UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN ELÉCTRICA - ELECTRÓNICA

SISTEMA DE ADQUISICIÓN DE DATOS Y REGISTRO DE EVENTOS PARA MONITOREO VÍA REMOTA DE ESTACIONES SISMOLÓGICAS

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO ELÉCTRICO – ELECTRÓNICO

P R E S E N T A N

ADRIÁN ISRAEL HERNÁNDEZ RUIZ

MAURICIO MARTÍNEZ MONTERO

D I R E C T O R

ING. ALEJANDRO SOSA FUENTES

Ciudad Universitaria, México D.F. Noviembre 2013

Agradecimientos

Al Servicio Sismológico Nacional por el apoyo otorgado, además de facilitar sus instalaciones y equipos para el desarrollo de este proyecto.

Al Dr. Carlos Valdés González por la oportunidad de formar parte del grupo de trabajo del SSN y a la Dra. Sara Ivonne Franco Sánchez por el impulso que nos ha brindado a lo largo de todo el proceso.

A los ingenieros Jorge Alberto Estrada Castillo y José Luis Cruz Cervantes por el apoyo, las facilidades y la asesoría brindada para alcanzar los objetivos propuestos.

Al equipo de trabajo del SSN; los ingenieros Iván Rodríguez Rasilla, Fernando Navarro Estrada, Arturo Cárdenas Ramírez, Alejandro Hurtado Díaz y al Maestro Jesús Antonio Pérez Santana.

Al ingeniero Miguel Ángel Gómez Reali por el apoyo en el desarrollo del software de adquisición y por la amistad de tantos años.

Al ingeniero Alejandro Sosa Fuentes por el apoyo en la dirección de esta tesis así como a los sinodales por sus aportaciones y observaciones de este trabajo.

Dedicatorias

Adrián

Les dedico este trabajo y mi esmero a mis padres Carmen y Rafael, por su apoyo y comprensión que me brindaron a lo largo de toda mi vida, además de la motivación que me dieron para poder concluir mis estudios y nunca dejarme dar por vencido para así poder lograr todo lo que me he propuesto tanto en el ámbito estudiantil, emocional y profesional.

También quiero agradecer a mis hermanos Baltazar y Berenice, por el apoyo que me brindaron y la motivación a seguir mis metas hasta el final, así mismo a mi novia Nayeli por todo su apoyo y comprensión además de que siempre me alienta a realizar mis objetivos.

Mauricio

Quiero dedicarle este trabajo y todo mi esfuerzo a mis padres Maricela y Jaime quienes me apoyaron con mis estudios y mi educación; a mis abuelos Guadalupe (QEPD) y Gabriel quienes fueron parte importante en mi educación; a mi hermano y amigo Alejandro que me ha apoyado en toda mi carrera; a mi tío Miguel quien desde pequeño me despertó el gusto por la ingeniería y por quien quise estudiar esta carrera.

A mi novia Erika quien ha formado parte importante en mi vida, quien me ha apoyado en todo, la que me inspira, me alienta y con quien he compartido momentos maravillosos a lo largo de estos magníficos 3 años.

Índice

Índice de Figuras y Tablas

Figuras

Capítulo 1. Estación Sismológica

Capítulo 2. Sistema de Adquisición de Datos y Registro de Eventos

Capítulo 3. Descripción del Sistema

Capítulo 4. Diseño del Sistema

Tablas

Objetivo

Diseñar un sistema que permita monitorear de manera remota el comportamiento de las variables eléctricas de los sistemas instalados en las estaciones sismológicas así como la temperatura a la que se encuentran expuestas las estaciones sismológicas. Para poder elaborar diagnósticos más específicos y conocer los comportamientos de los sistemas instalados en campo.

Introducción

Los sistemas de adquisición de datos son muy útiles debido a que la información recabada puede servir para describir el comportamiento de algunos sistemas ante variables que pudieran afectarlos como son la temperatura.

Las formas de comunicación que existen entre algunos sistemas electrónicos y el usuario crecen cada vez más y los diferentes protocolos de comunicación son muchos pero los más usados en ocasiones no están disponibles en todas las computadoras personales, por lo que proponemos diseñar una tarjeta electrónica que ofrezca cubrir las características necesarias en protocolos de comunicación para la adquisición de datos así como puertos de comunicación como USB y Ethernet que actualmente existen en la mayoría de las computadoras personales.

El método empleado será el de investigar previamente los parámetros a medir, sus alcances o límites y posteriormente la metodología que a continuación se describe.

- Levantamiento eléctrico de la instrumentación de la estación.
- Análisis de variables y los rangos de esas variables
- Determinar la variación de la temperatura en los sitios de interés
- Seleccionar de los muchos sensores que existen en el mercado, los que cumplan con los parámetros de operación que se requiere monitorear.
- Diseñar la etapa de acondicionamiento de las señales para su procesamiento con el microcontrolador.
- Llevar a cabo el procesamiento y despliegue de la información.

Así en el capítulo 1 se observa el panorama general de las estaciones sismológicas en las que se van a trabajar así como las ya existentes por otras instituciones.

En el capítulo 2 se presentará el concepto de *Sistema de Adquisición de Datos y Registro de Eventos*, para establecer de manera clara el objetivo de este proyecto y alcances de un sistema de este tipo.

El capítulo 3 nos muestra los conceptos básicos de instrumentación, que debemos tener presentes para el diseño de nuestro sistema, los diferentes tipos de sensores que existen y la selección del más adecuado, las etapas de acondicionamiento, la selección del

microcontrolador, los protocolos de comunicación empleados y un diagrama a bloques del sistema en general.

Finalmente, en el capítulo 4 veremos el diseño de la tarjeta de desarrollo así como el software con la que será operada, cómo se lleva a cabo la recepción y almacenamiento de los datos y se incluirá el diagrama elemental del sistema.

El resultado final será contar con un dispositivo que permita monitorear de manera continua, remota y confiable, las variables eléctricas y de temperatura en las estaciones sismológicas.

Que estos dispositivos cuenten con puertos de comunicaciones disponibles en los equipos de cómputo actuales.

Registrar los eventos significativos ocurridos que pudieran ser causa de fallas en el sistema.

Poder contar con información suficiente para establecer las condiciones de operación reales en campo, comparar con las condiciones de laboratorio y poder elaborar diagnósticos y acciones correctivas más eficientes.

Capítulo 1. Estación Sismológica

1.1. Panorama actual de las estaciones sismológicas

La Red Sismológica de Banda Ancha está configurada para monitorear la sismicidad en las regiones de mayor potencial sísmico dentro de la República Mexicana. Sus estaciones se localizan, en su mayoría, a lo largo de las costas del Océano Pacífico, así como en el eje neo volcánico. La red cuenta con más de 40 estaciones en operación, las cuales transmiten sus datos en tiempo real a las instalaciones del Servicio Sismológico Nacional por medios de comunicación tales como:

- Línea telefónica directa/dedicada con servicio de Internet.
- Servicio de Internet a través de un enlace satelital.
- Enlace por medio de radio módems.

El diseño de las estaciones contempla que:

El sitio donde va a ser construida debe ser lejano a grandes poblaciones, para evitar el ruido cultural.

El acceso al sitio esté disponible todo el tiempo y que pueda tener cerca servicios disponibles, preferentemente energía eléctrica y línea telefónica con acceso a internet.
Se debe garantizar la seguridad de los equipos instalados en el sitio.

Descripción de las Instalaciones de un Observatorio Sismológico

En este punto solo haremos una breve descripción sin entrar en detalle de las características de la construcción. Si desea más información puede revisar la siguiente bibliografía "Definición de un Estándar Nacional de los Observatorios Sísmicos y Acelerográficos para su Integración a la Red Sísmica Mexicana" que elaboró el Servicio Sismológico Nacional.

Cuarto de instrumentos

Las dimensiones del cuarto de instrumentos son: 1.95 m de ancho, 3.6 m de largo y 3.6 m de alto. En el fondo de éste se construye una plancha de concreto a una altura de 95 cm sobre el nivel del piso. Las dimensiones de la plancha son: 1.95 m de largo, 70 cm de ancho y un espesor aproximado de 8 cm.

En este cuarto se encuentra el acceso principal al observatorio; este acceso debe contar con dos puertas, una exterior metálica, de 90 cm de ancho y 2.1 m de alto, y una interior de madera, de las mismas dimensiones. En una de las paredes que dan hacia el exterior de la caseta se construye una ventana cuadrada de vitroblock de 60 cm de lado, a una altura de 1.8 metros; se instala además, a 50 cm de separación de la ventana, un extractor para la ventilación interna.

Cuarto de Sensores

Las dimensiones del cuarto de sensores son: 2.1 m de ancho, 3.6 m de largo y 3.6 m de alto. En este cuarto, directamente sobre el basamento de roca firme, está construido el pilar de aproximadamente 1 m x 1 m de lado. La altura del pilar depende de la profundidad a la que se encuentre el basamento, pero debe sobresalir aproximadamente 25 cm por encima del nivel de piso; los sensores de velocidad y de aceleración están orientados e instalados sobre este pilar de concreto.

Hay una separación entre el piso y el pilar de aproximadamente 10 cm por cada lado; esta separación se hace con la finalidad de minimizar el efecto que pudiera causar la caseta en la señal registrada por los sensores. Los sensores deben estar protegidos con una caja de aluminio de 40 cm de lado y 37 cm de alto, la cual va anclada al pilar por medio de tornillos y en el interior esta forrada con material aislante para minimizar los efectos de temperatura. Es necesario que exista una puerta de madera que separe el cuarto de sensores con el de equipos.

Dentro de las estaciones se encuentran los siguientes instrumentos:

- Un Sismómetro STS-2. Estos sensores triaxiales permiten registrar ondas sísmicas en una amplia banda de frecuencias, con respuesta plana a la velocidad del suelo entre 0.01 a 30 Hz, y capacidad de registrar sismos en una amplia gama de magnitudes, desde sismos locales pequeños hasta sismos lejanos, sin problemas de saturación.

- Un acelerómetro FBA-23. Estos sensores triaxiales permiten registrar las aceleraciones del suelo dentro de un amplio espectro de frecuencias sin saturación de la señal para sismos grandes locales y regionales. Estas características de los sensores, permiten estimar con gran precisión la magnitud de sismos grandes que puedan ocurrir en el territorio nacional.

- Un registrador Quanterra con digitizador de 24 bits. Este equipo cuenta con 6 canales, 3 canales para velocidad (N-S, W-E y Z) y 3 para aceleración (N-S, W-E y Z). Las señales de ambos sensores son muestreadas (en forma continua a 20, 1 y 0.1 muestras por segundo *mps*), son convertidas a un formato digital, almacenadas en el disco duro del registrador Quanterra y enviadas al SSN.

La Fig. 1.1 muestra ambos cuartos, la ubicación del registrador Quanterra (1) y los sensores de velocidad (2) y aceleración (3) ubicados en el pilar de concreto (4).

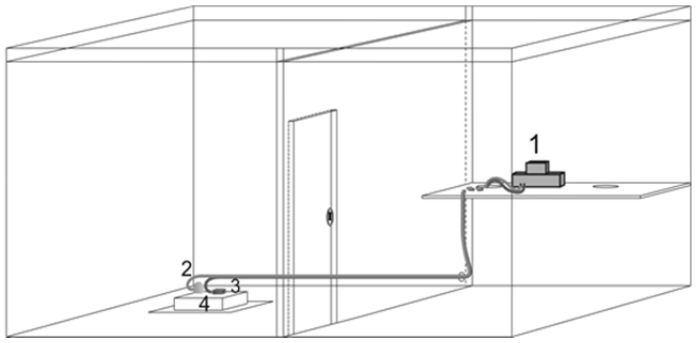


Fig. 1.1. Distribución de Cuartos de Sensores e Instrumentación.

Además de los equipos sísmicos, la estación alberga otros equipos que tienen diversas funciones como son, la regulación del voltaje de AC, energizar los equipos sísmicos y de GPS, protección contra descargas eléctricas, comunicación satelital y en la actualidad gran parte de la red de estaciones cuentan con una estación permanente de GPS.

A continuación se muestran imágenes de una estación sismológica (Fig. 1.2) y el interior de los cuartos de instrumentos (Fig. 1.3) y de sensores (Fig. 1.4).

Estación Sismológica de Hidalgo del Parral, Chihuahua
"HPIG"

Fig. 1.2.Vista Exterior.

Fig. 1.3.Interior Cuarto de Instrumentos.

Fig. 1.4.Interior Cuarto de Sensores.

Los equipos instalados actualmente en una estación sismológica de la Red de Banda Ancha son:

- Regulador de voltaje de 2 KVA.
- Rectificador de voltaje a 12V @ 12A con batería.
- DC-UPS a 12V @ 100W con batería.
- UPS de 1550 VA.
- Detector de tormentas.
- Módem de comunicación satelital.
- Registrador con digitalizador.
- Sensores sísmicos.
- GPS.
- Buffer telemétrico.

En conjunto todos los equipos instalados componen una estación sismológica. Las características de cada uno han sido especificadas debido a las necesidades que cada estación presenta o como consecuencia de los avances tecnológicos. En conjunto todo el sistema nos brinda en caso de falla en el suministro eléctrico un respaldo de poco más de 1 hora de comunicación satelital y hasta 10 días de operación bajo condiciones óptimas del sistema.

Esquema Básico de Operación de una Estación Sismológica

En el diagrama a bloques de la Fig. 1.5 podemos apreciar cada una de las etapas en las que se ubican los equipos mencionados anteriormente y todo ello con la finalidad de que exista una comunicación remota en todo momento, que permita tener acceso en tiempo real a los datos sísmicos y de GPS y precisar el estado actual de operación de la estación sismológica.

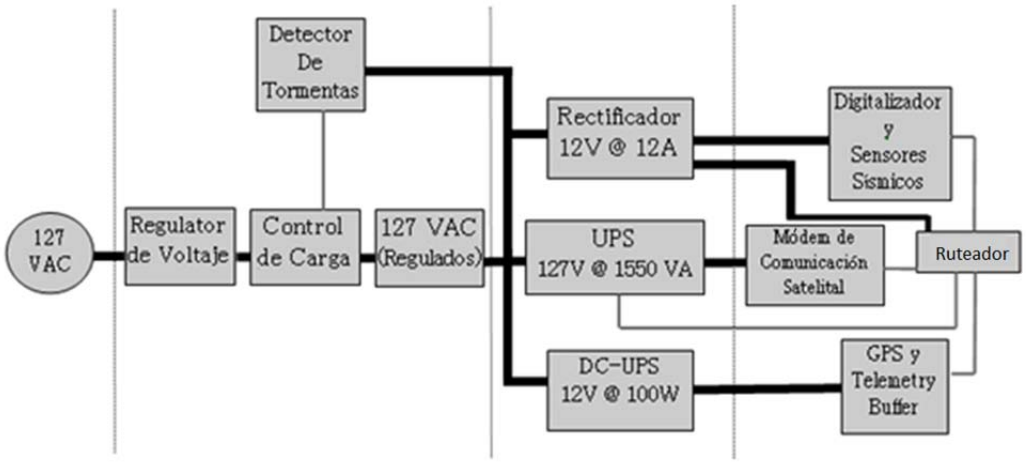


Fig. 1.5. Diagrama a bloques de una Estación Sísmica de Banda Ancha.

Desde las instalaciones del SSN es posible acceder de manera remota al módem de comunicación satelital, al UPS que le da respaldo de energía a dicho módem, al digitalizador y al buffer telemétrico. Sin embargo el estado de los demás instrumentos no es posible conocerlos hasta estar en el sitio ya que no contamos con herramientas que permitan de manera remota determinar su condición actual de operación.

1.2. Red Acelerográfica del Instituto de Ingeniería

La Red Acelerográfica del Instituto de Ingeniería se encuentra monitoreando principalmente en la región costera del pacifico mexicano y algunos otros puntos del interior del territorio, para poder monitorear y determinar la localización epicentral, así como la magnitud del mismo, ya que es importante el estudio de las ondas sísmicas desde su origen hasta su llegada a grandes asentamientos de población.

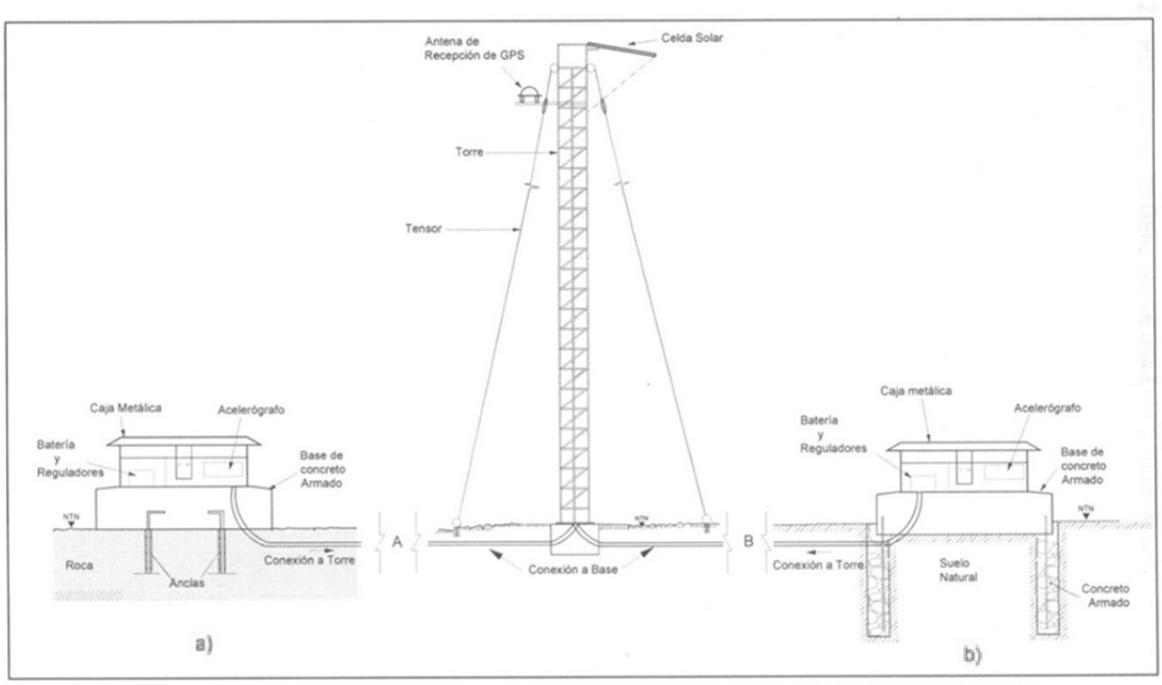


Fig. 1.6. Estación Acelerográfica Típica utilizada por la Red Acelerográfica del II-UNAM.
a) roca b) suelo

Las estaciones Acelerográficas en campo libre (Fig. 1.6) utilizadas actualmente cuentan con:

- Caja metálica empotrada en una base de concreto armado
- Torre metálica
- Acelerógrafo o registrador sísmico
- Regulador de corriente
- Baterías recargables
- GPS
- Celdas solares

Se busca el mejoramiento dela instalación de estas Estaciones Acelerográficas, ya que se obtienen algunos inconvenientes de operatividad sobre todo en la época de lluvias, al no poderse realizar las revisiones en condiciones óptimas.

Características del Terreno para la Instalación

La selección del terreno para la instalación de las estaciones debe de cumplir con ciertas características recomendables como son: acceso a servicios e instalaciones, seguridad del sitio, tipo de suelo, permisos y facilidad de acceso.

La instalación se puede realizar en dos tipos de suelo, dependiendo del fenómeno que se desea estudiar y estos pueden ser suelo blando o roca.

Las estaciones situadas en roca nos proporcionan datos muy importantes como son: localización epicentral, magnitud, duración, profundidad focal, etc.

Esto se debe de hacer en una roca firme no fracturada y preferentemente en afloramientos de rocas intrusivas, así se tendrá un mejor estudio de las ondas sísmicas.

Al instalar una estación acelerográfica en el suelo es porque se quiere conocer el movimiento del mismo y de las estructuras cercanas, estas se instalan en ciudades o asentamientos para el estudio de las estructuras que se encuentran en el sitio.

Infraestructura de la Estación Estándar

Se está buscando la estandarización para la integración a la Red Sísmica Mexicana, entre las principales características para que esto ocurra se debe de cumplir con un resguardo a los sistemas de alimentación y de transmisión de datos, además de ser seguras.

Estas consideraciones se tomaron al tener en cuenta que las estaciones instaladas poseen una base de concreto y una caja metálica empotrada en la misma con lo cual se tendría una doble protección, al construirles una caseta que alberge la estación. Además que en la parte de la azotea se puedan instalar celdas solares y una antena satelital para cubrir las necesidades de transmisión de datos en tiempo real.

La Estación Acelerográfica Estándar (Fig. 1.7) contara con lo siguiente:

- Caseta.
- Base acelerográfica.
- Losa de cimentación.
- Muros de concreto.
- Caja metálica.
- Repisa de concreto reforzado.
- Tubos de ventilación.
- Puerta metálica.

- Sistema de tierras.
- Ductos de interconexión entre la base, repisa y el exterior.
- Mufas.
- Pretil en azotea.
- Herraje para montura de celdas solares.
- Herraje para montura de antena parabólica.

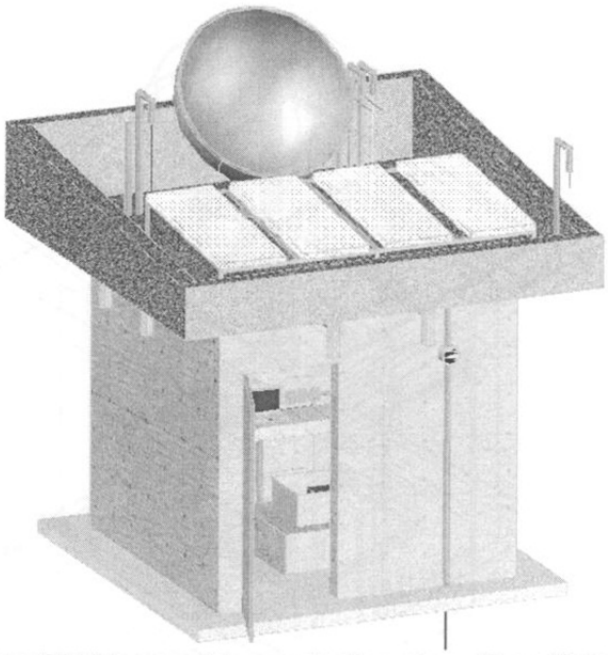


Fig. 1.7. Vista General de una Estación Acelerográfica Estándar.

La integración de estos componentes a las estaciones existentes da paso a una modernización e integración y a un estándar que ayudara a la adquisición de datos en tiempo real para su almacenamiento, análisis y estudio de los mismos, en caso de un sismo que pueda causar daños.

Esto también se debe hacer en estaciones regionales para su integración a la Red Sísmica Mexicana, con lo cual se tendrá una red más robusta capaz de adquirir y procesar datos en tiempo real con otros centros de registro.

Capítulo 2. Sistemas de Adquisición de Datos y Registro de Eventos

2.1. Objetivo

Con la ayuda de un sistema de adquisición de datos y registro de eventos se pretende obtener información suficiente que permita determinar el origen de las fallas que pudieran presentarse dentro de una estación sismológica. Así mismo se busca monitorear las condiciones reales de operación delos equipos instalados en campo ya que no existe un equipo dedicado a dicha tarea dentro de la estación.

2.2. Introducción

La Instrumentación electrónica es la técnica que se ocupa de la medición de cualquier magnitud física, de la conversión de la misma magnitud en señales eléctricas y de su acondicionamiento para proporcionar información adecuada a un sistema de control, a un operador humano o ambos.

Adquisición de datos

Proceso por el cual la señal a medir es adquirida y convertida en una señal eléctrica. Un sistema de adquisición de datos es el instrumento que nos sirve para obtener los datos de un proceso.

Sistema de Adquisición de Datos (SAD)

Es una interfaz entre el mundo real de parámetros físicos, y un sistema de cómputo y control digital. Con el énfasis actual en los sistemas digitales, la función de la interfaz se ha convertido en una muy importante, los sistemas digitales se utilizan ampliamente porque los circuitos complejos son de bajo costo, precisos y relativamente sencillos de implementar. Además, hay un rápido crecimiento en el uso de microprocesadores para realizar un control digital difícil y funciones de medición.

Los sistemas computarizados de control realimentados se utilizan en muchas industrias diferentes hoy en día con el fin de lograr una mayor productividad en nuestras sociedades. Algunas de las industrias que actualmente emplean estos sistemas automáticos se dedican a la fabricación de acero, procesamiento de alimentos, la producción de papel, refinación de petróleo, fabricación de productos químicos, la producción textil, fabricación de cemento, entre muchos otros.

Los dispositivos que realizan la función de interfaz entre los sistemas analógicos y digitales son los convertidores analógico-digitales (A/D). Algunas de las aplicaciones específicas en las que los convertidores están involucrados son: los sistemas de datos de telemetría, comunicaciones, sistemas automáticos de prueba, sistemas informáticos, sistemas de visualización de vídeo y procesamiento de señales, sistemas de registro de datos y sistemas de control de datos.

Etapas de un Sistema de Adquisición

Además de los convertidores A/D, para los sistemas de adquisición y distribución de datos se puede emplear uno o más de los siguientes circuitos:

- Sensores y transductores.
- Amplificadores.
- Filtros.
- Funciones analógicas no lineales.
- Multiplexores analógicos.
- Retenedor de muestras.

La interconexión de estos componentes se muestra en el diagrama de adquisición de datos de un sistema de realimentación de control computarizado en la Fig. 2.1.

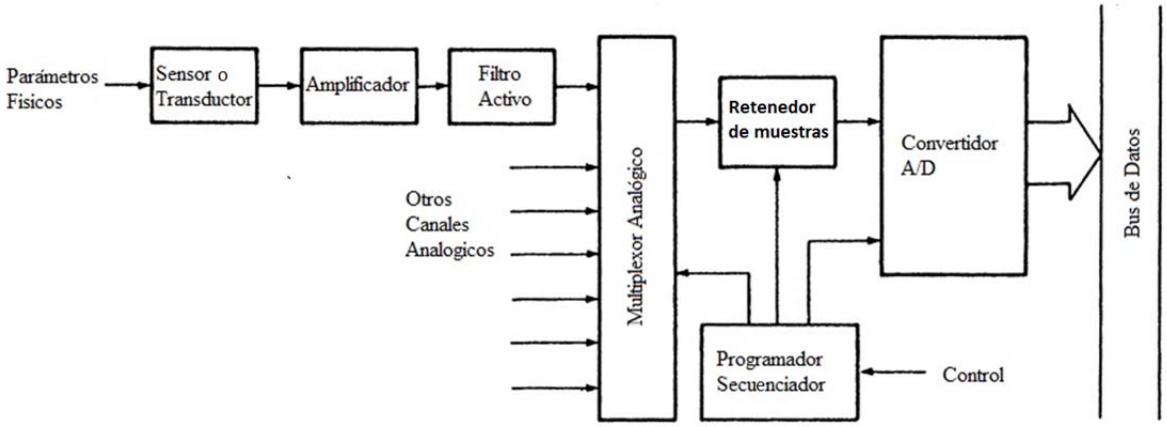


Fig. 2.1. Diagrama a bloques de un Sistema de Adquisición de Datos.

La entrada al sistema es un parámetro físico, que es convertido primero en una magnitud eléctrica por medio de un sensor o transductor, una vez en forma eléctrica, todo el procesamiento adicional se realiza mediante circuitos electrónicos.

Un sensores un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación.

Un transductor es un dispositivo que transforma magnitudes de una determinada variable en otras distintas, proporcionales a las anteriores.

A continuación, un amplificador aumenta la amplitud de la señal de salida del transductor a un nivel útil para su posterior procesamiento. Además, la salida del transductor puede ser una señal de alta impedancia, una señal diferencial de ruido en modo común, una corriente de salida, una señal superpuesta a una tensión alta, o una combinación de éstos. El amplificador, con el fin de convertir dichas señales en una tensión de alto nivel, puede ser uno de varios tipos especializados.

El amplificador es frecuentemente seguido por un filtro activo paso-bajas que reduce componentes de alta frecuencia de la señal y el ruido no deseado causado por interferencia eléctrica de la señal.

La señal analógica procesada va después a un multiplexor analógico, que cambia secuencialmente entre un número de diferentes canales de entrada. Cada una de ellas a su vez está conectada a la salida del multiplexor por un período de tiempo especificado por el interruptor de multiplexor. Durante este tiempo de conexión, un circuito de retención de la muestra adquiere el voltaje de la señal y a continuación mantiene su valor, mientras que un convertidor A/D convierte el valor en forma digital. La palabra digital resultante va a un bus de datos de la computadora a la entrada de un circuito digital.

Así, el multiplexor analógico, junto con la retención de la muestra, el tiempo de acciones del convertidor A/D con un número de canales de entrada analógicos, la temporización y control completa del SAD se realiza mediante un circuito digital llamado programador-secuenciador, que a su vez está bajo el control de la computadora. En algunos casos, la propia computadora puede controlar la totalidad del SAD. Esto es quizás la configuración más comúnmente utilizada del SAD, aunque hay otras alternativas.

Con la llegada de la PC y su adopción en múltiples campos de la ingeniería, la instrumentación dio un gran paso y un importante desarrollo que da origen al concepto de la *instrumentación virtual*, resultado que ofrece mayor productividad precisión y rendimiento.

Instrumentación Virtual y el Sistema de Adquisición de Datos y Registro de Eventos

Con la llegada de la instrumentación virtual y el potencial de los equipos de cómputo disponible para la adquisición de datos, es posible expandir los alcances del sistema. De esta manera registrar eventos significativos en el proceso de monitoreo es una tarea fundamental para el análisis de cualquier proceso y/o sistema. A partir de esto conoceremos a este conjunto como ***Sistema de Adquisición de Datos y Registro de Eventos (SADRE).*** Pero ¿qué es la Instrumentación Virtual?

Instrumentación Virtual

Un instrumento virtual consiste de manera general en una computadora equipada con un software y hardware económico y versátil que cumpla con las funciones de varios instrumentos de medición.

Los instrumentos virtuales le permiten al hardware dedicar toda su capacidad en la adquisición, acondicionamiento y digitalización de las variables a analizar; y al software aprovechar la potencia del cálculo, la conectividad con otros dispositivos y realizar procesos de control que actuarán posteriormente, si así lo desea el usuario.

Las ventajas de un instrumento virtual sobre instrumentos tradicionales es la ilimitada capacidad de almacenamiento y proceso que puede ofrecer una computadora. Gracias a esto, el análisis de los datos cada vez es más sencillo y ofrece mejores resultados.

La versatilidad de una computadora personal expande aún más la capacidad de análisis y proceso de los datos adquiridos en comparación con los que pudieran ofrecer instrumentos como multímetros, osciloscopios o cualquier instrumento de medición tradicional.

Reducción de Costos

Utilizando soluciones basadas en la instrumentación virtual, se pueden reducir los costos de inversión, desarrollo de sistemas y mantenimiento.

Software en la instrumentación virtual

El software es el componente más importante de un instrumento virtual. Con la herramienta de software apropiada los ingenieros y científicos pueden crear eficientemente sus propias aplicaciones, diseñando e integrando las rutinas que requiere un proceso en particular.

También pueden crear las interfaces de usuario que mejor satisfagan el objetivo de la aplicación y de aquellos que van a interactuar con ellas. Pueden definir cómo y cuándo la aplicación adquiere datos desde el dispositivo, cómo los procesa, manipula, almacena y cómo se presentan los resultados al usuario.

Un instrumento virtual (Fig. 2.2) se compone de las siguientes etapas:

- Sensores y transductores.
- Acondicionamiento de la señal (amplificación y filtrado).
- Digitalización (convertidor A/D).
- Interface de comunicación.
- Procesamiento de la información para su manejo posterior.
- Etapa de visualización de la información.

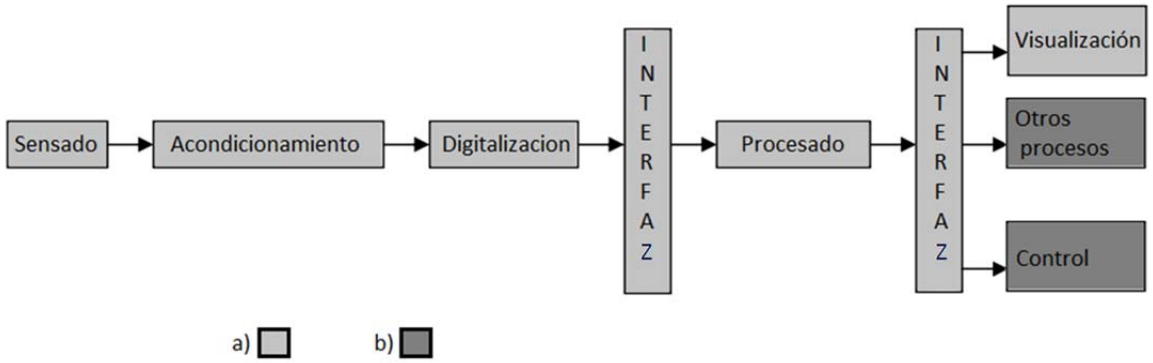


Fig. 2.2. Diagrama a bloques de un sistema de adquisición de datos basado en un instrumento virtual, a) Instrumento virtual y b) Procesos posteriores.

Sistema SCADA

Se le conoce como SCADA (Supervisory Control and Data Acquisition) a todo software que permita el acceso a datos remotos de un proceso y al control del mismo. Un sistema SCADA no solo es un sistema de control, sino un software de monitoreo que realiza la función de interfaz entre los módulos de control y la gestión del proceso.

Un sistema SCADA engloba los sistemas electrónicos y electromecánicos involucrados en la adquisición de señales y el procesamiento de los datos, necesarios para ofrecer una visualización del estado actual y el control del proceso que se está supervisando. Cumpliendo ciertos criterios que diferencian un sistema SCADA de un sistema de control.

Un sistema SCADA está conformado por:

- Interfaz Operador – Máquinas. Es el entorno visual que brinda el sistema para que el operador se adapte al proceso desarrollado. Permite la interacción del ser humano con los medios tecnológicos implementados.

- Unidad Central (MTU). Conocido como Unidad Maestra. Ejecuta las acciones de mando (programadas) con base de los valores actuales de las variables medidas. También se encarga del almacenamiento, procesado y ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos.

- Unidad Remota (RTU). Lo constituye todo elemento que envía algún tipo de información a la unidad central.

- Sistema de Comunicaciones. Se encarga de la transferencia de información del punto donde se realizan las operaciones, hasta el punto donde se supervisa y controla el proceso. Lo conforman los transmisores, receptores y medios de comunicación.

- Transductores. Son los elementos que permiten la conversión de una señal física en una señal eléctrica (y viceversa).

En la Fig. 2.3 se muestra un diagrama en el que se representa la distribución de estos sistemas dentro de un SCADA y la interacción entre ellos.

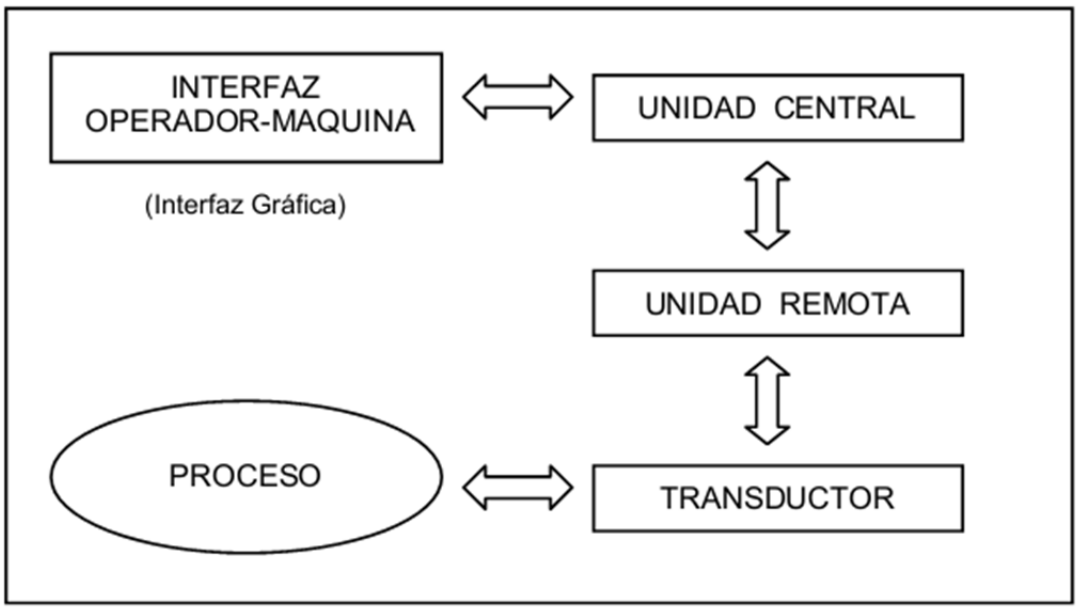


Fig. 2.3. Elementos de un sistema SCADA.

