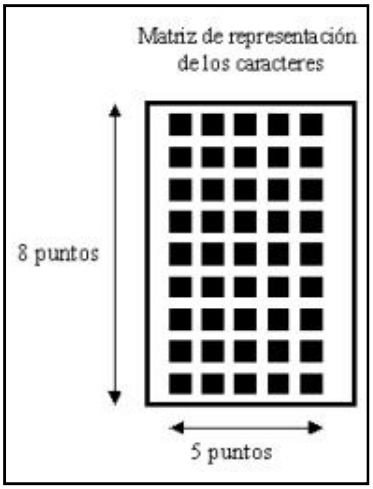



*Figura 2.13 Representación de caracteres*


*Figura 2.14 LCD 20x4*

El bus de datos de la LCD es de 8 bits, aunque para el diseño de la tarjeta de alta resolución se ocuparon solo 4 bits (D4, D5, D6 y D7) lo cual limita el número de caracteres pero también reduce el número de conexiones hacia el microcontrolador. La conexión del microcontrolador con la LCD se muestra en la figura 2.15.

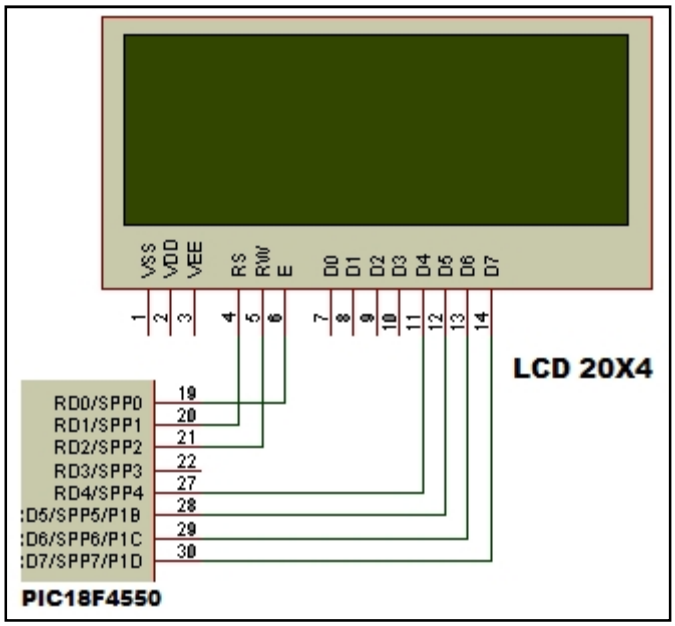



*Figura 2.15*

El pin RS se ocupa para indicarle al bus de datos de la LCD si la información que le llega es una instrucción o un carácter. Si RS=0 indica que en el bus de datos hay una instrucción, si RS=1 indica que en el bus hay un carácter.

El pin R/W es de lectura y escritura. Si R/W= 0 el modulo escribe en pantalla el dato que está en el bus de datos, Si R/W= 1 la LCD lee lo que hay en el bus de datos.

El pin E es el que le indica a la LCD cuando debe o no funcionar. Cuando E=0 no se podrá utilizar el display, si E=1 se podrá transferir información entre el microcontrolador y la LCD.

### 2.2.8. Módulo USB del PIC18F4550

El modulo USB del PIC18f4550 está conectado internamente a un regulador de voltaje de 3.3V. Físicamente la transmisión y recepción e datos entre el microcontrolador y la PC es por medio de USB. Para lograr esta comunicación se coloco un capacitor cerámico de 470nf entre el pin $V_{USB}$ del PIC y GND. Como se mostro en la sección 3.1 el voltaje de alimentación del circuito puede ser por medio de un voltaje externo de 5V o por medio del voltaje USB proveniente de la computadora. Los pines D+ y D- del PIC están puenteados directamente a un conector USB tipo B hembra en el circuito impreso, así como también el voltaje de alimentación de +5V y GND. Las conexiones se pueden observar en la figura 2.16.

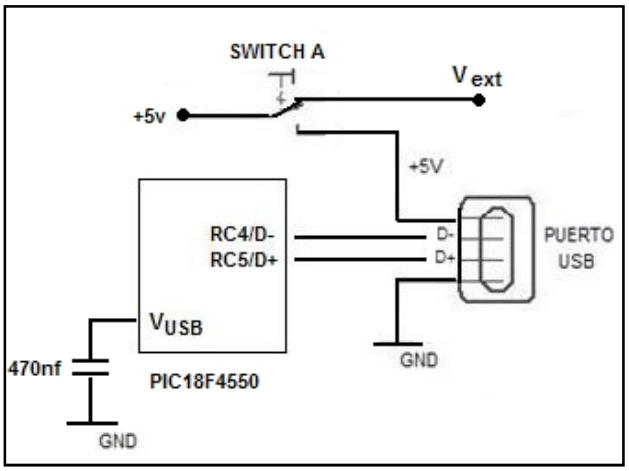



*Figura 2.16*

El módulo USB está configurado para emular la comunicación RS-232, ya que esta es la manera más sencilla de convertir una interface RS-232 a USB a través de la clase CDC (Clase de Dispositivo de Comunicación). Una ventaja de este método es que la aplicación en la PC ve la conexión USB como un puerto COM RS-232. Otra ventaja es que este método aprovecha controladores de Windows que ya están incluidos desde W´98 y versiones actuales, haciendo innecesario el desarrollo de un driver.

En la emulación RS-232, cuando se conecta el USB a la PC, la aplicación en Windows ve la conexión como un puerto COM virtual mediante el servicio provisto por los controladores usbser.sys y ccport.sys.

## 2.3 Acondicionamiento de la señal

Debido a que los voltajes entregados por los arreglos de strain gauges son muy pequeños (en el orden de los μV) se necesita un acondicionar a la señal para que esta pueda ser procesada digitalmente. Para esta se necesito del amplificador de instrumentación AD620.

### 2.3.1. Amplificador de instrumentación AD620

El AD620 es un amplificador de instrumentación fabricado por la empresa Analog Devices. Este dispositivo permite configurar su ganancia en el rango de 1 a 1000 con tan solo una resistencia externa por lo que es ideal para poder amplificar señales que están en el orden de los microvolts. Para este proyecto se ocuparon cuatro amplificadores AD620 uno para cada de entrada.

Este circuito integrado es utilizado para aplicaciones en donde se ocupan tarjetas de adquisición de datos, señales biomédicas o interfaces para transductores además de que es de bajo costo y tiene una baja corriente de polarización lo que permite que internamente sea de bajo ruido. El AD620 se polariza con un voltaje positivo de +5V y uno negativo de -5V, el voltaje negativo es obtenido del convertidor TC7660 como se observa en la figura 3.3.

Internamente el AD620 está compuesto por 3 amplificadores operacionales formando así el tradicional amplificador de instrumentación (mostrado en el capítulo 2). La ecuación que determina la ganancia en función de su resistencia externa es la siguiente:

$$G = \left( {}^{49.9\,kohms}/_{RG} \right) + 1$$

$$RG = {}^{(49.9\,Kohms)}/_{(G-1)}$$

Para este sistema de adquisición la ganancia del amplificador no es fija, sino que es controlada a través de un potenciómetro digital.

### 2.3.2. Potenciómetro digital MCP41010

El MCP41010 es un dispositivo fabricado por Microchip Technology y es un potenciómetro digital que puede variar su resistencia en un rango de 0 a 10K a través de la comunicación SPI. Su alimentación es con un voltaje de +5V y disponen de una memoria no volátil que les permite conservar su configuración una vez desconectados de su alimentación.

Estos dispositivos son ideales para una amplia variedad de aplicaciones de ajuste calibración, configuración, acondicionamiento de señales de control.

Este potenciómetro digital varía su resistencia gracias a un código que le envía el PIC18F4550 por medio de la SPI. La salida del potenciómetro (resistencia) está conectada directamente a las terminales RG de AD620. De tal manera que se ocuparon 4 MCP41010 para modificar la ganancia de los cuatro AD620, la figura 2.17 representa la conexión entre el potenciómetro y el amplificador AD620.

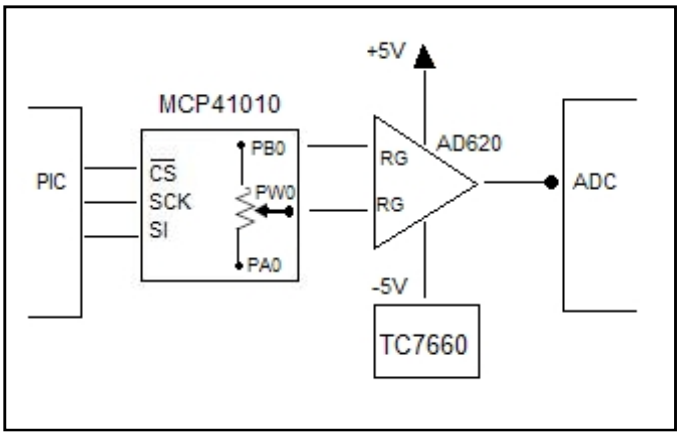



*Figura 2.17*

Con esta configuración se logro que la ganancia del amplificador pueda ser controlada a través del microcontrolador.

El MCP41010 se compone de una resistencia variable que es modificada a través de un registro interno de 8 bits, por lo que se puede tener 256 niveles diferentes de resistencia. Si por ejemplo se envía el código 00h desde el microcontrolador, la terminal PW estaría lo más cercano a la terminal PB, por lo que el dispositivo entregaría una resistencia muy baja, si se enviaran un código cercano a FFh la terminal PW estaría más cerca de la terminal PA y la resistencia seria cercana a los 10K. El cálculo de la resistencia se hace con la siguiente fórmula:

$$R_{WA}(DN) = [(R_{AB}) \cdot (256 - D_N)/256] + R_W$$

Por ejemplo, si se envía el código 192d desde el microcontrolador la resistencia será 7552 entre las terminales PB y PW como puede verse a continuación en la figura 2.18:

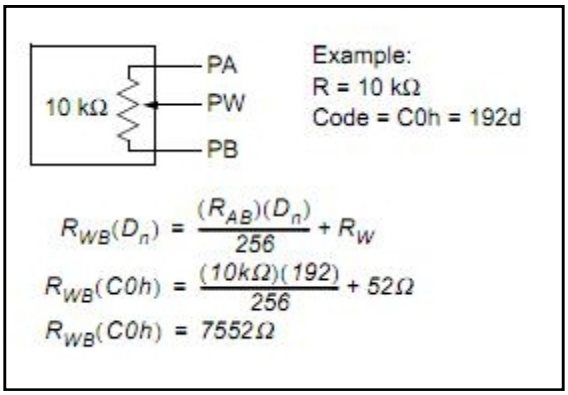



*Figura 2.18*

### 2.3.3. Interface SPI del MCP41010

Como ya se había mencionado la comunicación entre el microcontrolador y los potenciómetros es por medio de la interface SPI. Esta interface permite realizar los comandos;

- Escribir un nuevo valor al registro de datos del potenciómetro
- NOP (Comando de no operación)
- Modo shutdown (modo ahorro de energía)

Las terminales del MCP41010 que sirven como interface para la comunicación con el microcontrolador son:

**Chip Select (CS):** Este pin se utiliza para ejecutar un nuevo comando después de que es acomodado en por el registro de desplazamiento conectado en SDI.

**Serial Clock (SCK):** este pin es de entrada y espera una señal de sincronización que le envía el microcontrolador.

**Serial Data Input (SDI):** Este es el pin de entrada de datos, los comandos recibidos desde el microcontrolador son sincronizados en el registro de desplazamiento de este pin.

La ejecución de cualquier mando se logra poniendo CS en bajo y luego enviando un byte de comando seguido de un byte de datos. El comando de 16 bits se ejecuta cuando CS se pone en estado lógico alto. Los datos son registrados en flancos de subida de la señal de reloj (SCK) como se muestra en la figura 2.19:

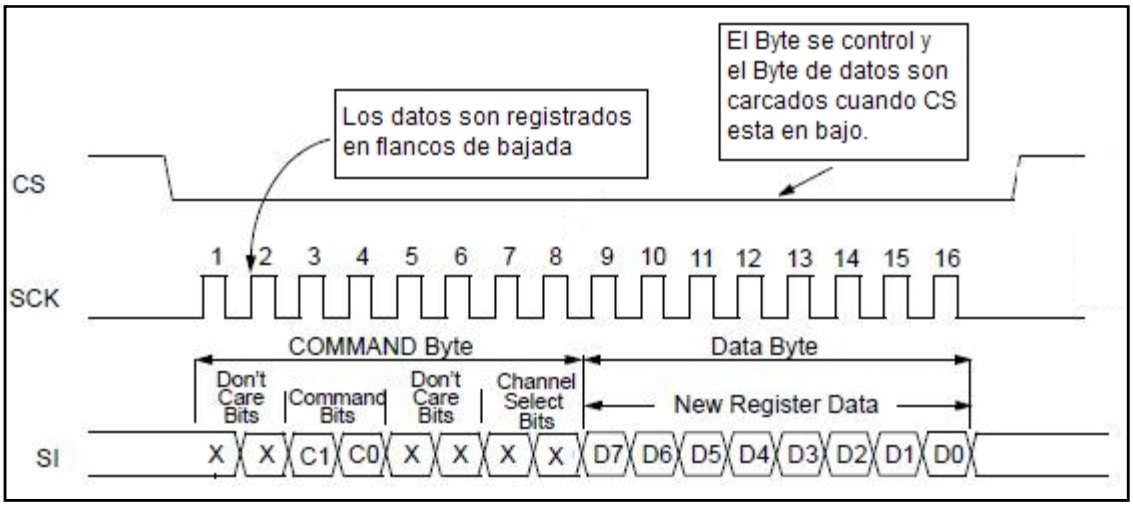



*Figura 2.19*

En el byte de comando solo se utilizan los bits C1 y C0, de manera que si se quiere escribir un código en el byte de datos entonces se tiene que escribir 0,1 en

estos bits. Si la combinación es 1,0 entonces el potenciómetro permanecerá en modo *shutdown,* en este comando el dispositivo permanece en una etapa de ahorro de energía. La siguiente tabla (2.1) de verdad muestra el funcionamiento del byte de comandos:

| C1 | C0 | Comando | Resumen del comando |
|---|---|---|---|
| 0 | 0 | Ninguno | Ningún comando se ejecuta |
| 0 | 1 | Escritura de datos | Escribe el dato contenido en el Byte de datos para modificar la resistencia del potenciómetro |
| 1 | 0 | Shutdown | El potenciómetro entra en "modo ahorro de energía" |
| 1 | 1 | Ninguno | Ningún comando se ejecuta |

*Tabla 2.1*

## 2.4 Procesamiento y transmisión de datos

Para que un sistema digital pueda procesar datos analógicos para su procesamiento en la computadora necesariamente debe contar con un conversor analógico digital (ADC). En este circuito se utilizo el convertidor MCP3901 para transformar la señal analógica proveniente de las galgas extensométricas a una señal digital, para que así pueda ser tratada por el microcontrolador y este a su vez transmita los datos a una pantalla de cristal líquido (LCD) o a una computadora por medio de USB.

### 2.4.1. Convertidor Analógico Digital MCP3901

El dispositivo MCP3901 tiene implementado dos convertidores A/D que pueden ser programados en cuanto a su resolución y velocidad. La velocidad puede llegar hasta las 64Ksps (Kilo muestras por segundo) y la resolución puede ser definida entre los 16 y 24 bits. Este convertidor es fabricado por Microchip Technology. Dentro de las aplicaciones para las que está diseñado este convertidor A/D están las de instrumentación portátil, señales biomédicas, automoción, medición de energía etc.

El MCP3901 dispone de dos pines para su alimentación digital y analógica, para el diseño de esta tarjeta de adquisición las dos señales se conectaron a +5V cada una con un capacitor de bypass como lo recomienda el fabricante. El convertidor A/D para que sea funcional necesita una señal de reloj. Esta señal se logra colocando en paralelo un cristal de 3.57Mhz y dos capacitores de 22pf en cada

terminal con referencia a tierra. El MCP3901 también dispone de un pin de RESET, por lo que esté se conecta directo a la señal de +5V, ya que si está conectado a tierra los registros internos del convertidor permanecerán resteados y no podrá existir la comunicación con el PIC. Estas conexiones se muestran en la figura 2.20.

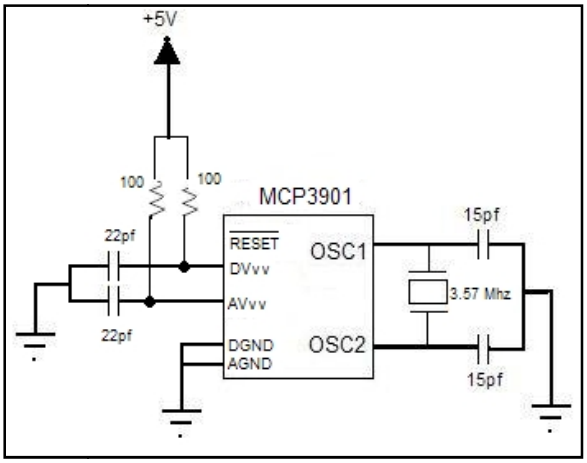



*Figura 2.20*

Internamente este dispositivo contiene dos convertidores A/D Delta-Sigma, dos amplificadores de ganancia programables, una interface SPI para la comunicación con un microcontrolador, referencia de tensión, compensación de retardo de fase y un bloque de salida con modulador. El diagrama a bloques del MCP3901 se puede observar en la figura 2.21.

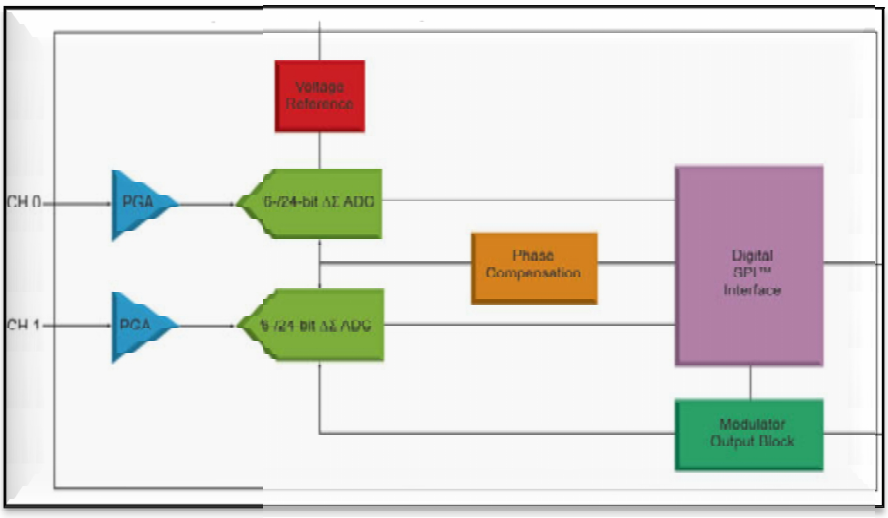



*Figura 2.21*

Los amplificadores operacionales y la referencia de tensión mejoran la capacidad del MCP3901 para medir señales de muy bajo voltaje, reduciendo así el número de componentes externos. Esto permite un diseño de menor tamaño y menor costo. La interface SPI proporciona una sencilla conexión al microcontrolador y gracias a esta comunicación se puede ajustar el convertidor A/D para controlar su resolución y su velocidad de muestreo.