Las principales características que debe cumplir un sistema SCADA son:

- Funcionalidad completa sobre cualquier PC.
- Arquitectura abierta que permita la combinación con aplicaciones estándar y de usuarios, que permita a los integradores crear soluciones optimizadas.
- Sencillez de instalación.
- Permitir la integración de herramientas ofimáticas y de producción.
- Facilidad de configuración y que sea escalable.
- Contener funciones de mando y supervisión.
- Sistemas de comunicación flexibles que permitan de manera transparente establecer comunicación entre sistemas y usuarios.

Este tipo de sistemas son comúnmente utilizados en procesos y líneas de producción como procesos químicos, farmacéuticos, ensamble equipos electrónicos, en el sector automotriz entre muchos otros. Con el fin de optimizar la productividad, detectar las pérdidas de materia prima que se generan a lo largo del proceso, reducir costos de mantenimiento y en el pago de servicios, calcular la eficiencia del proceso y localizar las áreas de oportunidad en las que se pueda atacar para mejorar los aspectos antes mencionados.

2.3. Análisis

Debido a nuestra necesidad vamos a utilizar un SADRE, ya que cumple con los requerimientos del proyecto a diferencia de un SCADA que es un sistema sobrado, puesto que no requerimos de un bloque de control que interactúe con algún proceso, lo único necesario para este caso es la adquisición de datos constante y su almacenamiento para un posterior análisis y estudio.

Por medio de un SADRE, se obtendrá la información suficiente para caracterizar las condiciones reales de operación a las que se encuentran los diferentes equipos instalados en una estación sismológica, además esta información será almacenada para posteriormente ser analizada y registrar a través del tiempo los eventos que pudieran ser la causa de posibles fallas.

Otra parte importante es la comunicación ya que gracias a ella podremos registrar los eventos de manera remota, permitiéndonos así, llevar un registro continuo de múltiples estaciones sismológicas de manera simultánea.

Capítulo 3. Descripción del Sistema

3.1. Conceptos Básicos de Instrumentación

Sistemas de Medida

Se denomina sistema a la combinación de dos o más elementos, subconjuntos y partes necesarias para realizar una o varias funciones. En los sistemas de medida, esta función es la asignación objetiva y empírica de un número a una propiedad o cualidad de un objeto o evento, de tal forma que la describa.

Objetiva se refiere a que el resultado de una observación debe ser completamente independiente del observador. Y empírica al experimento, que corresponde al evento que se encuentra en observación.

Los objetivos de la medida pueden ser: la vigilancia o el seguimiento de procesos, el control de un proceso y alguna necesidad de ingeniería experimental.

Transductores, Sensores y Acondicionamientos

Se denomina transductor en general, a todo tipo de dispositivo que convierte una señal de forma física en una señal correspondiente pero de otra forma física distinta. Es por tanto, un dispositivo que convierte un tipo de energía en otro.

Dado que hay seis tipos de señales: mecánicas, térmicas, magnéticas, eléctricas, ópticas y moleculares (químicas), cualquier dispositivo que convierta una señal de un tipo en una señal de otro tipo se considera un transductor, y la señal de salida podría ser de cualquier forma física útil.

Los sistemas de medida electrónicos ofrecen, entre otras cosas, las siguientes ventajas:

Cualquier variación de un parámetro no eléctrico de cualquier material viene acompañada por la variación de un parámetro eléctrico. Eligiendo el material adecuado, esto permite realizar transductores con salida eléctrica para cualquier magnitud física no eléctrica.

Además de la amplificación, existe una gran variedad de recursos en forma de circuitos integrados, para acondicionar o modificar señales eléctricas.

Un sensor es un dispositivo que, a partir de la energía del medio donde se mide, da una señal de salida transducida que es función de la variable medida.

Acondicionamiento de la Señal

Los acondicionadores de señal son los elementos del sistema de medida que ofrecen una señal apta para ser presentada o registrada o que simplemente permita un procesamiento posterior mediante un equipo o instrumento estándar. Consiste en circuitos electrónicos que ofrecen: amplificación, filtrado, acoplamiento de impedancias y modulación o demodulación.

Tipos de Sensores

Según el aporte de energía, los sensores se pueden dividir en moduladores y generadores. En los sensores moduladores, la energía de la señal de salida procede, en mayor parte, de una fuente de energía auxiliar. La entrada solo controla la salida. En los sensores generadores, en cambio, la energía de salida es suministrada por la entrada.

Los sensores, según su señal de salida, pueden ser clasificados. Su clasificación está en función de cómo presentan la señal transducible, en forma digital o analógica. Así mismo los sensores pueden ser de deflexión o de comparación. En el caso de los sensores que son por deflexión, la magnitud medida produce algún efecto físico que se refleja en un efecto similar pero opuesto en alguna parte del instrumento y se relaciona con una variable útil.

En los sensores que funcionan por comparación, se intenta mantener nula la deflexión mediante la aplicación de un efecto bien conocido, opuesto al generado por la magnitud a medir.

Las mediciones por comparación suelen ser más exactas porque el efecto conocido opuesto se puede calibrar con un patrón o magnitud de referencia. A continuación en la Tabla 1 se agrupan estas clasificaciones y se muestran algunos ejemplos de sensores.

Tabla 1. Clasificación de Sensores.

Criterio	Clases	Ejemplos
Aporte de Energía	Moduladores Generadores	Termistor Termopar
Señal de Salida	Analógicos Digitales	Potenciómetro Codificador de Posición
Modo de Operación	De Deflexión De Comparación	Acelerómetro de Deflexión Servoacelerómetro

Desde el punto de vista que nos interesa, es mejor clasificar a los sensores de acuerdo con un parámetro variable que para nosotros es conocido, y por ello, posible de manejar. Este parámetro variable puede ser: su resistencia, capacitancia e inductancia para luego añadir los sensores generadores de tensión, carga o corriente, al igual que otros tipos no incluidos en los anteriores grupos. A continuación se muestra en la Tabla 2 los diferentes tipos de sensores y los métodos de detección que existen actualmente.

Tabla 2. Tipos de sensores y métodos de detección que existen actualmente.

Sensores	Magnitudes								
	Posición Distancia Desplazamiento	Velocidad	Aceleración Vibración	Temperatura	Presión	Caudal Flujo	Nivel	Fuerza	Humedad
Resistivos	Potenciómetros Galgas Magneto-resistencias		Galgas + masa-resorte	RTD Termistores	Potenciómetro + tubo Bourdón	Anemómetros e hilo caliente Galgas + voladizo Termistores	Potenciómetro + flotador Termistores LDR	Galgas	Humistor
Capacitivos	Condensador diferencial				Condensador variable + diafragma		Condensador Variable	Galgas capacitivas	Dieléctrico variable
Inductivos y electromagnéticos	LVDT Corrientes Foucault Resolver Inductosyn Efecto Hall	Ley Faraday LVDT Efecto Hall Corrientes Foucault	LVDT + masa-resorte		LVDT + diafragma Reluctancia variable + diafragma	LVDT + rotámetro Ley Faraday	LVDT + flotador Corrientes Foucault	Magnetoelástico LVDT + célula de carga	
Generadores			Piezoeléctricos + masa-resorte	Termopares Piezoeléctricos	Piezoeléctricos			Piezoeléctrico	
Digitales	Codificadores incrementales y absolutos	Codificadores incrementales		Osciladores de cuarzo	Codificador + tubo Bourdón	Vórtices			SAW
Uniones *p-n*	Fotoeléctricos			Diodo Transistor ConvertidoresT/I			Fotoeléctricos		
Ultrasonido	Reflexión	Efecto Doppler				Efecto Doppler Tiempo Tránsito Vortice	Reflexión Absorción		

Exactitud, precisión y sensibilidad

La exactitud (en inglés, accuracy) es la cualidad que caracteriza la capacidad de un instrumento de medida de dar indicaciones que se aproximan al verdadero valor de la magnitud medida.

La exactitud de un sensor se determina mediante la denominada calibración estática. Consiste esta en mantener todas las entradas excepto una a un valor constante.

La discrepancia entre la indicación del instrumento y el verdadero valor de la magnitud medida se denomina "error". La diferencia entre la indicación del instrumento y el verdadero valor se denomina error absoluto.

$$Error\ absoluto = resultado - verdadero\ valor$$

Sin embargo, lo más común es especificar el error como cociente entre el error absoluto y el verdadero valor de la magnitud medida, cociente que se denomina error relativo.

$$Error\ relativo\ = \frac{Error\ absoluto}{Verdadero\ valor}$$

La fidelidad o precisión es la cualidad que caracteriza la capacidad de un instrumento de medida de dar el mismo valor de la magnitud medida, al medir varias veces en unas mismas condiciones determinadas prescindiendo de su concordancia o discrepancia con el valor real de dicha magnitud. La fidelidad o precisión es una condición necesaria pero no suficiente para la exactitud, ya que un instrumento puede ser preciso en cuanto a su capacidad de repetición pero no exacto en cuanto su aproximación al verdadero valor de la magnitud medida.

La sensibilidad o factor de escala es la pendiente de la curva de calibración que puede ser o no constante a lo largo de la escala de medida. Para un sensor cuya salida esté relacionada con la entrada *x* mediante la ecuación $y = f(x)$, la sensibilidad en el punto x_a,

$S(x_a)$, es:

$$S(x_a) = \left.\frac{dy}{dx}\right|_{x=x_a}$$

Linealidad y resolución

La linealidad expresa el grado de coincidencia entre la curva de calibración y una línea recta determinada. Según cuál sea dicha recta se habla de:

- Linealidad independiente: la línea de referencia se define por el método de mínimos cuadrados. De esta forma, el máximo error positivo y el mínimo error negativo son iguales, Fig. 3.1a.
- Linealidad ajustada al cero: la recta se define también por el método de los mínimos cuadrados, pero con la restricción adicional al pasar por cero, Fig. 3.1b.
- Linealidad terminal: la recta se defina por la salida sin entrada y la salida teórica máxima, correspondiente a la mayor entrada admitida, Fig. 3.1c.
- Linealidad a través de los extremos: la recta se define mediante la salida real cuando la entrada es la menor del alcance especificado, y la salida real cuando la entrada es la máxima del alcance especificado, Fig. 3.1d.
- Linealidad teórica: la recta es la definida por las previsiones teóricas formuladas al diseñar el sensor, Fig. 3.1e.

El interés de la linealidad está en que la conversión lectura-valor medido es más fácil si la sensibilidad es constante. Además, en instrumentos lineales la no linealidad equivale a la inexactitud.

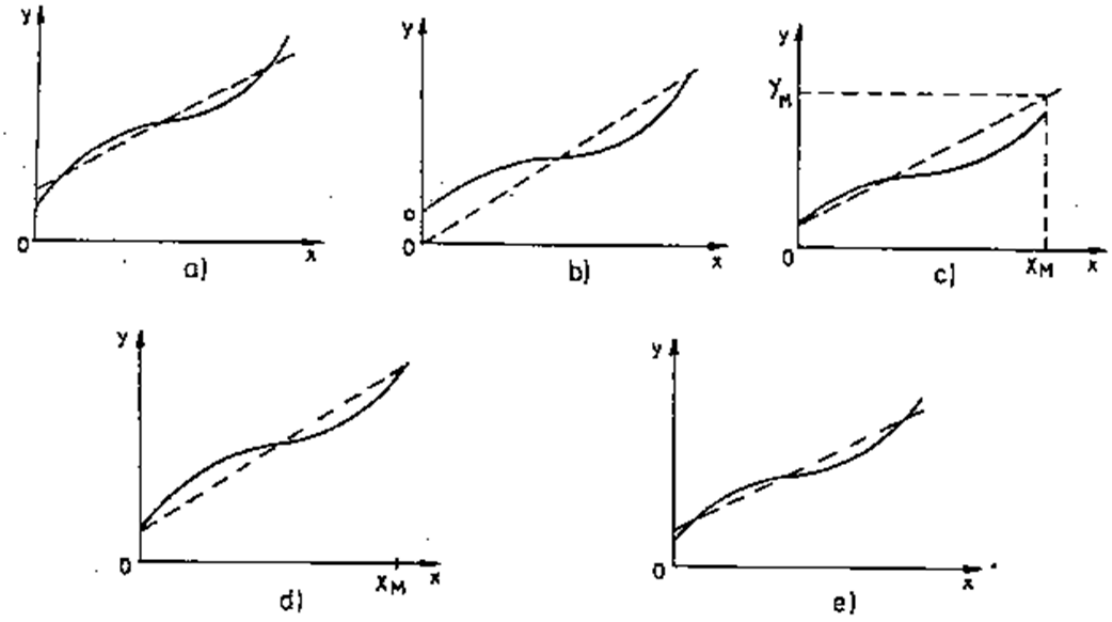


Fig. 3.1. Rectas para representar la linealidad; a) Mínimos cuadrados; b) Mínimos cuadrados ajustados al cero; c) Terminal; d) A través de los extremos; e) Teórica.

Los principales factores que influyen en la linealidad son: la resolución, el umbral y la histéresis. La resolución o discriminación es el incremento mínimo de la entrada para el que se obtiene un cambio en la salida. Cuando el incremento de la entrada se produce a partir de cero, se habla de umbral.

La histéresis se refiere a la diferencia en la salida para una misma entrada, según la dirección en que se alcance.

Errores sistemáticos

Se dice de un error que es sistemático cuando en el curso de varias medidas de una magnitud de un determinado valor, hechas en las mismas condiciones, o bien permanece constante en valor absoluto y signo, o bien varía de acuerdo con una ley definida cuando cambian las condiciones de medida.

Errores aleatorios

Los errores aleatorios se manifiestan cuando se miden repetidamente la misma magnitud, con el mismo instrumento y el mismo método, y presentan las propiedades siguientes:

- Los errores aleatorios positivos y negativos de igual valor absoluto, tienen la misma probabilidad de producirse.
- Los errores aleatorios son tanto menos probables cuanto mayor sea su valor.
- Al aumentar el número de medidas, la medida aritmética de los errores aleatorios de una muestra-conjunto de medidas- tiende a cero.
- Para un método de medida determinado, los errores aleatorios no exceden de cierto valor. Las medidas que lo superan deben repetirse y, en su caso, estudiarse por separado.

Los errores aleatorios se denominan también errores accidentales o fortuitos, y ello da a entender que pueden ser inevitables.

Cuando se realice una medida aislada, en las mismas condiciones, quedará solo la componente aleatoria del error.

3.2. Diagrama a bloques del sistema

La tarjeta electrónica se conforma de varios bloques y cada uno de ellos juega un papel fundamental en el sistema. Todos fueron desarrollados y probados de manera independiente.

De manera muy general podemos observar en la Fig. 3.2 como está compuesta la tarjeta electrónica.

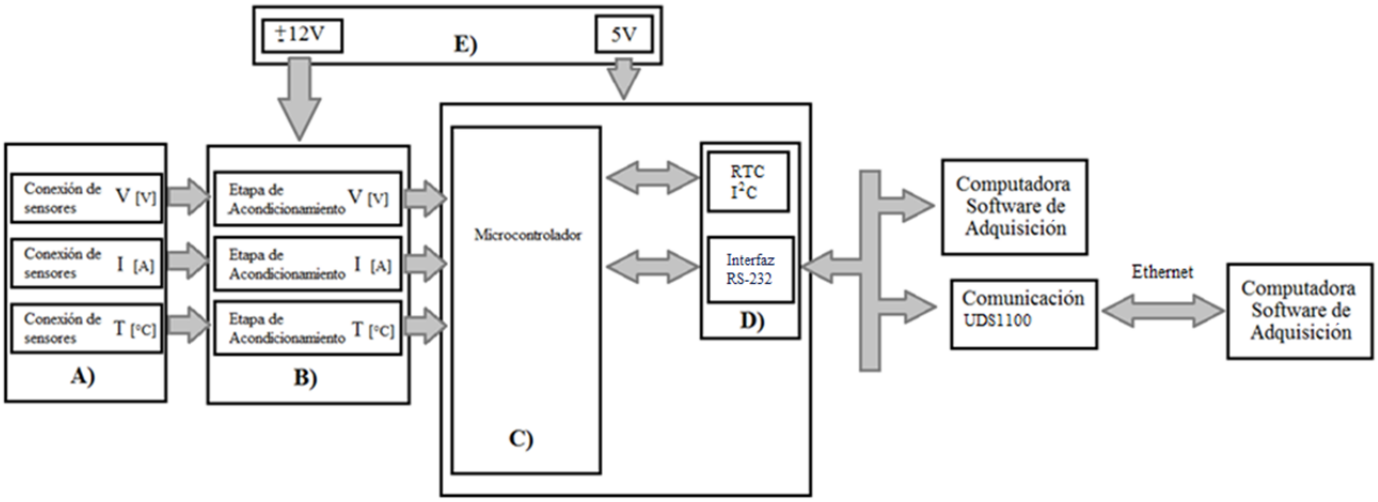


Fig. 3.2. Diagrama a bloques del sistema.

Los bloques fundamentales en esta figura son:

- Sensores.
- Acondicionamiento de los sensores.
- Microcontrolador o CPU.
- Periféricos de comunicación.
- Fuentes de voltaje de 5 y +/-12VDC.

A) Sensores

Este bloque está compuesto por los sensores del sistema, que para nuestro caso son de voltaje, corriente y temperatura. Para el voltaje es un divisor de tensión, el sensor de corriente es una dona de efecto de campo y para la temperatura es un detector de temperatura resistivo (RTD).

B) Acondicionamiento

En esta sección son acondicionadas las señales que proporcionan los sensores para así poder ser registrados adecuadamente por el convertidor A/D del microcontrolador, teniendo una óptima lectura de las señales sensadas.

C) Microcontrolador o CPU

Esta es una de las principales partes de la tarjeta que desarrollamos ya que en él se realizan tanto las lecturas de las señales sensadas como también el procesado de las mismas, para poder realizar la adquisición y envió a través del puerto de comunicación que posee este microcontrolador.

D) Periféricos de comunicación

Son los periféricos por los cuales el microcontrolador tiene comunicación con el exterior, el periférico I^2C es por el que se adquiere el dato de la hora y fecha a través del módulo del reloj en tiempo real (RTC), mientras que por la EUSART son enviados los datos y es recibida la configuración del microcontrolador a través de una hyperterminal o el software de adquisición.

E) Fuente de Alimentación

Esta es una parte fundamental de la tarjeta electrónica ya que el módulo del convertidor A/D del microcontrolador tiene como voltaje de referencia el voltaje de polarización, por lo tanto es importante tener un valor de voltaje constante y fijo en 5 [V]. Así mismo, es necesaria una alimentación ±12 [V] para el uso de los amplificadores operacionales y la electrónica utilizada en las etapas de acondicionamiento.

3.3. Tipo de Sensores para Voltaje, Corriente y Temperatura

Para la selección de los sensores se tomó en cuenta la magnitud de las variables que deseamos medir, que la respuesta del sensor sea lo más lineal posible en el rango de medición y que combinen los conceptos de exactitud y precisión. A continuación se describirá la selección de los sensores utilizados.

3.3.1. Sensor de Voltaje

Lo que se busca es sensar la magnitud del voltaje que reciben las estaciones sismológicas al igual que monitorear los equipos más importantes como son el regulador de voltaje y el rectificador.

Este voltaje es una señal sinusoidal con una magnitud de 127 [V] y una frecuencia de 60 [Hz], estos datos son proporcionados por la Comisión Federal de Electricidad.

Para hacer estas mediciones proponemos la implementación de un circuito divisor de voltaje, Fig. 3.5, con un bajo consumo de corriente, que nos ofrecerá una señal proporcional en magnitud al voltaje que se desea medir y que posteriormente acondicionaremos para su digitalización.

Tomando en consideración lo anterior utilizaremos el diseño de nuestro divisor de voltaje para una impedancia total de 1 [MΩ]. Considerando este valor tendremos una corriente circulando a través del arreglo resistivo de 127 [μA], el cual disipa una potencia de:

$$P=V*I = 127[V]*127[\mu A]= 16.13\ [mW]$$

Con este bajo consumo de potencia podemos garantizar que nuestro sistema no influye en las lecturas de los demás sensores; a continuación se muestra los cálculos respectivos al divisor de voltaje.

Si consideramos que necesitamos una salida de 1 [V_{rms}] y nuestra entrada máxima esperada es de 130 [V_{rms}] y sabiendo que nuestra resistencia total es de 1 [MΩ] tenemos que:

$$R_T = R_1 + R_2$$

$$R_2 = \left(\frac{V_1}{V_T}\right) R_T$$

Si sabemos que:

V_1=1 [V_{rms}]
V_T=130 [V_{rms}]
R_T=1 [MΩ]

Sustituyendo los valores anteriores en la ecuación anterior, obtenemos un valor de R_2=7.69 [kΩ] y una R_1=992.31 [kΩ], el divisor de tensión se ve representado en el diagrama de la Fig. 3.3 mostrado a continuación.

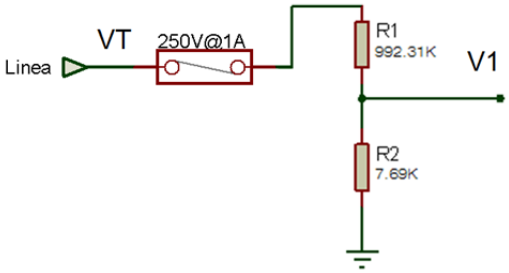


Fig. 3.3. Divisor de tensión empleado como sensor de voltaje.

Para facilitar la implementación y garantizar un ajuste manual en los valores se decidió utilizar trimpots (Fig. 3.4) que nos permitan la obtención de los valores antes mencionados, asegurando así su buen funcionamiento.

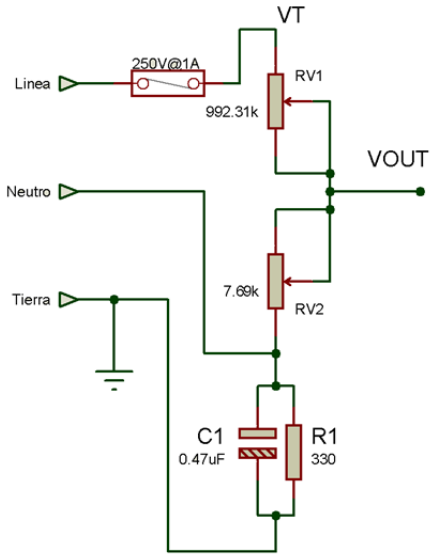


Fig. 3.4. Divisor de tensión usando Trimpots.

3.3.2. Sensor de Corriente

Para este sensor se toma la consideración de que se requiere medir el consumo total o parcial de una estación sismológica, para ello se tomó en cuenta que la corriente a medir es alterna con una frecuencia de 60 [Hz] y no sobrepasa los 25 [A] en la gran mayoría de las estaciones.

Se determinó que el sensor a utilizar fuera de tipo de efecto de campo ya que son muy utilizados en la industria para sistemas de monitoreo de energía, sistemas de control entre otras aplicaciones, ya que son muy eficientes, bajo costo, una alta precisión y un desfasamiento muy pequeño de la señal.

Un sensor de efecto de campo (Fig. 3.5) funciona mediante el flujo constante de una corriente a través de un conductor, la cual induce un campo magnético en el núcleo de material ferromagnético de nuestro sensor. Cuando este fenómeno sucede el dispositivo de efecto de campo produce una diferencia de potencial (Voltaje) proporcional a la corriente que se desea medir.

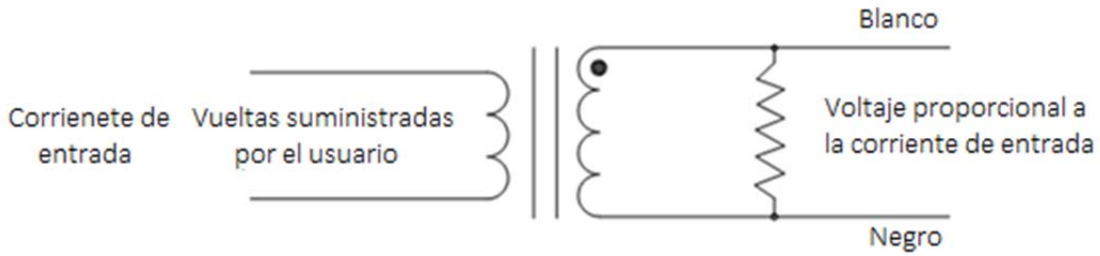


Fig. 3.5. Modelo de sensor de efecto de campo.

El sensor de corriente Split-Core AC, cumple con todo lo antes mencionado, cabe destacar que se seleccionó el modelo SCT-0400-025[1] (Fig. 3.6) ya que es capaz de medir una corriente máxima de 25 [A] correspondiente a un valor de voltaje de 0.333 [V_{RMS}] que proporciona el sensor.

Fig. 3.6. Sensor de corriente SCT-0400-025.

[1].- http://mexico.newark.com/magnelab/sct-0400-025/current-transformer-splitcore-25a/dp/76R0503

A partir de lo antes mencionado se obtiene una expresión $V = f(i)$, que es la función que representa al voltaje proporcionado por el sensor con respecto a la corriente que fluye a través del conductor y es la siguiente:

$$V_{out} = \frac{(0.333)I_{in}}{25}$$

En la siguiente Fig. 3.7 mostramos la respuesta del sensor de corriente obtenida de la expresión anterior.

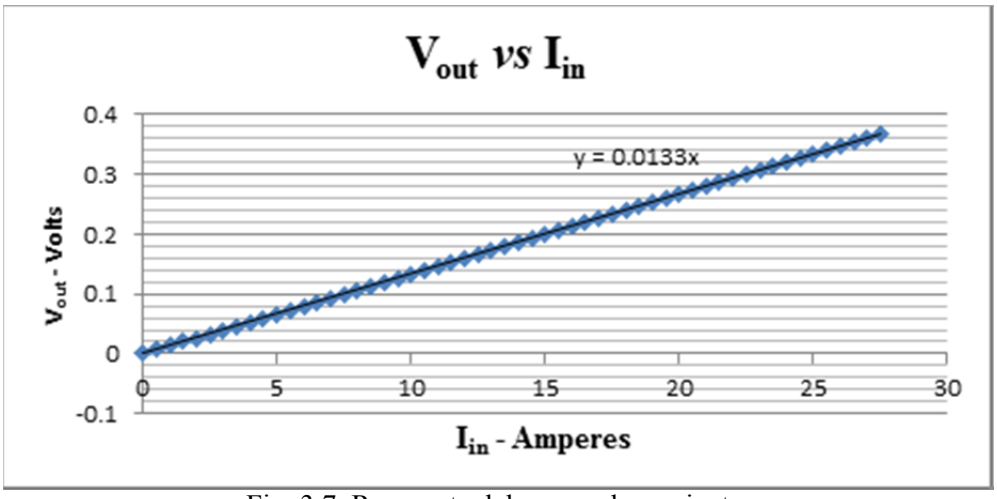


Fig. 3.7. Respuesta del sensor de corriente.

3.3.3. Sensor de Temperatura

La red de estaciones sismológicas se encuentra distribuida a lo largo del territorio nacional y esto significa que las condiciones bajo las que están sometidas son muy variadas. La temperatura es una variable que juega un papel muy importante en la electrónica de un sistema y por eso es que requerimos realizar lecturas de la misma.

Si tomamos en cuenta los mínimos y máximos valores de temperatura, que vemos en el Atlas Climático de México, podemos encontrar que a lo largo de nuestro país el rango de temperaturas oscila entre los 40 y -5 [°C] promedio con extremos de hasta 60 y -15 [°C] en algunas partes del país.

Para realizar estas mediciones hemos elegido un sensor de tipo resistivo RTD (resistance temperature detector) de platino ya que presenta una excelente linealidad y estabilidad en el rango de temperaturas en que deseamos medir. Este dispositivo entre sus características presenta una constante de tiempo menor a 4 segundos lo que permite realizar las lecturas necesarias ante los cambios de temperatura que se presenten en los sitios, ya que se desea medir solamente temperatura ambiente.

Estos sensores están basados en la variación de la resistencia de un conductor con la temperatura. Su símbolo es el siguiente, en el que se indica una variación lineal con coeficiente de temperatura positivo.

Al calentarse un metal habrá una mayor agitación térmica aumentando la resistencia. A mayor temperatura, mayor agitación, y mayor resistencia.

La variación de la resistencia del dispositivo con respecto a la temperatura se puede expresar de la siguiente manera:

$$R_T = R_0(1 + AT + BT^2 - 100CT^3 + CT^4)^{[2]}$$

donde:

R_T = Resistencia [Ω] a temperatura T [°C]
R_0 = Resistencia a 0 [°C]
T = Temperatura [°C]

$$A = \alpha + \frac{\alpha\delta}{100} \qquad B = \frac{-\alpha\delta}{100^2} \qquad C_{T<0} = \frac{-\alpha\beta}{100^4}$$

α = 0.00375 [°C]
δ = 1.605 [°C]
β = 0.16 [°C]

A partir de las expresiones anteriores obtenemos la gráfica de la Fig. 3.8 en la que podemos observar su comportamiento el cual es lineal en un amplio rango de temperaturas y se adapta perfectamente a nuestra necesidad.

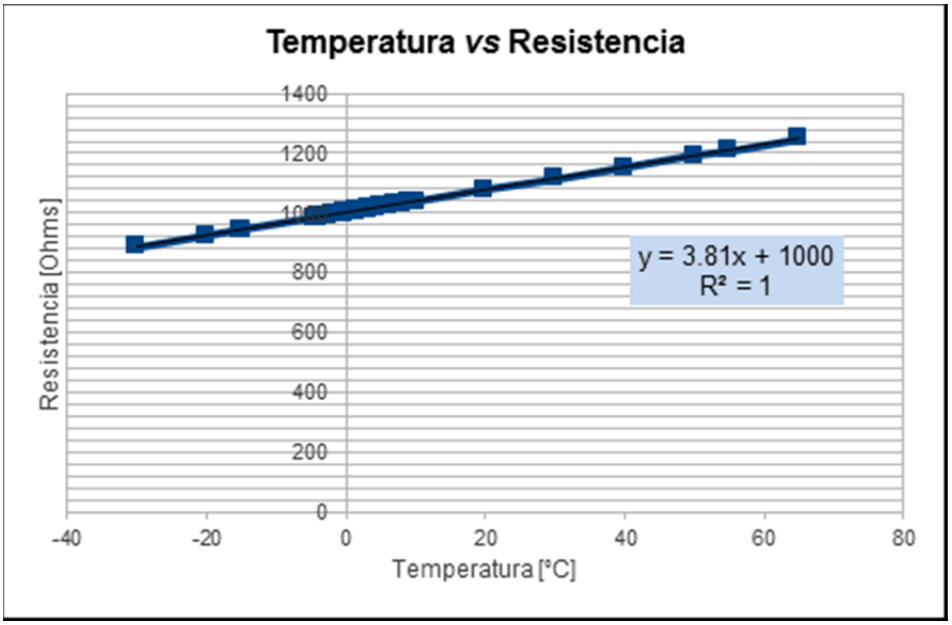


Fig. 3.8. Grafica de Temperatura *vs* Resistencia.

[2].-http://www.honeywellscportal.com//product%20page?pr_id=23055

A continuación en las figuras 3.9 y 3.10 se muestran algunos ejemplos de los elementos que conforman un sensor RTD.

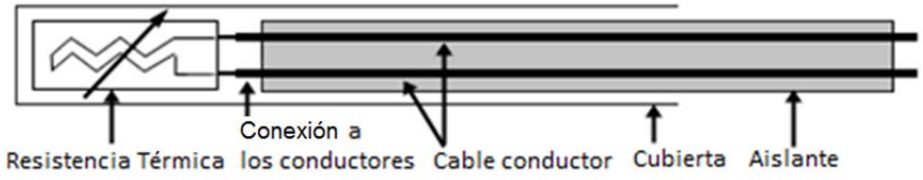


Fig. 3.9. Diagrama interno de un sensor RTD.

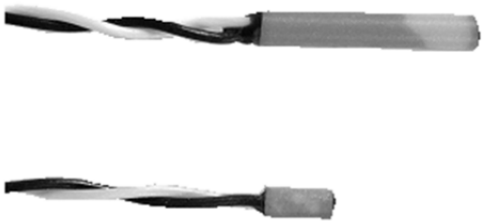


Fig. 3.10. Encapsulado de un sensor RTD.

3.4. Acondicionamiento de las señales

El acondicionamiento de señal se refiere al proceso de linealización de la señal, el filtrado de la misma, la amplificación de la señal de salida, etc. Dependiendo del tipo de sensor y la respuesta obtenida, deberemos enfocarnos en la mejor forma de adecuar la señal para poder homogeneizar las entradas del convertidor A/D.

En el proceso de acondicionamiento de las señales el uso de amplificadores operacionales y de instrumentación es conveniente debido a que estos dispositivos en sus diferentes configuraciones nos ofrecen tanto el filtrado de la señal como la amplificación requerida de la misma, además de funcionar como una interfaz entre el sensor y el digitalizador para acoplar las 2 etapas.

Amplificador de Instrumentación

El amplificador de instrumentación es un amplificador diferencial, cuya ganancia puede establecerse de forma muy precisa y que ha sido optimizado para que opere de acuerdo con sus propias especificaciones aun en un entorno hostil. Es un elemento esencial de los sistemas de medida, en los que se ensambla como un bloque funcional que ofrece características propias e independientes de los restantes elementos con los que interacciona.

Estos circuitos amplifican la diferencia entre dos señales de entrada y rechazan cualquier señal que sea común a ambas señales. Se utilizan principalmente para amplificar señales diferenciales muy pequeñas en muchos procesos industriales, medición, adquisición de datos y aplicaciones médicas.

Estos circuitos cumplen los siguientes requisitos generales:

- Ganancia: seleccionable, estable y lineal.
- Entrada diferencial: con CMMR alto.
- Error despreciable debido a las corrientes y tensiones de offset.
- Impedancia de entrada alta.
- Impedancia de salida baja.

Otra parte importante para el acondicionamiento de las señales en este proyecto es el uso de un rectificador ideal o convertidor de Corriente Alterna a Corriente Directa (CA-CD), para describir un rectificador ideal o CA-CD comenzaremos hablando del rectificador de precisión.

Rectificadores de Precisión

Un rectificador de media onda (RMO) es un circuito que pasa sólo la porción positiva (o sólo la porción negativa) de una onda, mientras que bloquea la otra porción. La característica de transferencia del RMO, está dada por:

$$\begin{cases} v_O = v_I & para\ v_I > 0 \\ \\ v_O = 0 & para\ v_I < 0 \end{cases}$$

Un rectificador de onda completa (ROC), además de pasar la porción positiva, se invierte y después pasa también la porción negativa. Su característica de transferencia es, $v_O = v_I$ para $v_I > 0$, y $v_O = -(-v_I)$ para $v_I < 0$, o en forma más concisa,

$$v_O = |v_I|$$

Un ROC también se denomina *circuito de valor absoluto*.

Rectificador de media onda

El análisis del circuito de la Fig. 3.11 se facilita si consideramos por separado los casos $v_I > 0$ y $v_I < 0$.

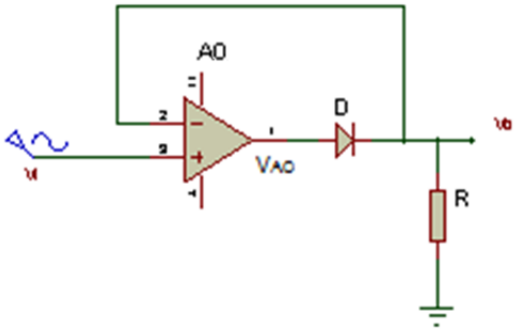


Fig. 3.11. Rectificador de media onda básico.

$v_I > 0$: En respuesta a una estrada positiva, la salida del *AO* v_{AO} también se volverá positiva, encendiendo el diodo por lo tanto la ruta de realimentación negativa mostrada en la Fig. 3.12a. Esto permite aplicar el principio del corto virtual y escribir $v_O = v_I$. Se observa que para hacer que v_O siga a v_I, el *AO* aplica a su salida una caída de diodo por encima de v_O, esto es $v_{AO} = v_O + V_{D(enc.)} \cong v_O + 0.7\,V$. Al colocar el diodo en el ciclo de realimentación se elimina cualquier error debido a su caída de voltaje. Para enfatizar este efecto dramático de la realimentación negativa, la combinación diodo-*AO* se conoce como *superdiodo*.

$v_I < 0$: Ahora la salida del *AO* se vuelve negativa, apagando el diodo y por ende causando que la corriente a través de *R* se convierta en cero. Por lo tanto, $v_O = 0$. Como se ilustra en la Fig. 3.12b, el *AO* ahora está operando en el modo de lazo abierto, y como $v_P < v_N$, la salida se satura en $v_{AO} = V_{OL}$. Con $V_{EE} = -15\,V$, $v_{AO} \cong -13\,V$.

a) b)

Fig. 3.12. Circuitos equivalentes del RMO básico para entrada a) positiva y b) negativa.

Rectificadores de onda completa

Una forma de sintetizar el valor absoluto de una señal es mediante la combinación de la misma señal con su versión rectificada de media onda invertida en una relación 1 a 2, como se muestra en la Fig. 3.13.

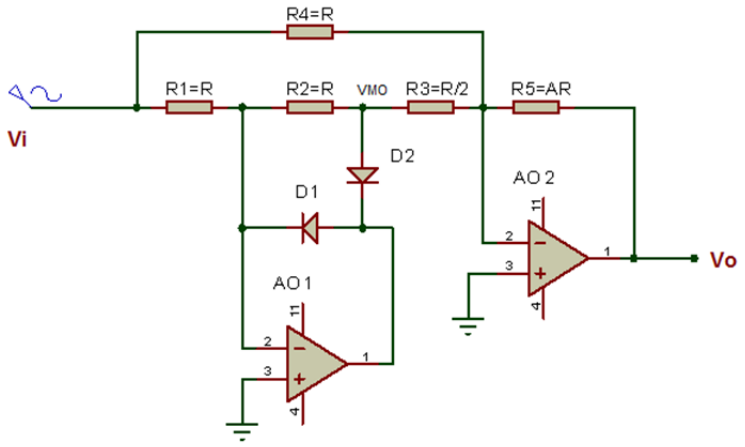


Fig. 3.13. ROC de precisión o circuito de valor absoluto.

Aquí AO_1 proporciona rectificación de media onda inversora, y AO_2 suma a v_1 y a la salida RMO v_{MO} en una relación 1 a 2 para obtener $v_O = -(R_5/R_4)v_I - (R_5/R_3)v_{MO}$. Considerando que $v_{MO} = -(R_2/R_1)v_I$ para $v_I > 0$, y $v_{MO} = 0$ para $v_I < 0$, se puede escribir:

$$\begin{cases} v_O = A_p v_I & para\ v_I > 0\ V \\ v_O = -A_n v_I & para\ v_I < 0\ V \end{cases}$$

donde

$$A_n = \frac{R_5}{R_4} \qquad A_p = \frac{R_2 R_5}{R_1 R_3} - A_n$$

Se desea que ambas mitades de la onda de entrada sean amplificadas por la misma ganancia $A_p = A_n = A$, para ello se puede escribir $v_O = A_{v_I}$ para $v_I > 0$ y $v_O = -A_{v_I}$ para $v_I < 0$, o en forma concisa

$$v_O = A|v_I|$$

Convertidores de CA-CD

La aplicación más común de los circuitos de precisión de valor absoluto es la conversión de CA-CD, esto es, la generación de un voltaje de CD proporcional a la amplitud de una onda de CA dada. Para realizar esta tarea, primero se rectifica con onda completa la señal de CA, y después esta se pasa por un filtro paso bajas para sintetizar un voltaje de CD. Este voltaje es el *promedio* de la onda rectificada,

$$V_{prom} = \frac{1}{T}\int_0^T |v(t)|dt$$

Donde *v(t)* es la onda de CA y *T* es el periodo. Sustituyendo $v(t) = V_m sen 2\pi f t$, donde V_m es la amplitud del pico y $f = 1/T$ es la frecuencia, se obtiene:

$$V_{prom} = (2/\pi)V_m = 0.637V_m$$

Un convertidor de CA-CD se calibra para que cuando sea alimentado con una señal de CA proporcione como resultado el valor de la *raíz cuadrática media (rms)*,

$$V_{rms} = \left(\frac{1}{T}\int_0^T v^2(t)dt\right)^{1/2}$$

Al asumir $v(t) = V_m sen2\pi ft$ e integrando se obtiene:

$$V_{rms} = V_m/\sqrt{2} = 0.707V_m$$

En la Fig. 3.14 se muestran las relaciones entre los valores promedio, el *rms* y el valor pico. Estas relaciones, que se aplican para las ondas senoidales pero no necesariamente para otras formas de onda, indican que con la intención de obtener V_{rms} a partir de V_{prom}, se necesita multiplicar este último por el cociente $V_{rms}/V_{prom} = 1.11$. Por lo tanto cumple perfectamente con nuestra aplicación.

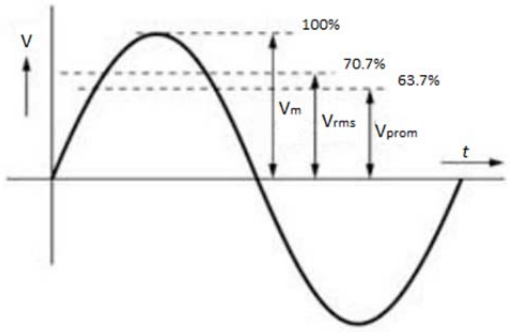


Fig. 3.14. Relación entre V_{rms} y V_p, y entre V_{prom} y V_p.

En la Fig. 3.15 observamos el circuito convertidor de CA-CD que fue utilizado en el diseño de la etapa de acondicionamiento.

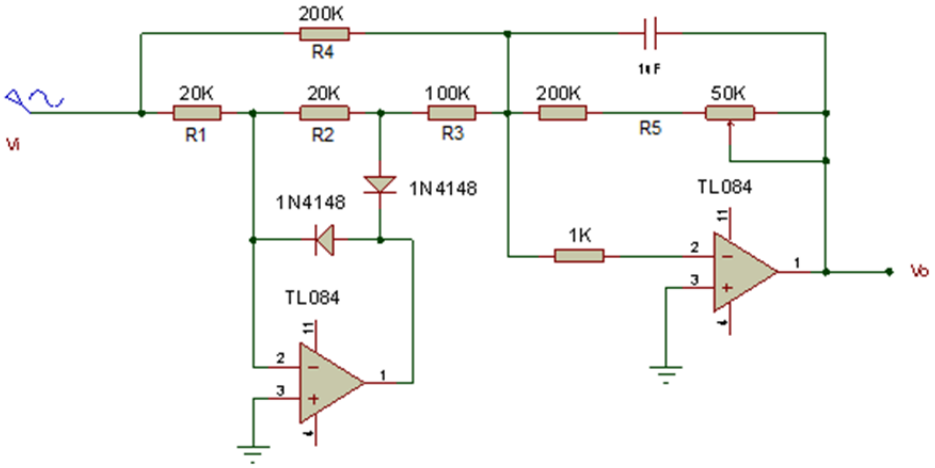


Fig. 3.15. Convertidor de CA-CD[3].

[3].- Franco, Sergio. "Diseño con Amplificadores Operacionales y Circuitos Integrados Analógicos"; sección 9.4. Rectificadores de Precisión.

3.4.1. Acondicionamiento de la señal de Voltaje

A partir de las características del sensor de Voltaje de AC que se muestra en la Fig. 3.6, es necesario amplificar la señal para su posterior entrada al convertidor A/D. La ganancia del amplificador está dada por la relación de la magnitud de la señal de entrada con respecto a la señal de salida deseada, sabemos que la señal máxima a la salida de nuestro sensor es de 1 [V] y nuestra entrada del convertidor A/D es de 0-5 [V] por lo tanto nuestra ganancia requerida es de G=5.

Para alcanzar la amplificación deseada es necesario apoyarnos en un dispositivo activo que nos ofrezca una ganancia ajustable y bajo ruido, al igual que una alta impedancia a la entrada de la señal y una baja impedancia de salida; esto lo podemos conseguir utilizando un amplificador de instrumentación debido a que este componente cumple con todos los requisitos antes mencionados además de presentar un menor error de medida.

A continuación se muestra (Fig. 3.16) el diagrama en el cual se representa de manera gráfica la interconexión de la etapa del sensor con la de acondicionamiento.

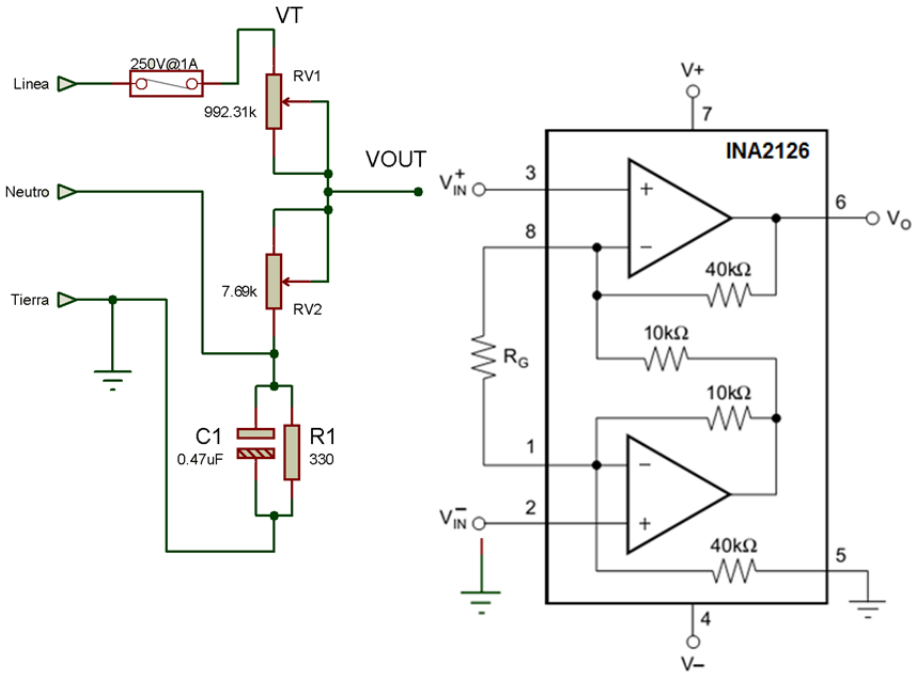


Fig. 3.16. Interconexión de la etapa del sensor de voltaje y el amplificador de instrumentación (INA2126[4]).

[4].- http://www.ti.com/product/ina2126

Dentro de la Fig. 3.17 en el bloque correspondiente al amplificador de instrumentación (INA2126) podemos observar que dada su configuración interna, el voltaje a la salida y la ganancia están representados mediante las siguientes expresiones:

$$V_0 = (V_{IN}^{+} - V_{IN}^{-})G$$

$$G = 5 + \frac{80\ [k\Omega]}{R_G}$$

El fabricante en las especificaciones del amplificador de instrumentación INA2126 menciona que si la resistencia R_G no es conectada la ganancia mínima es de 5, de esta manera solo es necesario polarizar el amplificador de instrumentación e introducir la señal a acondicionar.

Dado que los voltajes de referencia del módulo del convertidor A/D del microcontrolador se encuentran entre 0 y 5 [V] es conveniente hacer uso de un circuito convertidor de CA-CD que nos permita con mayor facilidad la lectura de los valores *rms* de la señal de interés, para poder lograr este objetivo se utilizó el circuito antes mencionado en la Fig. 3.16, el diagrama a bloques del acondicionamiento del sensor de voltaje se observa en la Fig. 3.18.

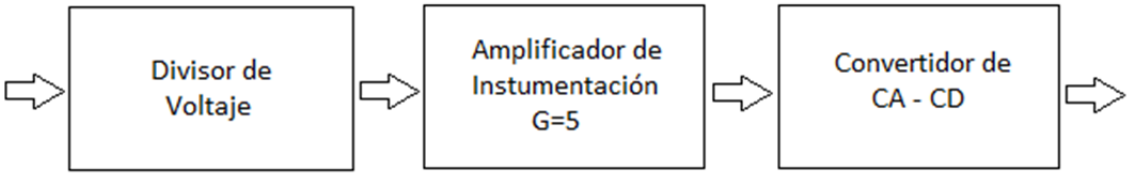


Fig. 3.17. Diagrama a bloques del acondicionamiento de la señal de voltaje.

En la gráfica de la Fig. 3.18 se muestra el resultado teórico de la salida de voltaje en DC en función a una entrada de voltaje AC que deseamos medir.

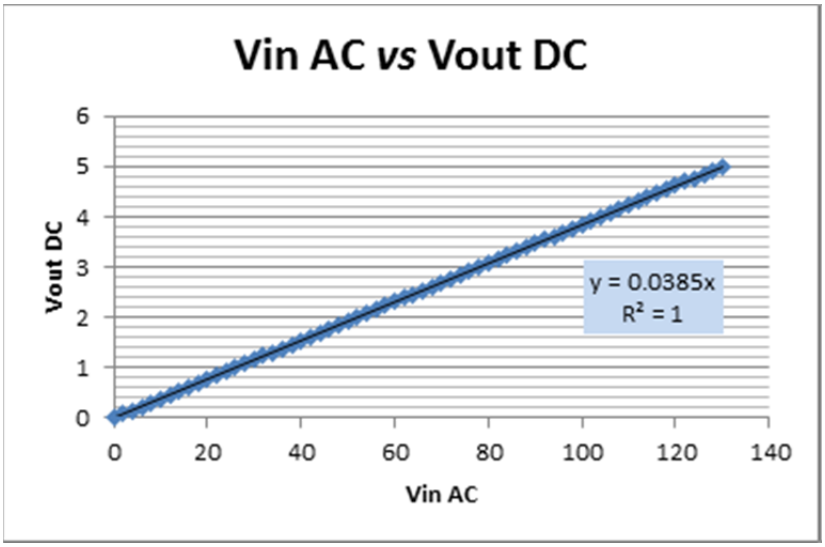


Fig. 3.18. Gráfica de V_{in}AC *vs* V_{out}DC 0-5vdc.

3.4.2. Acondicionamiento de la señal de Corriente

Dada la señal proporcionada por el sensor de corriente solo se requiere de una amplificación para su acondicionamiento. Debido a que el valor máximo entregado por nuestro sensor es de 0.333 [V_{rms}] y la entrada del convertidor A/D tiene un voltaje de referencia positivo de 5 [V_{DC}], por lo tanto la ganancia necesaria es de G=15.015.

La grafica de la Fig. 3.19 se obtuvo a partir de los valores teóricos obtenidos de la función $V_{in} = \left[\frac{(0.333)I_{in}}{25}\right][15.015]$ en la que se muestra la relación del voltaje de entrada al microcontrolador con respecto a la corriente que fluye a través del conductor sensado.

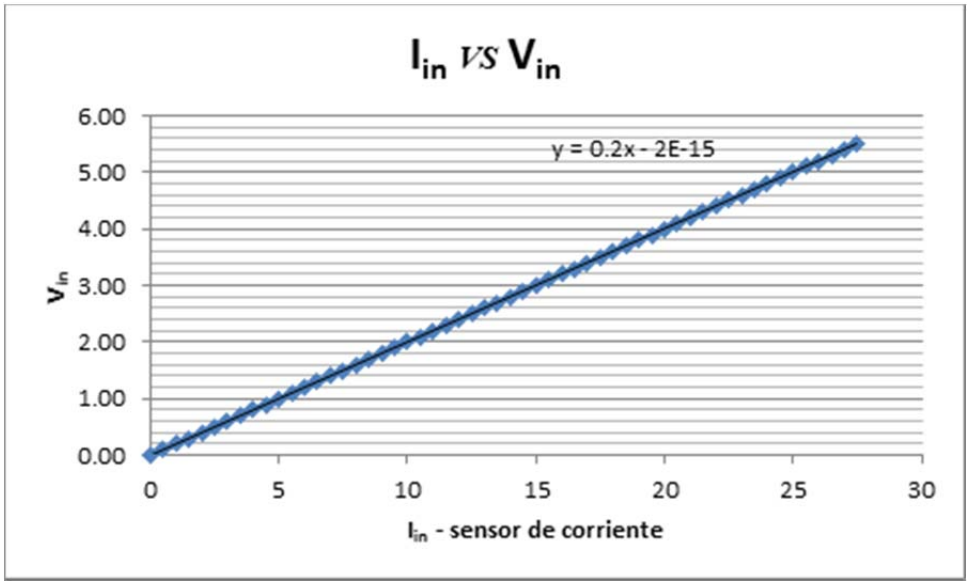


Fig. 3.19. I_{in} *vs* V_{in} del microcontrolador.

Como se mencionó en el punto anterior el dispositivo que cumple con los requerimientos para esta aplicación es el amplificador de instrumentación INA2126 ya que como todo amplificador de instrumentación presenta una alta impedancia a la entrada, una baja impedancia a la salida, una ganancia ajustable además son ideales para trabajar con señales pequeñas y tienen un alto rechazo en modo común.

Sabiendo que:

$$R_G = \frac{80\ [k\Omega]}{(G-5)}$$

Con una G=15.015 obtenemos un valor de R_G=7.988 [kΩ] que se ajustará con la ayuda de un trimpot para así asegurar que la ganancia sea lo más cercana al valor calculado.

A continuación en la Fig. 3.20 se muestra la interconexión entre el sensor de corriente y su etapa de acondicionamiento.

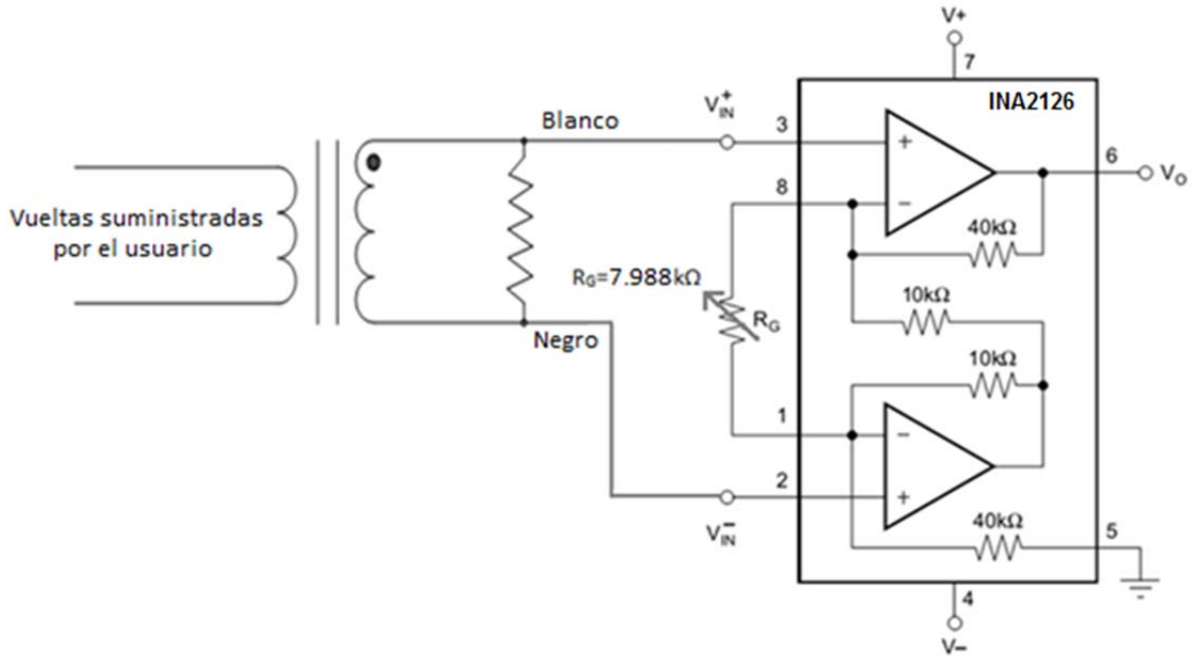


Fig. 3.20. Interconexión de la etapa del sensor de corriente y el amplificador de instrumentación.

De la misma forma que en el acondicionamiento de la señal de voltaje es conveniente hacer uso de un circuito convertidor de CA-CD que nos permita con mayor facilidad la lectura de los valores *rms* de la señal de interés, para poder lograr este objetivo se utilizó el circuito antes mencionado en la Fig. 3.15, el diagrama a bloques del acondicionamiento del sensor de voltaje se observa en la Fig. 3.21.

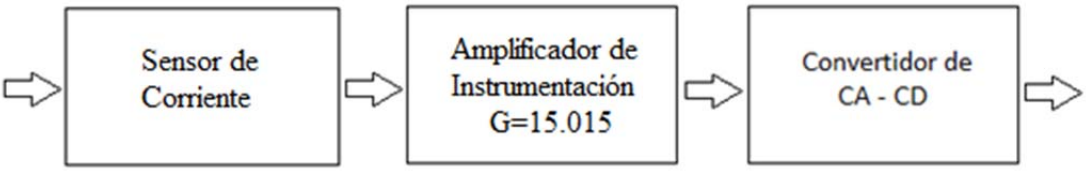


Fig. 3.21. Diagrama a bloques del acondicionamiento de la señal de corriente.

3.4.3. Acondicionamiento de la señal de Temperatura

Para este sensor se utilizó una variación del circuito propuesto por el fabricante que consta de la interconexión de dos *AOs*, uno en configuración no inversora (Fig. 3.22) y el otro es un restador inversor (Fig. 3.23) la entrada del primer *AO* V_{in} es un voltaje fijo que proviene de un divisor de voltaje que proporciona 1 [V_{CD}], debido al rango de operación requerido para este proyecto, se modificó el punto de referencia para que en una temperatura de -15 [°C] nos diera una salida de 0 [V], así pudimos fijar el cero del convertidor A/D con nuestra temperatura mínima, y para nuestra temperatura máxima de 65 [°C] nos proporciona una salida de 0.305 [V], esto se logra cambiando el valor de voltaje que entra en la terminal negativa del *AO* que es un restador inversor (Fig. 3.25), este valor debe de ser 1.943 [V].

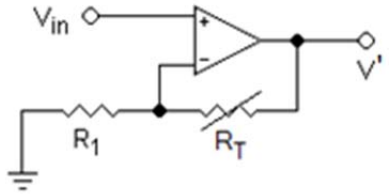


Fig. 3.22. *AO* no inversor con ganancia variable por R_T.

V_{in}= 1 [V]
R_1 = 1 [kΩ]
R_T = *RTD*

$$V' = V_{in}\left(1 + \frac{R_T}{R_1}\right)$$

Sustituyendo:

$$V' = 1 + (R_T * 10^{-3})$$

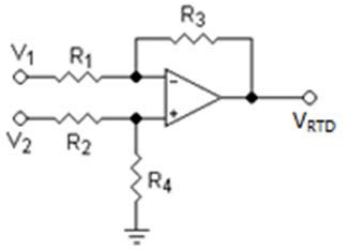


Fig. 3.23. *AO* restador.

V_1 = 1.943 [V]
V_2 = V' [V]
$R_1 = R_2 = R_3 = R_4$ = 10 [kΩ]

$$V_{RTD} = V_2\left(\frac{(R_3 + R_1)R_4}{(R_4 + R_2)R_1}\right) - V_1\left(\frac{R_3}{R_1}\right)$$

dado que todas las resistencias tienen el mismo valor y sustituyendo V_1 y V_2 obtenemos la expresión que nos da la relación entre el sensor *RTD* y el voltaje

$$V_{RTD} = V' - 1.943$$

Así bien, para que el convertidor A/D funcione en todo su rango de operación es necesario proporcionarle un rango de voltaje de 0 a 5 [V], para esto, se debe de amplificar la salida que proporciona el circuito anterior (el valor proporcionado por el mismo va de 0 a 0.305 [V]), esto se lleva a cabo con un *AO* en una configuración no inversora (Fig. 3.24), con una G=16.39

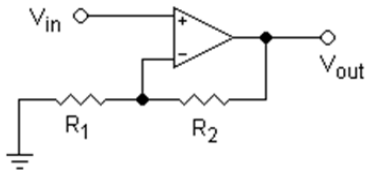


Fig. 3.24. *AO* no inversor para obtener un rango de voltaje de 0 a 5 [V].

$V_{in} = V_{RTD}$

$R_1 = 3.3$ [kΩ]

$R_2 = 50.787$ [kΩ]

$$V_{out} = V_{in}\left(1 + \frac{R_2}{R_1}\right)$$

De este modo obtenemos la expresión que nos da la relación entre el sensor de temperatura y el voltaje que se tiene en la entra del convertidor A/D de nuestro microcontrolador.

$$V_{out} = V_{RTD} * 16.39$$

La gráfica de la Fig. 3.25 muestra la relación del voltaje de salida V_{DC} con respecto a la temperatura registrada, a partir de la expresión $V = [R_T(0.001) - 0.943]16.39$ se obtuvo los valores de la misma.

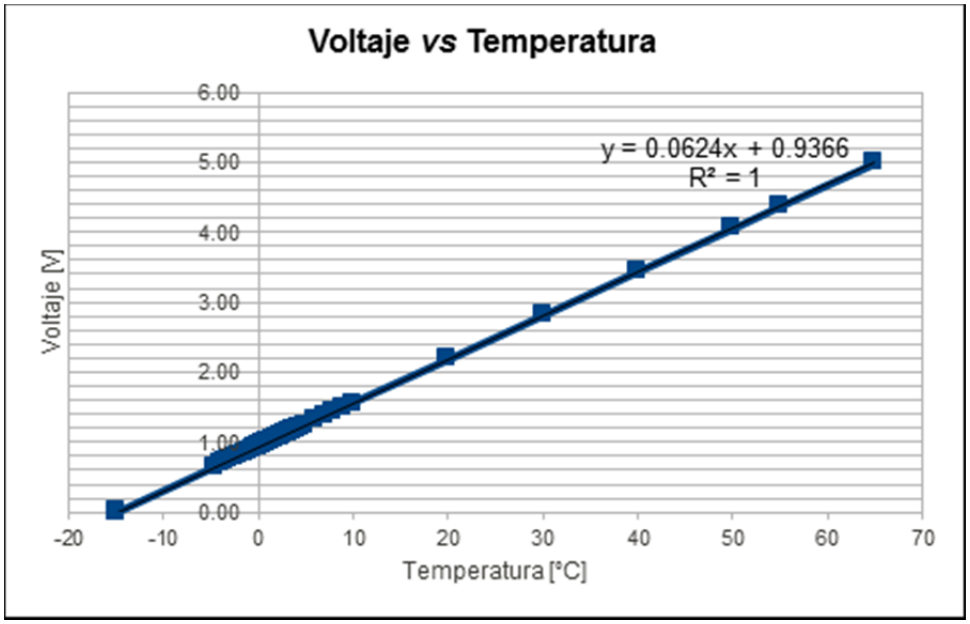


Fig. 3.25. Gráfica de Temperatura *vs* V_{out} 0-5vdc.

3.5. Uso de un microcontrolador para la adquisición de datos

En este tema hablaremos acerca de cómo decidimos en este proyecto hacer uso de un microcontrolador para la adquisición de datos, para ello haremos una descripción general del sistema propuesto. Hablaremos de cada uno de los bloques que integran el sistema de adquisición, haciendo énfasis en las características y la arquitectura del microcontrolador como núcleo del sistema, y de su integración con los bloques antes descritos.

3.5.1. Propuesta del sistema

Hemos propuesto el diseño de una tarjeta electrónica que a través de diversos sensores sea capaz de capturar datos correspondientes al voltaje, corriente y temperatura que se deseen medir y a su vez comunicarlos a computadoras personales de manera local o remota.

Nuestro sistema está dividido en varios bloques y de esta misma manera ha sido construido a lo largo del proceso. Los bloques que la componen son:

- Conexión de los sensores
- Etapa de acondicionamiento
- Microcontrolador (digitalización, procesamiento de los datos, etiquetado de los datos con fecha y hora de adquisición, envío de la información en formato ASCII en comunicación serial)
- RTC *(reloj en tiempo real con respaldo de batería externa y comunicación I^2C)*
- Interfaz RS-232

Y se muestran a continuación en la Fig. 3.26.

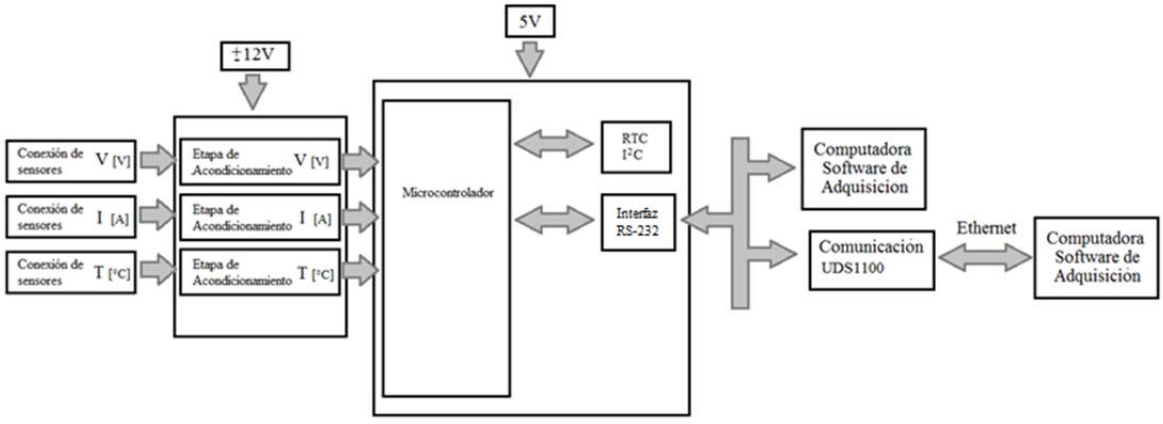


Fig. 3.26. Diagrama a bloques de sistema propuesto.

En este capítulo hablaremos brevemente de que es un microcontrolador, sus ventajas sobre un microprocesador, su arquitectura y posteriormente haremos referencia al microcontrolador Microchip PIC18F4550 y las razones por las que se decidió trabajar con él para este diseño.

¿Qué es un microcontrolador?

Un microcontrolador es un circuito integrado programable que contiene todos los componentes y periféricos de una computadora.

Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es lo que le confiere la denominación de sistema embebido.

El microcontrolador es una computadora dedicada. En su memoria sólo se encuentra almacenado un programa destinado a gobernar una aplicación específica; sus líneas de entrada/salida establece en la comunicación de los sensores y actuadores del dispositivo a controlar, y todos los recursos complementarios disponibles tienen como única finalidad atender sus requerimientos. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada.

Dada la descripción anterior podemos decir que el microcontrolador es una herramienta útil para el diseño de soluciones en tareas específicas en las cuales el proceso de interés puede ser automatizado sin requerir de una supervisión o de la atención dedicada de una persona durante el tiempo que requiera dicha tarea.

Otra de las características importantes de un micro controlador es que en un mismo circuito integrado encontramos periféricos suficientes que reducen el costo del diseño y del tamaño del circuito. En ocasiones existen microcontroladores que se ajustan perfectamente a las necesidades que se desean cubrir y el caso de las tareas que son muy específicas el hardware adicional puede ser fácilmente incorporado al diseño.

El uso de microcontroladores para la automatización y monitoreo de procesos cada día es mayor y hoy en día el número de productos electrónicos en el mercado que funcionan a base de un microcontrolador es incalculable ya que van desde electrodomésticos, juguetes, telecomunicaciones, la industria automotriz, sistemas de seguridad, procesos industriales, en el campo de la medicina, de la docencia y por supuesto en la investigación.

Microprocesador y microcontrolador

Un microprocesador es un circuito integrado que contiene en su interior una unidad central de proceso (CPU) y una unidad de control. Dichas unidades se comunican al exterior con los demás periféricos y memoria a través de buses de datos que pueden ser uno, dos, o más dependiendo de su arquitectura.

Por el contrario un microcontrolador contiene en su interior la CPU, la unidad de control, memoria de datos, memoria de programa y una gran variedad de periféricos que le permiten al microcontrolador ajustarse a las necesidades de cualquier aplicación.

Algunos de los periféricos que pueden encontrarse en un microcontrolador son la unidad aritmética y lógica (ALU), convertidores analógico/digital y digital/analógico, puertos de comunicación en paralelo y/o serial, temporizadores, etc.

Aunque el microcontrolador aparentemente es una herramienta muy poderosa, la elección entre un microcontrolador y un microprocesador depende de la tarea que se desea cubrir.

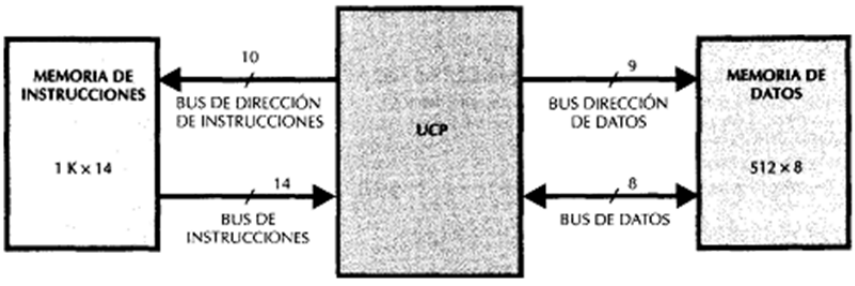
Arquitectura de un microcontrolador

Los microcontroladores poseen componentes que complementan las tareas de la CPU pero no pueden ser modificados o escalados dado que se encuentran en el interior del encapsulado.
Las partes importantes que se encuentran en un microcontrolador son:

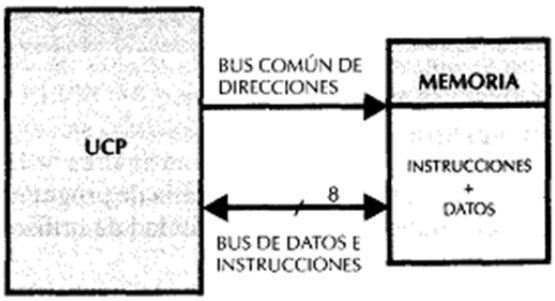
- Procesador.
- Memoria no volátil (que contiene el programa).
- Memoria volátil (para almacenar información temporal).
- Líneas de E/S para comunicarse con el exterior de manera serial o paralela.
- Módulo de conversión analógico-digital y/o digital-analógico.
- Temporizadores.

Existen 2 arquitecturas en los procesadores, Harvard y Von Neumann. En el caso de la arquitectura Von Neumann la CPU se comunica a través de 2 buses (bus de datos e instrucciones y bus de direcciones) a una memoria única en la que se almacenan datos e instrucciones. A diferencia de la arquitectura Von Neumann, en la arquitectura Harvard existen 2 memorias, una memoria de datos y una memoria de instrucciones. Esto se traduce en un mejor desempeño del sistema para muchas aplicaciones debido a que puede existir un paralelismo durante la ejecución de un programa.

En la Fig. 3.27 podemos observar un ejemplo de la arquitectura Harvard (a) y de la arquitectura Von Neumann (b).



Arquitectura Harvard.



b) Arquitectura Von Neumann.

Fig. 3.27. Ejemplo de arquitectura Harvard y Von Neumann.

Microcontroladores PIC

Existen muchos fabricantes de circuitos integrados y por supuesto de microcontroladores. De la misma forma cada fabricante produce un gran número de familias de microcontroladores de las cuales podemos elegir una que cubra perfectamente las necesidades primarias de nuestro diseño.

Microchip ha desarrollado a lo largo de muchos años los PIC, que hoy en día tienen una gran aceptación entre los diseñadores, técnicos, estudiantes y aficionados ya que brindan muchas características que son importantes en la elección del núcleo de nuestro sistema.

Las características más importantes que hemos apreciado nosotros, algunos autores, académicos y diseñadores en el área, son las siguientes:

- Sencillez de manejo.
- Bastante información.
- Parámetros que se encuentran en el promedio (tamaño, voltaje de polarización, bajo consumo entre otros).
- Herramientas de desarrollo accesibles (gratuitas o de bajo costo).
- Disponibilidad en el mercado.
- Compatibilidad del software con todos los modelos de la misma gama.
- Bajo costo de los CI y de las herramientas de programación.