El bloque de compensación de retardo de fase permite que este dispositivo compense las diferencias de fase en aplicaciones de energía trifásica y el bloque de salida con modulador permite mantener la linealidad de la señal mediante un algoritmo, sin embargo estos dos bloques no se programaron para esta aplicación.

### 2.4.1.1. Registros internos del MCP3901

El MCP3901 tiene ocho registros internos, dos de ellos son de solo lectura y seis son de configuración (lectura y escritura). Todos estos registros tienen un ancho de palabra de ocho bits y se puede acceder a cada uno por separado. En la tabla 2.2 se definen los ocho registros del MCP3901.

REGISTER MAP

| Address | Name | Bits | R/W | Description |
|---|---|---|---|---|
| 0x00 | DATA_CH0 | 24 | R | Channel 0 ADC Data <23:0>, MSB First |
| 0x03 | DATA_CH1 | 24 | R | Channel 1 ADC Data <23:0>, MSB First |
| 0x06 | MOD | 8 | R/W | Delta-Sigma Modulators Output Register |
| 0x07 | PHASE | 8 | R/W | Phase Delay Configuration Register |
| 0x08 | GAIN | 8 | R/W | Gain Configuration Register |
| 0x09 | STATUS/COM | 8 | R/W | Status/Communication Register |
| 0x0A | CONFIG1 | 8 | R/W | Configuration Register 1 |
| 0x0B | CONFIG2 | 8 | R/W | Configuration Register 2 |

*Tabla 2.2*

En la lectura de datos los registros se pueden definir por grupos o tipos definiendo así entre una lectura de datos continua o definir un bucle con un conjunto de direcciones. Para definir el modo de acceso a los registros internos se debe configurar los bits 7 y 6 (READ <1:0>) del registro STATUS. En la tabla 2.3 se muestra el agrupamiento de los registros:

**REGISTER MAP GROUPING FOR CONTINUOUS READ MODES**

| Function | Address | READ<1:0> = 01 | READ<1:0> = 10 | READ<1:0> = 11 |
|---|---|---|---|---|
| DATA_CH0 | 0x00 | GROUP | TYPE | LOOP ENTIRE REGISTER MAP |
| | 0x01 | | | |
| | 0x02 | | | |
| DATA_CH1 | 0x03 | GROUP | | |
| | 0x04 | | | |
| | 0x05 | | | |
| MOD | 0x06 | GROUP | TYPE | |
| PHASE | 0x07 | | | |
| GAIN | 0x08 | | | |
| STATUS/ COM | 0x09 | GROUP | | |
| CONFIG1 | 0x0A | | | |
| CONFIG2 | 0x0B | | | |

*Tabla 2.3*

### 2.4.1.2. Resolución del convertidor A/D

Los dos canales del MCP3901 se presentan en 23 bits o 15 bits más un bit que indica si la señal es positiva o negativa. El formato en que se presentan los datos es en complemento a dos.

La resolución del ADC depende de la configuración de los bits 13 y 12 (OSR <1:0>) del registro CONFIG 1. La resolución es la misma para ambos canales. En la tabla 2.4 se muestra la configuración del OSR=256 lo que equivale a programar los dos canales del convertidor a 24 bits.

**OSR = 256 OUTPUT CODE EXAMPLES**

| ADC Output Code (MSB First) | Hexadecimal | Decimal |
|---|---|---|
| 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | 0x7FFFFF | + 8,388,607 |
| 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 | 0x7FFFFE | + 8,388,606 |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0x000000 | 0 |
| 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | 0xFFFFFF | -1 |
| 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 | 0x800001 | - 8,388,607 |
| 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0x800000 | - 8,388,608 |

*Tabla 2.4.*

Como se muestra en la tabla anterior, el bit más significativo (MSB) es el bit de signo, los siguientes 23 bits representan el dato de la señal. Entonces para esta aplicación se están utilizando 23 bits de resolución más un bit de signo.

### 2.4.1.3 Interface SPI del MCP3901

La transmisión de datos entre el microcontrolador y los dos MCP3901 es mediante la interface SPI, esta conexión se puede ver en la figura 2.4.1.3. Las terminales del MCP3901 que sirven como interface para la comunicación con el microcontrolador son:

**Chip Select (CS):** Este pin habilita la comunicación SPI, cuando se encuentra en estado alto no existe la comunicación, esta sucede hasta que haya un flanco de bajada y la comunicación termina cuando exista un flanco de subida.

**Serial Data Clock (SCK):** Este pin es de entrada y espera una señal de sincronización que le envía el microcontrolador, la velocidad máxima del reloj es de 20Mhz.

**Serial Data Input (SDI):** Este pin es de entrada de datos. Los datos son registrados en flancos ascendentes de la señal SCK.

**Serial Data Output (SDO):** Este pin es de salida de datos. Los datos son transmitidos hacia el microcontrolador en flancos descendentes con respecto a la señal SCK.

El MCP3901 es compatible con la con los modos de comunicación 0,0 y 1,1 de la interface SPI. Estos dos modos de comunicación de datos son prácticamente iguales y para el diseño de la tarjeta de adquisición se ocupo la comunicación 0,0 en donde los datos son enviados en flancos de bajada de la señal SCK y los datos son registrados por la señal de entrada en flancos ascendentes con respecto a la señal SCK. La representación de estas señales se muestra en la figura 2.22.

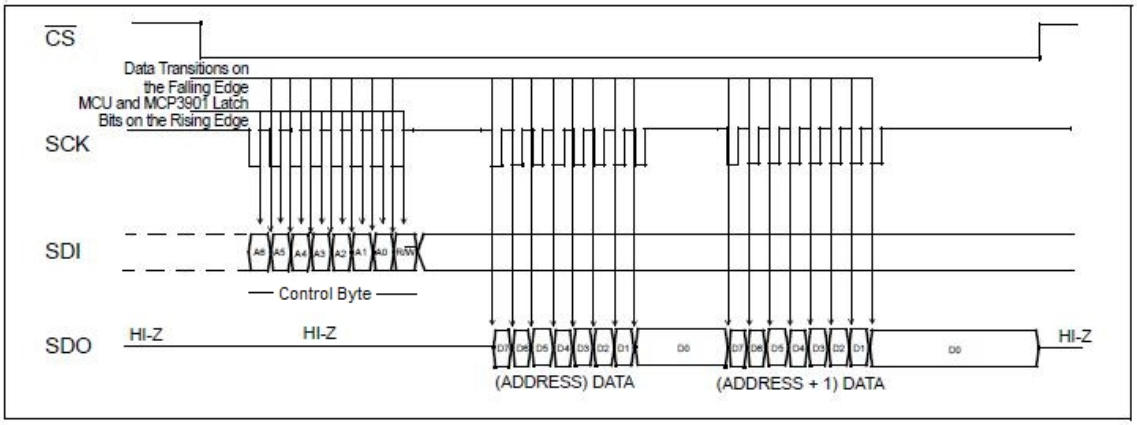


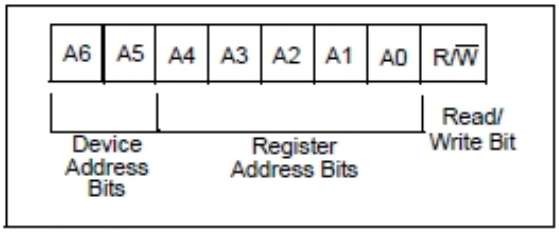

*Figura 2.22*

La comunicación SPI comienza cuando CS se pone en estado bajo y termina con CS en estado alto. La interface del MCP39001 tiene una estructura de comandos simple, el primer byte a transmitir siempre es el "Control Byte" y es seguido de bytes de datos que sirven para su configuración.

### 2.4.1.4 Byte de Control

El "Control Byte" del convertidor A/D contiene dos bits A<6:5> los cuales sirven para direccionar múltiples dispositivos MCP3901. Existen cuatro tipos de dispositivos MCP3901, cada uno con un código diferente que depende de la combinación de los bits A<6:5>. Sin embargo cabe mencionar que antes del diseño final de este sistema digital se trato de conseguir los cuatro modelos directamente con la empresa Microchip Technology y la respuesta fue que de esos cuatro modelos, tres de ellos no estaban en fabricación aun por ser un dispositivo prácticamente nuevo en el mercado. El único modelo que estaba en fabricación es el que tiene código 0,0.

Los bits A4, A3, A2 y A0 sirven para indicarle al convertidor a cuál de los ocho posibles registros se requiere acceder. Después de haber accedido a un registro determinado, se necesita indicar si ese registro será leído o escrito, esta operación se hace con el bit R/W. Cuando se requiere leer la información capturada por los dos convertidores A/D el bit R/W será igual a cero y si se necesita escribir sobre un registro para la configuración del MCP3901 el bit R/W debe ser igual a 1. La estructura de byte control se puede observar en la figura 2.23.




*Control Byte.*

*Figura 2.23*

Después del Byte de control, los siguientes bytes a los cuales se puede tener acceso son los de la tabla 2.2.

## 2.5 Desarrollo del código del microcontrolador

En esta sección se explicara la estructura del código de programación para el PIC18f4550. Este código es llamado *firmware* el cual es grabado en la memoria flash del microcontrolador y su función es la de control y comunicación del hardware. Se considera parte del hardware por estar integrado en la electrónica de la tarjeta de adquisición, pero también es parte del software porque proporciona la lógica y está escrito en un lenguaje de programación. El código completo se muestra en el Anexo 1.

### 2.5.1. Estructura del código de programación

La programación del PIC18F4550 se realizo en lenguaje C por medio del compilador CCS (Custom Computer Services). Este compilador ha sido desarrollado específicamente para microcontroladores PIC obteniendo la máxima optimización del compilador con estos dispositivos de Microchip Technology. Dispone de una amplia librería de funciones predefinidas, comandos de preprocesado y ejemplos. También suministra los *drivers*para la comunicación con diversos dispositivos como una LCD, convertidores AD, memorias EEPROM etc.

La estructura general del diagrama de flujo se muestra en la figura 2.24 en donde se puede observar que el diagrama tiene dos ciclos principales que permiten al sistema de adquisición mostrar los datos por medio de una PC o una LCD. Cada uno de los procesos de la estructura general contiene otros diagramas de flujo que se irán explicando a lo largo de esta sección.

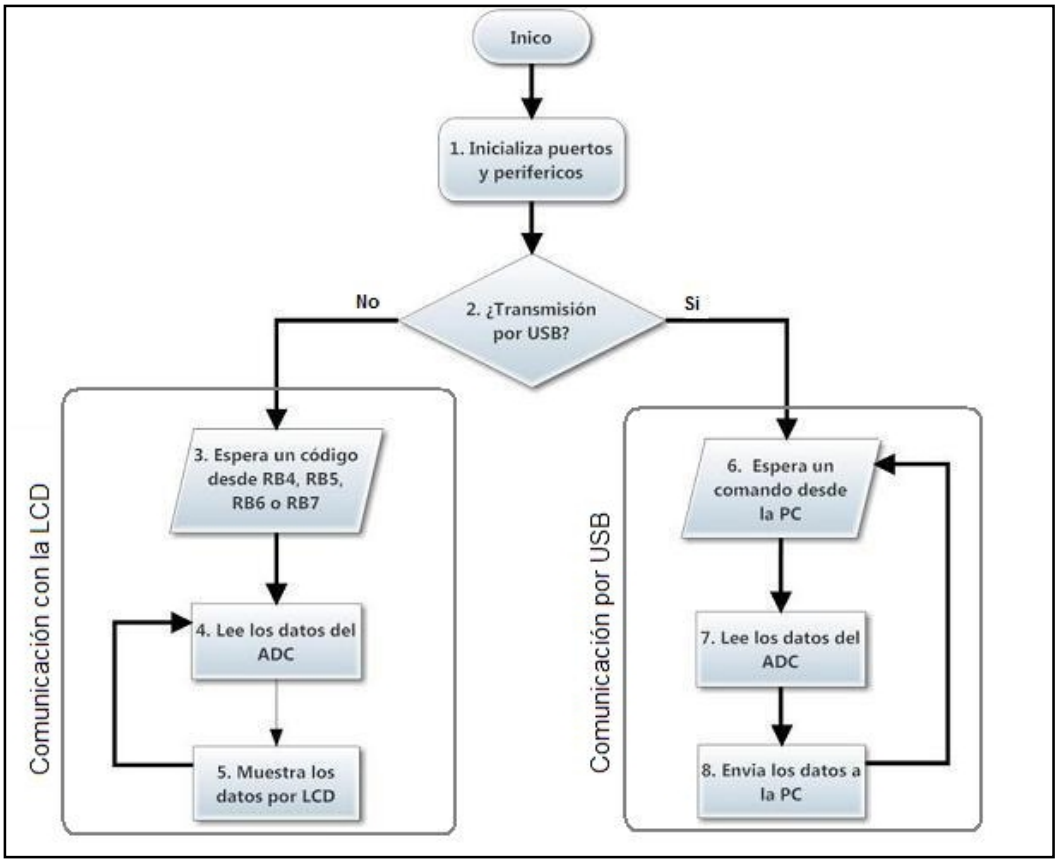



*Figura2.24*

### 2.5.1.1. Inicialización de puertos y periféricos.

En este proceso se configuran los pines del microcontrolador ya sea como entradas o salidas, se configura el módulo USB y SPI además de incluir las librerías para el uso de la LCD y de los potenciómetros. En el código de programación lo primero que se debe escribir es el modelo del microcontrolador con el que se trabaja, para esto se utiliza la directiva:

*#include<18F4550.h>*: Esta directiva define los registros internos del microcontrolador.

Después se le indica al microcontrolador cuáles son los fusibles internos que serán programados utilizando la directiva *#fuses.* Los fusibles activados se muestran a continuación y se hace una breve descripción de la función de cada uno de ellos:

*#fuses HSPLL, PLL5, USBDIV, NOWDT, PROTECT, NOLVP, NODEBUG, VREGEN,*

*HSPLL:* Indica que el cristal que se ocupa es de tipo HS alta velocidad.

*PLL5:* Para que el microcontrolador trabaje a 48Mhz se necesita que el preescalador PLL sea igual a 5.

*USBDIV:* Significa que el clock se tomara del PLL/2 = 96Mz/2 = 48Mhz.

*NOWDT:* No se utiliza el Watchdog.

*PROTECT*: El código del microcontrolador está protegido contra lecturas.

*NOLVP*: Queda desactivada la programación a bajo voltaje.

*NODEBUG*: No se utiliza el modo debug.

*VREGEN*: Se habilita el regulador interno de 3.3V para el módulo USB.

En esta parte del código también se declaran algunas de las variables y se inicializan los pines como entrada o salida de datos.

#### 2.5.1.2. Configuración del módulo USB

El compilador CCS ya suministra las librerías para comunicar el microcontrolador PIC con la PC utilizando la interface USB, por lo que se utilizo la especificación CDC (Clase de Dispositivo de Comunicación) para emular la comunicación RS-232.

El total de ancho de banda que posee el bus del PIC18F4550 es de 1.5Mbps, sin embargo es imposible que el dispositivo se comunique con el Host a esta velocidad debido a que existen pérdidas de tiempo por el protocolo y por los procesos que el microcontrolador debe atender con los ADC y el procesamiento de la información.

La librería que se ocupa para la configuración del PIC18F4550 es:

*Usb_cdc.h*: Incorpora el driver que permite utilizar una clase de dispositivo CDC para emular un dispositivo RS-232 y lo muestra como un puerto virtual COM en Windows.

Las funciones utilizadas para la programación del PIC son:

*Usb_init*: Inicializa el módulo USB. Espera un bucle infinito hasta que el periférico USB es conectado al bus.

*Usn_task()*: Cuando el PIC es conectado o desconectado del bus, esta función inicializa el puerto USB stack o resetea el puerto virtual.

*Usb_enumerated():* Devuelve un TRUE si el dispositivo ha sido enumerado correctamente por la PC, en este caso el dispositivo entra en un modo de operación normal por lo que puede enviar y recibir paquetes de datos.

*Usb_cdc_getc()*: Lee un carácter (8 bits) por el buffer de recepción, si no hay un dato en el buffer se espera indefinidamente hasta que un carácter haya sido recibido.

*Usb_cdc_putc()*: Envía un carácter (8 bits) por el buffer de transmisión, en caso de que este lleno, espera hasta haya espacio para poder enviarlo.

### 2.5.1.3. Configuración del módulo SPI

Al igual que en el módulo USB, CCS suministra la librería para el manejo de la comunicación SPI. Con esta librería solo se controlan los dos convertidores A/D ya que para el control de los cuatro potenciómetros digitales se realizo una propia.

La función *setup_spi()* sirve para inicializar y configurar la interface SPI, ya que define el modo de trabajo de esta comunicación.

En la directiva *#include<18F4550.h>*están definidas una serie de etiquetas que están asociadas a los modos de funcionamiento de la SPI y se explican a continuación:

*setup_spi(SPI_MASTER|SPI_L_TO_H|SPI_XMIT_L_TO_H|SPI_CLK_DIV_4);*

*SPI_MASTER*: El microcontrolador será el dispositivo maestro.

*SPI_L_TO_H|SPI_XMIT_L_TO_H:* El microcontrolador transmite datos en flancos de subida.

*SPI_CLK_DIV_4: Define la frecuencia a la que oscilara el reloj.*

Las funciones utilizadas para la programación del PIC son:

*Spi_write():* Esta función transmite un carácter por la interface SPI

*Spi_read():* Devuelve un carácter leído de la interface SPI

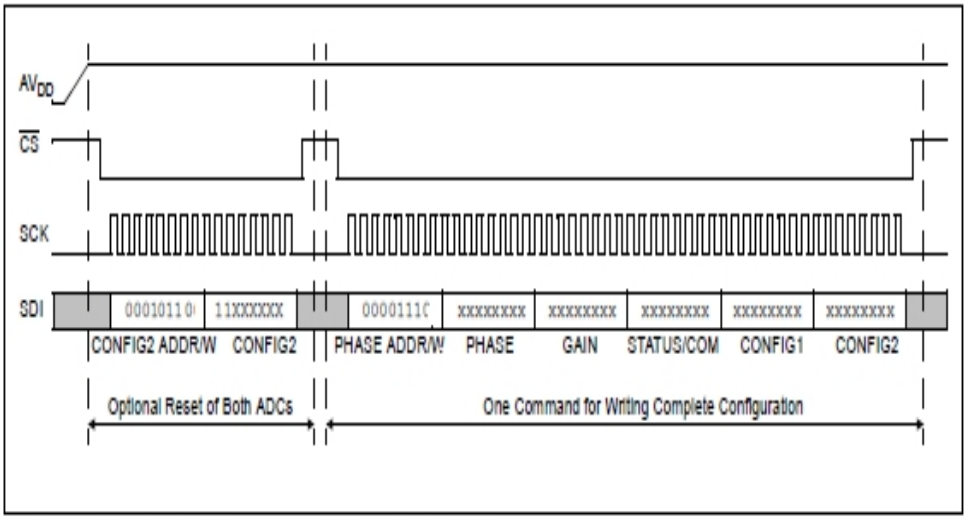
### 2.5.1.4. Configuración de los convertidores MCP3901

Los convertidores MCP3901 pueden ser configurados a través de sus registros internos de programación. Estos dispositivos fueron programados con una

ganancia de 2 y una resolución de 24 bits por lo que no fueron modificados todos los registros. Para facilitar el entendimiento a continuación, se explica la trama que se le envía solo a un convertidor A/D aunque la configuración es la misma para los dos.