Las familias que existen en el mercado actualmente son:

- Para 8 bits. PIC10, PIC12, PIC16 y PIC18.

- Para 16 bits. PIC24.

- Para 32 bits. PIC32.

Actualmente existen muchas aplicaciones que pueden desarrollarse con sistemas de 8 bits dado que no son aplicaciones en tiempo real o no desempeñan procesos de imágenes o vídeo.

Aplicaciones como la medición de variables físicas para un proceso de monitoreo y/o de control remoto puede ser una tarea muy pequeña y ser ejecutada por un microcontrolador de 8 bits, o lo suficientemente grande para implementar un sistema SCADA, todo depende de la complejidad del proyecto.

En nuestro caso existen variables físicas que requerimos monitorear para establecer las condiciones reales de operación a las cuales se encuentran sometidos los equipos electrónicos dentro de los observatorios sismológicos de la red de estaciones del Servicio Sismológico Nacional.

Las condiciones reales de las cuales hablamos son el voltaje suministrado por la Comisión Federal de Electricidad "CFE", la temperatura en el interior de los cuartos de instrumentos y de sensores, y la corriente que se requiere en algunos puntos específicos.

Dado que la mayoría de las estaciones cuenta con los mismos equipos, como parte de un estándar que se quiere lograr, se desean monitorear las condiciones de operación de los sistemas y contar con una herramienta que nos ayude en el diagnóstico de alguna posible falla.

En condiciones de laboratorio las variables antes mencionadas son muy estables y presentan valores que incluso se acercan a las condiciones que recomienda el fabricante pero la realidad es que todas esas condiciones pueden estar muy por arriba o muy por debajo de ellas pero no cambian de manera precipitada, a menos que se presenten ciertos eventos como lo es una interrupción de la energía eléctrica, por lo que el muestreo de todas las variables no es a una razón muy alta.

Todas estas características, la necesidad de contar con un sistema que sea compatible con equipos de cómputo actuales, desarrollar con elementos que sean de bajo costo y que podamos tener fácil acceso a su información nos llevó a decidirnos por la familia PIC18.

3.5.2. Arquitectura de los microcontroladores PIC

Los microcontroladores PIC fueron los primeros microcontroladores con un set de instrucciones reducido (RISC) y como todos los microcontroladores bajo este esquema ofrecen simplicidad en los diseños permitiendo más características a bajo coste.

Poseen arquitectura Harvard, esto es que en su interior presentan una memoria de datos y una memoria de programa las cuales se encuentran en comunicación con la unidad de control a través de sus respectivos buses, independientes entre las memorias. Esto le permite acceso simultáneo al programa y a los datos ofreciendo un mejor desempeño en la ejecución de las tareas.

Una característica que debemos tomar en cuenta es la segmentación de instrucciones, esta se refiere a dividir la ejecución de las instrucciones en varias fases, al final cada instrucción se realiza en 4 ciclos de reloj (excepto las instrucciones de salto).

Todas las instrucciones son ortogonales por lo que pueden manejar cualquier elemento de la arquitectura como fuente o destino.

Formato de instrucciones constante que permite optimizar el uso de la memoria.

Arquitectura basada en banco de registros, todos los objetos del sistema están implementados como registros.

Las diferentes familias de los PICs se ubican en gamas diferentes, comienzan la familia más baja (PIC12xx), baja (PIC16C5xx), media (PIC16xxx), alta (PIC17xxx) y mejorada (PIC18xxx). Esta última tiene memoria de programa de hasta 1M palabras, sus instrucciones en su mayoría son de 16 bits e incluso hay de 32.

En particular escogimos el modelo del PIC 18F4550 ya que cuenta con las siguientes características:

- 13 convertidores A/D.
- 1 puerto full speed USB 2.0.
- 1 USART.
- 1 puerto SPI/I2C.
- Voltaje de operación de 2 a 5.5 V.

Estas características superiores a los otros microcontroladores de la misma familia, además de las muchas otras ventajas que presentan sobre otros fabricantes.

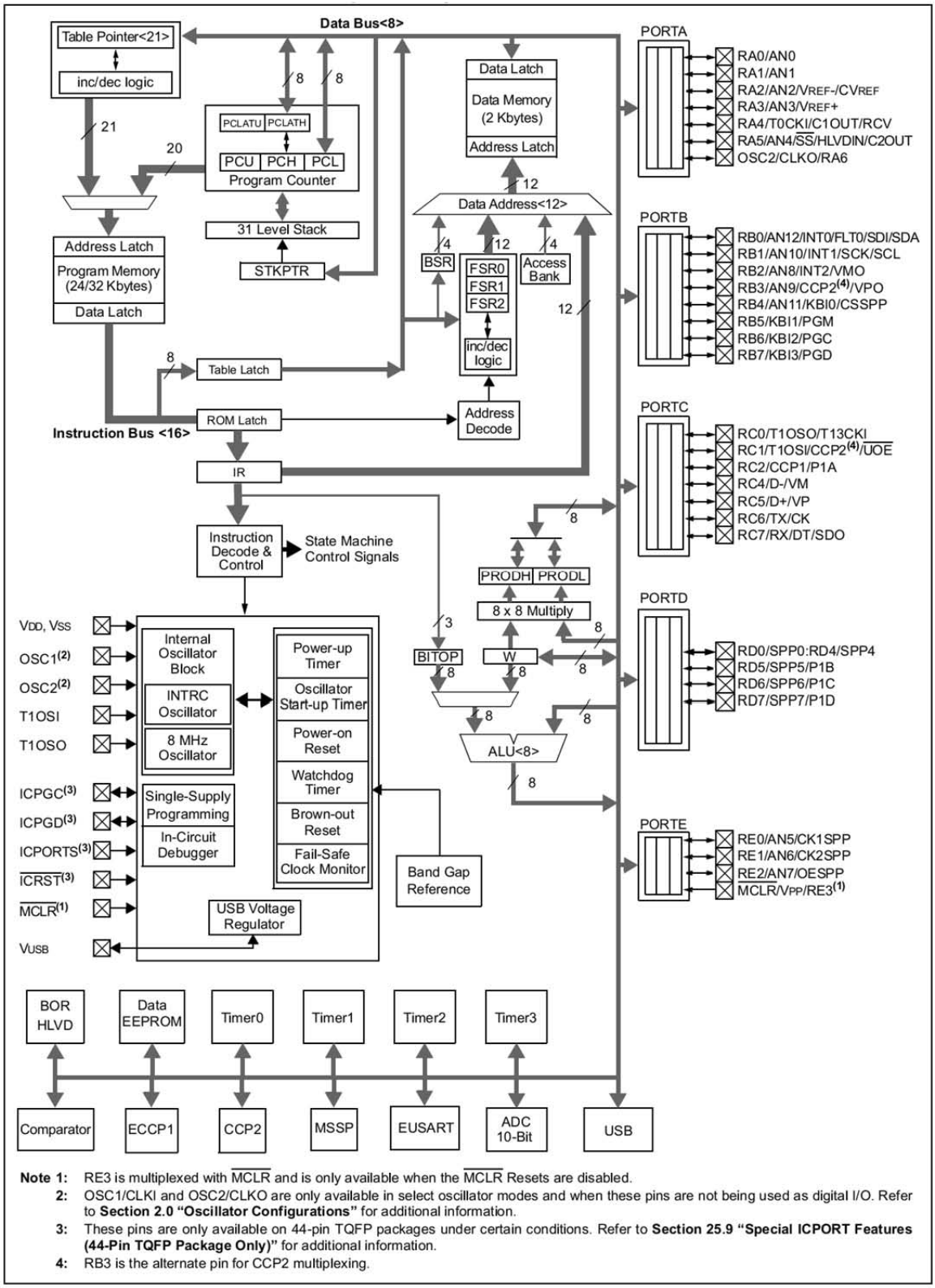
En la siguiente Fig. 3.28 se muestran las especificaciones de los dispositivos que ofrecen características similares dentro de la misma familia.

Features	PIC18F2455	PIC18F2550	PIC18F4455	PIC18F4550
Operating Frequency	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
Program Memory (Bytes)	24576	32768	24576	32768
Program Memory (Instructions)	12288	16384	12288	16384
Data Memory (Bytes)	2048	2048	2048	2048
Data EEPROM Memory (Bytes)	256	256	256	256
Interrupt Sources	19	19	20	20
I/O Ports	Ports A, B, C, (E)	Ports A, B, C, (E)	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers	4	4	4	4
Capture/Compare/PWM Modules	2	2	1	1
Enhanced Capture/ Compare/PWM Modules	0	0	1	1
Serial Communications	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART
Universal Serial Bus (USB) Module	1	1	1	1
Streaming Parallel Port (SPP)	No	No	Yes	Yes
10-Bit Analog-to-Digital Module	10 Input Channels	10 Input Channels	13 Input Channels	13 Input Channels
Comparators	2	2	2	2
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled
Packages	28-Pin PDIP 28-Pin SOIC	28-Pin PDIP 28-Pin SOIC	40-Pin PDIP 44-Pin QFN 44-Pin TQFP	40-Pin PDIP 44-Pin QFN 44-Pin TQFP

Fig. 3.28. Comparativa entre dispositivos similares al PIC18F4550.

Gracias al gran número de pines es posible tener espacio suficiente para interconectar un gran número de dispositivos externos a nuestro microcontrolador sin tener que multiplexar los pines por hardware y software para aprovechar el mayor número de periféricos posible en nuestro microcontrolador.

En la siguiente Fig. 3.29 se muestra la arquitectura del microcontrolador PIC18F4550 y la interconexión de los periféricos a través de su bus correspondiente.



Note 1: RE3 is multiplexed with $\overline{\text{MCLR}}$ and is only available when the $\overline{\text{MCLR}}$ Resets are disabled.
2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.
3: These pins are only available on 44-pin TQFP packages under certain conditions. Refer to **Section 25.9 "Special ICPORT Features (44-Pin TQFP Package Only)"** for additional information.
4: RB3 is the alternate pin for CCP2 multiplexing.

Fig. 3.29. Diagrama a bloques de los periféricos que componen al microcontrolador 18F4550.

Otra virtud de estos dispositivos es que el ambiente de desarrollo es gratuito y que con ayuda de otra paquetería (incluso de diferente fabricante) es posible compilar programas en lenguaje ensamblador (bajo nivel) y lenguaje C.

3.5.3. Programación del Microcontrolador con Lenguaje C, compilador CCS y Ambiente de desarrollo MPLAB

El lenguaje C es un lenguaje de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos. Gracias a estas características es que la programación de un microcontrolador se vuelve menos compleja, ya que le ofrece al programador las herramientas necesarias para acceder a los registros de memoria, puertos periféricos e incluso acceso a nivel de bits de los registros de control del microcontrolador.

Los compiladores de C ofrecen funciones dentro de sus librerías que vuelven las tareas mucho más simples, evitando mostrar código dentro de ellas que puede ser difícil de entender o que en ocasiones no necesitamos tener presente. Un programa codificado en lenguaje C resulta muy útil en la aplicación de microcontroladores, dado que su compilación es bastante eficiente y óptima acercándose a la codificación de lenguaje de máquina. Lo descriptivo de la sintaxis permite elaborar de mejor forma los algoritmos.

Compilador CCS

El compilador CCS ofrece los comandos básicos y las librerías estándar de C, y en particular es un compilador dedicado a la programación de microcontroladores por lo que tiene librerías dedicadas a muchos modelos de las distintas familias de Microchip. Es por ello que se escogió este compilador para realizar la programación de nuestra aplicación.

CCS ofrece además del compilador una ayuda bastante completa con temas acerca de las funciones y ejemplos del uso de ellas. En el archivo de ayuda podemos encontrar una explicación detallada del uso de los puertos, configuración de los módulos periféricos del microcontrolador, configuración de registros, funciones dentro de sus librerías y archivos de ejemplos, que explican y ejemplifican el uso de las funciones para llevar a la práctica su uso dentro de nuestro proyecto en desarrollo.

La redacción del código, la compilación y el grabado del programa dentro del microcontrolador lo realizamos a través de un ambiente de desarrollo integral o Integrated Development Enviroment por sus siglas en ingles IDE, propio de Microchip en el cual se puede elegir si se desea compilar con las herramientas que ofrece el fabricante o con algún otro, y es ahí en donde elegimos a CCS como herramienta de programación.

Ambiente de desarrollo MPLAB

MPLAB IDE v8.80 es un programa que corre en sistema operativo Windows, este IDE ofrece un editor de texto, un compilador, herramientas de depuración y de programación con las cuales se puede grabar el microcontrolador a través del programador, en nuestro caso utilizamos el PICKit3.

Dentro de este programa vamos a crear un proyecto en el que se incluyen el archivo .C que será donde redactemos el código y archivos de cabecera, los archivos de las librerías, la ventana "Output" donde visualizamos los resultados de la compilación y de la programación, y la ventana "watch" que sirve para ver los registros del microcontrolador y las variables declaradas así como su comportamiento cuando se ejecuta una simulación del programa dentro del MPLAB.

En la Fig. 3.30 se muestra una pantalla con la distribución de los elementos básicos que podemos observar dentro de un "workspace" en MPLAB.

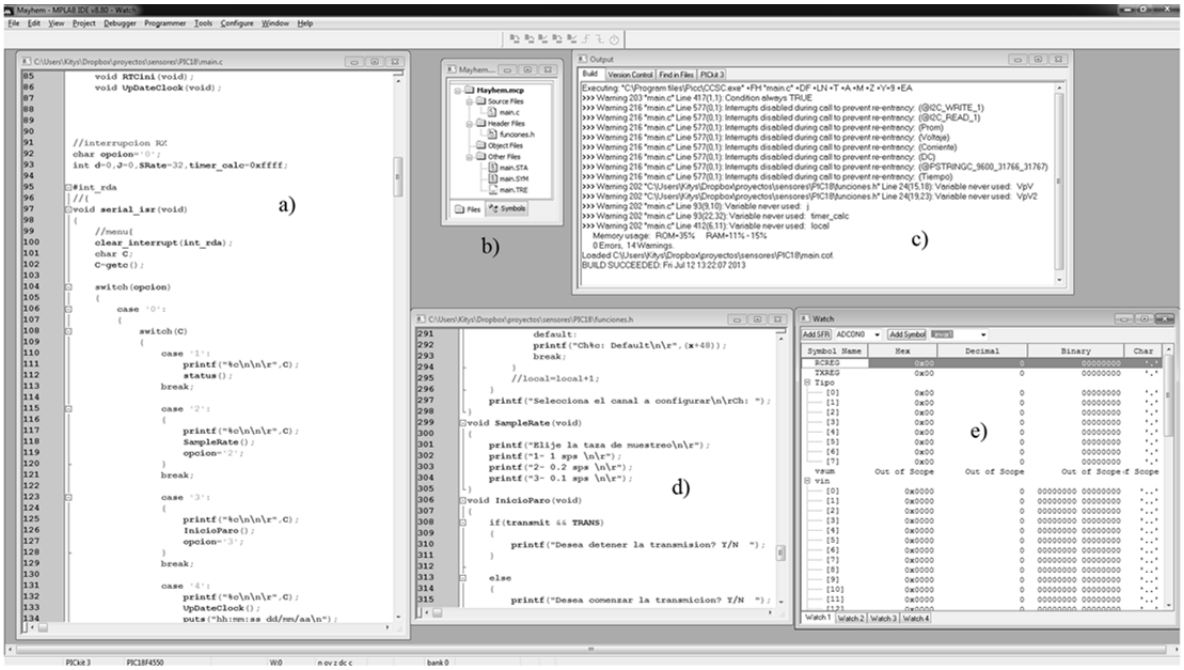


Fig. 3.30. Pantalla capturada de un workspace en MPLAB en el que se muestran las ventanas a) main.c, b) project, c) output, d) archivos .h y e) ventana de herramienta watch.

El uso de un IDE ofrece múltiples ventajas ya que como su nombre lo indica, integra múltiples herramientas que nos facilitan la redacción, la compilación, simulación del código y la programación en el microcontrolador desde un mismo software.

3.5.4. Configuración del microcontrolador

Oscilador

El microcontrolador PIC18F4550 incorpora un oscilador y sistema de reloj diferente que los dispositivos anteriores de la misma familia PIC18F. La adición de módulo USB hace que sea necesaria una fuente de reloj estable que sea compatible con las especificaciones para el USB de baja y alta velocidad.

Para adaptarse a estos requisitos, el PIC18F4550 incluye un nuevo bloque para proporcionar un reloj de 48 [MHz] que es necesario en la operación del USB de alta velocidad.

Dado que existe una fuente de reloj primaria, un sistema adicional de pre-escaladores y pos-escaladores se han añadido para poder obtener una gran variedad de frecuencias a las que puede operar el microcontrolador. Otras características del oscilador es el bloque del oscilador interno el cual nos brinda la oportunidad de reducir el costo del diseño.

El oscilador interno ofrece como frecuencia máxima 8 [MHz] y como mínima 31 [kHz] los cuales pueden ser elegidos a través de las funciones que ofrece el compilador CCS a través de su librería 18F4550.h.

Para este proyecto hemos utilizamos el reloj de 8 [MHz] y el código utilizado para esta configuración es la siguiente:

```
#include<18F4550.h>          //librería del microcontrolador usado
.
.
.
#FUSES INTEC_IO              //Habilitar la opción de oscilador interno
.
.
.
#use delay(clock=8000000)  /*Indica al compilador la velocidad del procesador y permite el uso de las funciones integradas: delay_ms () y delay_us ().*/
void main()
{
setup_oscillator(OSC_8MHZ|OSC_INTRC|OSC_PLL_OFF);          /*la configuración del oscilador es lo primero que hay que configurar en la función main*/
.
.
.
```

Módulo de Conversión Analógico Digital

El módulo de conversión analógico digital de este microcontrolador cuenta con 13 entradas y con una resolución de 10 bits. Posee 3 registros de control y 2 registros para almacenar el resultado de la conversión.

El voltaje de referencia puede ser seleccionado por software entre el voltaje de alimentación del dispositivo o de los pines Vref+ y Vref-, y el resultado de la conversión es a través de aproximaciones sucesivas.

La conversión inicia configurando el registro de control ADCON1 en el que se especifica el voltaje de referencia, después en el registro de control ADCON0 se indica el canal del cual se va a tomar la lectura para realizar la conversión. Posteriormente se enciende el modulo A/D, se espera el tiempo de conversión que es de aproximadamente de 3 [μs] y se inicia la conversión.

Al finalizar la conversión el bit GO/DONE cambia de estado y en ese momento es posible leer el resultado de los registros ADRESH:ADRESL.

Dentro de los registros de configuración es importante indicar que la resolución del convertidor A/D es de 10 bits y esto se hace justificando el resultado a la derecha, quedando la parte alta de la palabra en ADRESH y la parte baja en ADRESL. De esta manera aprovechamos la máxima resolución en la conversión.

A continuación vemos como se encuentran redactadas las líneas de código para la configuración inicial de nuestro modulo A/D.

```
#deviceadc=10      // en los archivos de cabecera se utiliza este comando, automáticamente el compilador hace la justificación a la derecha para utilizar los 10 bits del convertidor A/D y usar ambos registros, ADRESH:ADRESL


//main(){
setup_adc(ADC_CLOCK_INTERNAL);          // Built-in A/D setup function
setup_adc_ports(AN0_TO_AN7);                    /*se habilitan los 8 primeros puertos del convertidor A/D, puesto que son los que comenzamos a utilizar en este momento, si es necesario, quedan disponibles del AN8 y AN9 para ser utilizados.*/
//}
```

Cabe destacar que para utilizar los convertidores AN10, 11 y 12 es necesario hacer configuraciones en software y hardware que permitan multimplexar los pines entre AN10 y AN12 con el puerto de comunicación SPI/I^2C que actualmente usamos para el uso del reloj en tiempo real. Es por ello que nos limitamos al uso de 8 canales en nuestro diseño y con posibilidad de 10.

Comunicación Serial

La EUSART o el módulo de comunicación serial, genéricamente conocido como Interfaz de Comunicación Serial (por sus siglas en ingles SCI) puede ser configurado de dos maneras, de forma asíncrona full duplex para comunicarse con terminales CRT y computadoras personales, y síncrona half duplex para establecer comunicación con dispositivos periféricos tales como convertidores A/D o D/A, memorias EEPROMS, etc.

Las terminales que están destinado para la entrada y salida de datos de este módulo están compartidos con el PORTC por lo que hay que realizar la programación correspondiente para poder ser utilizados.

Para lograr los niveles de voltaje requeridos por el estándar RS-232 nos apoyamos al CI MAX232 con el cual es posible realizar la comunicación entre el microcontrolador y una computadora a través de un puerto serie.

En nuestro programa el envío de datos se realiza con el llamado de las siguientes funciones:

- printf()
- putc()
- puts()

en las cuales se especifica que se desea enviar, una cadena de texto o un carácter. Para hacerlo solo es necesario mandar llamar a la función en el momento que se desea mandar la información, dentro de la función main() o de alguna función cualquiera.

Para el caso de la recepción de la información, dado que no se sabe cuándo será transmitida (comunicación asíncrona) el uso de las funciones de recepción deberán ir dentro de una rutina de interrupción, la cual será ejecutada cuando la bandera de interrupción correspondiente a la comunicación serie este en alto. Dentro de dicha instrucción se deberá limpiar la bandera de interrupción para asegurar que no haga llamadas a interrupción en falso y la adquisición del dato recibido. Además de ejecutar la operación que deba ser realizada en consecuencia de la recepción de la información.

A continuación mostramos la configuración del módulo de comunicación serial y la función de interrupción en donde se usa la función de recepción de datos.

```
#use delay(clock=8000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)        /* en esta función declaramos el uso de la comunicación serial, el baudaje y los pines de recepción y transmisión (RX y TX respectivamente)*/
#int_rda
//{
void serial_isr(void)   //función de interrupción por recepción de datos por comunicación serie
{
clear_interrupt(int_rda);
char C;
C=getc();     /*getc(): esta función espera un carácter que se recibe a través del pin de recepción y devuelve el dicho carácter como resultado*/

void main()
{
enable_interrupts(int_rda);    //Declaración de interrupciones en la función main
enable_interrupts(global);
```

Dentro de nuestro programa hacemos uso de otro tipo de interrupción, esta es con el uso de un timer, en el caso hicimos uso del TIMER1.

El TIMER1

Este timer se incrementa en uno con cada ciclo de instrucción, es decir con cada cuatro ciclos de reloj del oscilador principal. Este timer está configurado actualmente en modo de 16 bits y el pre escalador en 1, y al momento de su desborde la bandera de interrupción correspondiente es puesta en 1 (TMR1F=1), de esta manera la rutina de interrupción es ejecutada.

El diagrama a bloques del Timer1 es el siguiente. Fig. 3.31.

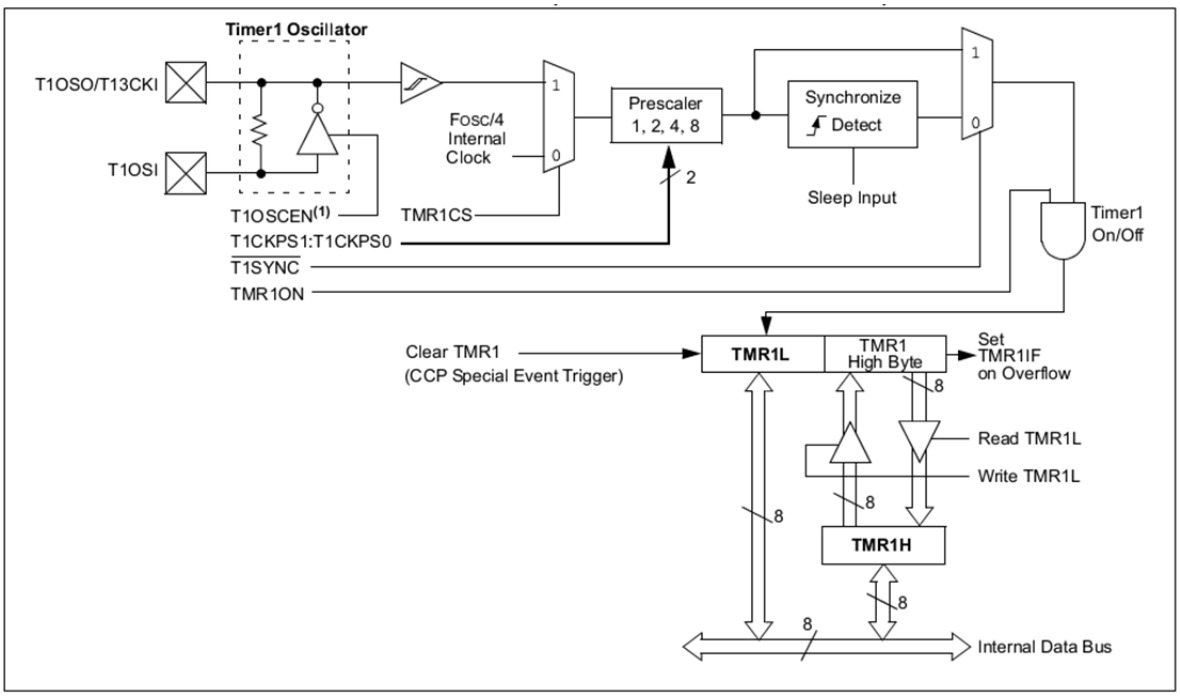


Fig. 3.31. Diagrama a bloques del Timer1 en modo de 16 bits.

El código con la configuración es el siguiente:

```
setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);   // clk interno y preescalador en 1
enable_interrupts(INT_TIMER1);        //              habilita la interrupción del TIMER1
```

posterior a esta configuración, lo único necesario para establecer el tiempo que deseamos establecer, debemos almacenar en TIMER1L:TIMER1H el dato correspondiente a los ciclos de reloj necesarios para alcanzar este tiempo.

Por ejemplo: sabemos que tenemos un reloj interno de 8 [MHz], que cada 4 ciclos de reloj el timer es incrementado en 1, sabemos que el pre escalador tiene un valor de 1 y dado que es un registro de 16 bits, quiere decir que ejecuta 65535 instrucciones antes de desbordarse el registro del timer. Todo esto significa que en un segundo el timer se desborda aproximadamente unas 30.5180437 veces. Si ejecutáramos dentro de una rutina 30.5 veces el desborde del timer tendríamos una interrupción cada 0.99940875 segundos esto significa

que existe un error que provocaría una deriva de tiempo que en algún momento sería necesario corregir.

Nosotros elegimos un número entero que representa el número de veces que deberá ejecutarse la rutina y así eliminar este error.

Si tenemos que:

$$\frac{8\ [MHz]}{4} * \frac{1}{65535} = 30.5180437\ desbordamientos/segundo$$

Despejando y aproximando este valor a 32 tenemos que:

$$\frac{8\ [MHz]}{4} * \frac{1}{32} = 62500\ ciclos/desbordamiento$$

Por lo que se necesitan 62500 ciclos de instrucción para desbordar el timer y una rutina que lo haga 32 veces para tener la ejecución de una rutina en 1 segundo exacto.

El código necesario para esta configuración es el siguiente:

```
set_timer1(0x0BDB);
```

nótese que 0x0BDB es igual a 3035, este es el dato que debe cargarse en el timer ya que los ciclos necesarios para desbordarlo son 62500.

$$65535 - 62500 = 3035 = \boldsymbol{0x0BDB}$$

Módulo de comunicación I^2C

El sistema está diseñado para contar con un reloj de tiempo real que cuenta con una batería de respaldo la cual garantiza que el dispositivo sea configurado una sola vez y mantenga actualizada su información. Este dispositivo es el DS1307, actualmente es el único dispositivo que se comunica a través del bus I^2C dentro de nuestro sistema.

Reloj de tiempo real DS1307

El DS1307 es un reloj en tiempo real que únicamente requiere de manera adicional un cristal para generar la base de tiempo con el cual es posible contar segundos, minutos, horas, así como fecha y año.

Este Circuito Integrado cuenta con 56 bytes de memoria RAM los cuales almacenan la configuración inicial y mantienen su información gracias a una batería de respaldo.

Actualmente este dispositivo lo integramos al sistema como un módulo, el cual es posible colocar y retirar de manera sencilla ya que se puede comprar en una presentación lista para ser conectada a través de sus terminales de conexión "headers".

El RTC (Fig. 3.32) por sus siglas en inglés (Real Time Clock) cuenta con una interfaz serial I^2C con la cual podemos establecer comunicación entre él y el microcontrolador. El RTC opera como un esclavo dentro del bus serial. El acceso se realiza implementando la condición de INICIO seguida de la dirección con la que se identifica el dispositivo. Se puede acceder a varios de sus registros de manera secuencial, en modo de lectura, hasta que una condición de PARO es ejecutada.

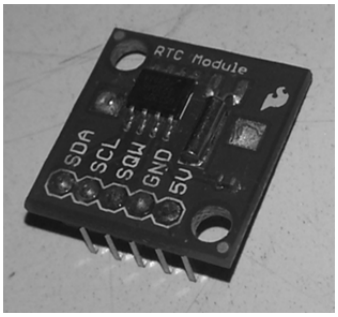


Fig. 3.32. Módulo del Reloj en Tiempo Real.

Cuando el Vcc del circuito cae por debajo de 1.25 veces el voltaje V_{BAT}, el dispositivo termina el acceso que este en progreso y da un reset al contador de direcciones. Las entradas no son reconocidas en este momento para prevenir errores en la escritura del dispositivo, de esta manera evitamos que pierdan los datos de la fecha y hora y garantizamos que ante un corte de energía el RTC se mantendrá en operación.

Para establecer comunicación entre el microcontrolador y el RTC *(reloj en tiempo real)* es necesario realizar la configuración dentro del microcontrolador correspondiente a la comunicación por I^2C, a continuación mostramos la línea de código necesaria para ello.

```
#use I2C(master, sda=PIN_B0, scl=PIN_B1)        /* indicamos que el microcontrolador está en
                                                   modo maestro y los pines correspondientes
                                                   a la comunicación, SDA y SCL*/
```

posteriormente mostraremos el código necesario para realizar la comunicación con el dispositivo.

3.6. Protocolos de comunicación

Son un conjunto de reglas y normas que permiten que dos o más entidades de un sistema de comunicación se comuniquen entre ellos para transmitir información por medio de cualquier tipo de variación de una magnitud física. Se trata de las reglas o el estándar que define la sintaxis, semántica y sincronización de la comunicación, así como posibles métodos de recuperación de errores. Los protocolos pueden ser implementados por hardware, software, o una combinación de ambos.

3.6.1. Protocolo RS-232 (Serial)

El puerto serie RS-232C, presente en todos los ordenadores actuales, es la forma más comúnmente usada para realizar transmisiones de datos entre ordenadores. El RS-232C es un estándar que constituye la tercera revisión de la antigua norma RS-232, propuesta por la EIA (Asociación de Industrias Electrónicas), realizándose posteriormente un versión internacional por el CCITT, conocida como V.24. Las diferencias entre ambas son mínimas, por lo que a veces se habla indistintamente de V.24 y de RS-232C (incluso sin el sufijo "C"), refiriéndose siempre al mismo estándar.

El RS-232 define especificaciones mecánicas, eléctricas, funcionales y de procedimientos típicos de un protocolo orientado al enlace físico punto a punto. Este estándar se basa en comunicación asíncrona, es decir que los datos pueden ser transmitidos en cualquier momento por lo que deben tomarse precauciones para sincronizar la transmisión y recepción. Como puede verse en el propio título del estándar, en la comunicación serie se distinguen dos tipos de dispositivos: Los equipos terminales de datos DTE ("Data Terminal Equipment"), y los equipos de comunicación de datos DCE ("Data Communication Equipment").

Especificaciones mecánicas

El conector empleado DB-9 (Fig. 3.33) es el más utilizado en el mercado, el estándar nos define que el conector hembra se situará en los DCE y el macho en el DTE. Aunque es fácil encontrar excepciones. También es frecuente que muchas interfaces sólo incorporen parte de los circuitos descritos en la especificación.

Para conseguir establecer la comunicación serial entre dos equipos se puede emplear los siguientes tipos de alambrado:

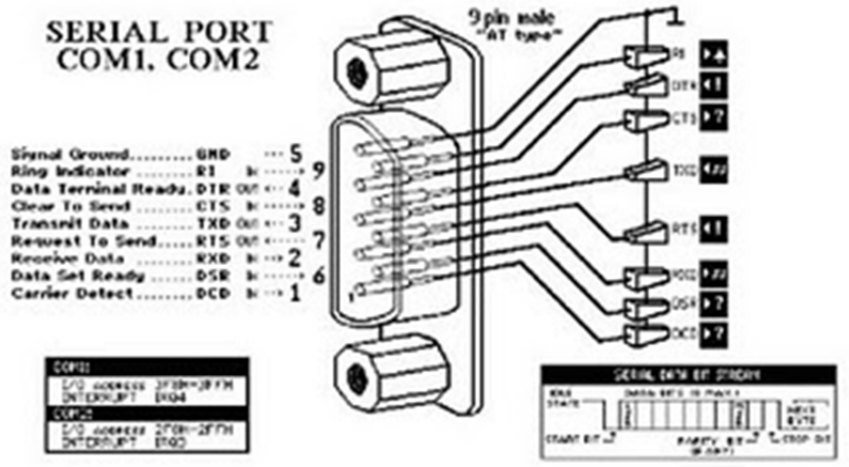
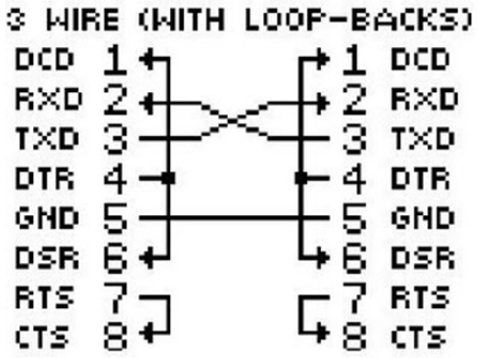


Fig. 3.33. Conector tipo DB-9 Macho.

MODEM ACTIVO CON HADSHAKING, FULL DUPLEX (DB-9)
MODEM NULO, FULL DUPLEX (DB-9)



Especificaciones eléctricas

La interfaz eléctrica utiliza una conexión eléctrica asimétrica con circuitos no equilibrados, todos referenciados a tierra.

Los estados lógicos son definidos por los siguientes niveles de voltaje en la Fig. 3.34.

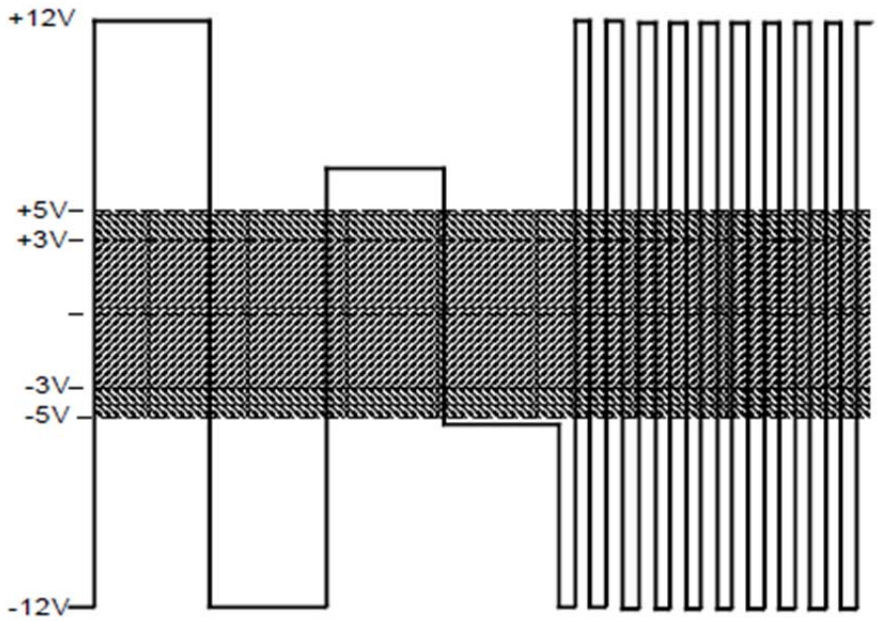
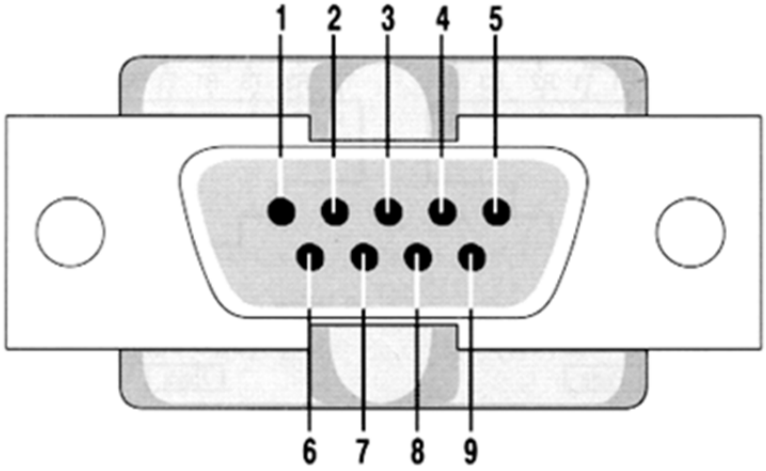


Fig. 3.34. Niveles eléctricos para una interface E/S *RS232C/V24-V28.*

La interfaz se utiliza a una razón de menos de 20 kbps para una distancia menor de 15m. En la práctica se pueden exceder estos límites utilizando cables de baja capacidad en entornos eléctricamente poco ruidosos.

Especificaciones funcionales

El RS-232C consiste en un conector tipo DB-9 de 9 pines, para cierto tipo de periféricos (como el ratón serie dela PC).Cada pin puede ser de entrada o de salida, teniendo una función específica cada uno de ellos (Fig. 3.35).



DB-9 Pin No.	Abreviatura	Nombre completo
Pin 3	TXD	Transmit Data
Pin 2	RXD	Receive Data
Pin 7	RTS	Request To Send
Pin 8	CTS	Clear To Send
Pin 6	DSR	Data Set Ready
Pin 5	SG	Signal Ground
Pin 1	DCD	Carrier Detect
Pin 4	DTR	Data Terminal Ready
Pin 9	RI	Ring Indicator

Fig. 3.35. Distribución de pines en un conector DB-9.

Las señales TXD, DTR y RTS son de salida, mientras que RXD, DSR, CTS y DCD son de entrada. La masa de referencia para todas las señales es SG (Tierra de Señal).

Descripción de las señales

SG (Signal Ground) o GND: Esta línea debe estar conectada al chasis dela PC y desde ahí hacer tierra.

DTR (Data Terminal Ready): Esta señal prepara al modem para conectarse a una línea de comunicación y mantener la conexión establecida. Una vez que modem está conectado a la línea este pin debe estar encendido para mantener la conexión, si DTR es apagado, provoca una desconexión de la línea, interrumpiendo el enlace de datos en progreso. DTR también

puede mantenerse encendido para efectuar la transmisión de datos por el pin TXD. Básicamente DTR es el pin de control maestro del modem.

DSR (Data Set Ready): Es la línea que indica que el modem está preparado. Usualmente el DSR está todo el tiempo encendido porque este valor es un indicador de que el MODEM esta encendido y listo.

RTS (Request To Send): Es la línea que dice al modem que la PC quiere enviar datos. El estándar RS-232 dice que RTS condiciona al modem para la transmisión, en realidad esta es solo una función que sirve de interruptor a un modem half duplex para transmitir o recibir. Mientras el modem half duplex está recibiendo, el DTE guarda el RTS apagado (cero), Cuando le toca al DTE el turno de transmitir, este informa al modem que desea transmitir poniendo el RTS encendido (uno). El DTE no puede comenzar inmediatamente con el envío de datos al modem porque este, no puede cambiar inmediatamente el modo de transmisión a recepción. Después de poner a uno RTS el DTE monitorea el pin CTS el cual es apagado por el modem que está en modo de recepción, cuando el modem está listo para transmitir, este enciende el CTS en función al DTE que está listo para el envío de datos. Este RTS-CTS (hand shaking), también es permitido en sentido contrario, es decir cuando el transmisor vuelve a recibir. Como en una comunicación full duplex hay dos canales no se necesita el RTS-CTS, de esta forma un modem full duplex pone permanentemente el pin CTS conectado al pin DCD (Data Carrier Detect).

CTS (Clear To Send): Es la línea que indica que el modem está preparado para recibir datos desde la PC.

DCD (Data Carrier Detect): Es la línea que indica que el modem tiene de verdad conexión remota. Este pin es también llamado "Receiver Time Signal Detect", es encendido cuando el modem recibe una señal remota y se mantiene encendido durante el enlace. En conexiones de modem half duplex el DCD es encendido solamente por el MODEM que está en recepción.

TXD (Transmit Data): Es la línea de transmisión de datos serie al modem. El TXD no puede transmitir datos a menos que los siguientes circuitos hayan sido encendidos:

- DTR
- DSR
- RTS (listo para enviar).
- CTS (listo para recibir).

RXD (Receive Data): Es la línea de recepción de datos serie desde el modem.

RI (Ring Indicator): Es la línea que indica que el modem ha detectado la señal de "llamada" (se pone en uno).

RTxC (Transmit/Receive Clock): Reloj común para transmisiones síncronas (solo existe en algunas PC's).

Además para que dos dispositivos puedan hacer efectivo el intercambio de información, se requiere que cada uno de ellos utilice las mismas características de transmisión, entre estas características están la velocidad de transmisión, que pueden ser de: 110bps, 300bps, 600bps, 900bps, 1200bps, 2400bps, 4800bps, 9600bps, 19200bps. Estas velocidades han sido ampliadas en la versión RS-232-E, que son: 28800bps, 38400bps, 57600bps, y 115200bps.

Este tipo de conexión lo utilizamos para comunicarnos con el microcontrolador y así introducir la configuración de operación, también nos sirve para mandar los datos a una hyperterminal, en la cual, el software desarrollado almacenara en un archivo, para su posterior análisis y estudio.

3.6.2. Protocolo I^2C

El I^2C (Inter Integrated Circuits) es un bus de comunicaciones serial síncrono de dos líneas que fue originalmente desarrollado por Philips Semiconductors (ahora nxpsemiconductors) desde los inicios de los '80.

El bus I^2C (Fig. 3.36) permite la comunicación entre múltiples dispositivos (en teoría más de 1000), todos conectados paralelamente a las dos líneas. Las transferencias de datos siempre se realizan entre dos dispositivos a la vez y en una relación *maestro – esclavo*.

Los dispositivos maestros son normalmente los microcontroladores y los dispositivos esclavos pueden ser memorias, conversores DAC y ADC, controladores de LCD, sensores de todos los tipos, etc.; Las líneas SDA (serial Data) y SCL (serial Clock) son bidireccionales, conectadas al positivo de la alimentación a través de las resistencias de pull-up. Cuando el bus está libre, ambas líneas están en nivel alto.

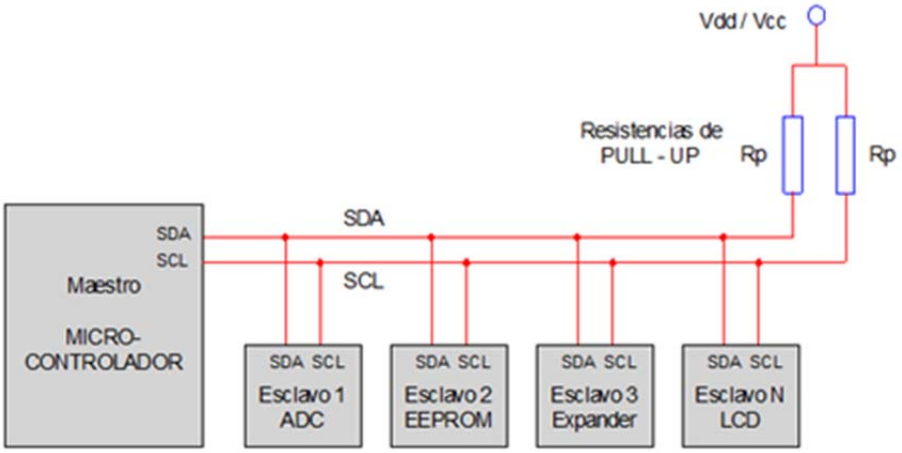


Fig. 3.36. Modelo de comunicación del bus I^2C.

Consideremos las siguientes características:

Las transferencias de datos se llevan a cabo mediante dos líneas: línea serial de datos SDA y línea serial de reloj SCL. Ambas son bidireccionales. SDA se encarga de conducir los datos entre el dispositivo maestro y los esclavos. SCL es la señal de reloj que sincroniza los datos que viajan por la línea SDA.

El dispositivo maestro (microcontrolador) es quien siempre tiene la iniciativa de la comunicación: el maestro genera la señal de reloj y controla cuando se transmiten o reciben los datos.

Puede haber varios esclavos en la red I^2C, pero el maestro solo se comunica con uno a la vez. Por eso cada dispositivo esclavo debe ser identificado por una dirección única.

Transferencias de Datos

Los datos que se transfieren por el bus I^2C deben viajar en forma de paquetes, también llamados transferencias. Como se ve en la Fig. 3.37, una transferencia empieza con un START y termina con un STOP. Entre estas señales van los datos propiamente dichos. Cada dato debe ser de 8 bits (1 byte) y debe ir seguido de un noveno bit, llamado bit de reconocimiento (ACK o NACK).

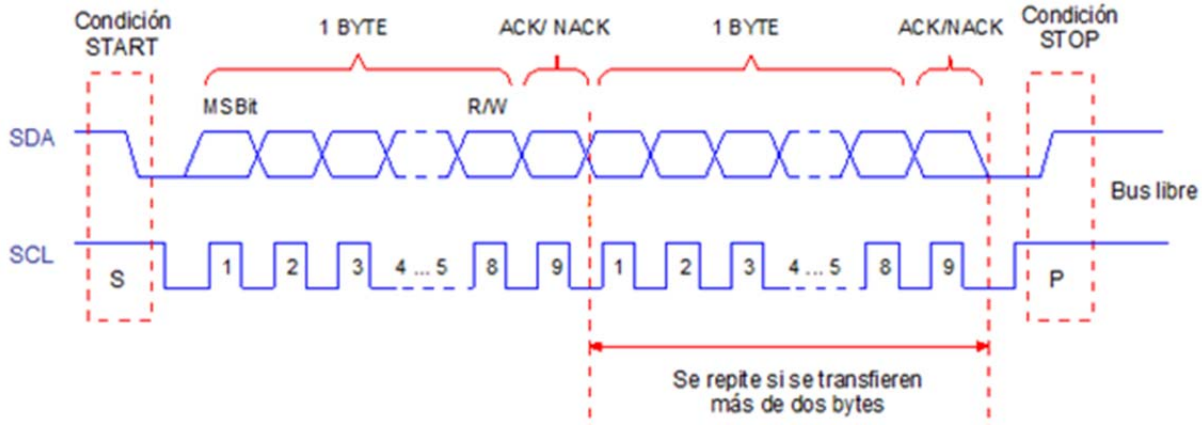


Fig. 3.37. Transferencia de datos por el bus de I^2C.

Los datos son transferidos por la línea SDA y son acompañados y sincronizados por los pulsos de reloj de la línea SCL. Para transmitir un bit primero hay que poner la línea SDA a 1 o 0 según sea el caso, y luego colocar un pulso en la línea SCL.

Condiciones START, STOP y START repetida

Los paquetes de datos transferidos por el bus I^2C deben ir enmarcados por un Start y un Stop. Ambas señales son generadas por el maestro, como se muestra en la Fig. 3.38.

- Una condición START es una transición de Alto a Bajo en la línea SDA cuando SCL está en Alto. Se le representa por la letra S. Después de Start el bus se considera ocupado.

- Una condición STOP es una transición de Bajo a Alto en la línea SDA mientras SCL está en Alto. Está simbolizada por la letra P. Después de Stop las dos líneas están en Alto y el bus se considera libre. Se usa Stop para cerrar la transferencia de un paquete de datos o para abortar una transferencia previa que quedó truncada.

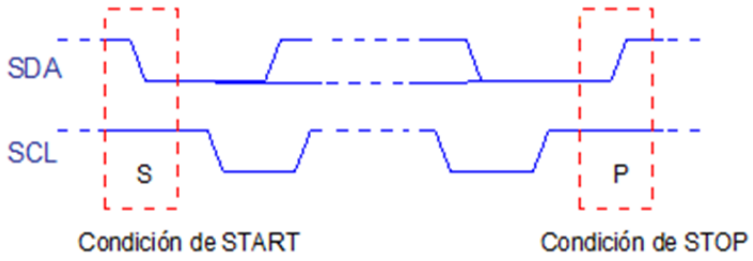


Fig. 3.38. Condición START y STOP.

- La señal de una condición START repetida es exactamente igual a la de START. La diferencia es de tipo "ocasional": aunque en principio cada transferencia debe ir enmarcada por un Start y un Stop, el estándar contempla la posibilidad de iniciar una nueva transferencia sobre una anterior que no ha sido cerrada con un Stop. El Start de la nueva transferencia se llama entonces Start Repetida y su símbolo es Rs.

Bit de Reconocimiento (ACK o NACK)

Cada byte transferido debe ir seguido de un noveno bit, llamado Acknowledge bit (bit de reconocimiento, en inglés). Este bit siempre debe ser devuelto por el dispositivo receptor (maestro o esclavo) tras cada byte recibido.

- Si el bit de reconocimiento es 0 significa que el dato fue reconocido y aceptado. Este bit se denomina ACK (Acknowledge).

- Si el bit de reconocimiento es 1 significa que el dato recibido aún no es aceptado. Se usa este mecanismo para indicar que el receptor está ocupado realizando alguna tarea interna. Este bit se denomina NACK (Not Acknowledge).

Byte de Control

Es el maestro (microcontrolador) quien ordena con cuál esclavo se va a comunicar o si los siguientes datos se van a transmitir o recibir; esa orden viaja en el primer byte de cada transferencia y es más conocido como byte de control.

En la Fig. 3.39 se muestra que 7 bits del byte de control contienen la dirección del esclavo con el cual se desea entablar la comunicación y el bit R/W establece si los siguientes bytes serán de lectura o escritura. Como siempre, R/W = 0 indica una escritura y R/W = 1 indica una lectura.

Fig. 3.39. Formato del byte de control.

Velocidad de Transferencia de Datos

Como cada bit de dato transferido sobre la línea SDA debe ser validado por la señal de reloj SCL, podemos deducir que la velocidad de transferencia está determinada por la frecuencia de la señal de SCL. El estándar del bus I^2C soporta cuatro modos de operación:

- *Standard Mode*, con una velocidad de hasta 100 kbit/s.
- *Fastmode*, con una velocidad de hasta 400 kbit/s.
- *Fastmode plus*, con una velocidad de hasta 1 Mbit/s.
- *High-speedmode*, con una velocidad de hasta 3.4 Mbit/s.

La comunicación de I^2C es empleada para interconectar un módulo de reloj en tiempo real con el microcontrolador, esto nos sirve para etiquetar los datos que son obtenidos con una fecha y hora adecuados, independiente mente si se presenta alguna perdida de alimentación, ya que el modulo del reloj cuenta con una batería externa que mantiene su funcionamiento.

3.6.3. Protocolo Ethernet

También se conoce como IEEE 802.3, es el estándar más popular para las LAN, usa el método de transmisión de datos llamado *Acceso múltiple con detección de portadora y detección de colisiones* (CSMA/CD). Antes de que un nodo envíe algún dato a través de la red, primero escucha y se da cuenta si algún otro nodo está transfiriendo información; de no ser así, el nodo transferirá la información a través de la red. Todos los otros nodos escucharán y el nodo seleccionado recibirá la información. En caso de que dos nodos traten de enviar datos por la red al mismo tiempo, cada nodo se dará cuenta de la colisión y esperará una cantidad de tiempo aleatoria antes de volver a hacer el envío.

Cada paquete enviado contiene la dirección de la estación destino, la dirección de la estación de envío y una secuencia variable de bits que representa el mensaje transmitido. El dato transmitido viaja a 10 millones de bits por segundo y el paquete varia en una longitud de 64 a 1518 bytes, así el tiempo de transmisión de un paquete de Ethernet está en un rango de 50 a 1200 microsegundos dependiendo de su longitud.

La dirección de la estación de destino normalmente es referida por una única interfaz de red. Cada estación recibe una copia de cada paquete, pero ignora los paquetes que son dirigidos a otras computadoras y procesa solamente los que son dirigidos a ella.

Las velocidades de envío de paquetes utilizando la tecnología Ethernet son de 10 Mbps (Ethernet estándar), 100 Mbps (Fast Ethernet – 100BASEX) y de 1000 Mbps utilizando el Gigabit Ethernet cuya especificación se encuentra respaldada por la IEEE con número 802.3z.

Las redes Ethernet tienen un esquema de direccionamiento de 48 bits. A cada computadora conectada a una red Ethernet se le asigna un número único de 48 bits conocido como *dirección Ethernet*. Debido a que el direccionamiento Ethernet se da entre dispositivos de hardware, a estos se les llama *direccionamientos o direcciones físicas*.

La trama de Ethernet es de una longitud variable pero no es menor a 64 bytes ni rebasa los 1518 bytes (encabezado, datos y CRC), cada trama contiene un campo con la información de la dirección de destino. En la Fig. 3.40 se muestra una trama Ethernet. Además de la información que identifica la fuente y el destino, cada trama transmitida contiene un preámbulo, un campo tipo, un campo de datos y un campo para verificación por redundancia cíclica (CRC- *Cyclic Redundancy Check*). El preámbulo consiste en 64 bits que alternan ceros y unos para ayudar a la sincronización de los nodos de recepción. El CRC de 32 bits ayuda a la interfaz a detectar los errores de transmisión: el emisor calcula el CRC como una función de los datos en la trama y el receptor calcula de nuevo el CRC para verificar que el paquete se reciba intacto.

Preámbulo	Dirección destino	Dirección fuente	Tipo	Datos	CRC
8 bytes	6 bytes	6 bytes	2 bytes	46-1500 bytes	4 bytes

Fig. 3.40. Formato de una trama (paquete) de Ethernet.

El campo de tipo de trama contiene un entero de 16 bits que identifica el tipo de dato que se está transfiriendo en la trama. Desde el punto de vista de Internet, este campo es esencial porque significa que las tramas se auto identifican. Cuando una trama llega a una máquina dada, el sistema operativo utiliza el tipo de trama para determinar qué módulo de software de protocolos se utilizará para procesar la trama. La mayor ventaja de que las tramas se auto identifiquen es que éstas permiten que múltiples protocolos se utilicen juntos en una sola máquina y sea posible entremezclar diferentes protocolos en una sola red física sin interferencia. Los protocolos TCP/IP utilizan tramas Ethernet auto identificables para hacer una selección entre varios protocolos.

3.7. Puertos de comunicación

Como habíamos visto anteriormente en nuestro diagrama a bloques del sistema propuesto en la Fig. 3.28 la tarjeta de adquisición cuenta con 2 formas de comunicarse con un equipo de cómputo y es básicamente a través de un puerto RS-232 y Ethernet.

Comunicación Serial

Como lo mencionamos, la comunicación RS-232 se logra por medio del módulo de comunicación EUSART del microcontrolador y acondicionando los niveles de voltaje con la ayuda de un circuito MAX232. Sin embargo, no todos los equipos de cómputo cuentan con interfaz de este tipo, afortunadamente existen en el mercado muchos dispositivos que nos permiten comunicarnos a través de ellos. Estos dispositivos Fig. 3.41B permiten la conectividad entre equipos, comunicándolos entre sí a través de un puerto USB del equipo mostrado en la Fig. 3.41A y a la USART del dispositivo Fig. 3.41C.

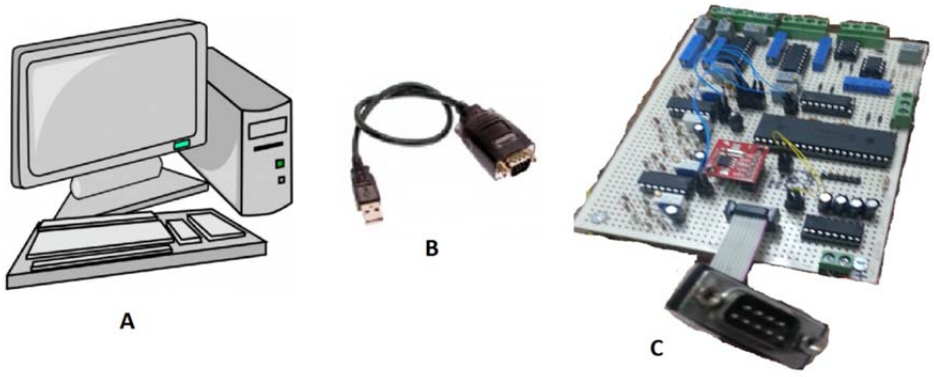


Fig. 3.41. A) Equipo de cómputo que recibe los datos a través de un puerto USB, B) Cable Adaptador de USB a Serial RS-232 y C) Sistema de adquisición de datos que envía la información a través de un puerto RS-232.

El uso de este tipo de accesorios, como el cable adaptador USB a Serial, es de gran ayuda aunque lo más correcto sería hacer uso del puerto USB que ofrece el propio microcontrolador. Sin embargo el uso de este puerto queda como una de las opciones de expansión y mejoras ya que primero se evaluara lo más importante que son la calidad de los datos y la adquisición de los mismos.

La comunicación de nuestro sistema de adquisición de datos a través de una interfaz Ethernet fue un tema que se discutió al principio del proyecto y que fue parte fundamental en la elección de los componentes del sistema. En un principio se había contemplado el uso de una tarjeta de evaluación de la compañía Texas Instruments, esta tarjeta incorpora entre varios de sus periféricos una interface Ethernet que le daría solución a la implementación del hardware y entre sus muchas librerías hace posible la implementación de protocolos TCP y UDP en IP con la ayuda de su compilador, el cual permite el uso de lenguaje C para la programación del microcontrolador.

Sin embargo aunque es un microcontrolador muy poderoso no presenta las características de los microcontroladores PIC que nos permitieron elegirlos y desarrollar en ellos. Unas de las principales causas de no elegir el dispositivo fueron:

- El número de puertos del módulo de conversión A/D es muy pequeño.
- El software no es del todo Gratuito.
- El costo del material para implementar el proyecto se vuelve muy alto.

Dado que el costo de implementar el proyecto se vuelve más costoso decidimos ver que otros dispositivos podrían satisfacer nuestras necesidades y permitir la evaluación, y la viabilidad de un sistema desarrollado por nosotros.

Con el apoyo del Servicio Sismológico Nacional fue posible realizar la implementación de la interface Ethernet por medio de un dispositivo que actualmente se encuentra detenido. El equipo que utilizamos junto con otros similares que se encuentran en las mismas condiciones fue utilizados en otra aplicación dentro del SSN, sin embargo fueron sustituidos ya que no cubrían completamente con el propósito y es por ello que fue posible utilizar uno para realizar las pruebas en el laboratorio. Estos equipos son los UDS (Universal Serial Server).

UDS y comunicación Ethernet

El equipo UDS es un dispositivo servidor de un solo puerto que nos proporciona de manera rápida el acceso a dispositivos a distancia y que solo cuentan con interface serial RS-232, a través de TCP o UDP por medio de una IP.

Usando un método llamado “Serial Tunneling” el UDS encapsula los datos en paquetes de datos y los envía por Ethernet.

Lantronix quien es el fabricante de estos dispositivos nos ofrece la descarga gratuita de un software que nos ayuda a complementar el circuito, de tal manera que con él es posible generar COMs virtuales en ambiente Windows que permiten tener acceso remoto a los dispositivos UDS en la red a través de ellos.

En la siguiente Fig. 3.42 se muestra un ejemplo de cómo se establece la comunicación entre una computadora y un dispositivo cualquiera de manera remota.

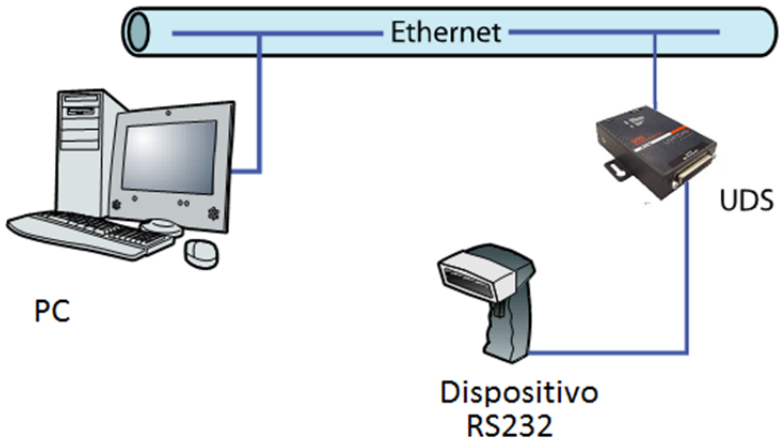


Fig. 3.42. Conexión entre dispositivo de comunicación serial y una PC a través de un UDS.

La instalación del software es a través de un wizard que nos lleva paso a paso por todo el proceso y no presenta ninguna complicación, para más información visite la guía de usuario del UDS en www.lantronix.com/support en donde se detalla todo el procedimiento de instalación del software. Una vez instalado el software Com Port Redirector, que se puede descargar de la página http://www.lantronix.com/support/downloads podemos configurar un puerto virtual que nos permita tener acceso a cualquier dispositivo que utilice RS-232 como medio de comunicación utilizando desde una hyperterminal o cualquier software similar. El número del COM lo asignamos nosotros dentro de la instalación y es posible identificarlo a través del administrador de dispositivos de Windows en el apartado de puertos, Fig. 3.43.

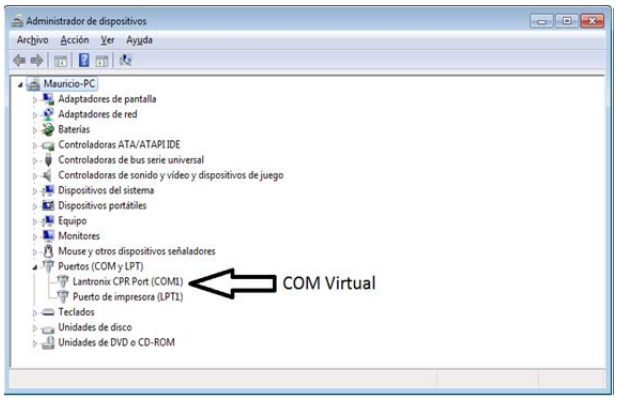


Fig. 3.43. COM virtual visto desde el administrador de dispositivos.

Es importante tomar en cuenta que para poder visualizar este COM es necesario instalar el software Com Port Redirector y que no es necesario que el UDS esté conectado a la red para que aparezca visible en el administrador de dispositivos.

Lantronix también nos ofrece gratuitamente el software DeviceInstaller que nos permite gestionar los dispositivos activos en la red y por el cual es posible asignar una dirección IP a nuestro UDS para poderlo conectar a la red local en la cual vamos a utilizarlo. Este programa también lo podemos descargar de la página Lantronix http://www.lantronix.com/support/downloads. Ya instalado nos permitirá realizar la asignación de la dirección IP y la configuración del puerto RS-232.

En las siguientes figuras mostraremos las principales configuraciones del dispositivo, las cuales nos permitirán tener comunicación con nuestra tarjeta de adquisición y para la gestión del UDS.

La Fig. 3.44 muestra la pantalla principal en la configuración del equipo.

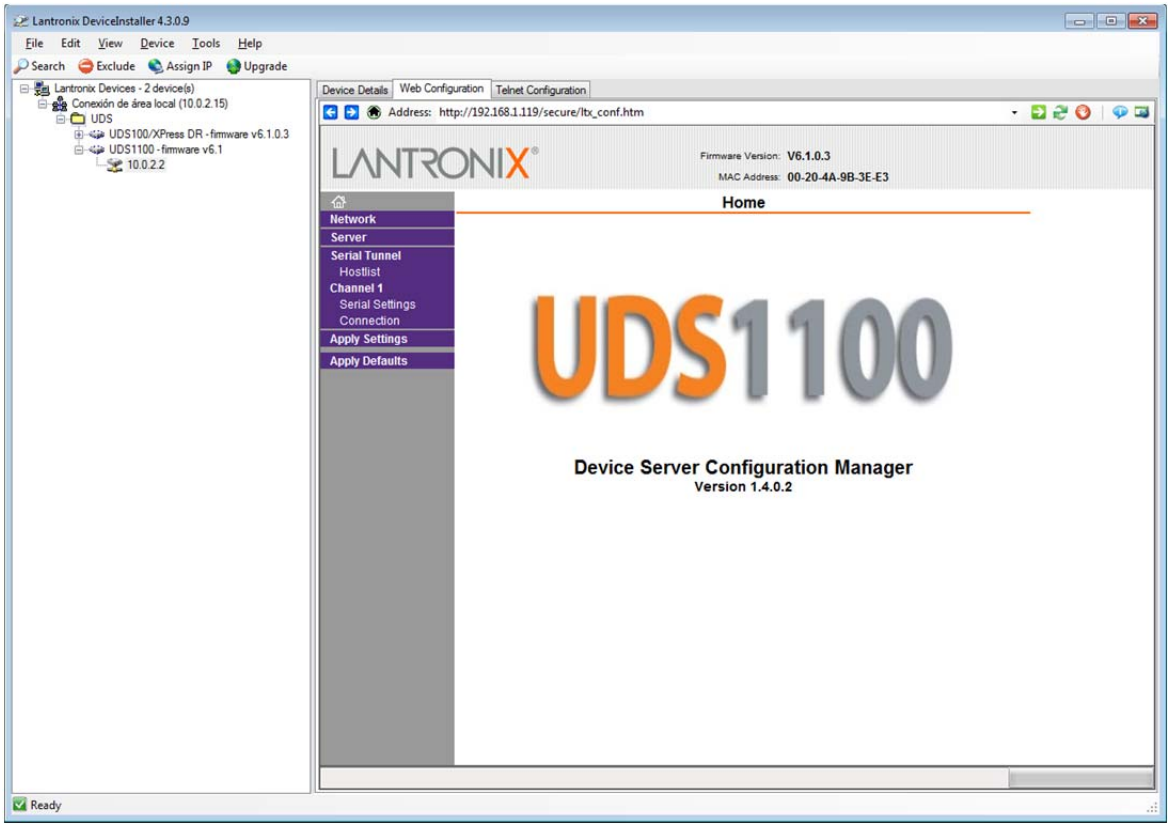


Fig. 3.44. Configuración del UDS a través del software DeviceInstaller.

En ella se pueden hacer los cambios a través de los links Channel 1 → Serial Settings para la configuración del puerto de comunicación serial del UDS, y en Network la configuración de la dirección IP utilizada para la interface Ethernet.

Configuración de los parámetros de Red

En este apartado podemos observar las opciones que nos ofrece el dispositivo y que podemos seleccionar para un mejor rendimiento del dispositivo.

En la mayoría de los casos lo mejor es utilizar una dirección IP (*Internet Protocol*) fija y libre dentro de nuestra red local, en el caso de nuestras pruebas solicitamos al área de sistemas una dirección IP nos asignaron los siguientes datos:

- Dirección IP 192.168.1.119
- Máscara de subred (SM) 255.255.255.0
- Puerta de enlace (GW) 192.168.1.4

Los cuales ingresamos en la pantalla como lo muestra la siguiente Fig. 3.45.

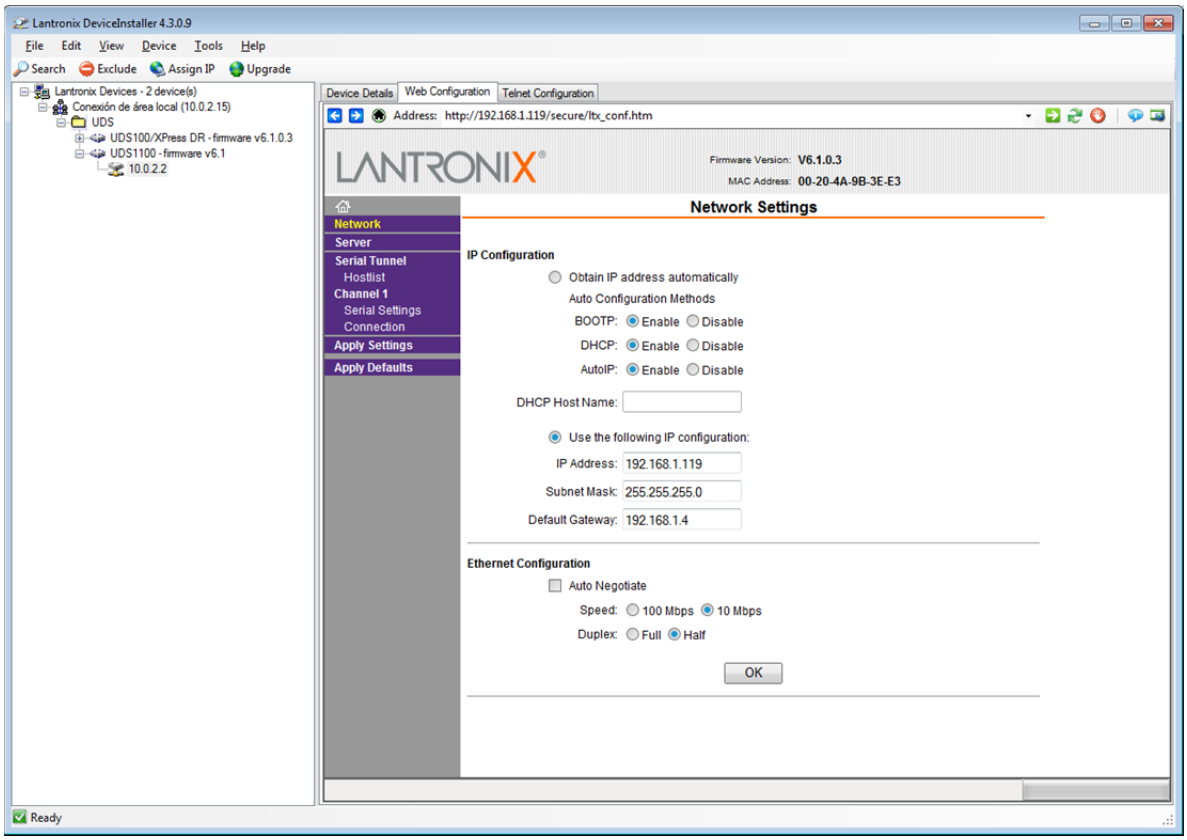


Fig. 3.45. Configuración de los parámetros de red en DeviceInstaller.

Configuración de la comunicación serial

Lo siguiente que debemos configurar es los parámetros necesarios para establecer la comunicación entre nuestro sistema de adquisición y el UDS. Es importante que estos parámetros coincidan si no, la comunicación no podrá establecerse.

Los parámetros configurados por default en nuestro microcontrolador son:

- Baud rate: 9600
- Bits de datos: 8
- Paridad: no
- Bits de paro: 1
- Control de flujo: no

Y los ingresamos como lo muestra la Fig. 3.46.