Los primeros dos bytes que se le envían al MCP3901 son para resetearlo ya que esta configuración es recomendada por el fabricante. Como se puede observar en la figura 2.25, el byte *CONFIG2 ADDR/W* es el byte de control. Su bit menos significativo se pone a "0" indicando con esto que se requiere hacer una operación de escritura en el convertidor. En sus bits $b_1$, $b_2$, $b_3$ y $b_4$ se escribe la dirección del registro al cual se requiere acceder, en este caso la dirección es 11 (0x0B) que corresponde al *CONFIG2*. En este registro los bits $b_7$ y $b_6$ se ponen a "1", ya de acuerdo con la hoja de datos del MCP3901 con esta configuración se garantiza que los dos canales de un convertidor A/D serán reseteados.

Enseguida de resetear el convertidor se envían un byte de control *(PHASE ADD/W)* con la operación de escritura y la dirección 7 (0X07) que corresponde a la dirección del registro PHASE. A partir de que se tiene acceso a este registro inmediatamente después, los demás registros pueden ser configurados sin necesidad de indicar su dirección. Es decir que si la comunicación no se interrumpe (poniendo en alto CS) se puede tener acceso a cada uno de los registros ya que el MCP3901 va incrementando las direcciones en 1.




*Recommended Configuration Sequence at Power-up.*

*Figura 2.25*

De acuerdo a la hoja de datos del MCP3901 el código que se escribió para enviar la trama de datos de la figura 2.24 es el siguiente:

```
//**********************Reset en el ADC**************************************
output_low(CS);                  //Inicio la comunicación con CS' en bajo
spi_write(0b00010110);          //Dirección del CONFIG2 (0x0B)
spi_write(0b11000000);          //Acceso al registro CONFIG2
output_high(CS);                //Finalizo la comunicación con CS en alto.
delay_us(1);                    //Se crea un reset
//**********************Programación del ADC*********************************
output_low(CS);                 //Inicio la comunicación con CS en bajo
spi_write(0b00001110);      //Dirección de la PHASE
spi_write(0b00000000);    //Escribo en PHASE
spi_write(0b00000001);    //Escribo en GAIN
spi_write(0b10100011);    //Escribo en STATUS
spi_write(0b00111100);    //Escribo en CONFIG1
spi_write(0b00000000);    //Escribo en CONFIG2
output_high(CS);            //Finalizo la configuración con CS en alto.
```

#### 2.5.1.5. Librería SPI para el control de los potenciómetros digitales

Como se comento en la sección anterior, se creó una librería para el control de los cuatro potenciómetros digitales. Esta librería se llama *potenciomentro.c*y fue diseñada para transmitir datos en flancos de subida. Para incluirla en el código se manda llamar de la siguiente manera:

*#include<potenciometro.c>:* Librería que permite el control de los cuatro dispositivos MCP41010. En esta librería se definen los pines de comunicación entre el micrcocontrolador y los potenciómetros.

Para escribir un dato a los potenciómetros se ocupa set_pot():

*Set_pot(dato)*:escribe un dato de 8 bits en cualquiera de los cuatros potenciómetros.

Si un material está muy deformado, el voltaje que se mida puede ser relativamente grade y el MCP3901 estará limitado para leer estas tensiones. Una forma de resolver este problema es disminuyendo la ganancia en los amplificadores y corrigiendo el valor obtenido por software. Sin embargo este sistema de adquisición de datos sigue estando limitado para medir voltajes mayores a +-0.8V.

Al inicio del programa la ganancia de los amplificadores de instrumentación AD620 es de 100, por lo tanto la resistencia de configuración debe ser aproximadamente de 500ohms.

$$RG = {}^{49.4K}/_{G} - 1$$

$$RG = {}^{49.4K}/_{(100 - 1)} = 498.99\, ohms$$

Entonces la programación de los potenciómetros MCP41010 para obtener la resistencia de 500ohms debe ser:

$$DN = {}^{[(RW - 52) \times 256]}/_{10K}$$

$$DN = {}^{[(500 - 52) \times 256]}/_{10K}$$

$$DN = 11.47 \approx 12$$

Por lo tanto el código que se escribe por SPI es de 12. Con este valor la resistencia de los potenciómetros no será exactamente de 500, por lo que su resistencia real es:

$$RWB = [(10K \times DN) \div 256] + 52$$

$$RWB = [(10K \times 12) \div 256] + 52$$

$$RWB = 520\,ohms$$

Esta resistencia de 520ohms es la que configura a los amplificadores AD620. La ganancia real será de:

$$G = [(49.4K) \div RG] + 1$$

$$G = [(49.4K) \div 520] + 1$$

$$G = 96$$

La ganancia de 96 es la que entregaran los amplificadores AD620 al inicio del código. Esta ganancia se modificara en el programa solo cuando el voltaje de entrada en los convertidores A/D sea mayor a +-0.8V.

**2.5.1.6. Configuración de la LCD**

Para el manejo de la LCD se tomo como referencia la librería LCD420.C del compilador CCS. Por default esta librería funciona para el puerto D de la familia de microcontroladores PIC16F por lo que se modifico para poder ser utilizada por el micrcocontrolador PIC18F4550. En la memoria RAM de la familia PIC16F el puerto D tiene la dirección 0x08, esta dirección se remplazo por 0xF83 que es la dirección del puerto B del PIC18F4550. Esta librería se incluye al código de programación escribiendo:

*#include<lcd4x20.h>:* incluye la librería para el uso de la LCD.

Este archivo también dispone de varias funciones ya definidas:

*Lcd_init():* Esta función inicializa la LCD.

*Lcd_gotoxy(x,y):* Con esta función se coloca el cursor en la parte que se desee de la pantalla, el parámetro “x” indica la columna y el parámetro “y” las filas.

*Printf(name, string, values):* Con esta función se pueden escribir variables en la LCD así como una cadena de caracteres

### 2.5.2. Envió de datos

El envió de datos puede ser hacia la LCD o hacia la computadora por medio del puerto USB. Esta selección se hace leyendo el pin B2 del microcontrolador en el cual está conectado un interruptor que permite al usuario seleccionar como quiere visualizar los datos. Si existe un uno lógico en B2 el microcontrolador ejecuta el código que de la función *display()* por lo que enviara los datos por la LCD, en caso contrario si B2 está en cero lógico el microcontrolador ejecuta la parte del código que está en la función *usb().*

A continuación se muestra la parte del código de programación que permite ingresar a cada una de las funciones:

```
if( input(PIN_B2) )

display();

else

usb();
```

### 2.5.3. Comunicación por USB

Para que ocurra la comunicación por USB se necesita que el pin B2 del microcontrolador se encuentre en estado lógico cero. Si esto ocurre se ejecuta el código de la función *usb()*. La estructura de esta parte del código se puede observar en la figura 2.26.

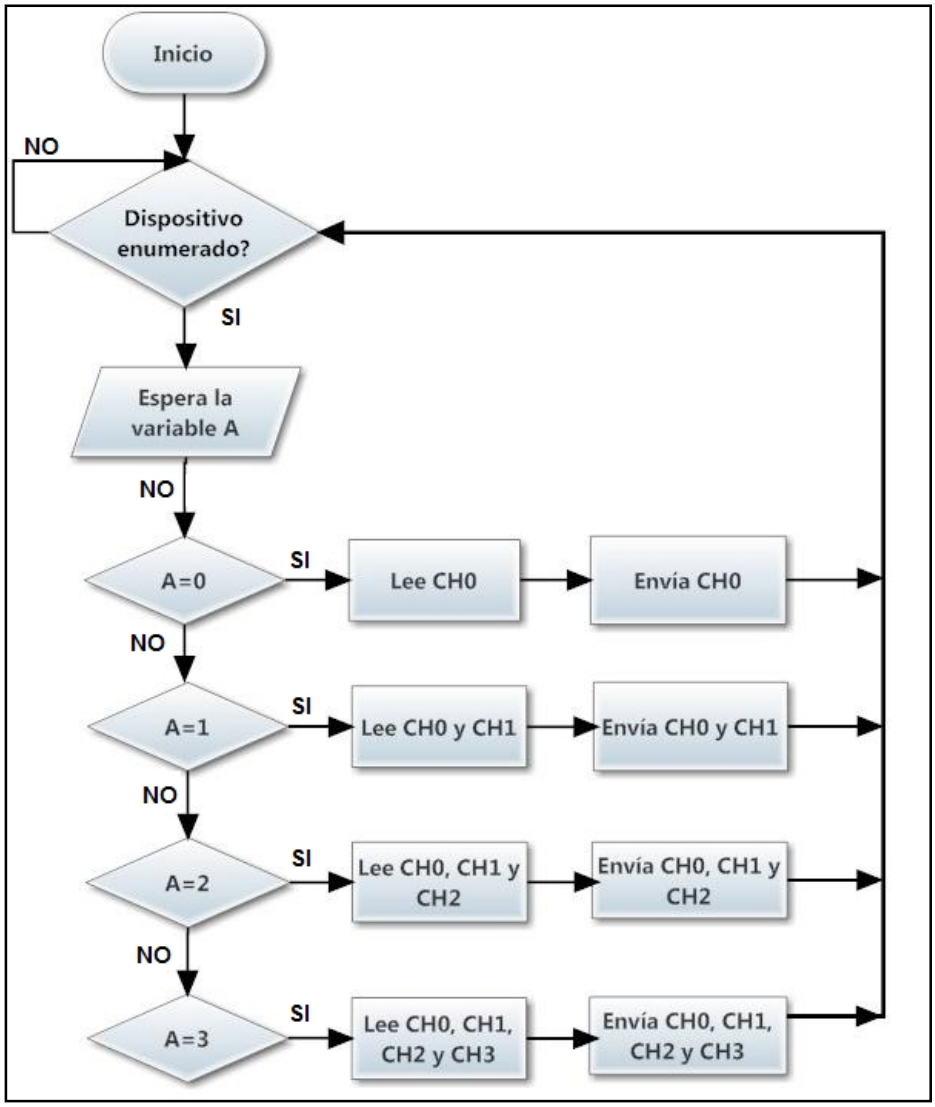



*Figura 2.26*

En esta parte del código si el microcontrolador ha sido correctamente enumerado por la PC se ejecuta el código, de lo contrario estará en un bucle infinito preguntando hasta que haya sido enumerado correctamente. Después el microcontrolador espera leer un dato de 8 bits que guarda en la variable A, este comando es recibido desde LabVIEW e indica el número de canales que el PIC deberá de leer.

Al estar el dato en la variable A, esta se evalúa en el programa y dependiendo de la constante se ejecuta el código correspondiente. La variable A podrá tener hasta cuatro constantes diferentes, cada una le indicara el número de canales que el micrcocontrolador debe de leer. Cuando el microcontrolador ha leído la información, este envía por medio del módulo USB el mismo número de canales

que se le ha indicado en la variable A. Cabe mencionar que cada canal es de 24 bits, por lo tanto la información de cada canal se envía en paquetes de 3 bytes. El proceso de concatenación de los 3 bytes para construir un dato se hace por software con LabVIEW ya que la computadora es más rápida que el microcontrolador, la parte del software se explica en el siguiente capítulo. A continuación se presenta una parte del código en donde el caso es A=0, lo que significa que solo se lee un canal del ADC.

```
voidusb()                          //Función para la comunicación USB
{
while(1)
  {
usb_task();
if (usb_enumerated())          //Esta correctamente enumerado el puerto?
{
    A=usb_cdc_getc();          //Recibe un comando por USB y lo guarda en A
switch(A)                      //Se evalúa A
      {
case '0':                      //Lee el canal CH0

output_low(CS);                //Inicia la comunicación con CS' en bajo
spi_write(0b00000001);     //Byte de control, selecciono la ADDR de CH0

       byte2= spi_read();    //Lee el byte más significativo (CH0)
byte1= spi_read();           //Lee el byte intermedio (CH0)
byte0= spi_read();           //Lee el byte menos significativo (CH0)

output_high(CS);             //Finaliza la comunicación

usb_cdc_putc(byte2);       //Envía el byte más significativo (CH0)
usb_cdc_putc(byte1);       //Envía el byte intermedio (CH0)
usb_cdc_putc(byte0);       //Envía el byte menos significativo (CH0)

break;                       //Sale del caso
}//switch
      }//if
}  //while
 }  //voidusb()
```

### 2.5.4. Comunicación con la LCD

Cuando en el sistema de adquisición muestra los datos por la LCD, es porque el pin B2 del microcontrolador se encuentra en estado lógico uno y por lo tanto el PIC ejecuta el código de la función lcd(). Esta función se explicará a continuación mediante dos diagramas de flujo.

Lo primero que se ejecuta en la función lcd() es un mensaje de inicio en la pantalla LCD y después ahí mismo se muestra un mensaje en donde se le dice al usuario que seleccione el número de canales del sistema de adquisición que quiere usar tal como se muestra en la figura 2.27. La selección se hace por medio de cuatro botones, cada uno de ellos representa un número de canales a utilizar.

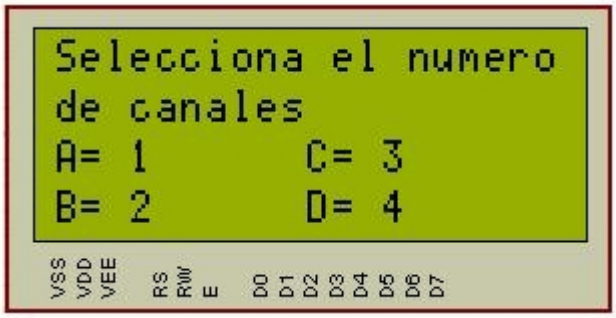



*Figura 2.27*

Para la lectura del estado de los pines, el PIC ejecuta un bucle infinito que pregunta un número indeterminado de veces por el estado de los pines B4, B5, B6 y B7. En el momento en que se detecta un cero lógico en cualquiera de estos bits, se lee el número de canales definidos.

Como se vio en el diagrama de flujo anterior, al seleccionar un número determinado de canales el microcontrolador ejecuta una función especificada.

A continuación solo se explicara el código de la función *1_canal()*. Esta es la más simple de las cuatro ya que solo se lee un canal de los dos dispositivos MCP3901.

Para tener mayor precisión en la medición, se toman 50 muestras y se promedian. Esto se logra con un ciclo *for*, el cual debe estar leyendo los 3 bytes del canal y concatenándolos hasta que se cumpla n=50

En esta parte del código, el procesamiento de las señales debe ser en microcontrolador a diferencia de la función *usb()*que la información es procesada por la PC.

La concatenación de los 3 bytes se ilustra en la figura 2.*28* para un solo canal y se explica a continuación:

1. Se crean 4 variables de 32 bits:
   *Int32 byte2, byte1, byte0, dato;*

2. Se leen los tres bytes:
   *byte2= spi_read(); //Leeel byte más significativo de CH0*
   *byte1= spi_read(); //Lee el byte intermedio de CH0*
   *byte0= spi_read(); //Lee el byte menos significativo de CH03*

3. Se elimina el bit más significativo de byte2, ya que este es el bit de signo y no debe sumarse al dato final.

   *byte2=byte2 && 0b01111111; //Se utiliza el operador AND para eliminar el // bit más significativo*

4. Se recorre 16 posiciones hacia la izquierda el byte más significativo y 8 posiciones hacia la izquierda el byte intermedio.

   ```
   for(i=0;i<=15;i++)
    {
   rotate_left(&byte3,3);  //Recorre a la izq el byte mas sig 16 posiciones
    }

   for(i=0;i<=7;i++)
   {
   rotate_left(&byte2,3);  //Recorre a la izq el byte intermedio 8 posiciones
   }
   ```

5. Se suman los tres bytes:
   *dato= byte2 + byte1 + byte0;*

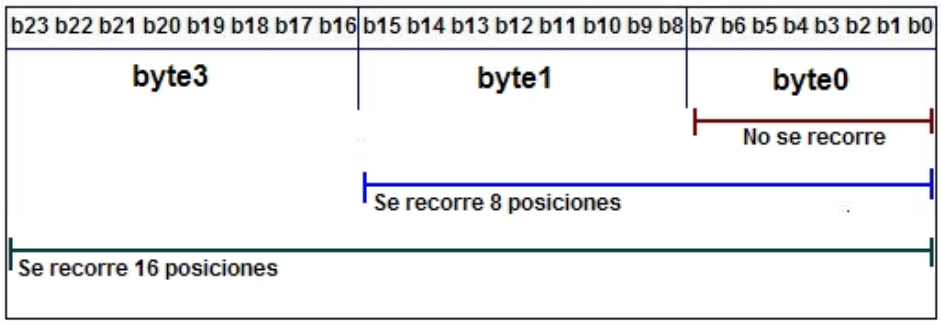



*Figura 2.28*

# CAPÍTULO 3

## Diseño de Software para el procesamiento de datos

En este capítulo se explica el desarrollo del programa para el procesamiento de datos y se hace una breve descripción del software *LabVIEW* que es el utilizado para monitorear los datos provenientes del sistema de adquisición.

### 3.1 Introducción a LabVIEW

Para el sistema de alta resolución, el software utilizado para la visualización y almacenamiento de datos en la PC es LabVIEW el cual es acrónimo de *Laboratory Virtual EnginneringWorkbench*y es un entorno de programación grafica en el que se pueden crear aplicaciones de una manera rápida y sencilla. La empresa National Instruments es la propietaria de LabVIEW y en 1983 comenzó su desarrollo que después saldría a la venta en 1986.

Dentro de las áreas en donde se utiliza LabVIEW están las de control y automatización, medición e instrumentación, análisis de ruido y vibraciones, visión artificial etc. En los últimos años este software ha estado creciendo en nuevas áreas estratégicas relacionadas con nuevos campos de trabajo como simulación de circuitos, diseño de control, sistemas embebidos en tiempo real (FPGAs, DSPs, microprocesadores), algoritmos matemáticos avanzados etc.

### 3.2 Programación en LabVIEW

LabVIEW usa un lenguaje de programación grafico también conocido como lenguaje G. Originalmente este programa estaba destinado a aplicaciones de control de instrumentos electrónicos usados en el desarrollo de sistemas de instrumentación, lo que se conoce como instrumento virtual. Por este motivo los programas creados en LabVIEW se guardaran en con la extensión *.VI*que significa instrumento virtual. Los Instrumentos virtuales están compuestos principalmente por tres partes: el Panel Frontal, el Diagrama a Bloques y el icono/conector.