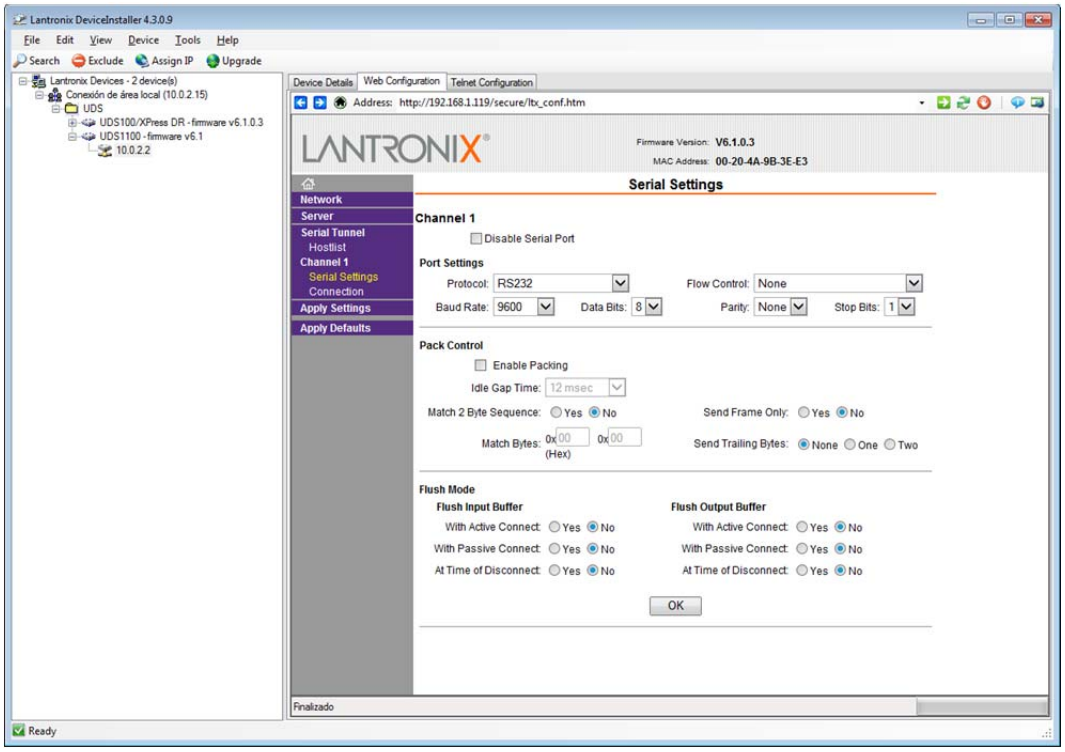


Fig. 3.46. Configuración de los parámetros de la comunicación serial en DeviceInstaller.

Los demás parámetros mostrados en la figura no los estamos utilizando en este momento por lo que no realizamos ningún otro cambio.

Estos parámetros son los únicos que debemos realizar para establecer una comunicación remota entre la computadora que recibirá los datos y el sistema de adquisición.

3.8. Transmisión y recepción de datos

La transmisión de datos por parte de la tarjeta de adquisición es solo por RS-232, sin embargo la computadora que va a recibir los datos puede hacerlo a través de 2 formas, y son, por RS-232 o Ethernet.

3.8.1. Transmisión de datos

La tarjeta de adquisición de datos cuenta con un módulo de comunicación serial que hace posible establecer la comunicación con cualquier computadora. Este módulo está basado en el circuito integrado MAX232 que hace muy fácil y de manera práctica su implementación, a través de él es posible acceder a la tarjeta desde cualquier computadora que cuente con un puerto serial RS-232 y una hyperterminal o similar. Sin embargo las computadoras que no cuentan con este puerto se ven limitadas en ese aspecto; actualmente contamos con cables que acondicionan estas señales y comunican a nuestra tarjeta o a cualquier otro dispositivo similar por medio de un puerto USB disponible en la computadora. Estos cables son muy comerciales sin embargo es un aditamento más que no siempre se cuenta con él y es por eso que se plantea en un futuro desarrollar el modulo USB que contiene el microcontrolador.

Otra forma de acceder a nuestra tarjeta de adquisición es por medio de un UDS, hablamos en particular del UDS1100 de Lantronix, por el cual es posible acceder al dispositivo, si éste se encuentra dentro de la misma red local que la computadora de adquisición.

Es así como podemos definir como 2, las posibles formas de acceder al sistema de adquisición y son:

- RS-232
- Ethernet

Formato de los datos

Actualmente la programación del microcontrolador está diseñada para arrojar todos los datos provenientes de la tarjeta en código ASCII, de manera que el menú de configuración y los datos adquiridos pueden ser leídos directamente desde una hyperterminal ya que se pensó en que la programación puede no ser siempre desde el software de adquisición.

La cadena de datos que contiene la información de las variables sensadas puede ser leída fácilmente ya que en la cadena se encuentran separados los datos con un carácter separador, este carácter es **";"**.

Los datos dentro de la cadena vienen ordenados de la siguiente manera:

sensor0;sensor1;sensor2;sensor3; sensor4; sensor5;sensor6;sensor7;hora;fecha;

Los sensores vienen numerados del 0 al 7 siguiendo la numeración de los puertos del convertidor A/D, que están enumerados comenzando desde el AN0 hasta el AN7 los canales que son utilizados actualmente.

El formato de la hora es HH:MM:SS, donde la hora está en formato 24 horas y la fecha tiene como formato DD/MM/AA.

Los canales del convertidor A/D pueden ser configurados para leer cualquiera de los 3 tipos de sensores disponibles o especificar si no se van a utilizar

Este es un ejemplo de una cadena con datos adquiridos:

120.19; 121.4; 2.05; 1.1; 32.08; 27.89; no; no; 14:30:5; 12/08/13;

En este ejemplo observamos lecturas de seis sensores y dos que no están conectados, la hora en formato de 24 horas y la fecha.

Este formato fue pensado en que el dispositivo puede ser leído por cualquier computadora y así saber la magnitud de las variables físicas leídas, la hora y fecha de la adquisición del dato.

3.8.2. Recepción de datos

La recepción de los datos es posible mediante dos formas, cada una presenta características diferentes pero pueden ser útiles en la aplicación que se desee. Una de ellas es a través de un programa hyperterminal o semejante, en el cual, de manera práctica se puede configurar la tarjeta para adquirir datos y desplegarlos en pantalla, para el monitoreo en pruebas de laboratorio de equipos que estén siendo probados para su instalación en campo. En estas pruebas pueden ser leídas múltiples variables y tener la lectura en el mismo instante de tiempo.

De la misma forma si se desea utilizar en el laboratorio o de manera remota nuestra tarjeta de adquisición, los datos recibidos deben ser almacenados para monitorear el comportamiento de uno o varios equipos, esto se puede hacer mediante un programa de adquisición, el cual se encarga de almacenar los datos en archivos de hoja de cálculo para posteriormente analizar el comportamiento de las variables.

A través de este programa es posible realizar varias acciones sobre la tarjeta de adquisición, estas acciones se realizan de la misma forma que en la hyperterminal pero de una manera más amigable para el usuario final. El registro de las variaciones que nosotros cataloguemos como eventos serán detectados por el software de adquisición y almacenados en un archivo, de esta manera es complementada la adquisición y el registro de datos que cumplan la condición de evento que el usuario delimite. Algunos ejemplos claros pueden ser:

- Un sobre/bajo voltaje de suministro por parte de CFE.
- Una interrupción en el suministro eléctrico.
- Una sobre corriente debido a la carga de las baterías después de haber sufrido una descarga prolongada.
- Temperaturas registradas por encima de los 45 [°C] o por debajo de 0 [°C].

3.9. Sistema operativo de la tarjeta electrónica

La adquisición de datos y registro de eventos son dos tareas, cada una de ellas se lleva a cabo con sistemas diferentes y que en conjunto forman un solo sistema.

La adquisición de datos y la transmisión de los mismos, se realiza con la tarjeta electrónica cuyo núcleo es un microcontrolador PIC18F4550. La recepción de los datos, su almacenamiento y el registro de eventos es hecha por un software de adquisición desarrollado en Labview instalado en una computadora con sistema operativo Windosws.

Como vimos anteriormente el acondicionamiento de las señales que se desean sensar se realiza en diversas etapas dentro de la tarjeta electrónica. Sin embargo, todas llegan a alguno de los 8 canales del convertidor A/D disponibles actualmente (capacidad de expandirse a 10 canales) y el programa que ejecuta el microcontrolador se encarga de mostrarnos la magnitud, en la escala correspondiente, de la variable en cuestión.

El diseño del sistema operativo se realizó tomando como base que no se requiere de un muestreo alto, es decir, el tiempo entre muestras es muy largo.

Dado que el muestreo de los canales se requiere únicamente al cumplir un lapso de tiempo, la rutina principal se ejecutara al generarse una interrupción. Dicha interrupción nos indicara cuando se cumpla el tiempo de realizar la captura de las muestras y de enviarlas a través de la EUSART hacia la computadora que recibirá los datos.

La estructura de la función *main()* se divide en 3 partes, la primera consta de la declaración del cristal interno y el valor que adopta, (el cual es de 8 MHz) así como de la inicialización de interrupciones y del módulo ADC, la segunda es la llamada de la función *RTCini()* que se encarga de inicializar el modulo del reloj en tiempo real a través del bus I^2C y la tercera es un *while(1)* que se ejecuta constantemente y en el cual está contenida la función principal del sensado.

Dentro de la función *while(1)* existe una condición *if(transmit==1 &&Trans==1)*, para que éstas variables sean válidas es necesario cumplir con las condiciones de "transmisión" las cuales dependen de la rutina de interrupción o del usuario. Cuando ambas variables son válidas, un ciclo *for{x=0;x<=7;x++}* ejecuta la lectura de los 8 sensores y dependiendo del valor de *x* es la rutina que ejecutara, ya que existe un arreglo llamado *Tipo[x]* que hace referencia al tipo de sensor que está leyendo. Dentro de *Tipo[x]* se encuentra un dato en ASCII que puede ser cualquiera del *'1'* al *'4'*, el cual indica si el sensor es de Vac, Iac, Temperatura o si esta deshabilitado respectivamente. De esta manera y por medio de un *switch{}* el microcontrolador decide qué función se ejecuta para hacer el cálculo y la transmisión del dato. Las funciones dentro del *switch{}* son *Prom(), Voltaje(), Corriente() y Temperatura().*

Al concluir el *switch{}* se ejecuta la función *Tiempo(1)* quien se encarga de comunicar al microcontrolador con el RTC y leer los datos correspondientes a la fecha y hora que en ese momento le solicite el microcontrolador.

Al final y para concluir la rutina principal, está un segundo *switch{}* el cual únicamente asigna el valor correspondiente a la tasa de muestreo a la que se generaran los datos a enviar. La asignación de este valor será dado por el usuario. A continuación se muestra el diagrama de flujo correspondiente de la función *main()*, Fig. 3.47.

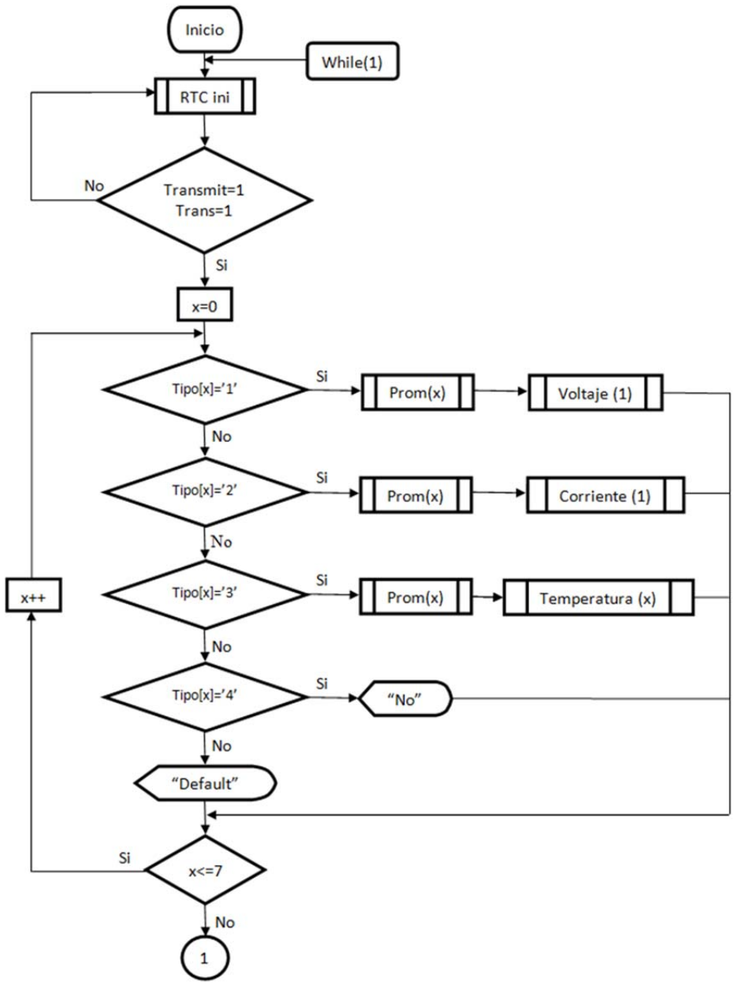


Fig. 3.47. Diagrama de flujo de la función *main()*.

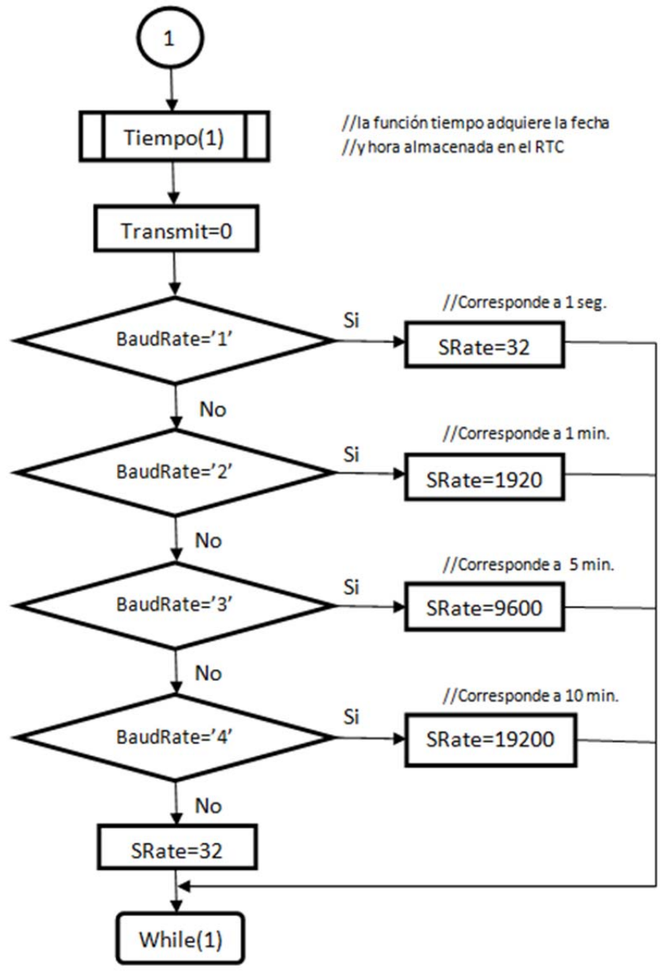


Fig. 3.47. *Continuación.*

Función RTCini()

Esta función tiene como objetivo cargar una configuración inicial por default en el caso de ser la primera vez que se inicia la tarjeta electrónica y no se haya configurado la hora y fecha actual de manera manual. La decisión es tomada a partir de la lectura de un registro llamado RTC dentro de la memoria RAM del circuito integrado DS1307, en este registro se aloja un dato que nos indica si ya ha sido configurado manualmente la hora y fecha. Este dato es introducido por la función *SetDate()* al momento de capturar los datos introducidos por el usuario. Si la condición nos indica que no ha sido actualizada la hora, ejecuta una rutina en la que cargara una fecha y hora por default, y el registro RTC se mantiene en el mismo valor, de manera que si el sistema se reinicia la fecha y hora que muestre será esta misma.

Si por el contrario, el registro RTC cumple con la condición de que ya ha sido configurado el dispositivo anteriormente, la rutina simplemente no hará ningún cambio en los valores de los registros.

El siguiente diagrama muestra el algoritmo de la función *RTCini()* Fig. 3.48.

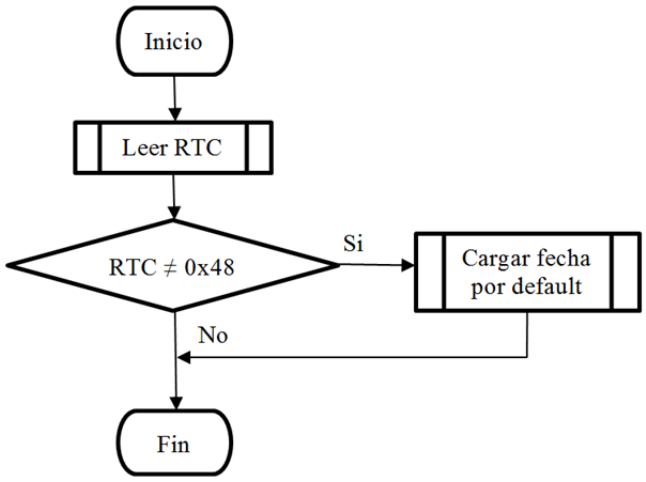


Fig. 3.48. Diagrama de flujo de la función *RTCini()*.

El valor de la variable RTC mostrado en el diagrama de flujo corresponde a un dato que es guardado en la memoria del Reloj en Tiempo Real, de esta forma se sabe cuándo el dispositivo ha sido configurado el modulo previamente, de lo contrario se cargará una fecha por *default*.

Función Prom(char ch)

La función *Prom()*se encarga de sacar un promedio de 240 muestras tomadas del canal del convertidor A/D que se le indica a través de *"ch"*. Posterior a la selección del canal se inicia un *for{x=0;x<=240;x++}* el cual captura el dato proveniente del convertidor A/D y lo almacena en un buffer. De este buffer se toman los datos y por medio de otro *for{x=0;x<=240;x++}* se realiza la suma de todos los elementos para después sacar un promedio de las 240 muestras tomadas. A continuación se muestra el diagrama de flujo de la función *Prom()*, Fig. 3.49.

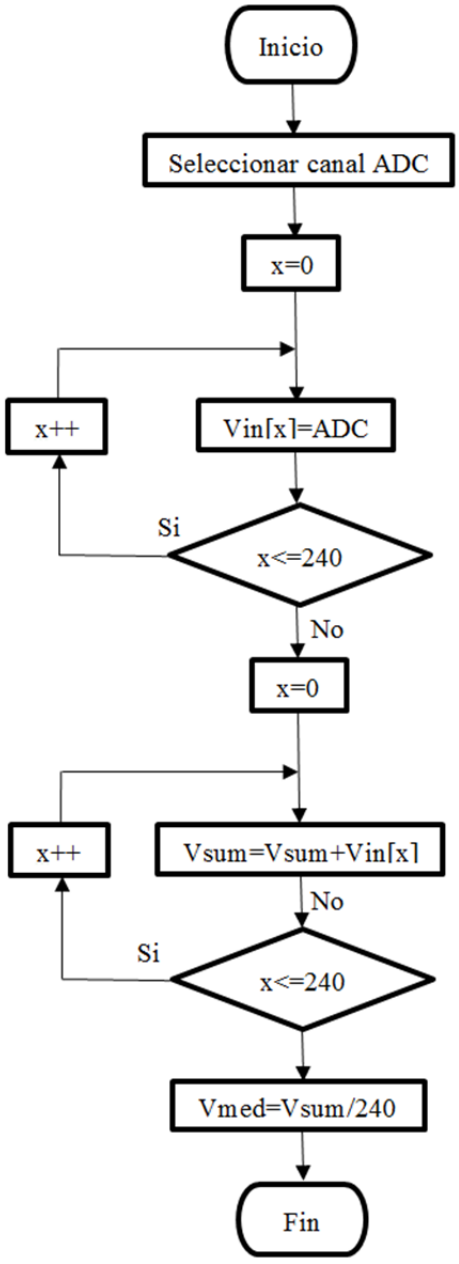


Fig. 3.49. Diagrama de flujo de la función *Prom(char ch).*

Función Voltaje(char tr) y Corriente(char tr)

Ambas funciones cuentan con la misma estructura. Se cuenta con una variable de tipo flotante la cual inicialmente adopta el valor promedio de las lecturas hechas en la función *Prom()*, posteriormente se realiza la siguiente operación:

$$V_{float} = \frac{(V_{float} * Max)}{1023}$$

Donde Max representa el valor máximo de dentro de la escala de la variable correspondiente, sea 130 para voltaje y 25 para corriente, dado que serán 130 volts en voltaje y 25 amperes máximo los que se puedan leer por nuestros sensores.

Si *tr=1* indica que la función fue ejecutada para mandar un string con el valor del sensor de voltaje o corriente y al terminar se le concatena un carácter *";"* cumpliendo con el formato de transmisión que hemos establecido.

Al final, se indica que *Vmed=0* para que en un futuro cuando se utilice la variable ésta se encuentre limpia y no afecte alguna lectura de otra función que la necesite.

A continuación los diagramas de flujo de las funciones *Voltaje()* y *Corriente()*, Fig. 3.50.

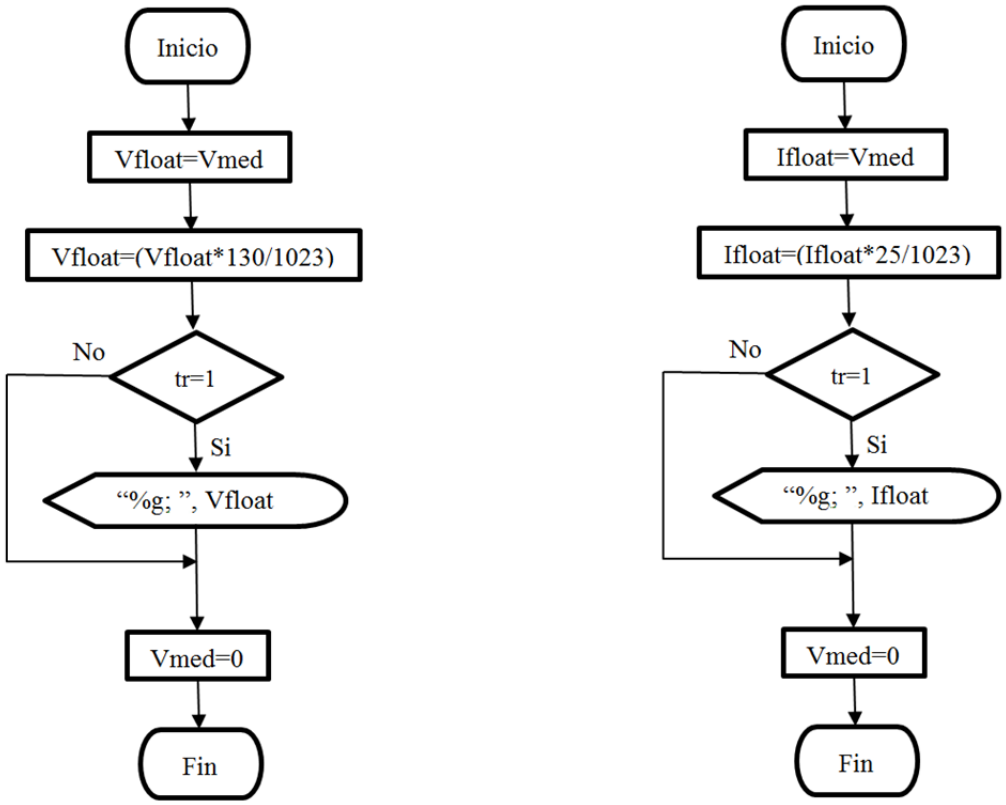


Fig.3.50. Diagrama de flujo de las funciones *Voltaje(char tr)* y *Corriente(char tr).*

Función Temperatura(char tr)

La función *Temperatura()* se encarga de calcular el valor correspondiente a la temperatura, comienza asignando el valor de la función promedio a una variable de tipo flotante y después realizamos la siguiente operación:

$$V_{float} = \frac{(V_{float} * 80)}{1023} - 15$$

Donde 80 es el número máximo de grados que deseamos sensar y -15 el ajuste del cero, para que la escala de la temperatura vaya de -15 a 65 grados Celsius.

El valor de *tr* nos indica si se le concatena o no el carácter separador *";"* para ser enviado.

A continuación se muestra el diagrama de flujo de la función *Temperatura()* Fig. 3.51.

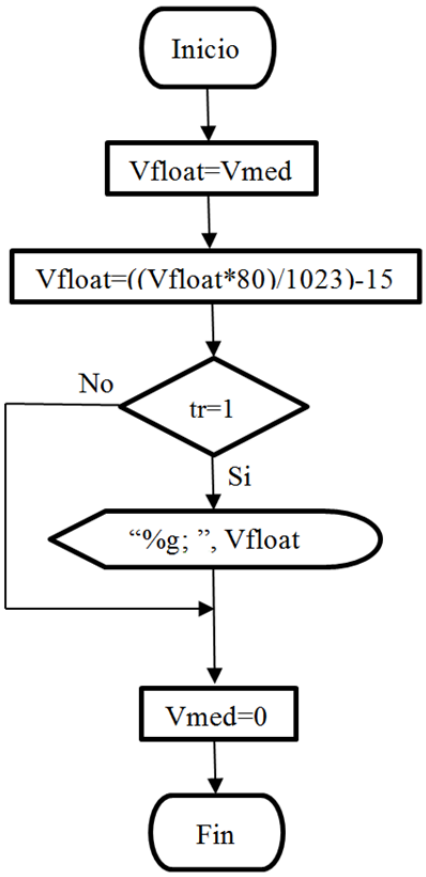


Fig. 3.51. Diagrama de flujo de la función *Temperatura()*.

Función Tiempo(char x)

La función cuenta con 4 ciclos *for{}*, 2 de ellos dedicados a la lectura del RTC para obtener la fecha y hora, y otros 2 para cambiar el formato de los valores obtenidos y enviarlos con sus caracteres separadores, *":"* y *"/"*.

Al inicio de la función se establece comunicación entre el microcontrolador y el RTC, se inicia en modo de lectura y se coloca en la dirección 0 en donde se encuentra el vector de la hora, seguido de minutos y segundos. Siendo inicializado así el ciclo *for{x=0;x<=2;x++}* y se almacena en una variable temporal los valores leídos. Al concluir la lectura se detiene la comunicación. Después, otro ciclo *for{}* se encarga de convertir de entero a ASCII los valores y se imprimen junto con su carácter separador.

De la misma forma se realiza la lectura de los valores correspondientes a la fecha, la conversión de entero a ASCII y él envío de los datos junto con sus respectivos caracteres separadores para el caso de la fecha.

El diagrama de flujo se muestra a continuación, Fig. 3.52.

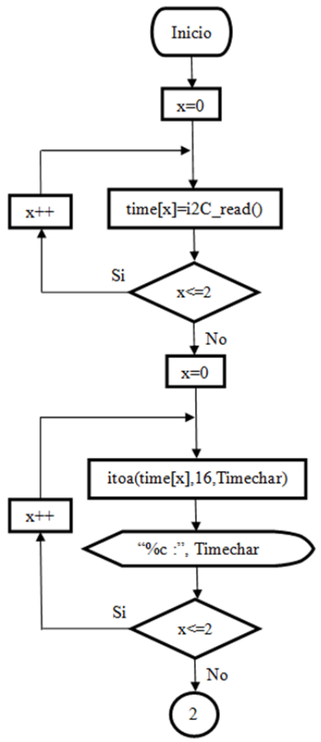


Fig. 3.52. Diagrama de flujo de la función *Tiempo(char x).*

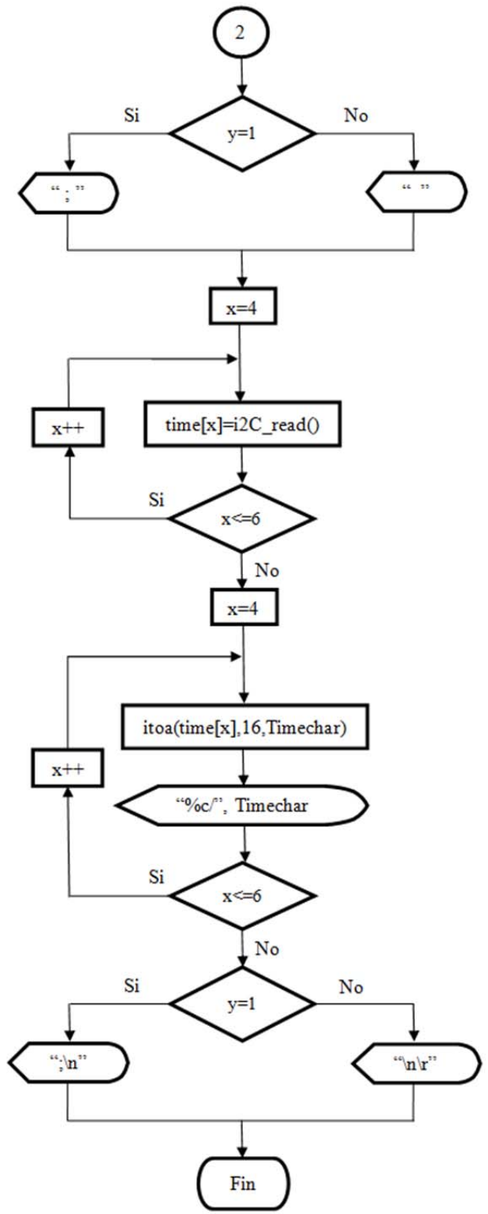


Fig. 3.52. Continuación.

Interrupción por Timer1

La ejecución de la mayoría de las funciones está condicionada a la generación de una interrupción. En este caso el *Timer1* es un contador de 16 bits que puede generar una interrupción cuando ocurra un desbordamiento en su registro. Esto es una característica que nos permite hacer un cálculo del tiempo que requerimos para comenzar la lectura de los canales y la transmisión de la cadena de datos, es decir, que cumplan las condiciones "transmisión" dentro la función *main().*

Al inicio de todo el programa existe una variable *"d=0"* la cual se irá incrementando en uno cada que entre a la función de interrupción. Antes de ser incrementada se realiza una comparación con la variable *SRate.* El resultado de la comparación determina si el tiempo transcurrido es el necesario para la transmisión o si se incrementa la variable *"d"* que al ser mayor que *SRate* indica que *Transmit=1* y comienza la transmisión del dato.

El valor 0x0BDB es el valor que le asignamos al registro del *Timer1*, si sustituimos este valor en la formula

$$\left(\frac{4}{8\ [MHz]}\right)(preescalador\ 1{:}\ 1)(0x0BDB) = 0.03125\ [s]$$

nos da como resultado un tiempo de 0.03125 segundos, este es el tiempo que hay entre una interrupción y otra. Si ejecutamos 32 veces la misma interrupción tenemos una base de tiempo exacta de 1 segundo. De esta manera podemos definir nuestra fórmula de la siguiente manera:

$$\left(\frac{4}{8\ [MHz]}\right)(0x0BDB)(32) = 1\ [s]$$

El diagrama de flujo de la función de interrupción es el siguiente, Fig. 3.53.

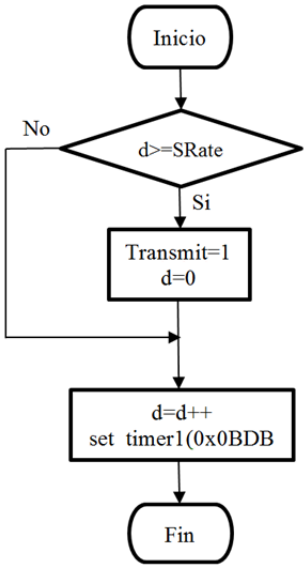


Fig. 3.53. Diagrama de flujo de la función de *interrupción por Timer1*.

Capítulo 4. Diseño del Sistema

4.1. Diseño de la tarjeta de adquisición

Al inicio del proyecto se especificó la necesidad de un instrumento que fuera capaz de adquirir datos correspondientes al voltaje, la corriente y la temperatura a las cuales se encuentra operando una estación sismológica. El rango de valores de medición, fue delimitado tomando en consideración los parámetros de operación de los diferentes equipos, que constituyen una estación sismológica, de los cuales ya se tienen un conocimiento previo, además de las especificaciones que provee del fabricante.

Las variables que se deseaban registrar son:

- Voltaje de AC.
- Corriente de AC.
- Temperatura.
- Voltaje de DC.
- Corriente de DC.

Todas estas variables fueron consideradas como importantes para registrar en qué condiciones de operación se encuentran trabajados los equipos instalados. Al tomar en cuenta el voltaje de AC suministrado por CFE, el consumo de los equipos y la temperatura del ambiente en sitio, se delimito que el rango de operación de los sensores dedicados a estas variables, sería superior a los valores máximos contemplados.

El rango de medición utilizado para las variables que fueron tomadas como más relevantes para su monitoreo fueron; el voltaje de AC, la corriente de AC y la temperatura, para las cuales se delimitó un rango de medición de 0 a 130 Volts para el voltaje, 0 a 25 Amperes para la corriente y de -15 a 65 grados Celsius para la temperatura.

La selección del microcontrolador comenzó por cubrir las características principales del sistema (número de canales del convertidor A/D, puertos de comunicación, periféricos, etc.), así mismo se consideró el costo que conlleva el desarrollar un sistema basado en él, ya que nos pareció importante tomar en cuenta las siguientes características:

- Costo del circuito integrado.
- Costo de la licencia del software de desarrollo.
- Costo del programador o accesibilidad a él.
- Disponibilidad en el mercado.
- Conocimiento y experiencia en el lenguaje de programación del microcontrolador.

Con el resultado de dichas consideraciones concluimos que el desarrollo de la tarjeta electrónica de nuestro sistema de adquisición lo haríamos con el microcontrolador PIC18F4550. Sin embargo existen una gran variedad de microcontroladores que superan en capacidad a éste, pero para el desarrollo del proyecto es más que suficiente.

Considerando los rangos de operación requeridos y las características del microcontrolador fue posible hacer la selección de los sensores adecuados para cubrir con los objetivos del diseño. Posterior mente se llevó a cabo el desarrollo de las etapas de acondicionamiento, con las cuales se obtuvieron los niveles de voltaje adecuados y así aprovechar todo el rango de conversión del digitalizador.

Otra parte importante en el desarrollo de la tarjeta electrónica son los puertos de comunicación con lo que se tiene acceso al sistema, debido a que su configuración debe ser a través de una computadora personal con la cual el usuario determine las características de adquisición, así mismo es el medio por el cual el usuario tiene acceso al sistema vía remota. Para lograr la interconexión de la tarjeta electrónica con una computadora es necesaria la implementación de una interfaz, como es el caso del CI MAX232, con el cual es posible realizarla.

Uno de los objetivos del desarrollo de la tarjeta es el poder comunicarse con el usuario de manera remota. Al principio del proyecto se planteó elegir un microcontrolador que incluyera un módulo de comunicación Ethernet entre sus periféricos, sin embargo las consideraciones costo/beneficio descartaron al microcontrolador que contaba con estas características. La solución a esta necesidad fue posible gracias a la implementación del módulo UDS1100 debido que en el Servicio Sismológico se cuenta con varios de estos equipos y el desarrollo no implicó un costo adicional.

Debido a la necesidad de etiquetar los datos adquiridos con una fecha y hora de adquisición, se buscó implementar un sistema que nos proporcionara esta información. Este sistema debe mantener comunicación directa con el microcontrolador y mantener su configuración aun cuando sufra una perdida prolongada de energía. Existe una amplia gama de circuitos integrados RTC que son capaces de comunicarse con nuestro microcontrolador por medio de puertos de comunicación como lo es el puerto I^2C y el puerto SPI, sin embargo en el mercado existen módulos que integran al circuito integrado y todos los componentes necesarios para su funcionamiento, así como una batería de respaldo que garantiza su funcionamiento ininterrumpido.

Decidimos utilizar el módulo RTC DS1307 ya que su costo y la fácil implementación con nuestro sistema nos permite obtener la información necesaria para etiquetar de manera adecuada los datos adquiridos y dicho módulo integra una batería de respaldo que nos asegura mantener el dato de hora y fecha lo más actualizada posible.

Para la polarización de los componentes que integran este sistema se tomó en consideración que la fuente primaria de alimentación podría ser de 12 o 24 volts, sin embargo los voltajes de operación de nuestros CI son 5V y +-12V por lo que requerimos de un dispositivo convertidor DC-DC que nos ofrezca:

- Vin: 9~36V
- Vout: 5v, 12v y -12v

Dado que el dispositivo que cumplía con estas características era demasiado costoso se implementó a través de 2 dispositivos DC-DC, uno que ofrece 5V de salida y otro que ofrece +-12V, estos componentes son el AM5TW-2405sz y el ZA05-24-12D.

Al final se llevó a cabo la integración de todos los módulos antes mencionados y se presentó como un prototipo de lo que puede ser un sistema de adquisición de datos basado en un microcontrolador.

En la Fig. 4.1 mostramos las tarjetas electrónicas que componen el sistema y el UDS1100 de Lantronix con el que realizamos la comunicación remota.

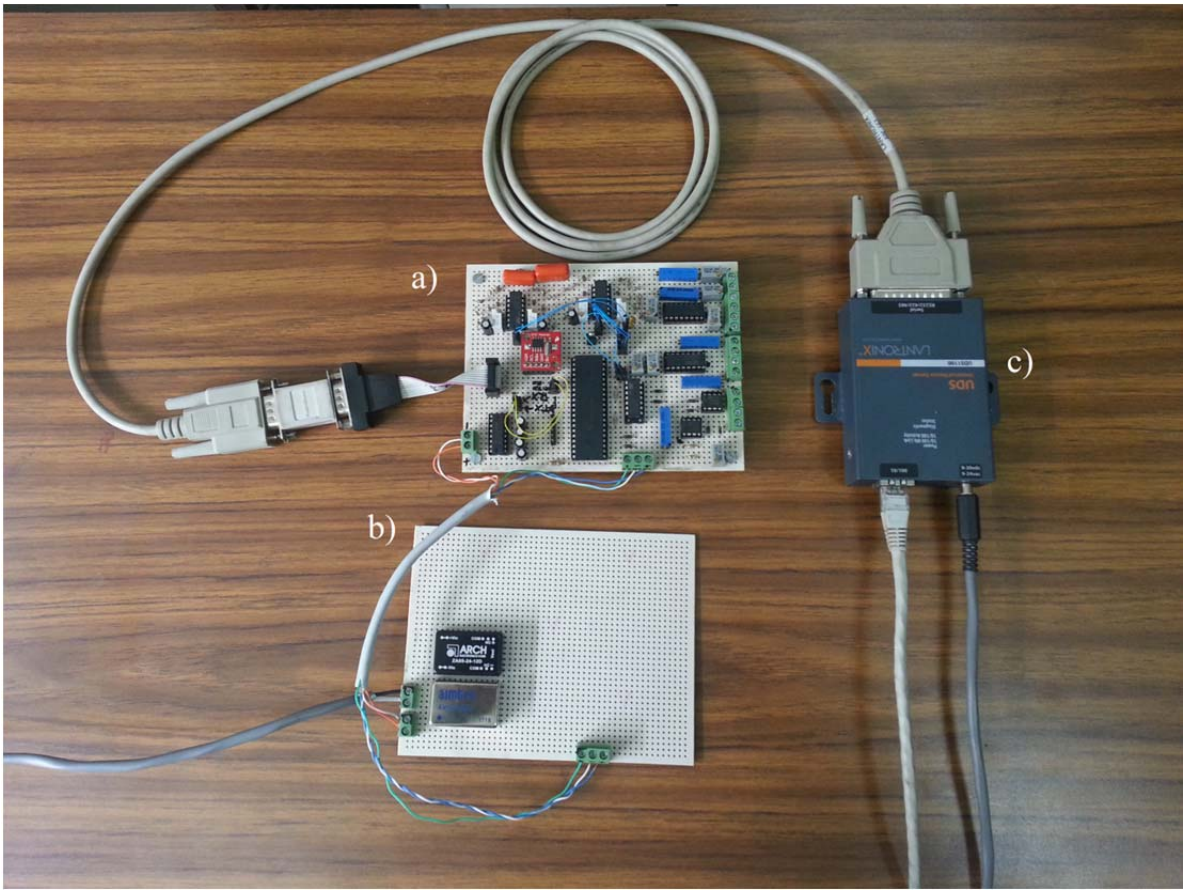
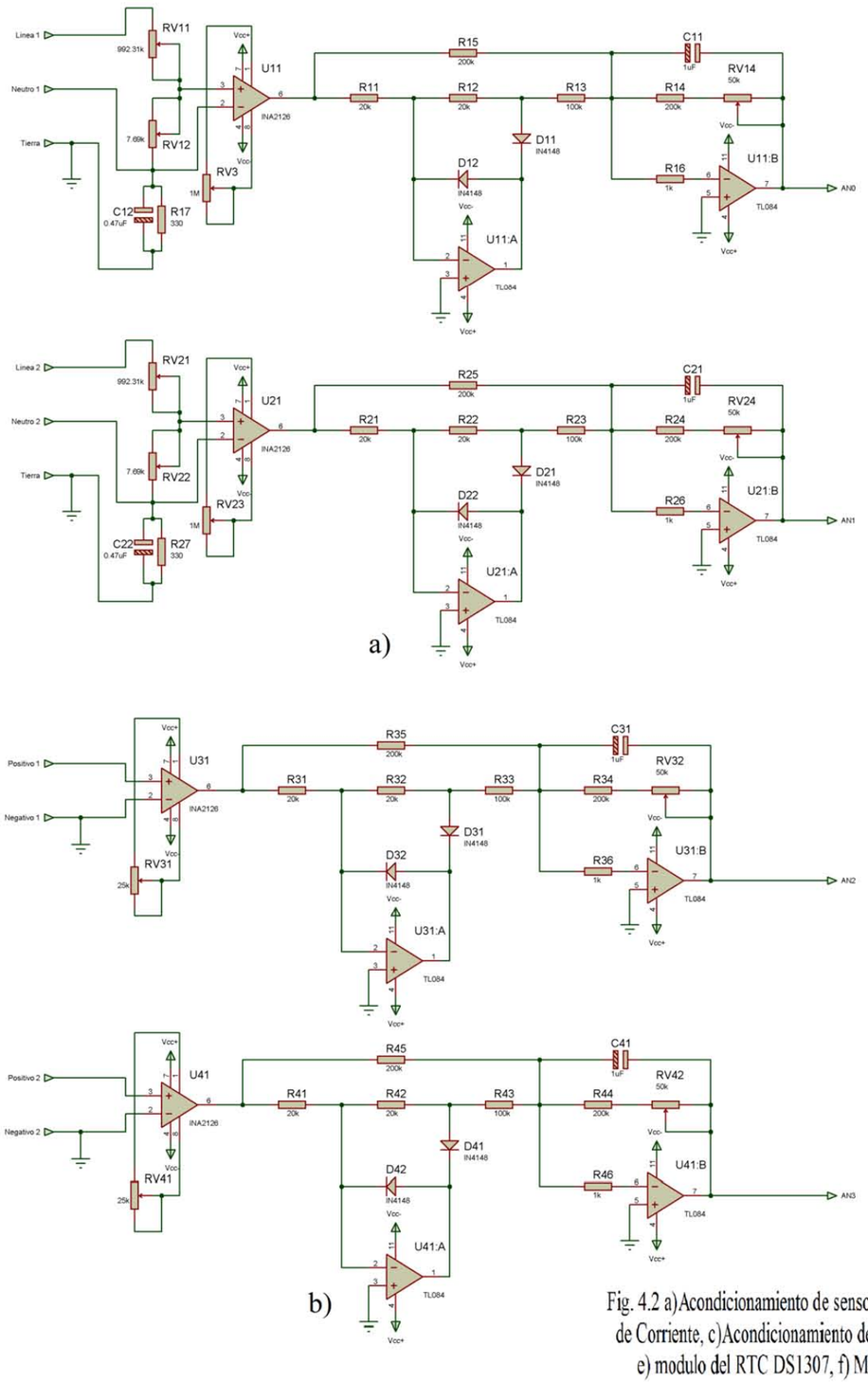
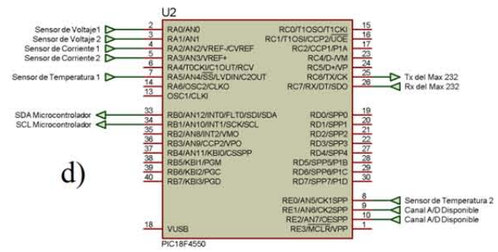


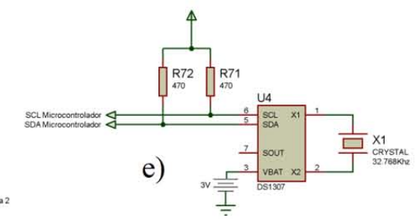
Fig. 4.1. a) Tarjeta electrónica basada en el microcontrolador PIC18F4550, b) Tarjeta que incluye los convertidores DC-DC y c) UDS1100.

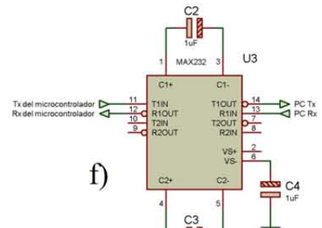
4.2 Diagrama Elemental del Sistema

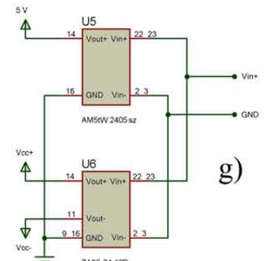
A continuación en la Fig. 4.2 se muestra el diagrama esquemático de los circuitos implementados en la tarjeta electrónica. Este diagrama fue elaborado en el programa ISIS7, este programa es muy útil en la elaboración de diagramas y en la simulación de circuitos.
En el diagrama se muestra la distribución de los bloques que conforman el sistema, para su mejor ubicación colocamos líneas punteadas que delimitan estos bloques.

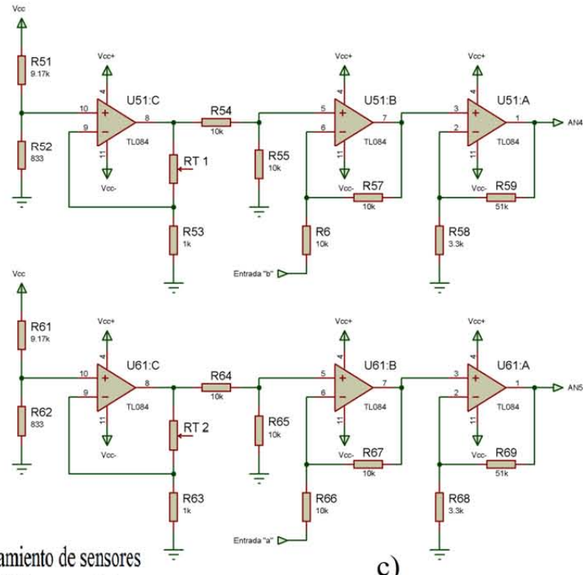












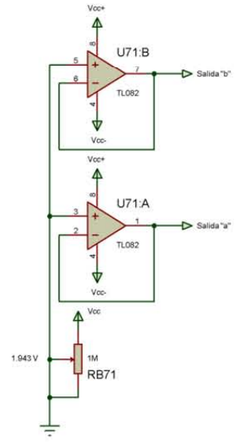


Fig. 4.2 a)Acondicionamiento de sensores de Voltaje, b)Acondicionamiento de sensores de Corriente, c)Acondicionamiento de sensores de Temperatura, d) Microcontrolador, e) modulo del RTC DS1307, f) Max232 y g)Circuitos convertidores DC-DC

4.3. Descripción del software y funciones básicas

El software de adquisición está desarrollado en Labview, de tal manera que nuestro programa es completamente personalizable y actualmente está limitado a los requisitos básicos como son:

- Interface amigable con el usuario.
- Programación básica de la tarjeta de adquisición.
- Adquisición de datos y almacenamiento de los mismos en archivos .csv.
- Registro de eventos y almacenamiento en archivos .csv.

En la Fig. 4.3 se muestran las pantallas que integran la interface de comunicación con el usuario y en las cuales se encuentran las opciones de configuración para la conexión de tipo serial RS-232, los botones de configuración, pantallas de estados y cajas de texto para ingresar datos manualmente.

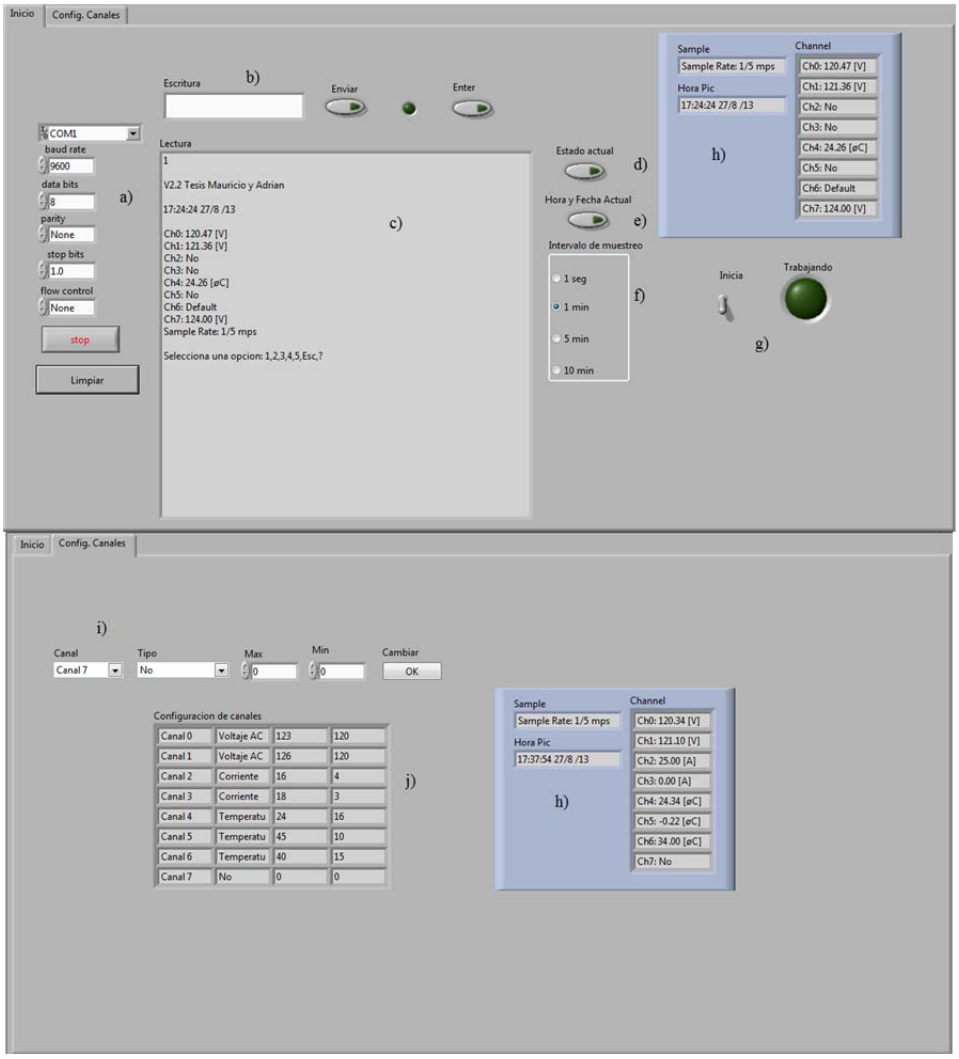


Fig. 4.3. Pantallas del software de adquisición de datos y registro de los eventos.

En la Figura 4.3 observamos varios incisos, a continuación presentamos la descripción de cada uno de ellos.

a) Parámetros de configuración serial RS-232

Una programación básica para nuestra tarjeta de adquisición comienza a partir de la conexión de la computadora con la tarjeta electrónica, para ello es necesario elegir de manera correcta los parámetros dentro del *a)*, de la Fig. 4.3 los cuales son Baud Rate de 9600, 8 bits de datos, sin paridad, 1 bit de paro y sin control de flujo. Justo debajo de estas opciones se encuentran el botón de "Stop" y "Limpiar" los cuales sirven para detener completamente la comunicación y liberar el puerto COM de la computadora, así como para borrar todos los datos de la pantalla de visualización de resultados (Fig. 4.3c), respectivamente.

b) Caja de texto para inserción de comandos de manera manual

Una vez establecida la comunicación entre la computadora y la tarjeta electrónica, es posible realizar la programación para la adquisición de datos de 2 maneras. Una de ellas es usando los botones de la interface (descritos en los incisos *d), e), f), g)* e *i)*) y la otra es ingresando los comandos necesarios en la caja de texto en *b),* con los cuales podemos realizar las mismas acciones. Los comandos necesarios son los siguientes:

1: Despliega un estado actual de las variables a medir, la hora, la fecha y la tasa de muestreo.
2: Selección de la tasa de muestreo.
3: Inicio de la adquisición y transmisión de datos.
4: Actualización de Hora y Fecha.
5: Selección de los canales que se desean utilizar y del sensor para cada uno de ellos.
?: Ayuda, despliega la lista de los comandos y su descripción.

c) Ventana de visualización de resultados

Con la ayuda de esta ventana, es posible saber el comando ingresado manualmente o por medio de algunos de los botones de la interface, y el resultado de dicha acción, ya que refleja todos los datos que son transmitidos hacia el microcontrolador y los que provienen de él. Es de mucha ayuda porque podemos monitorear el comportamiento de la tarjeta.

d) Botón de estado actual

Este botón nos despliega de una manera detallada la información hasta ese momento proveniente de la tarjeta electrónica y actualiza la ventana de estado actual *h).*

e) Botón de Hora y Fecha

Este botón nos ayuda a desplegar la hora y fecha actual, proveniente de nuestra computadora y de igual forma ingresa esa información directamente al reloj de tiempo real de la tarjeta. Es muy útil porque con un solo clik es posible actualizar esa información.

Nota: Debemos estar seguros que queremos ingresar la hora y fecha de nuestra computadora ya que si queremos una hora y fecha diferente será necesario ingresarla de manera manual.

f) Selección del intervalo de muestreo

Nuestro microcontrolador contiene una rutina de interrupción que tiene programada el sensado de las variables y él envió de los datos en intervalos de tiempo de 1seg., 1min, 5min y 10min. La acción se ejecuta de manera inmediata y solo necesita de darse un clik en la opción deseada.

g) Interruptor Inicio/Paro y Led indicador de adquisición de datos

Este interruptor comienza o detiene la adquisición (según sea el caso) y el Led indicador nos muestra el estado actual de la transmisión. Si esta encendido significa que el dispositivo se encuentra transmitiendo y por el contrario si está apagado la adquisición se encuentra detenida.

h) Estado actual

En esta ventana se encuentran los valores registrados de cada uno de los sensores hasta la última vez que se solicitó un "estado actual", por medio del botón descrito en él *d)* o ingresando el comando 1 en la caja de texto del *b).*

Dicha ventana se encuentra en ambas pestañas, la de inicio y en la de configuración.

i) Configuración de canales

La configuración de canales se lleva a cabo en la pestaña que lleva el mismo nombre. En ella se encuentran cajas de texto en la que se puede seleccionar el canal que se desea configurar, la variable que se desea medir y los límites superior e inferior, que se tiene contemplado como máximo y mínimo en las lecturas de la variable seleccionada. Una vez iniciada la adquisición el sistema evalúa la lectura y determina si es un evento, a partir de estos valores límite y crea un archivo llamado "Eventos.csv", en el que se almacenan todas las lecturas que tengan por lo menos un evento.

j) Estado actual de la configuración de los canales

En esta tabla de muestra la configuración de cada uno de los canales del convertidos A/D del microcontrolador, además de los limites superior e inferior que determinan los eventos.

4.4. Registro de eventos

Para poder registrar los eventos, el software de adquisición requiere de la creación de una carpeta en el disco C: con el nombre de "Datos", es muy importante la creación de esta carpeta ya que si no es realizada esta acción no será capaz el software de registrar los datos y los eventos ocurridos durante su funcionamiento.

Dentro de la carpeta "Datos" se crea el archivo "Config.csv" automáticamente cuando se configura por primera vez uno o todos los canales de la tarjeta electrónica a través del software en la Fig. 4.3i, dentro de él este muestran la configuración como aparecen en la Fig. 4.3j.

Al iniciar la adquisición de los datos se crea un archivo .csv con la siguiente estructura: "Datos*mmddaa*diadelasemana.csv"

Ejemplos:

"Datos082913jueves.csv"
"Datos082813miércoles.csv"

En los cuales se encuentran registradas las lecturas de cada canal, dependiendo de la tasa de muestreo que se haya definido por el usuario. El orden de los datos esta dado de izquierda a derecha, comenzando con el canal *Ch0* hasta *Ch7*, esto corresponde de la columna *"A"* hasta la *"H"* respectivamente, para este ejemplo las columnas *A y B*, corresponden a V_{AC}; *C y D*, a I_{AC}*; E y F* a Temperatura; *G y H*, no se usan en este momento.

En la columna *I* se muestra la hora a la que fue tomada esa cadena de datos con un formato de 24 horas (hh:mm:ss). Por último la columna *J* muestra la fecha en que fueron adquiridos los datos de esa cadena, esta fecha se encuentra en formato "dd/mm/aa".

La estructura antes mencionada se puede apreciar en la Tabla 3.

Tabla 3. Estructura en la que se almacenan los datos por el software de adquisición.

A	B	C	D	E	F	G	H	I	J
119.58	0.51	12.49	12.49	22.3	-0.77	No	No	23:59:49	3/8/13
119.58	0.25	12.49	12.44	22.22	-0.77	No	No	23:59:54	3/8/13
119.45	0.38	12.51	12.51	22.22	-0.77	No	No	23:59:59	3/8/13
119.58	0.38	12.51	12.49	22.22	-0.85	No	No	00:00:04	4/8/13
119.58	0.38	12.51	12.49	22.22	-0.85	No	No	00:00:09	4/8/13
119.45	0.25	12.49	12.51	22.15	-0.85	No	No	00:00:14	4/8/13
119.83	0.38	12.51	12.49	22.22	-0.77	No	No	00:00:19	4/8/13
119.71	0.38	12.51	12.46	22.22	-0.77	No	No	00:00:24	4/8/13

Además de los dos archivos antes mencionados se crea un tercer archivo en el cual son registrados los eventos definidos por el usuario. Esta cadena muestra la misma estructura que el archivo de adquisición de datos, además de presentarnos en la columna *K* dentro del mismo renglón el primer canal que haya registrado un evento, comenzando por el *Ch0* y en orden ascendente; en la columna *L* se muestra el tipo de sensor correspondiente a ese canal y en las siguientes 2 columnas los valores Máximo y Mínimo, respectivamente, que determinan los límites de un evento.

En la Tabla 4 se muestra un ejemplo de esta estructura.

Tabla 4. Estructura en la que se almacenan los eventos por el software de adquisición.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
120.85	126.19	No	No	22.46	No	No	No	22:09:08	26/8/13	Canal 1	Voltaje AC	126	120
119.96	124.28	No	No	21.36	No	No	No	07:51:21	27/8/13	Canal 0	Voltaje AC	123	120
120.34	121.61	No	No	24.02	No	130	130	15:28:54	27/8/13	Canal 4	Temperatura	24	18
120.34	121.61	No	No	24.02	No	130	130	15:29:9	27/8/13	Canal 4	Temperatura	24	18

Resultados

Al final del proyecto se logró establecer la comunicación entre la computadora que aloja el software de adquisición y la tarjeta electrónica por medio de una red local, bajo este esquema es posible instalar la tarjeta en sitio y ser monitoreada vía remota.

Se logró desarrollar el SADRE con el cual es posible monitorear las variables eléctricas y la temperatura a las que se encuentran operando los circuitos instalados en campo.

El software de adquisición es capaz de recibir y almacenar la información, configurar la tarjeta y registrar eventos a partir de los datos que genera la tarjeta.

Al realizar pruebas con los distintos sensores obtuvimos ciertas variaciones con respecto a las mediciones realizadas con nuestro instrumento de comparación (multímetro Fluke179), esto se debe principalmente a la resolución que posee nuestro convertidor A/D, así como a la calibración final de las etapas de acondicionamiento.

Dado que el resultado de la conversión analógico-digital está en función de la polarización del microcontrolador, es fundamental proporcionarle voltajes estables. Si esta condición no se cumpliera la calidad de todos los datos adquiridos se vería comprometida.

Se muestran a continuación (Tabla 5) los resultados obtenidos con cada sensor respecto a la lectura de nuestro instrumento base, que también fue nuestro punto de comparación para la calibración de los sensores.

Tabla 5. Comparación de lecturas Tarjeta *VS* Fluke179.

Canal	Tipo	Fluke179	Micro.	E. relativo	E. absoluto	Sensibilidad	Rango
Ch0	V_{AC} [V]	120.57	120.62	0.053	0.04%	0.127	0-130
Ch1	V_{AC} [V]	120.50	120.62	0.123	0.10%	0.127	0-130
Ch2	I_{AC} [A]	12.07	12.05	-0.019	0.16%	0.024	0-25
Ch3	I_{AC} [A]	12.15	12.14	-0.012	0.10%	0.024	0-25
Ch4	Temp. [°C]	30.00	30.53	0.53	1.75%	0.078	-15 a 65
Ch5	Temp. [°C]	30.20	29.67	-0.53	1.74%	0.078	-15 a 65

Los valores mostrados anteriormente se obtuvieron a partir del promedio de 10 muestras tomadas para cada variable, tanto por la tarjeta de adquisición como por el multímetro Fluke179.

Con los datos adquiridos por la tarjeta electrónica, el software de adquisición es capaz de generar archivos en los que se registra el comportamiento de las variables de interés así como los eventos ocurridos durante su adquisición, dejando al usuario su posterior análisis.

Conclusiones

Los resultados obtenidos son favorables porque es posible registrar la actividad de las variables a medir, acotamos las magnitudes de los datos recibidos y al final se logró el registro de eventos. En estos momentos existen ya ideas para mejorar el diseño, pero esta tesis es el inicio de un proyecto de largo alcance.

La calidad de los sensores es muy buena y aunque existen dispositivos con digitalizadores de mejor resolución, podemos decir que las variables que deseamos medir no requieren de una sensibilidad mucho mayor.

El proyecto tuvo un costo aproximado de $ 4,500.00 M.N. sin embargo, gracias a que nos apoyamos en las empresas fabricantes de los dispositivos solicitando muestras, fue posible reducir el mismo.

Debido a que el Servicio Sismológico cuenta con los equipos UDS1100 fue posible hacer pruebas de manera inmediata y aunque el costo del proyecto lo contempla, no hubo necesidad de comprarlo, gracias a esto y aunado a las muestras que se solicitaron se logró reducir un 40% aproximadamente del costo total.

Dado que todos los sensores se están utilizando en el rango donde el fabricante nos asegura la linealidad de los mismos y al utilizar todos nuestros componentes electrónicos en dicha región, por lo tanto, podemos decir que nuestros circuitos son lineales.

Bibliografía

Libros

1.- Pallas Areny, Ramón. *"Sensores y Acondicionamiento de Señal"*. 3era Edición, Ed. Alfa Omega

2.- Franco, Sergio. "*Diseño con Amplificadores Operacionales y Circuitos Integrados Analógicos"*. 3era Edición, Ed. Mc Graw Hill

3.- Schildt, Herbert. *"C Manual de Referencia"*. 4ta Edición, Ed. Mc Graw Hill

Manuales

4.- Servicio Sismológico Nacional. "*Definición de un Estándar Nacional de los Observatorios Sísmicos y Acelerográficos para su Integración a la Red Sísmica Mexicana"*

5.- Instituto de Ingeniería. *"Definición de un Estándar Nacional de las estaciones Sísmicas y Acelerográficas para su integración a la Red Sísmica Mexicana (Estaciones Acelerográficas)"*

Mesografía

6.- Hoja de datos de Microcontrolador PIC18F4550
http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010300

7.- Hoja de datos del sensor de corriente SCT 0400-025
http://mexico.newark.com/magnelab/sct-0400-025/current-transformer-splitcore-25a/dp/76R0503

8.- Hoja de datos del sensor de temperatura RTD, HEL-705-U-1-12-00
http://www.honeywellscportal.com//product%20page?pr_id=23055

http://mexico.newark.com/honeywell-s-c/hel-705-u-1-12-00/platinum-rtd-temperature-sensor/dp/15M0243?ref=lookahead

9.- Hoja de datos del Amplificador de instrumentación INA2126
http://www.ti.com/product/ina2126

10.- Hoja de datos del amplificador operacional TL084
http://www.ti.com/product/tl084

11.- UDS 1100 User Guide
http://www.lantronix.com/support/downloads/?p=UDS1100

Anexos

Hoja de datos del sensor de corriente

SCT-0400 Split-Core AC Current Sensor

Measures AC current passing through the center

Electrical Specifications

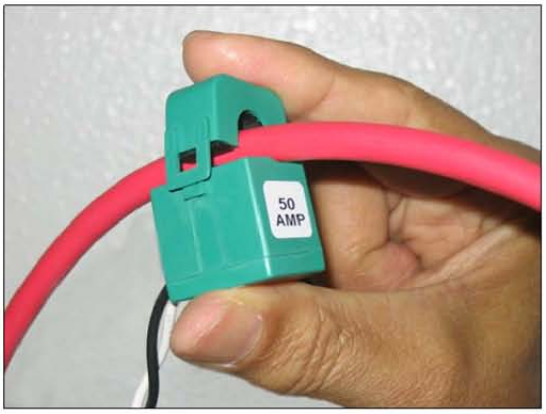
- Output 0.333 V at rated current
- Accuracy ±1%
- Phase angle < 2 degrees
- Rated accuracy at 10% to 130% of rated current
- Operational from 50Hz to 400Hz

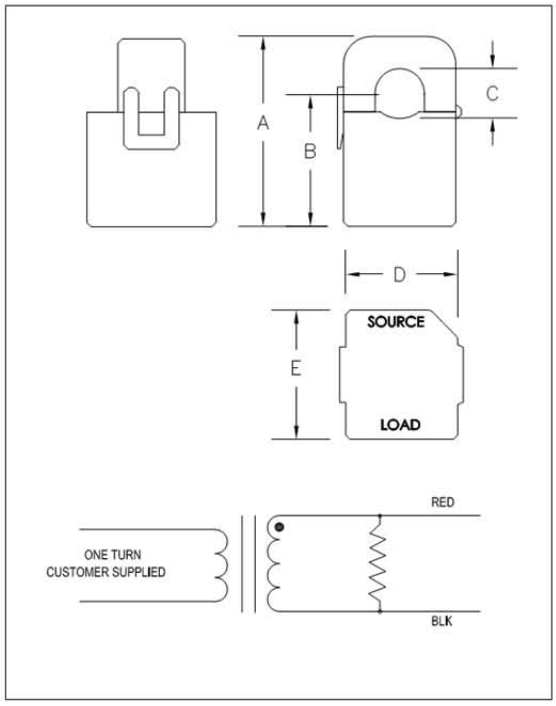
Mechanical Specifications

- 0.40" [10.16 mm] opening, split core
- Leads-8ft. [2.44 m] twisted pair, jacketed, 24 AWG

Features

- UL recognized, CE RoHS compliant
- Precision burden resistor
- Other ratings and specifications are available

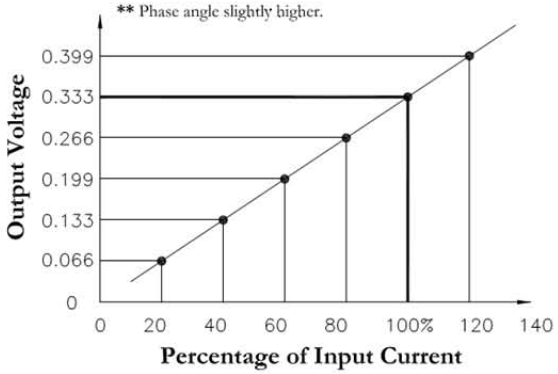




MAGNELAB P/N	RATED CURRENT (Amps)	DIMENSIONS INCHES [MILLIMETERS]				
		A	B	C	D	E
SCT-0400-000*	0-75	1.55 [39.55]	1.09 [27.69]	0.40 [10.16]	1.00 [25.40]	1.05 [26.67]
SCT-0400-005**	5					
SCT-0400-010	10					
SCT-0400-015	15					
SCT-0400-020	20					
SCT-0400-025	25					
SCT-0400-030	30					
SCT-0400-050	50					
SCT-0400-075	75					

* Products with -000 have no burden resistor, and the output voltage is clamped to 22 Volts.

** Phase angle slightly higher.



Phone: (303) 772-9100
Fax: (303) 772-9400
E-Mail: info@magnelab.com
Web Site: www.magnelab.com

MAGNELAB INCORPORATION
LONGMONT, COLORADO 80501

Hoja de datos del sensor de temperatura

Temperature Sensors
Platinum RTDs

HEL-700 Series

FEATURES
- Linear resistance vs temperature
- Accurate and Interchangeable
- Excellent stability
- Teflon or fiberglass lead wires
- Wide temperature range
- Ceramic case material

TYPICAL APPLICATIONS
- HVAC – room, duct and refrigerant equipment
- Instrument and probe assemblies – temperature compensation
- Process control – temperature regulation

HEL-700 Series elements are fully assembled, ready to use directly or in probe assemblies without the need for fragile splices to extension leads.

The 1000Ω, 375 alpha version, provides 10X greater sensitivity and signal-to-noise. Optional NIST calibrations improve accuracy to ±0.03°C at 0°C.

ORDER GUIDE

HEL-705	28 ga. TFE Teflon, 2-wire only				
HEL-707	28 ga. Fiberglass, 2-wire only				
HEL-711	28 ga. TFE Teflon (2-wire 1000Ω, 3-wire 100Ω)				
HEL-712	28 ga. Fiberglass (2-wire 1000Ω, 3-wire 100Ω)				
HEL-716	24 ga. TFE Teflon (2-wire 1000Ω, 3-wire 100Ω)				
HEL-717	24 ga. Fiberglass (2-wire 1000Ω, 3-wire 100Ω)				
	-U	1000Ω, 0.00375 Ω/Ω/°C			
	-T	100Ω, 0.00385 Ω/Ω/°C DIN Standard			
		-0	±0.2% Resistance Trim (Standard)		
		-1	±0.1% Resistance Trim (Optional)		
			-12	Lead wire length, 12 inches	
				-00	No NIST calibration
				-C1	NIST @ 0°C
				-C2	NIST @ 0 & 100°C
				-C3	NIST @ 0, 100 & 260°C

MOUNTING DIMENSIONS (for reference only)

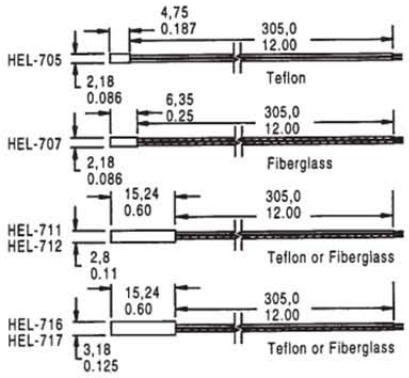


Fig. 1: Wheatstone Bridge 2-Wire Interface

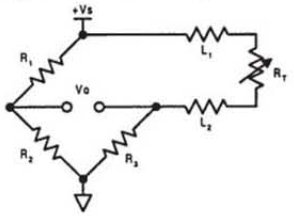


Fig. 2: Linear Output Voltage

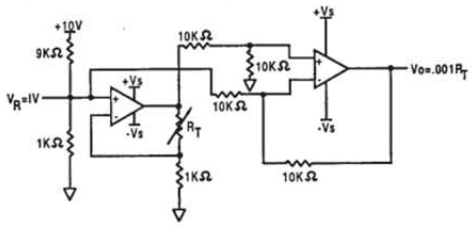
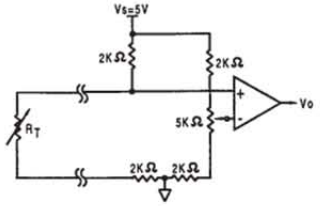


Fig. 3: Adjustable Point (Comparator) Interface



CAUTION
PRODUCT DAMAGE
The inherent design of this component causes it to be sensitive to electrostatic discharge (ESD). To prevent ESD-induced damage and/or degradation, take normal ESD precautions when handling this product.