El panel frontal es la interface de usuario a través de la cual se establecen los valores de entrada y se monitorean las señales de salida del Instrumento Virtual (VI). Las entradas de datos son llamadas controles y las salidas de datos son los indicadores. Se pueden utilizar una gran variedad de controles e indicadores, como por ejemplo lo pueden ser perillas, interruptores, botones, tablas, graficas etc. Un ejemplo de panel frontal de un VI se muestra en la figura 3.1.

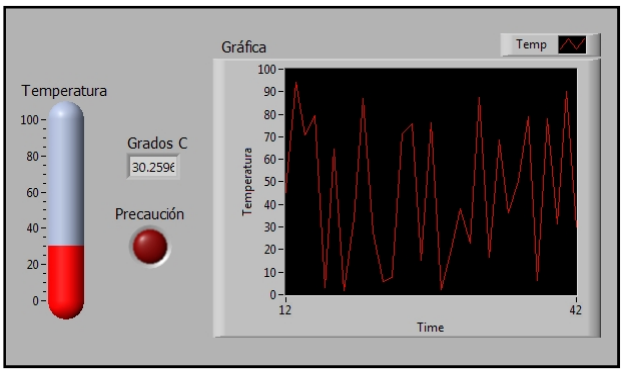



*Figura 3.1*

Cada panel frontal tiene un diagrama a bloques, el cual es el código de fuente de un Instrumento Virtual (VI). El diagrama a bloques se construye por medio de un lenguaje G de programación grafica. En lenguaje G la ejecución del programa es a través de un flujo de datos, el cual consiste en una serie de funciones unidas mediante cables por donde fluyen los datos. Una función solo podrá ejecutarse cuando tenga disponibles todos los datos que le sirven como entrada. Esta forma de ejecutar un programa favorece el paralelismo y es más apropiada para sistemas multiprocesador y multihilo.

El diagrama a bloques que corresponde al panel frontal de la figura 3.2 se muestra a continuación:

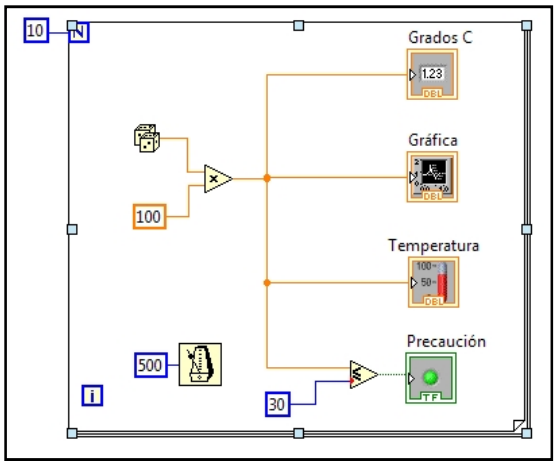



*Figura 3.2*

En muchas ocasiones un programa será de un tamaño tal que habrá que separarlo en parios ficheros o habrá una sección de código que convenga reutilizara varias veces. Un VI puede contener a otro de forma que el segundo seria un *subVI*del primero, el concepto es equivalente a las funciones de un lenguaje tradicional.

## 3.3 Software para el procesamiento de datos

Para definir el procesamiento de datos se desarrollo una interfaz en *LabVIEW* en la que el usuario puede seleccionar el número de canales que desea visualizar y si desea almacenar la información en una base de datos.

En la *figura 3.3* .Se muestra la estructura general del software sin embargo para facilitar su entendimiento, en esta sección solo se explicará el código para el procesamiento de la información de un solo canal de la tarjeta de adquisición de datos.

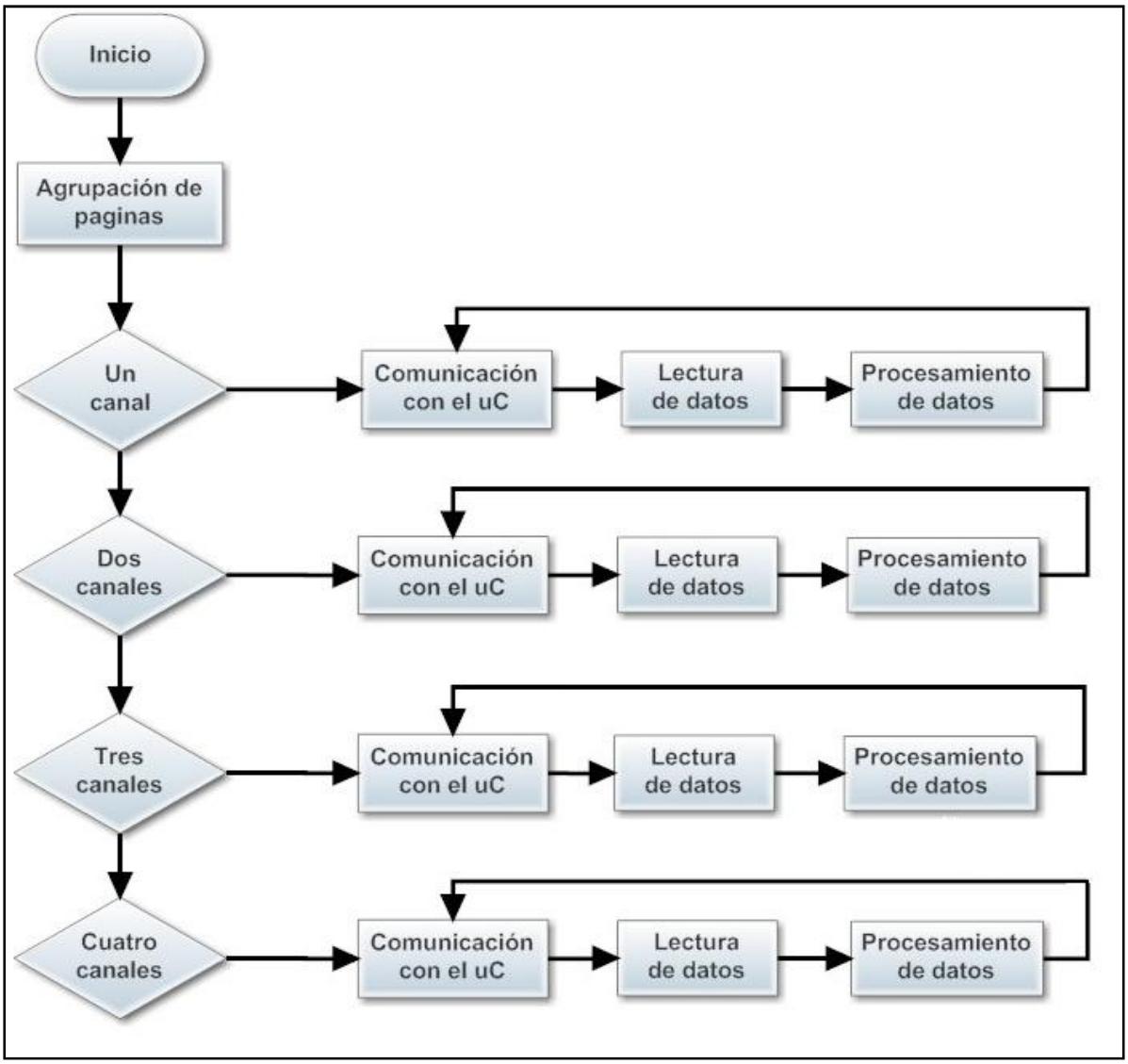



*Figura 3.3*

### 3.3.1. Selección del número de canales

Antes de ejecutar el programa el usuario tiene la opción de seleccionar el número de canales con los que requiere trabajar. Para separar el número de canales, estos se dividen en páginas por medio de un *Tab Control.* De tal manera que en el software hay cuatro páginas, cada una con un código similar para visualizar el número de canales que ha seleccionado el usuario.

Al seleccionar el número de canales con los que se desea trabajar, el *Tab control* entrega un dato que es evaluado por una estructura *Case Structure*. Esta estructura determina el código que debe ser ejecutado en *LabVIEW.* Esta parte del código se muestra en la figura 3.4.

### 3.3.2. Comunicación serial con el microcontrolador

Debido a que la transmisión y recepción de datos entre el microcontrolador y la PC se hace emulando la comunicación RS-232 a través de la configuración el módulo USB, en el software la configuración se realiza de tal manera que se pretenda manejar un puerto serie. En LabVIEW esta comunicación se logra con el uso de las librerías VISA (*Virtual Instrument Software Architecture*). En la figura 3.2.1.a. se muestra el uso de dichas librerías.

En el punto 1 se configura al puerto serie a una velocidad de 9600 baudios (bits por segundo), transmisión de 8 bits y sin bit de paridad. El puerto se selecciona con *VISA resourcename*que es un identificador lógico el cual sirve para comunicarse por medio de un puerto *COM,* esta configuración se ejecuta solo una vez y es al principio del código. En el paso 2 se escribe un comando de la PC al microcontrolador, este dato le indica al PIC el número de canales con los que se desea trabajar. Después en el punto 3 se hace un pequeño retardo de 1ms para permitirle al microcontrolador procesar la información. El punto 4 se lee el numero de bytes que se le indica al *VISA Read.* En el caso de la figura 3.2.1.a, como es un canal el que se lee solamente, el número de bytes que se leen son 3 ya que la resolución de cada canal es de 24 bits (3 bytes). En el paso 5 se cierra una sesión *VISA*, por lo tanto no habrá ninguna comunicación con el microcontrolador hasta que el programa termine de procesar la información.

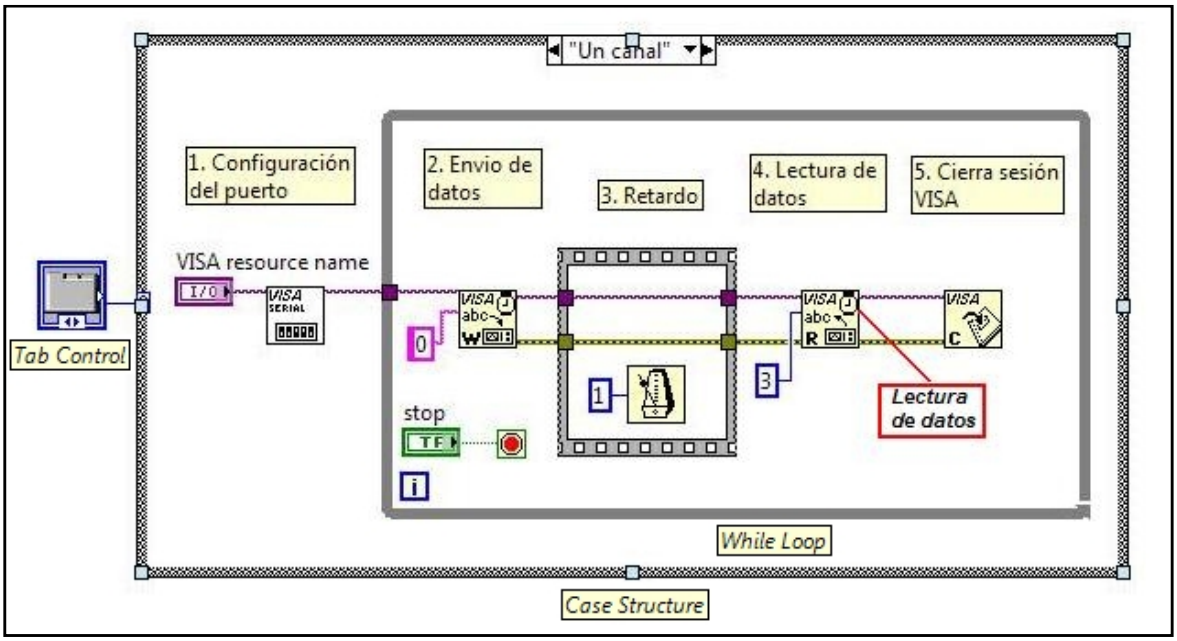



*Figura 3.4*

### 3.3.3 Lectura de datos

Después de haber configurado el puerto serie y de indicarle al microcontrolador el número de canales con los que debe trabajar, el siguiente paso es leer y agrupar la información que entrega la tarjeta de adquisición de datos. Este código se muestra en la figura 3.5.

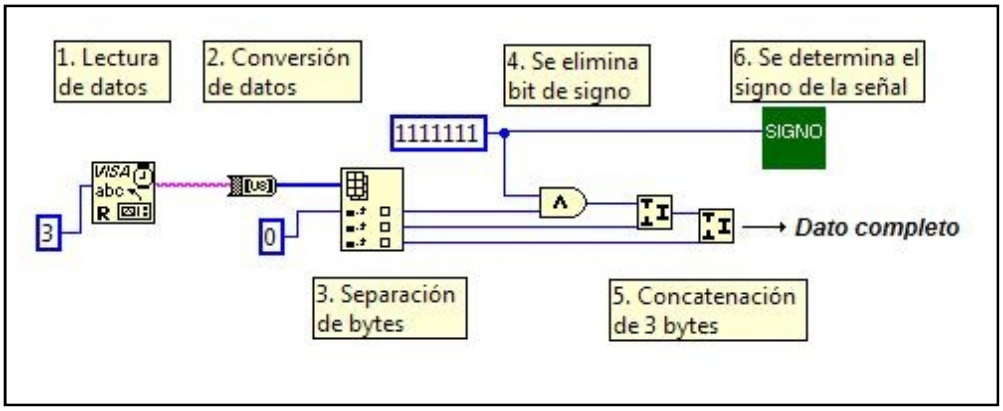



*Figura 3.5*

En el paso 1, el *VISA Read*lee todos los bytes que hay en el buffery entrega una cadena de datos (*string*). Estos datos de tipo *string* se convierten a un arreglo de bytes en el punto 2. En el punto 3, los bytes se separan del arreglo desde el byte más significativo al menos significativo. Y en el caso de que se lea más de un canal, primero se leen los 3 bytes del canal CH0 después los 3 bytes del canal CH1 y así sucesivamente. En paso número 4 se elimina el bit de signo para que no afecte el procesamiento de la información, esto se logra haciendo la operación lógica AND entre el byte más significativo de un canal y la constante 0x7F como se muestra a continuación en la figura 3.6:

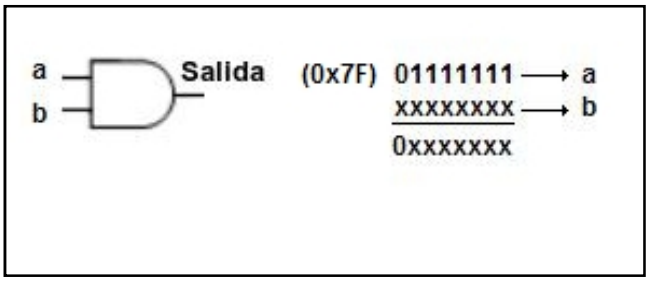



*Figura 3.6*

Después de haber eliminado el bit de signo, en el paso 5 se hace la agrupación de 3 bytes por canal para formar un solo dato. Con el bloque *JoinNumbers*se concatenan dos bytes, por lo que se necesitan dos de estos bloques para concatenar los 3 bytes de un canal.

Para saber cuando la señal es positiva o negativa, en el paso 6 se crea un *SubVI* en donde el byte más significativo de un canal de datos se convierte a un arreglo de boléanos ya que este byte es el que contiene el bit de signo. Después ya que se tiene el arreglo de boléanos, se lee solo el valor del bit 7 (bit de signo) y se hace una comparación, si el bit es igual a uno la señal es negativa y si es igual a cero la señal es positiva. Este *SubVI* se puede observar en la figura 3.7:

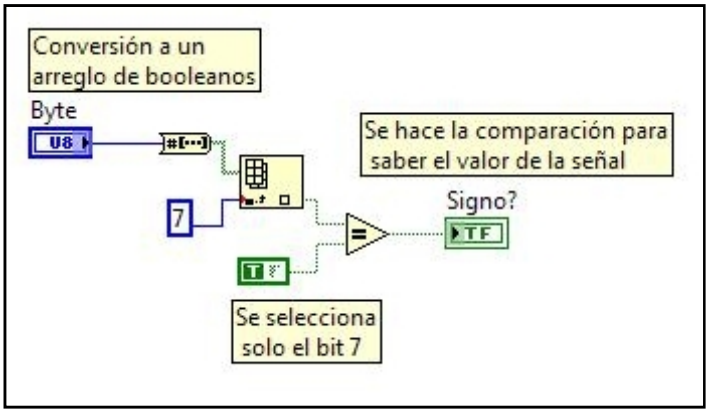



*Figura 3.7*

### 3.3.4 Procesamiento de datos

En esta sección se explica el código que se ocupa para una señal positiva y negativa, así como la formula que se utiliza para convertir los datos a su equivalente a microdeformaciones. La estructura general del código para el procesamiento de los datos se presenta a continuación en la figura 3.8.

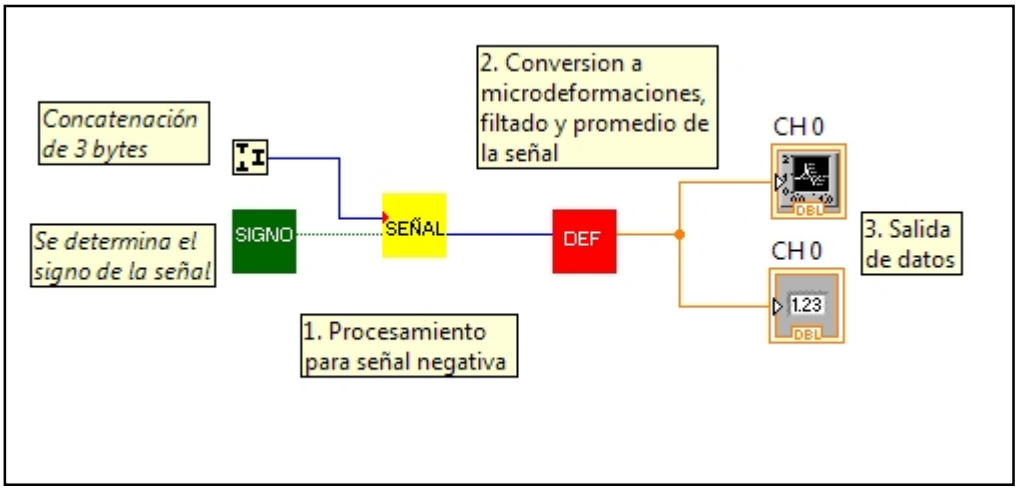



*Figura 3.8*

### 3.3.4.1 Procesamiento para señal negativa

El formato en el que el convertidor A/D entrega los datos es en complemento a dos. Como se explico en el capítulo uno, la representación de una señal positiva se queda igual en su representación binaria, pero una señal negativa está en complemento a dos. Por lo tanto, para conocer el valor de la señal negativa, se pasa a señal positiva haciendo el proceso inverso del complemento a dos, es decir que se invierte el valor de cada una de sus cifras y después se le suma uno. Al conocer el valor de la señal negativa, solo se multiplica por menos uno para agregarle el signo "-"a la señal. Este proceso se encuentra en un SubVI que se muestra en la figura 3.9.

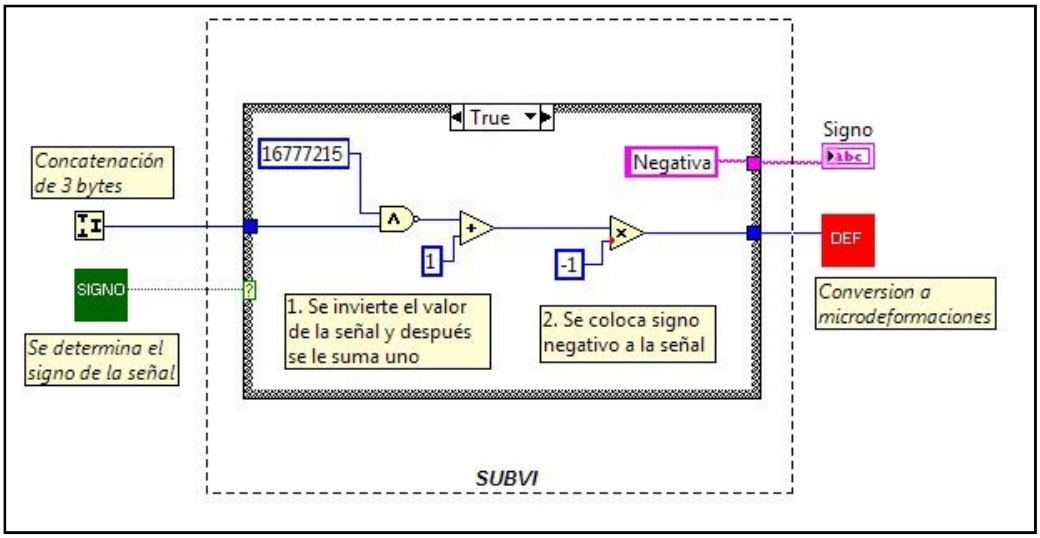



*Figura 3.9*

Este SubVI se encuentra dentro de una estructura *CASE*. El icono que determina el signo de la señal (icono verde) le indicara a la estructura el código que deberá ejecutar. Si es falso se trata de una señal positiva y la señal de entrada pasa sin ser alterada. Si es verdadero se ejecuta un código que permitirá trabajar con números negativos.

En el código para una señal negativa (figura 3.9), en el paso 1 se invierte el valor de cada una de las cifras utilizando una compuerta lógica *NAND*. En una de sus entradas se coloca el numero decimal 16777215 ($2^{24}$) que en binario es la representación de 24 unos, lo cual permite invertir la señal de entrada. A la salida de la NAND se le suma uno y después en el paso 2 la señal se multiplica por menos uno para agregar el signo menos.

### 3.3.4.2 Conversión a microdeformaciones

Para convertir los datos a su representación en microdeformaciones, primero el dato de entrada se convierte a voltaje ya que de acuerdo a las pruebas realizadas en el laboratorio de Mecánica Aplicada una microdeformación es aproximadamente equivalente a un microvolt. El *subVI* que hace esta operación se muestra en la figura 3.10.

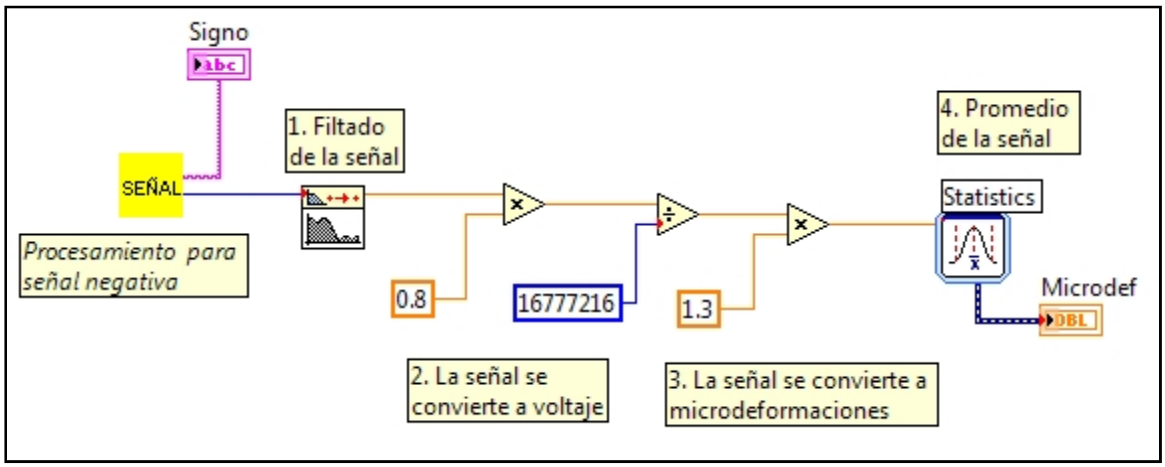



*Figura 3.10*

Por lo tanto, la formula que se ocupo para convertir los datos provenientes del convertidor A/D a microdeformaciones, es la siguiente:

$$uD = \frac{DATx0.8}{2^{24}} x\,1.3$$

En este SubVI, antes de hacer la conversión de la señal, se utiliza un filtro digital pasa bajas que ayuda a limpiar el ruido proveniente de la tarjeta de adquisición de datos. Después en el punto número 2 se hace la conversión de la señal a su equivalente en voltaje y en punto 3 se multiplica por un factor de 1.3 con lo cual se obtienen muestras muy similares a los aparatos comerciales utilizados para medir microdeformaciones. En el punto 4 se utiliza un icono llamado *Statistics* con el cual se promedia la señal con el fin de eliminar el ruido.

# CAPÍTULO 4

## Implementación y pruebas del sistema

En este capítulo se presenta el diseño y desarrollo de los circuitos impresos utilizados para el sistema. También se presentan las etapas en la que se llego al diseño final de la tarjeta de adquisición de datos, así como las pruebas que se realizaron al sistema desde sus primeras versiones.

### 4.1 Adaptación del convertidor MCP3901

Por las características deseadas para el desarrollo de este proyecto se ha decidido utilizar el convertidor Analógico Digital MCP3901 de Microchip. La reciente salida al mercado del convertidor MCP3901 hace que la mayoría de los programas para la elaboración de circuitos en PCB (*Printed Circuit Board)* no tengan entre sus librerías una adecuada para el tipo de encapsulado de este convertidor. Por esa razón se diseño la librería de acuerdo a las especificaciones del fabricante, para ello se utilizo el software para circuitos impresos EAGLE (*Easily Applicable Graphical Layout Editor*). Teniendo el diseño de la librería específica para el convertidor, se procedió a elaborar la tableta fenólica y al estañado de los pines a la tableta como se muestra en las dos siguientes figuras.

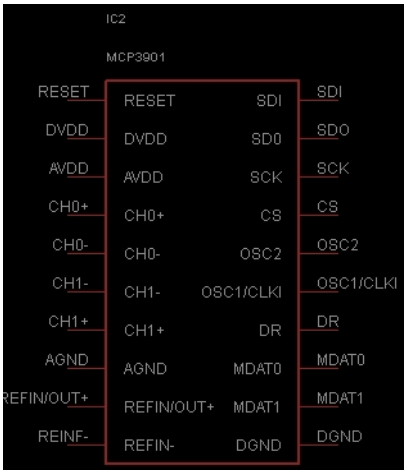



*Figura 4.1 Diseño del símbolo en el software EAGLE*

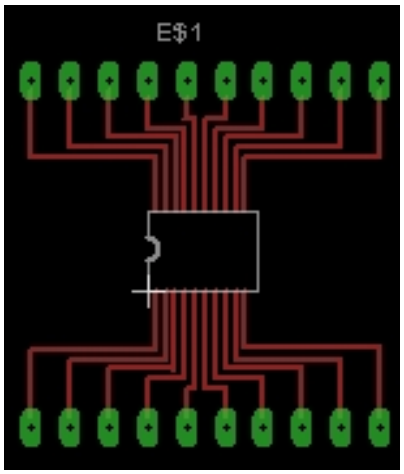



*Figura 4.2 Diseño final de la librería para la implementación del convertidor*

Ya que el fabricante solo cuenta con este convertidor con empaquetado SSOP de montaje superficial (*Plastic Shrink Small Outline*) se diseño y se diseño una tableta fenólica con las medias necesarias para la colocación y soldado de este convertidor.

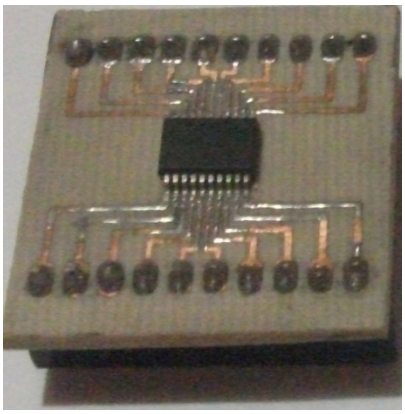


*Figura 4.3 Tableta fenólica diseñada e implementación con el convertidor MCP3901*

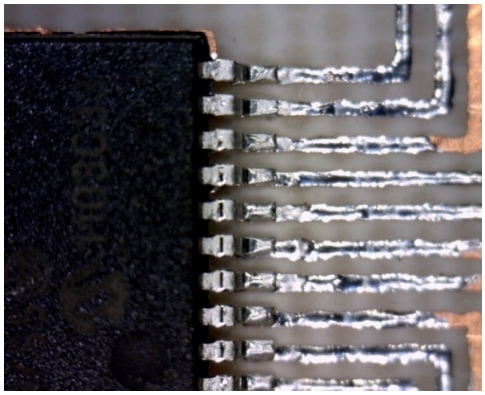


*Figura 4.4 Estañado de los pines a la tableta fenólica*

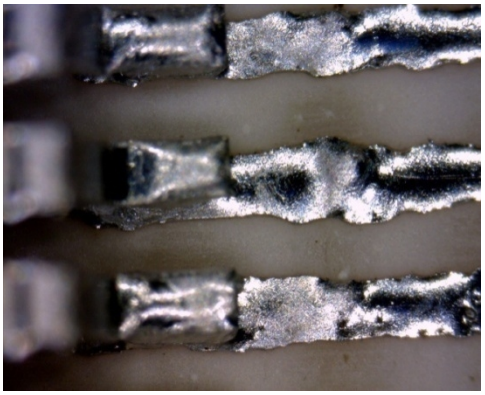


*Figura 4.5 Estañado de los pines a la tableta fenólica (Imagen tomada con microscopio)*

## 4.2 Pruebas de funcionamiento del convertidor MCP3901

Teniendo al convertidor adaptado a la tableta fenólica, se diseño una prueba en la que se demostraría que las configuraciones de software y hardware son las correctas para hacer funcionar a dicho convertidor.

Esta prueba se realizó de la manera más sencilla posible, ya que el objetivo principal es observar el funcionamiento del convertidor. Para esta prueba se tomaron las siguientes consideraciones:

- Programación de resolución del convertidor de 16 bits
- Las señales analógicas a convertir en digitales provienen de una fuente de alimentación controlada por un potenciómetro para no sobrepasar el voltaje máximo de operación
- La transmisión de datos vía RS-232, con visualización en display LCD

Las siguientes figuras muestran los prototipos para la realización de las pruebas:

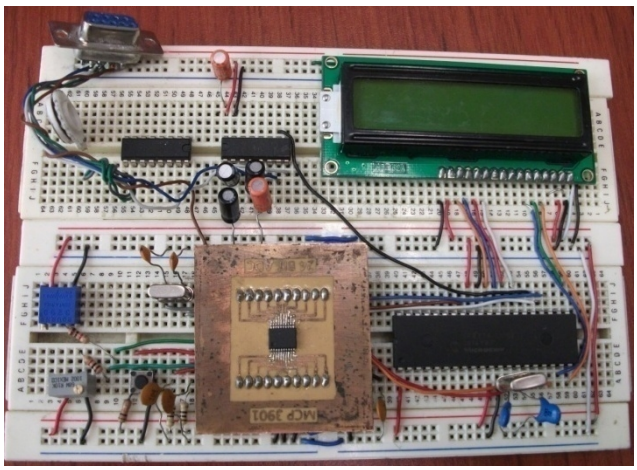


*Figura 4.6 Circuito realizado en protoboard*

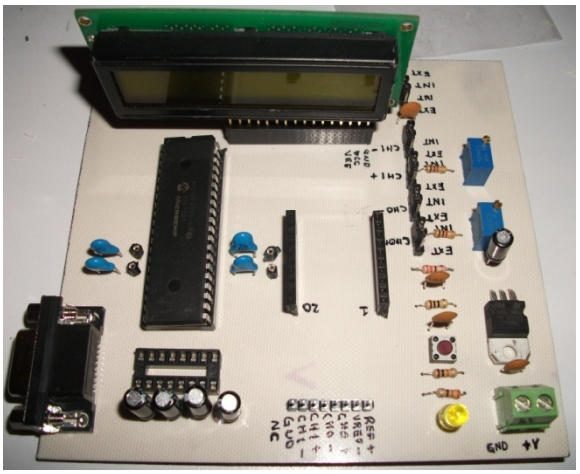


*Figura 4.7 Circuito realizado en tableta fenólica*

### 4.3 Resultados de la prueba de funcionamiento.

Para comprobar que el convertidor se encuentra funcionando se comparo la lectura mostrada por el display LCD con un multímetro digital de alta resolución marca HP.

Los primeros resultados nos permitieron comprobar que las configuraciones iníciales para la utilización del convertidor, fueron las correctas para hacerlo funcionar. Las siguientes figuras muestran la comparación entre ambas lecturas en los dos canales que posee el convertidor.

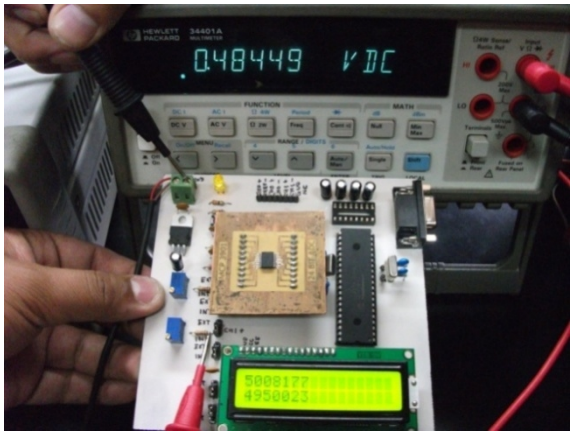



*Figura 4.8 Lectura del Canal 0 y comparativa entre lecturas*

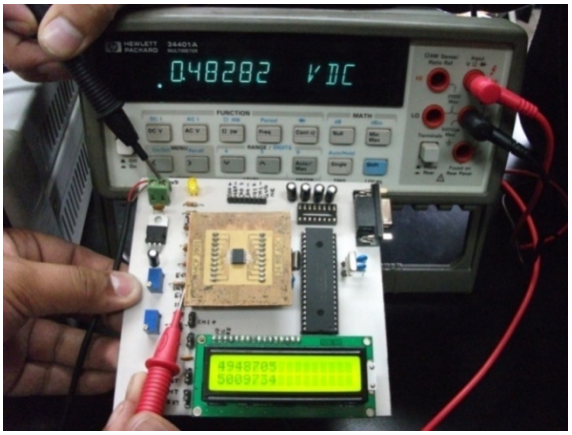



*Figura 4.9 Lectura del Canal 1 y comparativa entre lecturas*

De las pruebas que se hicieron con las tarjetas de la figura 4.8 y 4.9 se obtuvieron los siguientes resultados:

| Canal | Lectura multimetro HP | Lectura convertidor MCP3901 |
|---|---|---|
| Canal 1 | 0.48449 Volts | 0.48282 Volts |
| Canal 2 | 0.4950023 Volts | 0.4948705 Volts |

Con las lecturas anteriores se pudo comprobar que las muestras tomadas del convertidor son muy similares a las lecturas obtenidas con el multímetro HP, de esta manera se comprobó que el convertidor funciona correctamente con las configuraciones. La figura 4.10 muestra el diagrama esquemático con el que se construyeron los circuitos para las primeras pruebas.

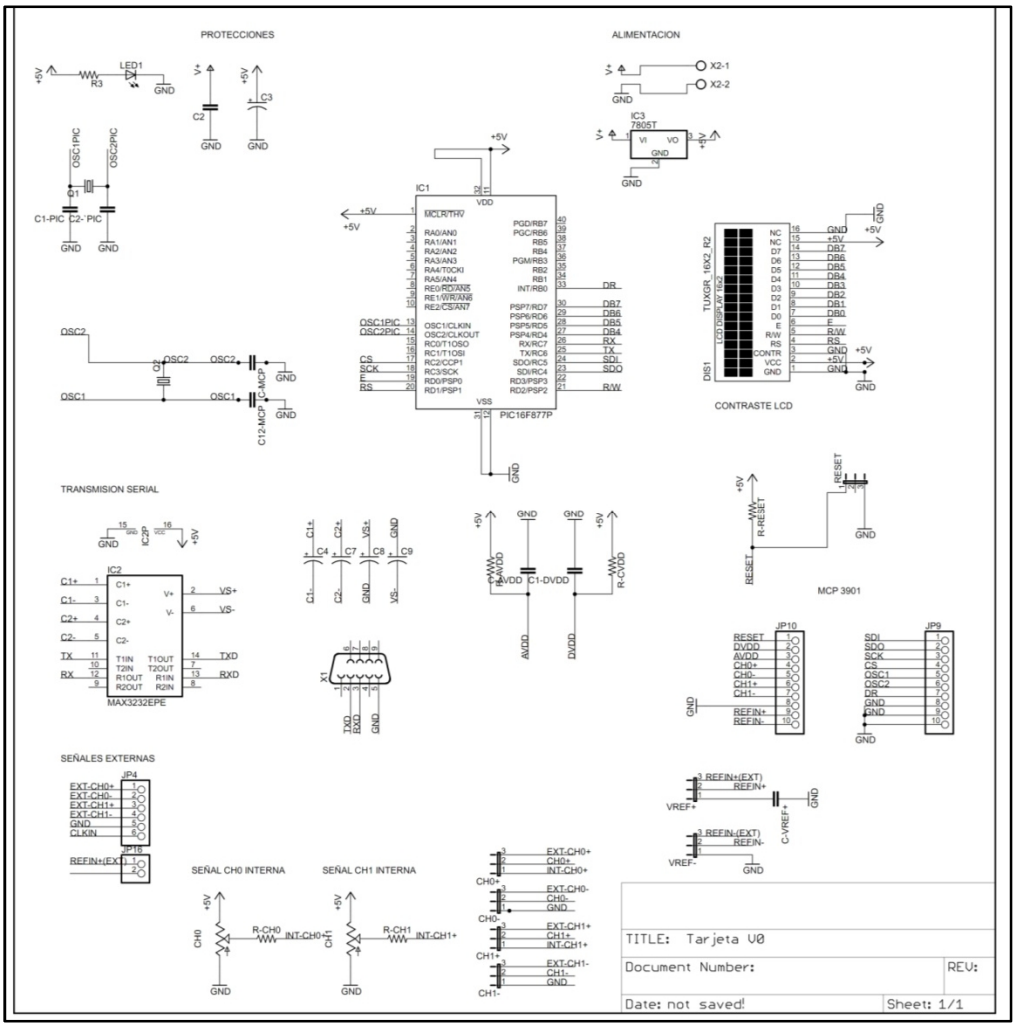



*Figura 4.10 Diagrama esquemático realizado en el programa EAGLE para la elaboración de las primeras pruebas con el convertidor MCP3901*

## 4.4 Arreglo en puente completo y acondicionamiento de la señal

La instrumentación con las galgas extensométricas se realizo en diafragmas de acero como lo muestra la figura 4.11.a en un arreglo de puente de Wheatstone en configuración puente completo.