Temperature

Temperature Sensors
Platinum RTDs

HEL-700 Series

FUNCTIONAL BEHAVIOR

$R_T = R_o(1+AT+BT^2-100CT^3+CT^4)$
RT = Resistance (Ω) at temperature T (°C)
R_o = Resistance (Ω) at 0°C
T = Temperature in °C

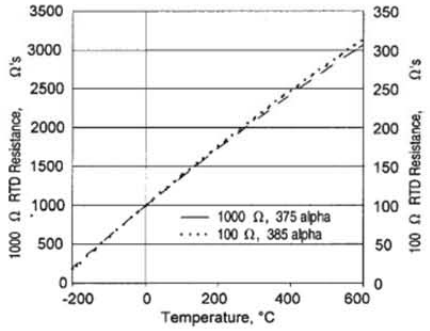
$A = \alpha + \frac{\alpha\,\delta}{100}$ $\quad B = \frac{-\alpha\,\delta}{100^2}$ $\quad C_{T<0} = \frac{-\alpha\,\beta}{100^4}$

CONSTANTS

Alpha, α (°C⁻¹)	0.00375 ±0.000029	0.003850 ±0.000010
Delta, δ (°C)	1.605 ± 0.009	1.4999 ± 0.007
Beta, β (°C)	0.16	0.10863
A (°C⁻¹)	3.81×10^{-3}	3.908×10^{-3}
B (°C⁻²)	-6.02×10^{-7}	-5.775×10^{-7}
C (°C⁻⁴)	-6.0×10^{-12}	-4.183×10^{-12}

Both β = 0 and C = 0 for T>0°C

RESISTANCE VS TEMPERATURE CURVE



ACCURACY VS TEMPERATURE

Tolerance	Standard ±0.2%		Optional ±0.1%	
Temperature (°C)	±ΔR* (Ω)	±ΔT (°C)	±ΔR* (Ω)	±ΔT (°C)
−200	6.8	1.6	5.1	1.2
−100	2.9	0.8	2.4	0.6
0	2.0	0.5	1.0	0.3
100	2.9	0.8	2.2	0.6
200	5.6	1.6	4.3	1.2
300	8.2	2.4	6.2	1.8
400	11.0	3.2	8.3	2.5
500	12.5	4.0	9.6	3.0
600	15.1	4.8	10.4	3.3

*1000Ω RTD. Divide Δ by 10 for 100Ω RTD.

NIST CALIBRATION

NIST traceable calibration provides resistance readings at 1, 2 or 3 standard temperature points to yield a resistance versus temperature curve with 10x better accuracy.

Calibration	1 Point	2 Point	3 Point
T (°C)	±ΔT (°C)	±ΔT (°C)	±ΔT (°C)
-200	0.9	—	—
-100	0.5	0.27	0.15
0	0.03	0.03	0.03
100	0.4	0.11	0.07
200	0.8	0.2	0.08
300	1.2	0.33	6.2
400	1.6	0.5	8.3
500	2.0	0.8	9.6
600	2.6	1.2	10.4

SPECIFICATIONS

Sensor Type	Thin film platinum RTD; R_o = 1000 Ω @ 0°C; alpha = 0.00375 Ω/Ω/°C R_o = 100 Ω @ 0°C; alpha = 0.00385 Ω/Ω/°C
Temperature Range	TFE Teflon: −200° to +260°C (−320° to +500°F) Fiberglass: −75° to +540°C (−100° to +1000°F)
Temperature Accuracy	±0.5°C or 0.8% of temperature, °C (R_o ±0.2% trim), whichever is greater ±0.3°C or 0.6% of temperature, °C (R_o ±0.1% trim), whichever is greater (optional)
Base Resistance and Interchangeability, $R_o \pm \Delta R_o$	1000 ± 2 Ω (±0.2%) @ 0°C 1000 ± 1 Ω (±0.1%) @ 0°C (optional)
Linearity	±0.1% of full scale for temperatures spanning −40° to +125°C ±2.0% of full scale for temperatures spanning −75° to +540°C
Time Constant	<0.5 sec. 0.85 inch O.D. in water at 3 ft/sec; <1.0 sec, 0.85 inch O.D. in still water
Operating Current	2 mA maximum for self heating errors of <1°C; 1 mA recommended
Stability	<0.25°C/year; 0.05°C per 5 years in occupied environments
Self Heating	<15 mW/°C for 0.85 O.D. typical
Insulation Resistance	>50 MΩ at 50 VDC at 25°C
Construction	Alumina case; Epoxy potting (Teflon leads); Ceramic potting (fiberglass leads)
Lead Material	Nickel coated stranded copper, Teflon or Fiberglass insulated

Hoja de datos del amplificador de instrumentación

Burr-Brown Products from Texas Instruments

INA126
INA2126

SBOS062A – JANUARY 1996 – REVISED AUGUST 2005

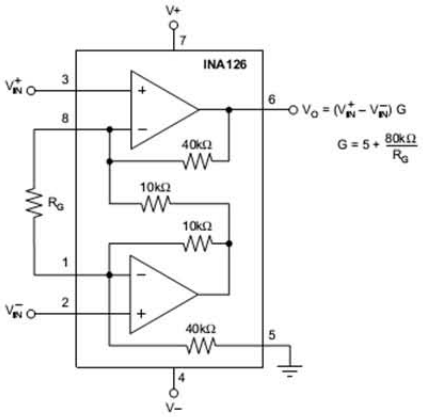
*Micro*POWER INSTRUMENTATION AMPLIFIER
Single and Dual Versions

FEATURES

- LOW QUIESCENT CURRENT: 175μA/chan.
- WIDE SUPPLY RANGE: ±1.35V to ±18V
- LOW OFFSET VOLTAGE: 250μV max
- LOW OFFSET DRIFT: 3μV/°C max
- LOW NOISE: 35nV/√Hz
- LOW INPUT BIAS CURRENT: 25nA max
- 8-PIN DIP, SO-8, MSOP-8 SURFACE-MOUNT DUAL: 16-Pin DIP, SO-16, SSOP-16

APPLICATIONS

- INDUSTRIAL SENSOR AMPLIFIER: Bridge, RTD, Thermocouple
- PHYSIOLOGICAL AMPLIFIER: ECG, EEG, EMG
- MULTI-CHANNEL DATA ACQUISITION
- PORTABLE, BATTERY OPERATED SYSTEMS

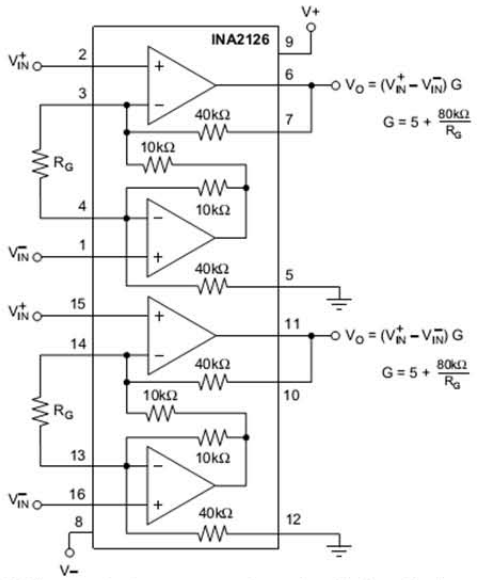


DESCRIPTION

The INA126 and INA2126 are precision instrumentation amplifiers for accurate, low noise differential signal acquisition. Their two-op-amp design provides excellent performance with very low quiescent current (175μA/channel). This, combined with a wide operating voltage range of ±1.35V to ±18V, makes them ideal for portable instrumentation and data acquisition systems.

Gain can be set from 5V/V to 10000V/V with a single external resistor. Laser trimmed input circuitry provides low offset voltage (250μV max), low offset voltage drift (3μV/°C max) and excellent common-mode rejection.

Single version package options include 8-pin plastic DIP, SO-8 surface mount, and fine-pitch MSOP-8 surface-mount. Dual version is available in the space-saving SSOP-16 fine-pitch surface mount, SO-16, and 16-pin DIP. All are specified for the –40°C to +85°C industrial temperature range.



Hoja de datos del reloj de tiempo real

DALLAS SEMICONDUCTOR

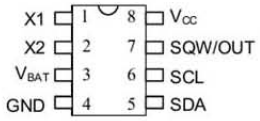
DS1307
64 x 8 Serial Real-Time Clock

www.maxim-ic.com

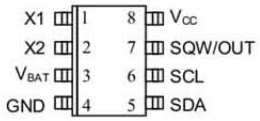
FEATURES

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range: -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized

PIN ASSIGNMENT



DS1307 8-Pin DIP (300-mil)



DS1307 8-Pin SOIC (150-mil)

ORDERING INFORMATION

DS1307	8-Pin DIP (300-mil)
DS1307Z	8-Pin SOIC (150-mil)
DS1307N	8-Pin DIP (Industrial)
DS1307ZN	8-Pin SOIC (Industrial)

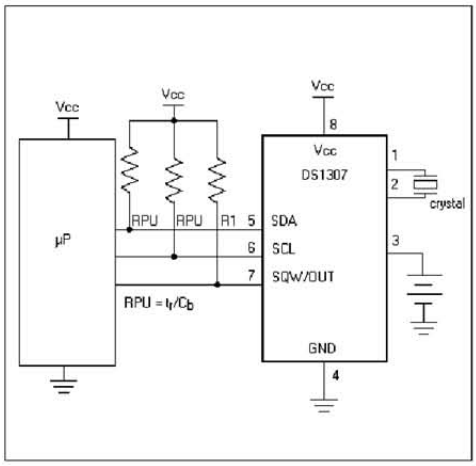
PIN DESCRIPTION

V_{CC}	- Primary Power Supply
X1, X2	- 32.768kHz Crystal Connection
V_{BAT}	- +3V Battery Input
GND	- Ground
SDA	- Serial Data
SCL	- Serial Clock
SQW/OUT	- Square Wave/Output Driver

DESCRIPTION

The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.

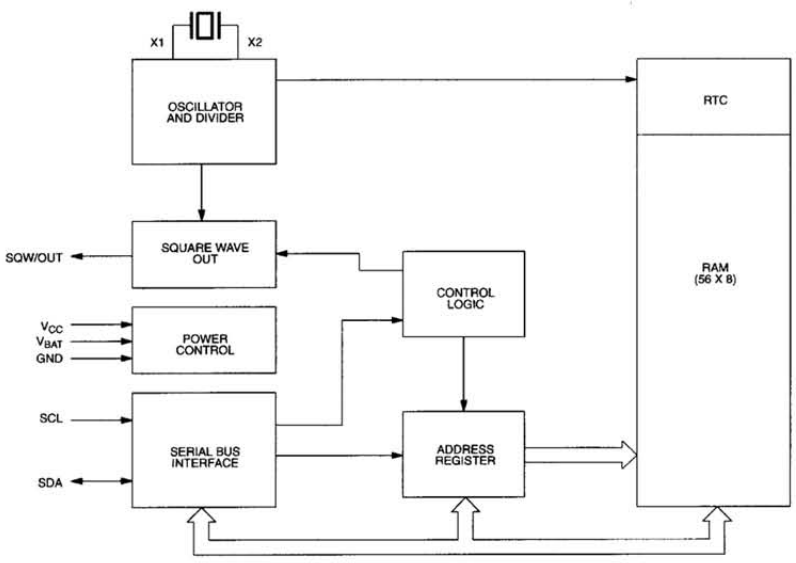
TYPICAL OPERATING CIRCUIT



OPERATION

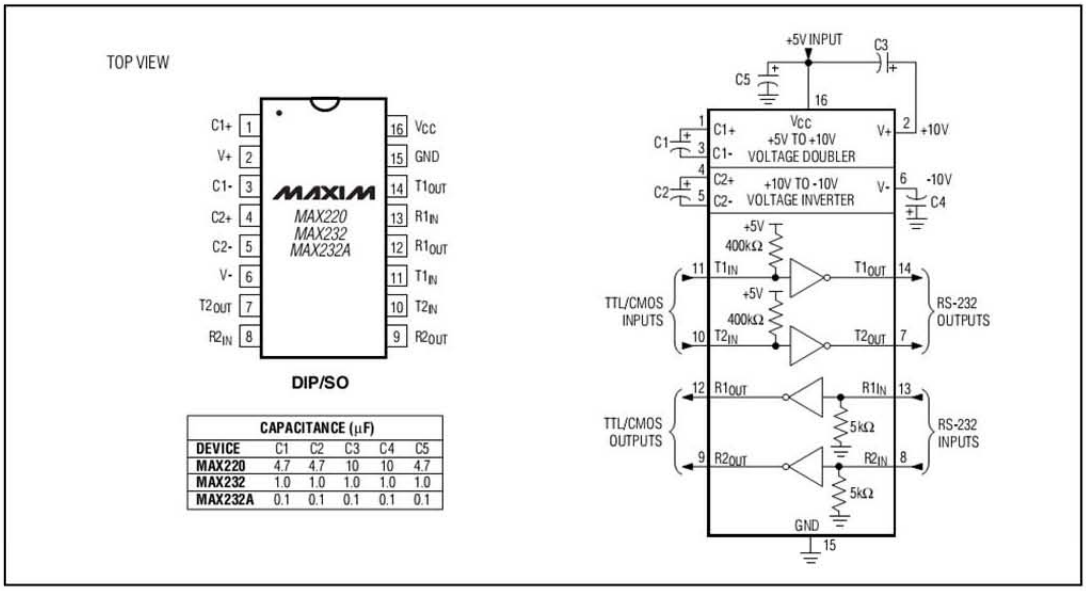
The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below 1.25 x V_{BAT} the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out of tolerance system. When V_{CC} falls below V_{BAT} the device switches into a low-current battery backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than V_{BAT} + 0.2V and recognizes inputs when V_{CC} is greater than 1.25 x V_{BAT}. The block diagram in Figure 1 shows the main elements of the serial RTC.

DS1307 BLOCK DIAGRAM Figure 1



Hoja de datos del acondicionamiento de niveles TTL a RS-232

+5V-Powered, Multichannel RS-232 Drivers/Receivers



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

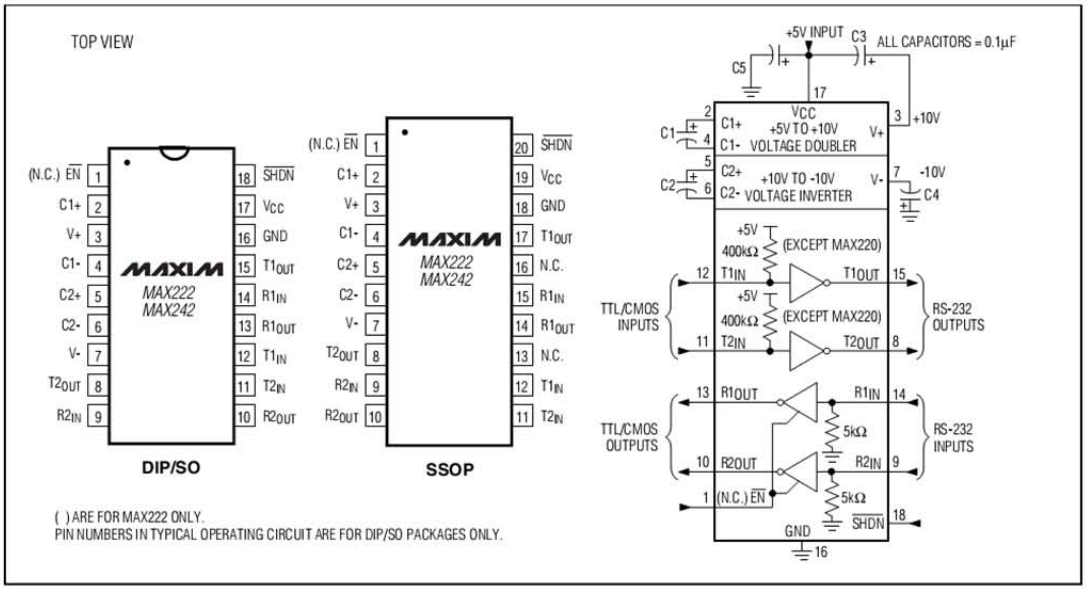
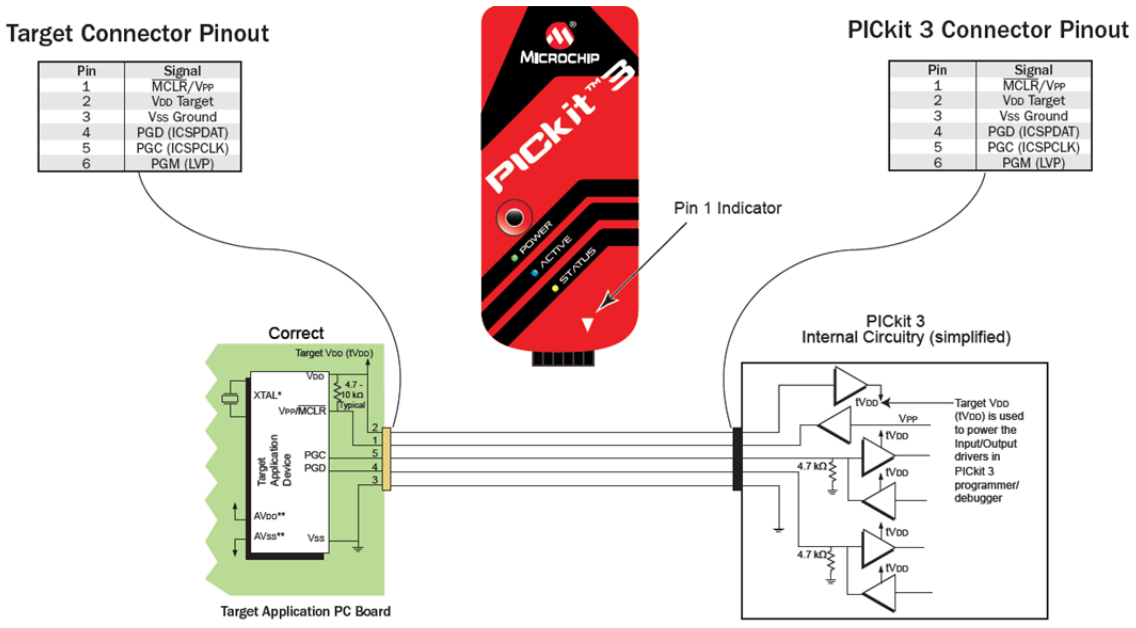


Diagrama de conexión para el uso del programador



A continuación se muestra el código de las funciones principales que integran la programación del microcontrolador.

Funciones de Cabecera

```
#fuses XT,NOWDT,NOPUT,NOPROTECT,NOBROWNOUT,MCLR,NOLVP
#device adc=10
#use delay(clock=4000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#use I2C(master, sda=PIN_B0, scl=PIN_B1)


#include <stdlib.h>
*/
#include <18F4550.h>


#FUSES NOWDT                 //No Watch Dog Timer
#FUSES WDT128                //Watch Dog Timer uses 1:128 Postscale
#FUSES PLL1                 //No PLL PreScaler
#FUSES CPUDIV1               //No System Clock Postscaler
#FUSES INTEC_IO              //Internal Clock, EC used by USB, I/O on RA6
#FUSES NOFCMEN                //Fail-safe clock monitor disabled
#FUSES NOIESO               //Internal External Switch Over mode disabled
#FUSES NOBROWNOUT                //No brownout reset
#FUSES NOVREGEN              //USB voltage regulator disabled
#FUSES NOLPT1OSC             //Timer1 configured for higher power operation
```

```
#FUSES NOLVP                    //No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
#FUSES NOXINST                  //Extended set extension and Indexed Addressing mode disabled (Legacy
mode)

#device adc=10
#use I2C(master, sda=PIN_B0, scl=PIN_B1)

#use delay(clock=8000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)

#include <stdlib.h>
#include "funciones.h"
//}
        //RTC
        void RTCini(void);
        void UpDateClock(void);

char opcion='0';
int J=0,timer_calc=0xffff;
unsigned int16 d=0, SRate=32;
```

Interrupción por comunicación serial

En esta interrupción se lleva a cabo todo el proceso que involucra la comunicación entre el microcontrolador y una PC a través de la interface serial

```
#int_rda
//{
void serial_isr(void)
{
        //menu{
        clear_interrupt(int_rda);
        char C, confi;

        C=getc();

        switch(opcion)
        {
                case '0':
                {
                        switch(C)
                        {
                                case '1':
                                        printf("%c\n\n\r",C);
                                        status();
                                        printf("\n\r%Lu",SRate);
                                break;

                                case '2':
                                {
                                        printf("%c\n\n\r",C);

                                        SampleRate();
                                        opcion='2';
                                }
```

```
                        break;

                        case '3':
                        {
                                printf("%c\n\n\r",C);

                                InicioParo();
                                opcion='3';
                        }
                        break;

                        case '4':
                                printf("%c\n\n\r",C);

                                UpDateClock();
                                puts("hh:mm:ss dd/mm/aa\n");
                                u=0;
                                opcion='4';
                        break;

                        case '5':
                                printf("%c\n\n\r",C);

                                Tipo_sensor();
                                opcion='5';

                        break;

                        case '?':
                                //printf("%c\n\n\r",C);

                                ayuda();
                        break;

                        case '6':
                                printf("transmit: %d \n\r TRANS: %d",transmit,TRANS);
                        break;

                        case 27:                        //ESC detiene la adquisición
                                transmit=0;
                                TRANS=0;
                                puts("\rSelecciona una opcion: 1,2,3,4,5,Esc,?    ");
                                opcion='0';
                        break;

                        default:
                                menu();
                                opcion='0';
                        break;
                        }
}
break;

//
case '2':
{
```

```
            printf("%c\n\r",C);
            printf("Selecciona una opcion: 1,2,3,4,5,Esc,?  ");
            BaudRate=C;//leer la variable
            opcion='0';


}
break;

case '3':
{
            printf("  %c\n\n\r",C);
            opcion='0';

            if(C=='Y' || C=='y')
            {

                        if(transmit && TRANS)
                        {
                                    transmit=0;
                                    TRANS=0;
                                    puts("Selecciona una opcion: 1,2,3,4,5,Esc,?\n");
                        }
                        else
                        {
                                    TRANS=1;
                                    transmit=TRANS;
                        }
            }

            else
            {
                        puts("Selecciona una opcion: 1,2,3,4,5,Esc,?\n");
            }

}
break;

case '4':
{
            printf("%c",C);
            updatechar[w]=C;
            w++;
            if(C!=0X0D)
            {
                        if(C!=':' && C!=' ' && C!='/')
                        {
                                    update[u][v]=C-0X30;
                                    v=!v;
                        }
                        else u++;
            }

            else if(u!=5)
            {
                        opcion='0';
```

```
                    u=0;
                    v=0;
                    w=0;
            }
            else
            {
                    puts("\n\rla Fecha y Hora son correctas?\n");
                    for(w=0;w<=16;w++)
                    {
                            printf("%c",updatechar[w]);
                    }
                    puts(" Y/N ?\n");
                    opcion='A';
                    u++;

                    v=0;
                    w=0;
            }
    }
    break;

    case '5':
    {
            printf("%c\n\r",C);
            if(C!=0X0D && C<='7' && C>= '0')
            {
                    x=C-0x30;
                    printf("\r\nIngresa la nueva configuracion\r\n");
                    printf("1.- Sensor de Voltaje (AC)\r\n");
                    printf("2.- Sensor de Corriente (AC)\r\n");
                    printf("3.- Sensor de Temperatura\r\n");
                    printf("4.- Deshabilitar\n\r");
                    opcion='B';
            }
            else
            {
                    opcion='0';
                    puts("Operacion No Valida!!!\r\n");
                    puts("Selecciona una opcion: 1,2,3,4,5,Esc,?   ");
            }

    }

    break;

    case 'A':
    {
            printf("   %c\n\n\r",C);
            if(C=='y' || C=='Y')
            {
                    SetDate();
            }
            else
            {
                    opcion='0';
```

```
                puts("Selecciona una opcion: 1,2,3,4,5,Esc,?\n");

        }

        opcion='0';
}
break;

case 'B':
{
        printf("\n%c\n\n\r",C);
        if(C!=0X0D && C<='4' && C>= '1')
        {
                Tipo[x]=C;//x es el número del canal
                switch(Tipo[x])
                {
                        case '1':
                                printf("Ch %c: Voltaje\n\r",(x+0x30));
                        break;

                        case '2':
                                printf("Ch %c: Corriente\n\r",(x+0x30));
                        break;

                        case '3':
                                printf("Ch %c: Temperatura\n\r",(x+0x30));
                        break;

                        case '4':
                                printf("Ch %c: Deshabilitado\n\r",(x+0x30));
                        break;

                        default:
                                printf("Ch %c: Deshabilitado\n\r",(x+0x30));
                        break;
                }

                puts("Desea configurar otro Canal? Y/N?   ");


                opcion='C';
        }
        else
        {
                opcion='0';
                puts("Operacion No Valida!!!\n");
                puts("Selecciona una opcion: 1,2,3,4,5,Esc,?\n");
        }


}
break;

case 'C':
{
        printf("\n%c\n\n\r",C);
```

```
                        if(C=='y' || C=='Y')
                        {
                                opcion='5';
                                Tipo_sensor();

                        }
                        else
                        {
                                opcion='0';
                                puts("Selecciona una opcion: 1,2,3,4,5,Esc,?\n");
                        }
                }
                break;

                default:
                {
                        opcion='0';
                }
                break;
        }
}
//}
```

Interrupción por Timer 1

En esta función se lleva a cabo la cuenta de ciclos que deben cumplirse para ejecutar la rutina de adquisición y envió de datos.

```
#INT_TIMER1
//{
void timer_interrupt()
{

        if(d>=SRate)
        {
                transmit=1;
                d=0;
                set_timer1(0x0BDB);
        }

        d=d+1;
        set_timer1(0x0BDB);
}
```

Función Main

En esta función se encuentra la configuración inicial de algunos parámetros del microcontrolador y del proceso principal que es la adquisición de datos y su transmisión.

```
void main()
{
        setup_oscillator(OSC_8MHZ|OSC_INTRC|OSC_PLL_OFF);

        enable_interrupts(int_rda);
```

```
enable_interrupts(global);

setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);   // setup interrupts
enable_interrupts(INT_TIMER1);

setup_adc(ADC_CLOCK_INTERNAL);           // Built-in A/D setup function
setup_adc_ports(AN0_TO_AN7);

RTCini(); //manda a llamar la inicialización del reloj

char local=0;
transmit=TRANS;
//BaudRate='1';

while(1)
{
        char x;

        if(transmit==1 && TRANS==1)
        {
                //output_c(1);
                for(x=0;x<=7;x++)
                {
                        switch(Tipo[x])
                        {
                                case '1':
                                Prom(x);
                                Voltaje(1);
                                break;

                                case '2':
                                Prom(x);
                                Corriente(1);
                                break;
                                case '3':
                                Prom(x);
                                Temperatura(x);
                                break;

                                case '4':
                                printf("No;");
                                break;

                                default:
                                printf("Default;");
                                break;
                        }

                }

                Tiempo(1);
                transmit=0;
        }

        switch(BaudRate)
        {
```

```
                        case '1':
                        {
                                SRate=32;//1 cada 1seg

                        }
                        break;

                        case '2':
                        {
                                SRate=1920;//1 cada min

                        }
                        break;

                        case '3':
                        {
                                SRate=9600;//1 cada 5 min

                        }
                        break;

                        case '4':
                        {
                                SRate=19200;//1 cada 10 min
                        }
                        break;

                        default:
                        {
                                SRate=32;//delay_ms(800);//1 cada 1seg
                        }
                        break;
                }

                //while(sleep && !TRANS && !transmit)
                //sleep();
        }
        }
```

Función de Inicialización del Reloj de Tiempo Real RTC

Esta función realiza la configuración inicial del módulo RTC y al ser ejecutada carga valores por default si el modulo no ha sido configurado previamente.

```
void RTCini(void)
{

        char rtc=0;

        i2c_start();        // Start
        i2c_write(0xD0);    // data direction
        i2c_write(0x08);    // data direction
        i2c_start();                    // ReStart
        i2c_write(0xD1); // direction RTC modo Lectura
```

```
        rtc=i2c_read(0);   //lectura
        i2c_stop();                             // paro


        if(rtc!=0x48)
        {
                        i2c_start();        // Start
                        i2c_write(0xD0);    // data direction
                        i2c_write(0x00);
                        i2c_write(0x50); //50 seg
                        i2c_write(0x46); //46 min
                        i2c_write(0x09); //9hrs formato 24hrs
                        i2c_write(0x02); //dia 2
                        i2c_write(0x10); //fecha
                        i2c_write(0x02); //mes
                        i2c_write(0x13); //año
                        i2c_write(0x10); //control
                        i2c_write(0x01); //este dato está en la localidad de la RAM
                        i2c_stop();
        }
}
```

Función de Configuración de Hora y Fecha

Esta función se ejecuta cuando se desea ingresar una hora y fecha determinada.

```
void SetDate(void)
{
        char condicion[6],updateRTC[6],x;

        u=0;

        condicion[0]=2;
        condicion[1]=5;
        condicion[2]=5;
        condicion[3]=3;
        condicion[4]=1;
        condicion[5]=9;

        for(x=0;x<=5;x++)
        {
                updateRTC[x]=update[x][0]*16+update[x][1];
        }

        i2c_start();        // Start
        i2c_write(0xD0);    // data direction
        i2c_write(0x00);

        i2c_write(updateRTC[2]);//50 seg
        i2c_write(updateRTC[1]);//50 seg
        i2c_write(updateRTC[0]);//50 seg
        i2c_write(0x01);
        i2c_write(updateRTC[3]);//50 seg
        i2c_write(updateRTC[4]);//50 seg
        i2c_write(updateRTC[5]);
```

```
        i2c_write(0x10); //control
        i2c_write(0x48); //este dato está en la localidad de la RAM
        i2c_stop();


}
```

Estas funciones son las que despliegan los textos del menú se colocan en esta librería para que sea más fácil la lectura y edición del main()

```
//variables del menu
void status(void);
void Tipo_sensor(void);
void SampleRate(void);
void InicioParo(void);
void ayuda(void);
void menu(void);

//variables de operacion
void Prom(char Ch);
void Voltaje(char tr);
void Corriente(char tr);
void Tiempo(char x);
void SetDate(void);
void Temperatura(char Ch);

int16 vin[60],VpV,VpV2,vmed=0;
float Vfloat=0,Ifloat=0;
char u=0,v,w,x=0,update[6][2],updatechar[18];
char  Tipo[8]={'1','1','2','2','3','3','4','4'},BaudRate='1',transmit=0,TRANS=0;//Tipo es el tipo de sensor del canal elegido va del Ch:0 al Ch:7
```

FunciónProm()

Esta función realiza el promedio de las lecturas adquiridas y almacenadas en un vector.

```
void Prom(char Ch)
{
char q=0,w=0;
int16 vsum=0,vprom[4];
  set_adc_channel(Ch);
  delay_ms(10);
for(w=0;w<=3;w++)
  {
for(q=0;q<=59;q++)
{
vin[q]=read_adc();//leer ADC y guardarlo en vin
    }
for(q=0;q<=59;q++)
    {
vsum=(vsum+vin[q]);
}
vprom[w]=vsum/60;
```

```
vsum=0;
  }
for(w=0;w<=3;w++)
  {
vmed=(vmed+vprom[w]);
  }
vmed=vmed/4;
}
```

Función Temperatura()

Esta función hace el cálculo de la temperatura.

```
void Temperatura(char tr)
{
  Vfloat=vmed;
  Vfloat=((Vfloat*80)/1023)-15;
if(tr)
  {
printf("%g; ",Vfloat);
  }

vmed=0;
}
```

Función Voltaje()

Esta función hace el cálculo del voltaje.

```
void Voltaje(char tr)
{
  Vfloat=vmed;
  Vfloat=(Vfloat*130)/1023;
if(tr)
  {
printf("%g; ",Vfloat);
}
vmed=0;
}
```

Función Corriente()

Esta función hace el cálculo del corriente.

```
void Corriente(char tr)
{
  Ifloat=vmed;
  Ifloat=(Ifloat*25/1023);
if(tr)
  {
printf("%g; ",Ifloat);
}
vmed=0;
}
```

Función Tiempo()

Esta función etiqueta la cadena de datos con la hora y fecha en la que se realizó la lectura de las variables.

```
void Tiempo(char x)
{
char i=0,TimeChar[2];
int8 time[3];


  i2c_start();         // Start
  i2c_write(0xD0);     // data direction
  i2c_write(0x00);     // data direction
  i2c_start();
  i2c_write(0xD1);

for(i=2;i>=1;i--)
  {
time[i]=i2c_read();

  }
time[0]=i2c_read(0);
  i2c_stop();


for(i=0;i<=1;i++)
  {
itoa(time[i],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);
putc(':');
  }

itoa(time[2],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);

if(x)
  {
printf("; ");

    i2c_start();         // Start
    i2c_write(0xD0);     // data direction
    i2c_write(0x04);     // data direction
    i2c_start();
    i2c_write(0xD1);

for(i=0;i<=1;i++)
    {
time[i]=i2c_read();

    }
time[2]=i2c_read(0);
```

```
    i2c_stop();


for(i=0;i<=1;i++)
    {
itoa(time[i],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);
putc('/');
    }

itoa(time[2],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);
puts("; \n");
  }
else
  {
putc(' ');

    i2c_start();        // Start
    i2c_write(0xD0);    // data direction
    i2c_write(0x04);    // data direction
    i2c_start();
    i2c_write(0xD1);

for(i=0;i<=1;i++)
    {
time[i]=i2c_read();

    }
time[2]=i2c_read(0);
    i2c_stop();


for(i=0;i<=1;i++)
    {
itoa(time[i],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);
putc('/');
    }

itoa(time[2],16,TimeChar);
printf("%c",TimeChar[0]);
printf("%c",TimeChar[1]);
puts("\n\r");
}



}
```

Función status()

Esta función despliega un estado actual de los sensores y muestra la hora, fecha y tasa de muestreo.

```
void status(void)
{
char x=0;

char g=0XF8;

puts("V2.2 Tesis Mauricio y Adrian \n");
Tiempo(0);

for(x=0;x<=7;x++)
{

switch(Tipo[x])
     {
case '1':
        Prom(x);
Voltaje(0);
printf("Ch%c: %g [V]\n\r",(x+48),Vfloat);
break;
case '2':
        Prom(x);
Corriente(0);
printf("Ch%c: %g [A]\n\r",(x+48),Ifloat);
break;
case '3':
        Prom(x);
Temperatura(0);
printf("Ch%c: %g [%cC]\n\r",(x+48),Vfloat,g);
break;
case '4':
printf("Ch%c: No\n\r",(x+48));
break;
default:
printf("Ch%c: Default\n\r",(x+48));
break;
     }
     //local=local+1;
  }


switch(BaudRate)
  {
case '1':
puts("Sample Rate: 1 muestra por segundo\n");
break;
case '2':
puts("Sample Rate: 1 muestra por minuto\n");
break;
case '3':
puts("Sample Rate: 1 muestra cada 5 minutos\n");
```

```
break;
case '4':
puts("Sample Rate: 1 muestra cada 10 minutos\n");
break;
default:
puts("Sample Rate: DEFAULT\n");
break;
  }
  //printf("%c",BaudRate);
printf("\rSelecciona una opcion: 1,2,3,4,5,Esc,?  ");
}
```

Función Tipo_sensor()

Esta función despliega en pantalla las opciones de configuración de los canales.

```
void Tipo_sensor(void)
{
  //char local='0';

char x;
puts("Canales y Tipos de sensores\n\n");
for(x=0;x<=7;x++)
  {

switch(Tipo[x])
    {
case '1':
printf("Ch%c: Voltaje\n\r",(x+48));
break;
case '2':
printf("Ch%c: Corriente\n\r",(x+48));
break;
case '3':
printf("Ch%c: Temperatura\n\r",(x+48));
break;
case '4':
printf("Ch%c: No\n\r",(x+48));
break;
default:
printf("Ch%c: Default\n\r",(x+48));
break;
    }
    //local=local+1;
  }
printf("Selecciona el canal a configurar\n\rCh: ");
}
void SampleRate(void)
{
printf("Elije la tasa de muestreo\n\r");
printf("1- 1 muestra por segundo \n\r");
printf("2- 1 muestra por minuto \n\r");
printf("3- 1 muestra cada 5 minutos \n\r");
printf("4- 1 muestra cada 10 minutos \n\r");
}
```

```
void InicioParo(void)
{
if(transmit && TRANS)
{
printf("Desea detener la transmision? Y/N  ");
  }

else
  {
printf("Desea comenzar la transmicion? Y/N  ");
  }

}
```

Función UpDateClock()

Esta función despliega la hora y fecha actual previa a su modificación por el usuario.

```
void UpDateClock(void)
{
puts("Actualizacion de la Hora y Fecha\n");
puts("Hora y Fecha actual\n");
Tiempo(0);
}
```

Función ayuda()

Esta función despliega el menú de ayuda con las posibles opciones que puede ingresar el usuario para la configuración de la tarjeta.

```
void ayuda(void)
{
puts("Ayuda\n");
puts("1 - Status\n");
puts("2 - Sample Rate\n");
puts("3 - Transmit\n");
puts("4 - Clock & Date\n");
puts("5 - Sensor Configuration\n");
puts("Esc - Stop Transmit\n");
puts("? - Help\n");
}
void menu(void)
{
printf("\n\rSelecciona una opcion: 1,2,3,4,5,Esc,?   ");
}
```