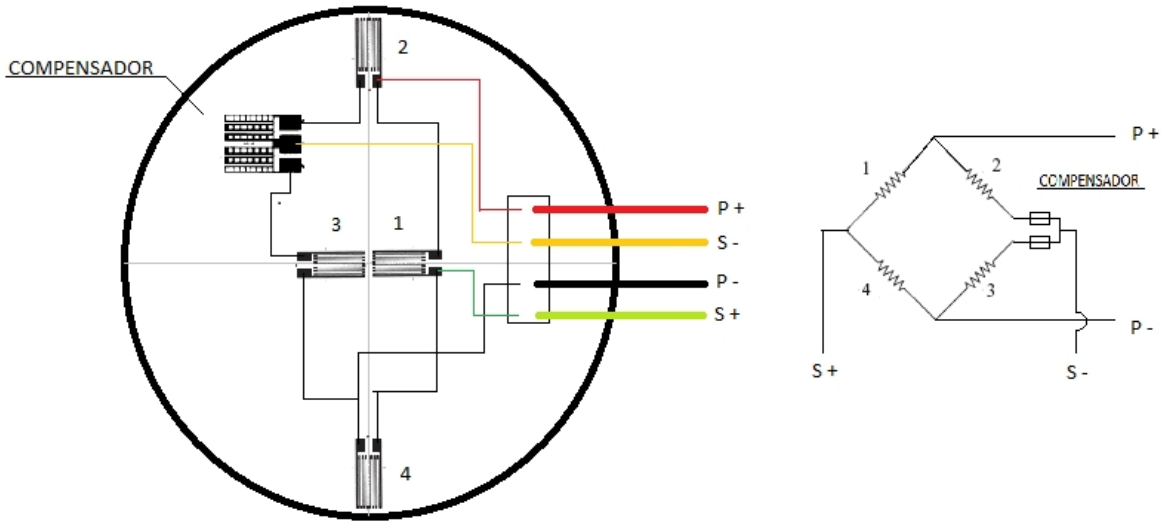



*Figura 4.11.a Diafragma instrumentado con galgas extensométricas en arreglo de puente completo*

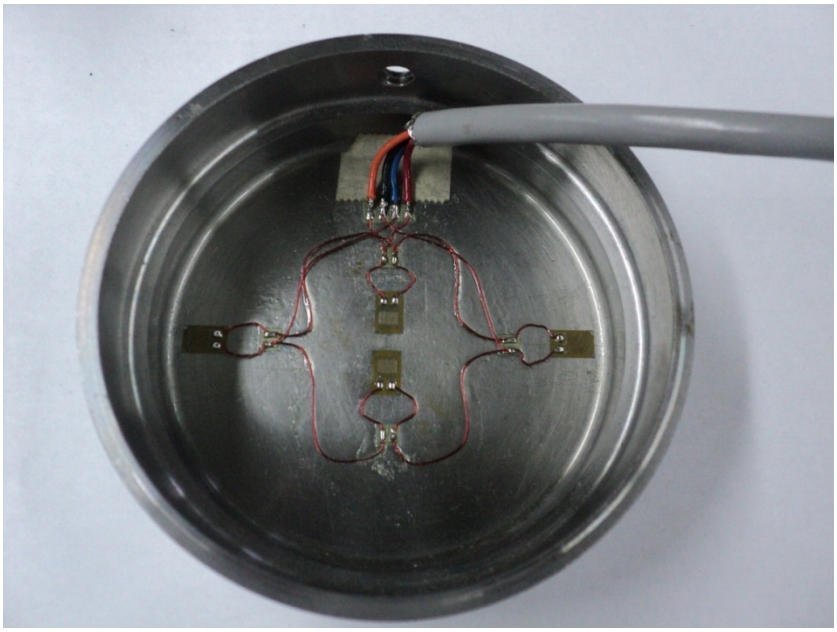


*Figura 4.11.b Diafragma instrumentado con galgas extensométricas en arreglo de puente completo diseñado en el laboratorio de Mecánica Aplicada*

El diafragma instrumentado se alimento con un voltaje de 2 V y se comprobó que el cambio mínimo de cualquier galga extensométricas provoca una diferencia de potencial (salida diferencial) en escala mínima. Para acondicionar la señal tan pequeña se colocaron los amplificadores de instrumentación AD620 y se programo la ganancia del convertidor A/D para que la amplitud de la señal fuera la necesaria para tomar lecturas. Las figuras 4.12 y 4.13 muestran la señal diferencial amplificada con una ganancia de 100 y de 1000 respectivamente.


Figura 4.12


Figura 4.13

Otra prueba de funcionamiento que se hizo, fue la de realizar un programa en LabVIEW que mostrar la forma de onda de la señal y los datos convertidos a voltajes. En la figura 4.14 se hace la comparación de la señal utilizando un multímetro FLUKE y el programa en LabVIEW.


Figura 4.14

Tomando como referencia las pruebas anteriores, se diseño una nueva tarjeta de adquisición que a diferencia de la primera, esta integra las siguientes características:

- Capacidad de leer 4 canales con arreglo de puente completo
- Amplificación programable de las señales diferenciales de cada uno de los canales.
- Transmisión de datos vía protocolo RS-232.
- Fuente de alimentación variable para tener 2 volts necesarios para alimentar a cada uno de los diafragmas.

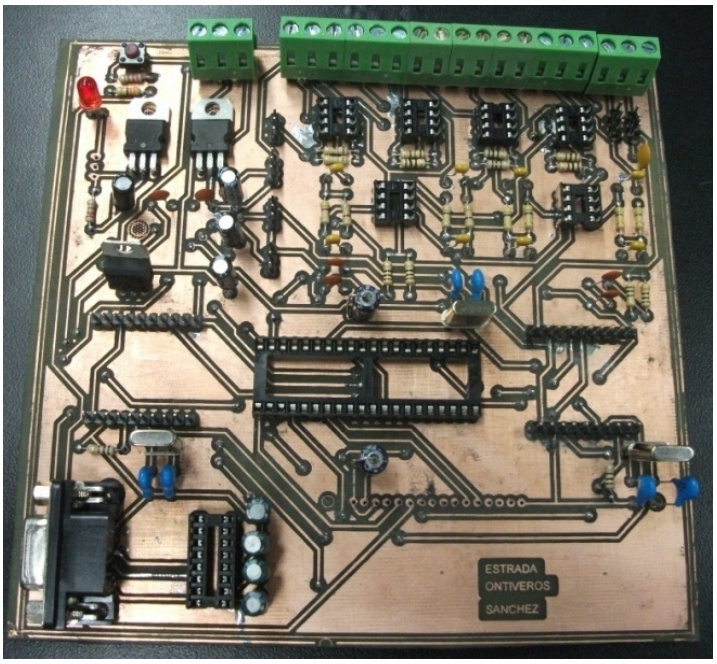



*Figura 4.15 Tarjeta diseñada con capacidad de lectura de 4 canales, amplificadores de instrumentación con ganancia fija de 100, transmisión de datos via RS-232 y visualización de datos por medio de display LCD*

## 4.5. Resultados obtenidos con el primer diseño del sistema

Los resultados obtenidos con el primer diseño de la tarjeta fueron satisfactorios en cuanto a las mediciones obtenidas.

De las características de diseño de la primera tarjeta se observó que había algunas que se podían mejorar y así mismo mejorarían el rendimiento del sistema de adquisición de datos. De entre las características que se podían mejorar de este primer diseño podemos mencionar las siguientes:

*De la fuente de alimentación:* En este diseño, la fuente de alimentación era una fuente externa bipolar que proporcionaba los valores de ±5 volts. Para eliminar la necesidad de esa fuente se agrego al diseño el circuito integrado TC7660 que convierte una entrada positiva a negativa, ideal para alimentar a los amplificadores de instrumentación. El sistema cuenta con dos maneras de alimentación; la primera será por medio del conector USB de la PC y la otra será por una fuente externa de +5 volts.

*De la ganancia programable del AD620:* La ganancia del amplificador de instrumentación se hace con una resistencia de precisión, en este diseño la ganancia es fija. Para que el sistema tenga la capacidad de seleccionar la ganancia del amplificador de instrumentación se cambio la resistencia fija por un potenciómetro digital de precisión que varia la resistencia por software y lo hace ideal para esta aplicación.

*De la transmisión de datos:* en este diseño la transmisión de datos es vía protocolo RS-232. Muchas de las computadoras actuales carecen del puerto serial para establecer este tipo de comunicación, es por eso que se ha decidido hacer la comunicación con la PC por medio de la emulación USB-Serial. Este tipo de transmisión de datos será activado cuando el sistema sea alimentado por el conector USB de la computadora.

*De la visualización de la información.* En este diseño los datos son enviados a la computadora y mostrados en una grafica con el software LabVIEW. Para hacer que el sistema sea funcional sin la necesidad de una PC, se agrego un display LCD que muestra la deformación de cada canal. Debido al consumo de corriente del display LCD, este solo será disponible cuando el sistema sea alimentado por una fuente externa de voltaje.

## 4.6 Diseño final del sistema

Con base en pruebas realizadas anteriormente se definieron las características con las que debe contar el sistema final de adquisición, que entre otras son las siguientes:

- Dos modos de alimentación al sistema
- Capacidad de lectura de 4 canales al mismo tiempo
- Resolución del convertidor Analógico-Digital programable vía software
- Ganancia del amplificador de instrumentación programable via software
- Transmisión de datos vía emulación USB-Serial
- Capacidad de tomar más de 50 muestras por segundo
- Visualización de datos por medio de un display LCD
- Procesamiento de la información por medio del entorno grafico LabVIEW.
- Capacidad de almacenar la información en una archivo de texto para posterior revisión de datos

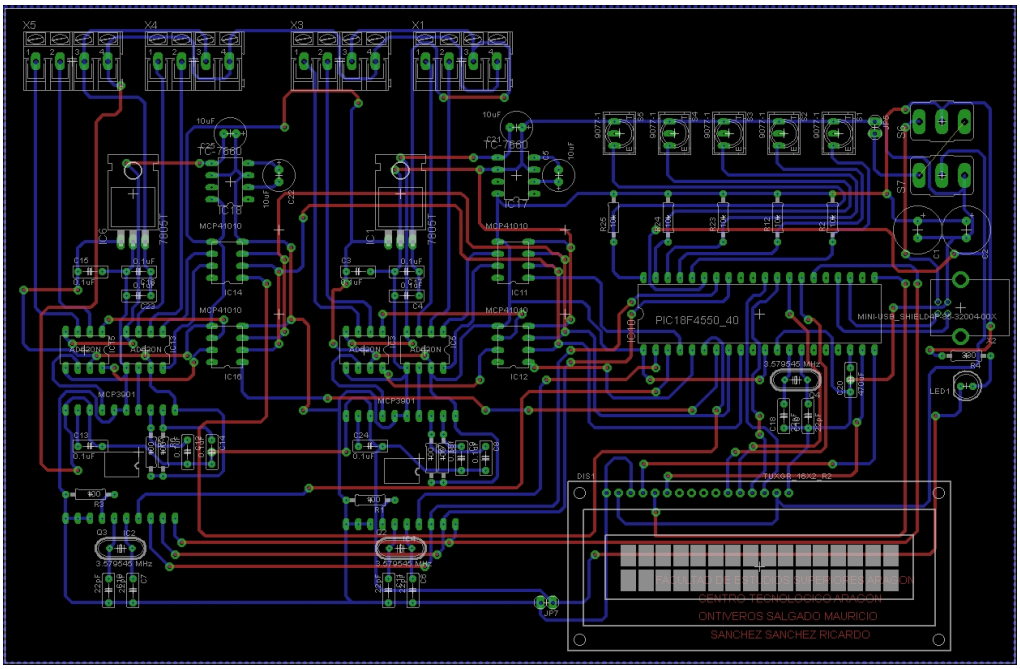



*Figura 4.16 Diseño final de la tarjeta con el software EAGLE. Vistas TOP y BOTTOM*

Las dos figuras anteriores muestran el diseño final del sistema con todas las características requeridas implementadas. De esta manera se tienen todas las características necesarias para hacer pruebas con el sistema final.

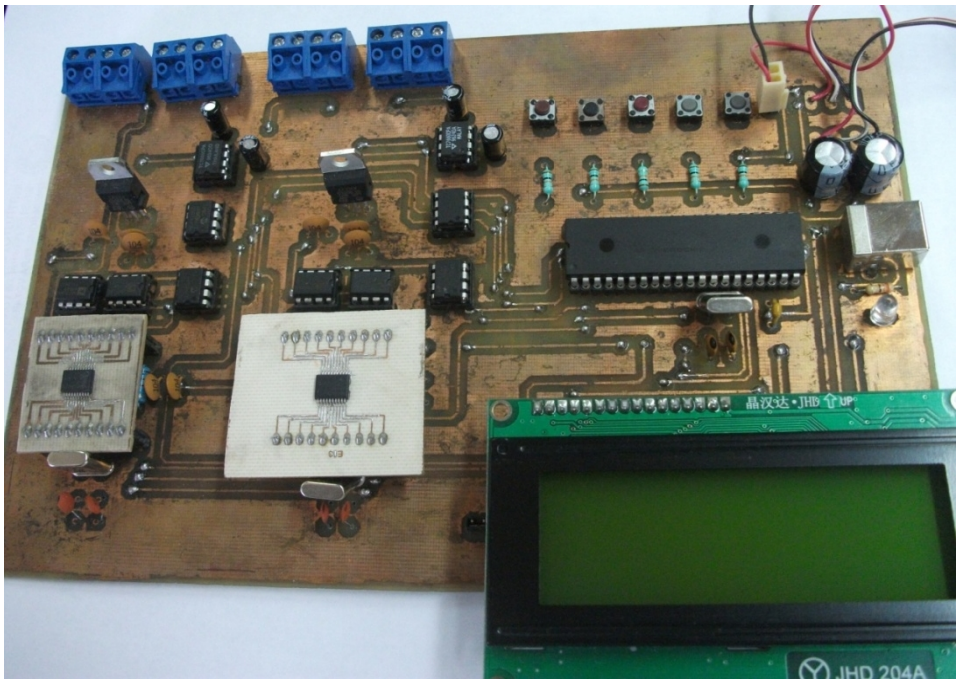



*Figura 4.17 Diseño final del sistema de adquisición de datos final con visualización de datos por medio del display LCD y transmisión de datos vía USB*

Las dos figuras anteriores muestran el diseño final del sistema con todas las características requeridas implementadas. De esta manera se realizaron las pruebas con el sistema final.

### 4.7 Pruebas al sistema

La operación del sistema se realizo con éxito y se compararon los resultados con el equipo comercial con el que se cuenta en el laboratorio de Mecánica Aplicada, a continuación se describen las imágenes 4.18 y 4.19 respectivamente.


*Figura 4.18*

En la figura 4.18 el sistema está operando con una fuente de alimentación externa, y los datos son visualizados por medio de un display LCD, la resolución del convertidor es de 24 bits y la velocidad es de 50 muestras por segundo. Esta imagen fue tomada cuando se media una barra de aluminio instrumentada con strain gages en arreglo de puente completo  por el canal 0 obteniendo la lectura: 1572 microdeformaciones.

En la figura 4.19 se muestra el sistema comercial de la marca *Vishay-Measurements P3* tomando lecturas de una barra de aluminio instrumentada con strain gages en arreglo de puente completo y visualización de datos por medio de un display LCD. En esta prueba se obtuvo la lectura de 1537 microdeformaciones.

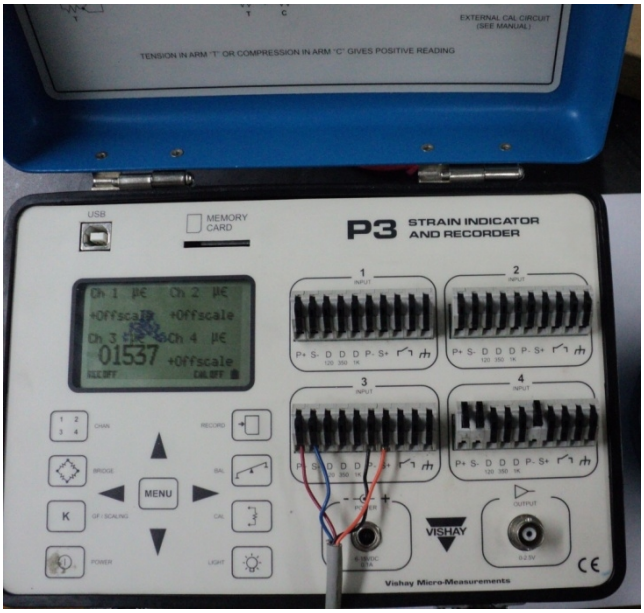



*Figura 4.19*

La lectura obtenida en el sistema digital de alta resolución y el sistema comercial *"P3"* son muy similares por lo que se pude asegurar que nuestro sistema trabaja de forma eficiente y precisa.

La siguiente prueba al sistema se hizo con el modo de operación USB alimentando al sistema con el voltaje de la Laptop (+5V). En esta prueba se visualizan los resultados de forma grafica.

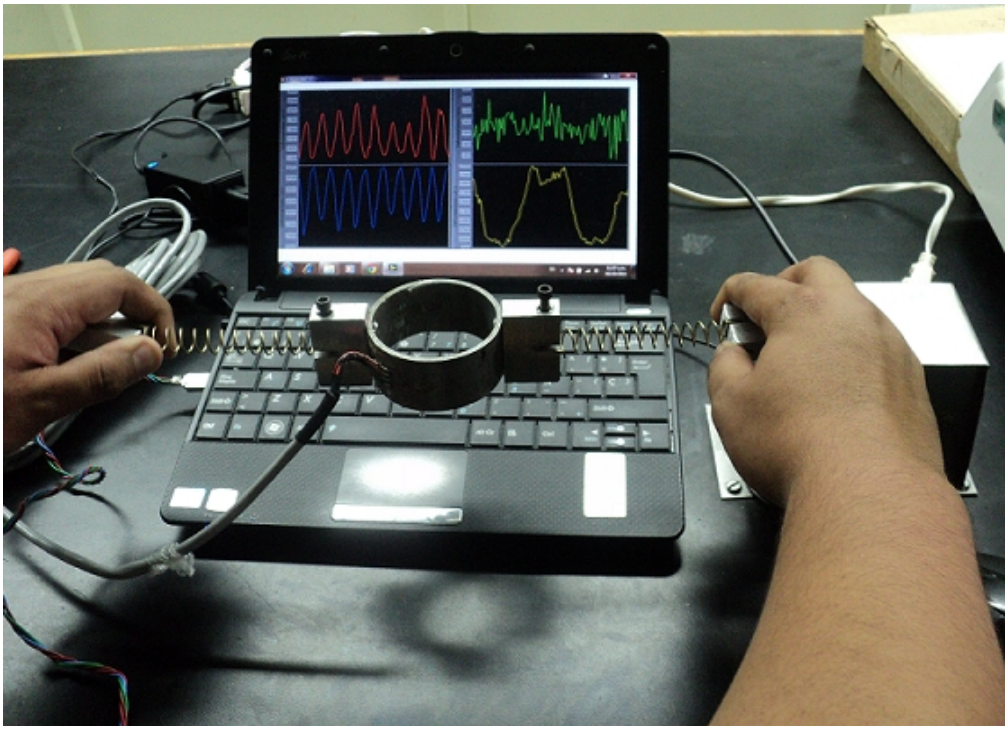


*Figura 4.20*

La figura 4.20, muestra el software desarrollado para el procesamiento de datos. Para este caso, cuando se le aplica una fuerza al resorte (en expansión) las galgas instrumentadas en puente completo mandan la señal a la tarjeta de adquisición de datos. La señal que corresponde al resorte es la amarilla, para las otras tres se conectaron: un diafragma y dos barras de aluminio de diferente tamaño por lo que puede apreciar cómo es que varían la graficas en cada esfuerzo aplicado a las barras, diafragma y resorte.

Con estas dos pruebas podemos concluir que el sistema es capaz de tomar, convertir, trasmitir y procesar las microdeformaciones para que estas sean representadas de manera grafica y puedan ser interpretadas por los ingenieros mecánicos para su análisis.

### 4.8 Comparación de lecturas con el equipo comercial P3

Las siguientes pruebas se hicieron en el laboratorio de Mecánica Aplicada. Se hizo la medición de las deformaciones de una barra de aluminio instrumentada en puente completo con el equipo comercial *P3 Strain Indicator and Recorder* y el sistema de alta resolución.

Durante la medición, la barra nunca fue sometida a ningún esfuerzo de compresión o deflexión sino que se dejo en total reposo para poder observar el ruido que se introduce en los canales de entrada. A continuación se hace una comparación entre los resultados obtenidos del equipo P3 y el sistema de alta resolución.

| P3 Strain Indicator and Recorder | | Sistema Digital de alta resolución | |
|---|---|---|---|
| Model P3 Strain Indicator and Recorder | | High Resolution System | |
| | | Operator Mauricio/Ricardo | |
| | | Date | 22/02/2012 |
| | | Time | 16:43:39,0625 |
| | | Channels 1 | |
| | | Samples | 1 |
| Date/Time | Ch 2 ue | Time | CH0 |
| 22/02/2011 04:56:44 p.m. | -00000 | 6.000480 | 0.468621 |
| 22/02/2011 04:56:45 p.m. | -00000 | 6.020520 | 0.858510 |
| 22/02/2011 04:56:46 p.m. | +00000 | 6.040635 | 1.645215 |
| 22/02/2011 04:56:47 p.m. | -00000 | 6.060778 | 0.889412 |
| 22/02/2011 04:56:48 p.m. | -00000 | 6.080812 | 0.064585 |
| 22/02/2011 04:56:49 p.m. | +00000 | 6.100952 | -0.190781 |
| 22/02/2011 04:56:50 p.m. | -00000 | 6.121020 | -0.259087 |
| 22/02/2011 04:56:51 p.m. | -00000 | 6.141096 | 0.851233 |
| 22/02/2011 04:56:52 p.m. | -00000 | 6.161153 | 1.254896 |
| 22/02/2011 04:56:53 p.m. | -0000.1 | 6.181209 | 2.047777 |
| 22/02/2011 04:56:54 p.m. | +000.1 | 6.201280 | 1.278514 |
| 22/02/2011 04:56:55 p.m. | -00000 | 6.221340 | 2.856541 |
| 22/02/2011 04:56:56 p.m. | -00000 | 6.241405 | 1.508667 |
| 22/02/2011 04:56:57 p.m. | -00000 | 6.261473 | 1.631713 |
| 22/02/2011 04:56:58 p.m. | -00000 | 6.281523 | 0.954115 |
| 22/02/2011 04:56:59 p.m. | -00000 | 6.301637 | -1.598614 |
| 22/02/2011 04:57:00 p.m. | -00000 | 6.321771 | -2.659044 |
| 22/02/2011 04:57:01 p.m. | -00000 | 6.341819 | -2.407990 |
| 22/02/2011 04:57:02 p.m. | -00000 | 6.361958 | -0.298749 |
| 22/02/2011 04:57:03 p.m. | -00000 | 6.382020 | -1.099546 |

En esta comparación se capturan solamente 20 muestras por cada equipo. Como puede observarse el P3 es un instrumento electrónico muy preciso ya que durante la medición prácticamente se mantuvo en cero deformaciones. El sistema de alta

resolución es menos preciso que el P3 pero más rápido en la captura de datos. El P3 obtiene una muestra por segundo mientras que el sistema de alta resolución captura aproximadamente 50 muestras por segundo. El obtener mayor número de muestras, permite al usuario observar lo que sucede durante un segundo, ya que en este tiempo podrían ocurrir muchas deformaciones.

En el sistema de alta resolución, cuando se activa un canal se obtiene 50 sps, cuando se activan dos canales se obtienen 42 sps, para tres canales se capturan 38 sps y para cuatro canales 34 sps.

### 4.9 Inicialización del sistema

Cuando el usuario selecciona el modo de visualizar los datos por medio de la PC, este debe primero abrir el software que permite establecer la comunicación con la tarjeta de adquisición de datos. Al abrirlo, se muestra una interface que permite al usuario configurar la comunicación de datos (figura 4.21).

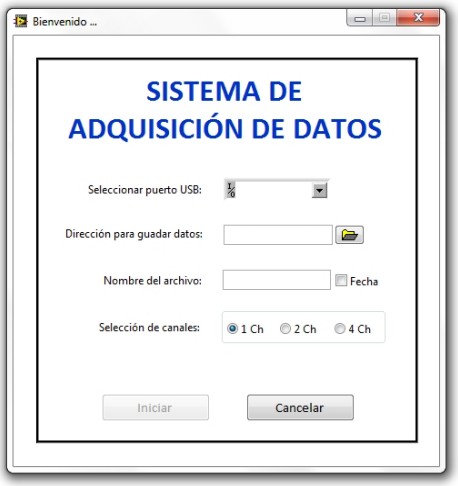



*Figura 4.21*

Por medio de la interface, el usuario puede seleccionar el puerto virtual, la ruta y nombre en donde se guardaran los datos y el número de canales con los que se desea trabajar.

El prototipo del chasis en donde estará la tarjeta de adquisición de datos se encuentra en desarrollo en el Laboratorio de Mecánica Aplicada del Centro Tecnológico Aragón por el Ing. Ángel Romero. El prototipo se muestra en las siguientes tres figuras:

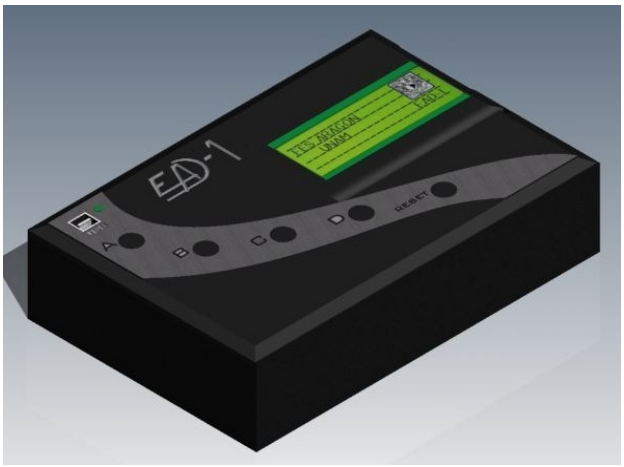



Figura 4.22


Figura 4.23


Figura 4.24

Como se puede observar, la caja tiene las 4 entradas para los canales, los 5 botones para la seleccione de canal y un reset. Para la posición de la LCD se podrá colocar de forma horizontal o a 45 grados según lo requiera el usuario.

La caja también tiene su entrada para la comunicación USB con una computadora, así como las ranuras para la alimentación por medio de un voltaje externo.

# Conclusiones

El sistema de alta resolución diseñado en el laboratorio de Mecánica Aplicada es una buena propuesta para realizar mediciones en la industria, ya que a diferencia del *P3 Strain Indicator and Recorder*, el sistema digital captura mas muestras por segundo. (50sps aproximadamente).

Esta tarjeta de adquisición no es tan precisa como los sistemas comerciales, pero si más rápida y mucho más económica comparada con el *P3 Strain Indicator and Recorder*.

Actualmente se está trabajando para construir un sistema más preciso con la implementación de filtros analógicos. También se está diseñando un nuevo sistema que permita al usuario la selección del tipo de arreglo de strain gages (puente completo, medio puente y un cuarto de puente) que necesite por cada canal.

Consideramos que el avance de esta tesis puede ser útil para las futuras generaciones de alumnos en el área de Electrónica ya que contiene las bases fundamentales para el diseño de una tarjeta de adquisición de datos con una velocidad y resolución bastante optimas para la medición de microdeformaciones.

Durante el desarrollo de este proyecto y con la ayuda de los profesores, Dr. Jacinto Cortes Pérez y el M. en C. Arturo Ocampo Álvarez, se tuvo la oportunidad de presentar este trabajo en el segundo Congreso Internacional de Instrumentación y Ciencias Aplicadas (ICIAS, *International Congress on Instrumentation and Apliedd Sciences*) que se realizo en la ciudad de Puebla en octubre de 2011. El cartel presentado en el congreso se puede localizar en el anexo 2.

# ANEXO 1

## CÓDIGO DEL MICROCONTROLADOR

```
//**********************Configuración del microcontrolador**********************
#include <18f4550.h>
#fuses HSPLL, PLL5, USBDIV, NOWDT, PROTECT, NOLVP, NODEBUG, VREGEN,
#use delay(clock=20000000)
#include <potenciometro.c>
#include <usb_cdc.h>
#include <lcd4x20.c>
#use standard_io(a)
#use standard_io(b)
#use standard_io(c)
#use standard_io(d)
#define CS   PIN_A0
#define CSB   PIN_A1
#define CS0   PIN_A2
#define CS1   PIN_A3
#define CS2   PIN_A4
#define CS3   PIN_A5
//***************************Declaración de variables************************
int8 i;
signed int32 byte5, byte4, byte3, byte2, byte1, byte0;
signed int32 byteB5, byteB4, byteB3, byteB2, byteB1, byteB0;
signed int32 dato0, dato1, dato2, dato3;
float voltaje0, voltaje1, voltaje2, voltaje3;
```

```
signed int32 deformacion0, deformacion1, deformacion2, deformacion3;
float m0=0, m1=0, m2=0, m3=0;
float p0=0, p1=0, p2=0, p3=0;
int16 ciclo;
//******************Función para un canal***********************************
ch_0()
{
 while (true)
 {
      for(ciclo=0;ciclo<=49;ciclo++)
      {
        output_low(CS);    //Inicio la comunicación con CS' en bajo
        spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
        byte2= spi_read(00000000);     //Leo el byte más significativo de CH0
        byte1= spi_read(00000000);     //Leo el byte intermediode CH0
        byte0= spi_read(00000000);     //Leo el byte menos significat  de CH0
        output_high(CS);   //Finalizo la comunicación
         for(i=0;i<=15;i++)
         {
          rotate_left(&byte2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH0
         }
        dato0=(byte2+byte1+byte0);             //Sumo los 3 bytes de la conversión de CH0
        voltaje0=(dato0*0.8)/16777215;       // Paso a voltaje
        voltaje0=voltaje1*1000;                //Multiplico X 1000 para eliminar el punto
        m0=voltaje0+m0;                        //Se suma 50 veces las muestras...
```

```
      }
          p0=m0/50;                              //y se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH0= %ld",p0);
       delay_ms(200);
 }
}
//******************Función para dos canales*********************************
ch_1()
{
 while (true)
 {
     for(ciclo=0;ciclo<=49;ciclo++)
     {
       output_low(CS);    //Inicio la comunicación con CS' en bajo
       spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
       byte5= spi_read(00000000);     //Leo el byte más significativo de CH0
       byte4= spi_read(00000000);     //Leo el byte intermedio de CH0
       byte3= spi_read(00000000);     //Leo el byte menos significativo de CH0
       byte2= spi_read(00000000);     //Leo el byte más significativo de CH1
       byte1= spi_read(00000000);     //Leo el byte intermedio de CH1
       byte0= spi_read(00000000);     //Leo el byte menos significativo de CH1
       output_high(CS);   //Finalizo la comunicación
        for(i=0;i<=15;i++)
        {
         rotate_left(&byte5,3); //Recorro a la izq el byte mas sig 15 posiciones de CH0
```

```
          rotate_left(&byte2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH1
         }
        dato0=(byte5+byte4+byte3);          //Sumo los 3 bytes de la conversión de CH0
        voltaje0=(dato0*0.8)/16777215;      // Paso a voltaje
        voltaje0=voltaje0*1000;             //Multiplico X 1000 para eliminar el punto
        m0=voltaje0+m0;                     //Se suma 50 veces las muestras...

        dato1=(byte2+byte1+byte0);          //Sumo los 3 bytes de la conversión de CH1
        voltaje1=(dato1*0.8)/16777215;      // Paso a voltaje
        voltaje1=voltaje1*1000;             //Multiplico X 1000 para eliminar el punto
        m1=voltaje1+m1;                     //Se suma 50 veces las muestras...
      }
       P0=m0/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH0= %ld",p0);

       P1=m1/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH1= %ld",p1);
       delay_ms(200);
 }
}
//******************Función para tres canales*********************************
ch_2()
{
 while (true)
 {
```

```
for(ciclo=0;ciclo<=49;ciclo++)
{
  //**********************Se lee el ADC_0*****************************
  output_low(CS);    //Inicio la comunicación con CS' en bajo
  spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
  byte5= spi_read(00000000);     //Leo el byte más significativo de CH0
  byte4= spi_read(00000000);     //Leo el byte intermedio de CH0
  byte3= spi_read(00000000);     //Leo el byte menos significativo de CH0
  byte2= spi_read(00000000);     //Leo el byte más significativo de CH1
  byte1= spi_read(00000000);     //Leo el byte intermedio de CH1
  byte0= spi_read(00000000);     //Leo el byte menos significativo de CH1
  output_high(CS);   //Finalizo la comunicación del ADC_0
 //**********************Se lee el ADC_1*****************************
  output_low(CSB);    //Inicio la comunicación con CSB en bajo
  spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
  byteB2= spi_read(00000000);     //Leo el byte más significativo de CH0
  byteB1= spi_read(00000000);     //Leo el byte intermedio de CH0
  byteB0= spi_read(00000000);     //Leo el byte menos significativo de CH0
  output_high(CS);   //Finalizo la comunicación del ADC_1
   for(i=0;i<=15;i++)
   {
    rotate_left(&byte5,3); //Recorro a la izq el byte mas sig 15 posiciones de CH0
    rotate_left(&byte2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH1
    rotate_left(&byteB2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH2
   }
```

```
      dato0=(byte5+byte4+byte3);          //Sumo los 3 bytes de la conversión de CH0
      voltaje0=(dato0*0.8)/16777215;      // Paso a voltaje
      voltaje0=voltaje0*1000;             //Multiplico X 1000 para eliminar el punto
      m0=voltaje0+m0;                     //Se suma 50 veces las muestras...

      dato1=(byte2+byte1+byte0);          //Sumo los 3 bytes de la conversión de CH1
      voltaje1=(dato1*0.8)/16777215;      // Paso a voltaje
      voltaje1=voltaje1*1000;             //Multiplico X 1000 para eliminar el punto
      m1=voltaje1+m1;                     //Se suma 50 veces las muestras...

      dato2=(byteB2+byteB1+byteB0);       //Sumo los 3 bytes de la conversión de CH2
      voltaje2=(dato2*0.8)/16777215;      // Paso a voltaje
      voltaje2=voltaje2*1000;             //Multiplico X 1000 para eliminar el punto
      m2=voltaje2+m2;                     //Se suma 50 veces las muestras...
   }
    P0=m0/50;                             //Se divide entre 50 para sacar el promedio
    printf(lcd_putc, "\fCH0= %ld",p0);

    P1=m1/50;                             //Se divide entre 50 para sacar el promedio
    printf(lcd_putc, "\fCH1= %ld",p1);

    P2=m2/50;                             //Se divide entre 50 para sacar el promedio
    printf(lcd_putc, "\fCH2= %ld",p2);
    delay_ms(200);
}
```

```
}
//******************Función para cuatro canales*****************************
ch_3()
{
 while (true)
 {
      for(ciclo=0;ciclo<=49;ciclo++)
      {
        //************************Se lee el ADC_0*****************************
        output_low(CS);    //Inicio la comunicación con CS' en bajo
        spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
        byte5= spi_read(00000000);     //Leo el byte más significativo de CH0
        byte4= spi_read(00000000);     //Leo el byte intermedio de CH0
        byte3= spi_read(00000000);     //Leo el byte menos significativo de CH0
        byte2= spi_read(00000000);     //Leo el byte más significativo de CH1
        byte1= spi_read(00000000);     //Leo el byte intermedio de CH1
        byte0= spi_read(00000000);     //Leo el byte menos significativo de CH1
        output_high(CS);   //Finalizo la comunicación del ADC_0
       //************************Se lee el ADC_1*****************************
        output_low(CSB);    //Inicio la comunicación con CSB en bajo
        spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
        byteB5= spi_read(00000000);     //Leo el byte más significativo de CH2
        byteB4= spi_read(00000000);     //Leo el byte intermedio de CH2
        byteB3= spi_read(00000000);     //Leo el byte menos significativo de CH2
        byteB2= spi_read(00000000);     //Leo el byte más significativo de CH3
```

```
byteB1= spi_read(00000000);     //Leo el byte intermedio de CH3
byteB0= spi_read(00000000);     //Leo el byte menos significativo de CH3
output_high(CS);   //Finalizo la comunicación del ADC_1
 for(i=0;i<=15;i++)
 {
  rotate_left(&byte5,3); //Recorro a la izq el byte mas sig 15 posiciones de CH0
  rotate_left(&byte2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH1
  rotate_left(&byteB5,3); //Recorro a la izq el byte mas sig 15 posiciones de CH2
  rotate_left(&byteB2,3); //Recorro a la izq el byte mas sig 15 posiciones de CH3

 }
dato0=(byte5+byte4+byte3);            //Sumo los 3 bytes de la conversión de CH0
voltaje0=(dato0*0.8)/16777215;        // Paso a voltaje
voltaje0=voltaje0*1000;               //Multiplico X 1000 para eliminar el punto
m0=voltaje0+m0;                       //Se suma 50 veces las muestras...

dato1=(byte2+byte1+byte0);            //Sumo los 3 bytes de la conversión de CH1
voltaje1=(dato1*0.8)/16777215;        // Paso a voltaje
voltaje1=voltaje1*1000;               //Multiplico X 1000 para eliminar el punto
m1=voltaje1+m1;                       //Se suma 50 veces las muestras...

dato2=(byteB5+byteB4+byteB3);         //Sumo los 3 bytes de la conversión de CH2
voltaje2=(dato2*0.8)/16777215;        // Paso a voltaje
voltaje2=voltaje2*1000;               //Multiplico X 1000 para eliminar el punto
m2=voltaje2+m2;                       //Se suma 50 veces las muestras...
```

```
        dato3=(byteB2+byteB1+byteB0);       //Sumo los 3 bytes de la conversión de CH2
        voltaje3=(dato3*0.8)/16777215;      // Paso a voltaje
        voltaje3=voltaje3*1000;             //Multiplico X 1000 para eliminar el punto
        m3=voltaje3+m3;                     //Se suma 50 veces las muestras...
      }
       P0=m0/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH0= %ld",p0);

       P1=m1/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH1= %ld",p1);

       P2=m2/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH2= %ld",p2);

       P3=m3/50;                            //Se divide entre 50 para sacar el promedio
       printf(lcd_putc, "\fCH2= %ld",p2);
       delay_ms(200);

 }
}
//**********************Función para la LCD*********************************
void display()
{
//**********************Espera a que se seleccione un canal******************
```

```
 printf(lcd_putc,"Selecciona el numero");
 lcd_gotoxy(1,2);
 printf(lcd_putc,"de canales");
 lcd_gotoxy(1,3);
 printf(lcd_putc,"A= 1      C= 3");
 lcd_gotoxy(1,4);
 printf(lcd_putc,"B= 2      D= 4");
while(1)
{
if(input(PIN_B7)==0)
  {
   delay_ms(300);
   ch_0();
  }
if(input(PIN_B6)==0)
  {
   delay_ms(300);
   ch_1();
  }
if(input(PIN_B5)==0)
  {
   delay_ms(300);
   ch_2();
  }
if(input(PIN_B4)==0)
```

```
    {
     delay_ms(300);
     ch_3();
    }
  }
}
//**********************Función para comunicación USB*******************
void usb()
{
int8 byteusb2;
int8 byteusb1;
int8 byteusb0;
  while(1)
   {
     usb_task();
     if (usb_enumerated())            //Esta correctamente activado usb?
      {

       id=usb_cdc_getc();
       switch(id)
        {
//***************Se lee un canal*********************************************
            case 0:    //
            output_low(CS);                  //Inicio la comunicación con CS' en bajo
            spi_write(0b00000001); //Byte de control, selecciono la ADDR de DATA_CH0
```

```
      byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH0)
      usb_cdc_putc(byteusb2);
      byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH0)
      usb_cdc_putc(byteusb1);
      byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH0)
      usb_cdc_putc(byteusb0);
      output_high(CS);                  //Finalizo la comunicación
      delay_ms(50);
      break;
//***************Se leen dos canales**********************************
case 1:
      output_low(CS);               //Inicio la comunicación con CS' en bajo
      spi_write(0b00000001);   //Byte de control, selecciono la ADDR de DATA_CH0
      byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH0)
      usb_cdc_putc(byteusb2);
      byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH0)
      usb_cdc_putc(byteusb1);
      byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH0)
      usb_cdc_putc(byteusb0);
      byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH1)
      usb_cdc_putc(byteusb2);
      byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH1)
      usb_cdc_putc(byteusb1);
      byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH1)
      usb_cdc_putc(byteusb0);
```

```
            output_high(CS);                    //Finalizo la comunicación
            delay_ms(50);
            break;

        //***************Se leen tres canales***********************************
            case 2:
            output_low(CS);             //Inicio la comunicación con CS' en bajo
            spi_write(0b00000001);   //Byte de control, selecciono la ADDR de DATA_CH0
            byteusb2= spi_read(00000000);    //Leo el byte más significativo (CH0)
            usb_cdc_putc(byteusb2);
            byteusb1= spi_read(00000000);    //Leo el byte intermedio (CH0)
            usb_cdc_putc(byteusb1);
            byteusb0= spi_read(00000000);    //Leo el byte menos significativo (CH0)
            usb_cdc_putc(byteusb0);
            byteusb2= spi_read(00000000);    //Leo el byte más significativo (CH1)
            usb_cdc_putc(byteusb2);
            byteusb1= spi_read(00000000);    //Leo el byte intermedio (CH1)
            usb_cdc_putc(byteusb1);
            byteusb0= spi_read(00000000);    //Leo el byte menos significativo (CH1)
            usb_cdc_putc(byteusb0);
            output_high(CS);                    //Finalizo la comunicación
            delay_ms(50);
    //**********************Se lee ADC1**********************************
            output_low(CSB);             //Inicio la comunicación con CS' en bajo
            spi_write(0b00000001);   //Byte de control, selecciono la ADDR de DATA_CH2
```

```
byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH2)
usb_cdc_putc(byteusb2);
byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH2)
usb_cdc_putc(byteusb1);
byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH2)
usb_cdc_putc(byteusb0);
output_high(CSB);                 //Finalizo la comunicación
break;


//***************Se leen cuatro canales*****************************
 case 3:
output_low(CS);               //Inicio la comunicación con CS' en bajo
spi_write(0b00000001);   //Byte de control, selecciono la ADDR de DATA_CH0
byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH0)
usb_cdc_putc(byteusb2);
byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH0)
usb_cdc_putc(byteusb1);
byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH0)
usb_cdc_putc(byteusb0);
byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH1)
usb_cdc_putc(byteusb2);
byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH1)
usb_cdc_putc(byteusb1);
byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH1)
usb_cdc_putc(byteusb0);
```

```
            output_high(CS);                    //Finalizo la comunicación
            delay_ms(50);
//**********************Se lee ADC1**********************************
            output_low(CSB);          //Inicio la comunicación con CS' en bajo
            spi_write(0b00000001);   //Byte de control, selecciono la ADDR de DATA_CH2
            byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH2)
            usb_cdc_putc(byteusb2);
            byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH2)
            usb_cdc_putc(byteusb1);
            byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH2)
            usb_cdc_putc(byteusb0);
           byteusb2= spi_read(00000000);     //Leo el byte más significativo (CH3)
            usb_cdc_putc(byteusb2);
            byteusb1= spi_read(00000000);     //Leo el byte intermedio (CH3)
            usb_cdc_putc(byteusb1);
            byteusb0= spi_read(00000000);     //Leo el byte menos significativo (CH3)
            usb_cdc_putc(byteusb0);
            output_high(CSB);                   //Finalizo la comunicación
            break;


        }//end switch
      }//end usb_enumerated?
   }//end while
}//end void usb
```

```
//*********************************Función principal*************************
void main()
 {
setup_spi(SPI_MASTER|SPI_H_TO_L|SPI_XMIT_L_TO_H|SPI_CLK_DIV_4);
  lcd_init();            //Inicializa la LCD
  usb_cdc_init();    //Configura puerto virtual
  usb_init();            //Inicializa en modo USB

 //***************************Programación de los convertidores***************
//************************Reset en el ADC_0*********************************
output_low(CS);                  //Inicio la comunicación con CS' en bajo
spi_write(0b00010110);        //Dirección del CONFIG2 (0x0B)
spi_write(0b11000000);       //Acceso al registro CONFIG2
output_high(CS);                //Finalizo la comunicación con CS en alto.
delay_us(1);                      //Se crea un reset
//***********************Programación del ADC_0****************************
output_low(CS);                //Inicio la comunicación con CS en bajo
spi_write(0b00001110);     //Dirección de la PHASE
spi_write(0b00000000);    //Escribo en PHASE
spi_write(0b00000001);    //Escribo en GAIN
spi_write(0b10100011);    //Escribo en STATUS
spi_write(0b00111100);    //Escribo en CONFIG1
spi_write(0b00000000);    //Escribo en CONFIG2
output_high(CS);              //Finalizo la configuración con CS en alto.
```

```
//**********************Reset en el ADC_1*********************************
output_low(CSB);                  //Inicio la comunicación con CS' en bajo
spi_write(0b00010110);         //Dirección del CONFIG2 (0x0B)
spi_write(0b11000000);        //Acceso al registro CONFIG2
output_high(CSB);                //Finalizo la comunicación con CS en alto.
delay_us(1);                          //Se crea un reset
//**********************Programación del ADC_1*****************************
output_low(CSB);                //Inicio la comunicación con CS en bajo
spi_write(0b00001110);     //Dirección de la PHASE
spi_write(0b00000000);    //Escribo en PHASE
spi_write(0b00000001);    //Escribo en GAIN
spi_write(0b10100011);    //Escribo en STATUS
spi_write(0b00111100);    //Escribo en CONFIG1
spi_write(0b00000000);    //Escribo en CONFIG2
output_high(CSB);           //Finaliza la configuración con CS en alto.


 //************************Mensaje de inicio en LCD************************
  lcd_gotoxy(1,1);
   printf(lcd_putc,"      U N A M     ");
   lcd_gotoxy(1,2);
   printf(lcd_putc,"     FES-ARAGON     ");
   lcd_gotoxy(1,3);
   printf(lcd_putc," CENTRO TECNOLOGICO   ");
   lcd_gotoxy(1,4);
```

```
printf(lcd_putc,"                    ");
delay_ms(3000);
printf(lcd_putc,"\f");
lcd_gotoxy(1,2);
printf(lcd_putc,"   S I S T E M A        ");
lcd_gotoxy(1,3);
printf(lcd_putc,"  DE  ADQUISICION  ");
lcd_gotoxy(1,4);
printf(lcd_putc,"   DE  D A T O S    ");
delay_ms(3000);
lcd_putc("\f");

//****************Configura los potenciómetros*****************************
 output_low(CS0);
 set_pot(12);
 output_high(CS0);
 delay_us(2);
 output_low(CS1);
 set_pot(12);
 output_high(CS1);
  delay_us(2);
 output_low(CS2);
 set_pot(12);
 output_high(CS2);
 delay_us(2);
```

```
   output_low(CS3);
    set_pot(12);
    output_high(CS3);
    delay_us(2);
//**********************elije entre Display  y LCD*****************************
      if( input(PIN_B2) )
        display();
       else
        usb();
}
```

# ANEXO 2

## CARTEL PRESENTADO EN EL SEGUNDO CONGRESO INTERNACIONAL DE INSTRUMENTACION Y CIENCIAS APLICAS

2nd International Congress on Instrumentation and Applied Sciences
Puebla, Puebla, Mexico. October 5-8, 2011

Data Acquisition system for a test that simulates conditions of confinement in drainpipes.

M. Ontiveros[1], R. Sánchez[1], A. Ocampo[1], J. Cortes[2]
[1] Facultad de Estudios Superiores Aragón UNAM. [2] Centro Tecnológico Aragón UNAM

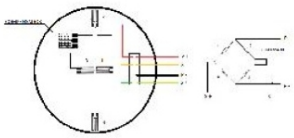
### Abstract

The design of an electronic circuit that allow to the measurements of deformations and efforts in structural elements through strain gauges, The sensors of deformations ( STRAIN GAUGES ) are dispositive whose resistance varies in small amounts when it stretches out or it becomes shorter . These sensors are related with a structure, so that the change percentages of length of the sensor of deformation and the structure are identical. What you must try on in a sensor is the change of resistance and this change has moral values of miliohms. A technique to turn the change of resistance into voltage is supposed to meet in order to measure these changes of resistance that you may measure oneself. With the measurement of the increment of resistance, the Mechanical Engineers can measure the pressure on a structure orchestrated with sensors of deformation in such a way that the pressure exercised in the aforementioned instrument can be common knowledge. The transducers of pressure used to measure the transmission of load through the filling that they were designed are based on the mechanical behavior of the mechanical so-called element diaphragm. The diaphragm is an element it is appropriate for orchestrating whose mechanical behavior, using strain gauges in a repair of complete Wheatstone bridge, a transducer of pressure. The typical diaphragms are designed for measuring pressure in fluids so that their construction generally consists of a hose that conduces that strain conducts the fluent to the intervening diaphragm gauges that it can be buried in the stuffing and you are able to measure the component ones belonging to pressure transmitted.

A card that you allow to has designed converting the change of resistance itself with this work and to become voltage that in turn we amplified and we conditioned to be able to convert that value of analogical voltage to a digital voltage that in turn is processed by means of a microcontroller and envoy of serial way to the PC for his processing.

### Introduction

The strain gauges are used as transducer. These sensors are placed in an arrangement known as Wheatstone bridge so that measurements can be performed by deflection, so the acquisition board measures the voltage difference between the output terminals of the sensor.



The output Vo of the bridge van be determined by treating the top and bottom parts of the bridge as individual voltage dividers. Thus

$$V_{AB} = \frac{R_3}{R_3 + R_4} V_S \quad and \quad V_{AD} = \frac{R_4}{R_2 + R_4} V_S \qquad \text{Eq (a)}$$

The out`put voltage Vo of the bridge is given by:

$$V_G = V_{BD} = V_{AB} - V_{AD} \qquad \text{Eq(b)}$$

Substituting Eqs. (a) into (b) yields:

$$V_o = \frac{R_3 R_2 - R_4 R_1}{(R_1 + R_3)(R_4 + R_2)} V_S \qquad \text{Eq(c)}$$

Equation (d) indicates that the initial output voltage will vanish (Vo= 0) if

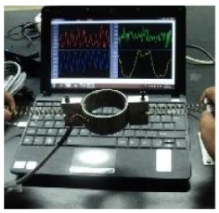
$$R_1 R_3 = R_2 R_4 \qquad \text{Eq(d)}$$

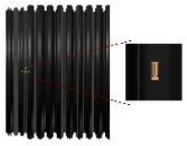
### Experimental Procedures

*Signal Conditioning*

The signals measured by the gauges are very small as they are in the order of microvolts, so the acquisition system account directly on their entries whit an instrumentation amplifier (AD620), this device was placed in order to increase tensions and thus obtained a more practical way to manipulate the captured information. The Gain of the AD620 amplifier Set with One External Resistor (Gain Range 1 to 1000).

When measuring small will be various signals that affect the electronic system, one of those signals is the frequency of alternating current, so the system also has the design of a low pass filter connected to the output of the amplifier (figure 3). This filter was designed whit the LF353 operational amplifier and is set to a cut off frequency of 10 Hz so removing all those frequencies that are above this value including the line frequency of 60 Hz.





*Analog-Digital Conversion.*

After signal conditioning, the information passes through the analog to digital converter (ADC) MCP3901 as its name implies transforms an analog voltage to a binary signals in order to process information. The device was chosen due to its high resolution, speed and internal supplements to make more accurate measurements. The ADC is programmed and controlled by SPI communications (Serial Peripheral Interface) with the PIC18f4550 microcontroller from Microchip

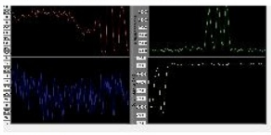
*Transmission Data*

The system has been designed to have the option to display channel data input through an LCD (Liquid Crystal Display) of lines 20x4 or to monitor data through a PC.


When the system is used as a portable device, the data are displayed on an LCD. The transmission of data to a computer is by the USB interface of the PIC18F4550 which has a program to send packet encoded bits of the 2 channels of each ADC. The computer creates a virtual USB port; this is RS-232 emulation in order to display data from the acquisition card.

*Information Processing.*

The information is processed on a PC using a program designed to plot and store data, the program is based on a graphical environment. This section is responsible for the basic description of the computer program which is called "User Interface". Starting the user interface displays a main window, which is due to enter information such as the USB connection port for the Data Acquisition Board, the file name and the address where to save samples and the number of channels to plot once information provided user interface to acquire samples and graphs when you want to end the test samples are automatically saved to a text file with the name and address listed in the main window..



*Conclusions.*

The system has presented good results and we are currently working to design a card of 8 input channels. The proposed circuit is a good alternative for measuring of straing gauges because it has a good resolutions and speed at once. Besides having the option to use the system as a portable device or a more elaborate system that can communicate with a PC..

### References

1. Experimental Stress Analysis - James W. Dally and William F. Riley
2. LabVIEW graphical programming environment – Jose Rafael Lajara
3. Embedded C Programming and the Microchip PIC – Richard Barnett, Larry O´cul

**Acknowledgements**

The authors wish to thank for financial support to the INNOVATEC del Consejo de Ciencia y Tecnologia No. 26096-806-11-V-10

# Bibliografía


- Enrique Mandado, Perfecto Mariño y Alfonso Lago

*Instrumentación Electrónica*

Editorial Alfa Omega 1996

- Albert D. Helfrick, William D. Cooper

*Instrumentación Electrónica Moderna y Técnicas de Medición*

Albert D. Helfrick, William D. Cooper

Editorial Prentice Hall Hispanoamericana

- James W. Dally, William F. Riley

*Experimental Stress Analysis Fourth Edition*

College House Enteprises, LLC

- Robert F. Coughlin, Frederick F. Dridcoll

*Amplificadores operacionales y circuitos integrados lineales*

Prentice Hall Hispanoamericana

- Bolton, William

*Mediciones y pruebas eléctricas y electrónicas*

*Marcombo S.A.*

- Ramon Pallás Areny

*Sensores y acondicionadores de señal*

Barcelona España 2003, Marcombo SA.

- Ramon Pallás Areny

*Adquisición y distribución de señales*

Barcelona España 1993, Marcombo SA.

- Huidobro Moya José Manuel.

*Sistemas telemáticos*

Madrid España Thompson Delmar Learning

- Lajara Vizcaino Jose Rafael, Pelegri Sebastiá (2007)

*LabVIEW. Entorno gráfico de programación*

Mexico DF, Alfaomega

- Antoni Mánuel Lázaro, Joaquín del Río Fernández

*LabVIEW, programación grafica para el control de Instrumentación*

Ed. Paraninfo 2001

- Angulo Usategui, José Maria. (1988)

*Microprocesadores: Arquitectura, Programación y desarrollo de sistemas*

Madrid: Paraninfo

- Valdez Pérez Fernando E (2007)

*Microcontroladores: fundamentos y aplicaciones con PIC*

Barcelona: Marcombo; México, Alfaomega.

- Garcia Breijo Eduardo

*Compilador C CCS y Simulador PROTEUS para Microcontroladores PIC*

Alfaomega Grupo Editor México, junio 2008.

- Richar Barnett, Larry O´Cull, Sara Cox

*Embedded C Programming and the Microchip PIC*

Thompson Delmar Learning